mathematics

MDPI

Article

An Economic Investment Strategy: Enhanced Golden Sine Optimization Algorithm for Global Optimization and Practical Engineering Applications

Zheming Zhang [1] and Hui Zhang [2,*]

[1] Tandon School of Engineering, New York University, New York, NY 10012, USA; z18221385248@hotmail.com
[2] School of Art and Design, Zhejiang Sci-Tech University, Hangzhou 310018, China
* Correspondence: zhanghui@zstu.edu.cn

Abstract

Cloud task scheduling is a critical optimization problem in cloud computing environments, aiming to allocate computational tasks to appropriate virtual machines while reducing execution time, balancing resource load, and minimizing scheduling cost. However, due to the high dimensionality, nonlinear characteristics, and complex constraints of cloud scheduling scenarios, traditional optimization methods often struggle to obtain high-quality solutions efficiently. To address these challenges, this paper proposes a Multi-strategy Improved Golden Sine Optimization Algorithm (MIGoldSA) for global optimization and cloud task scheduling problems. First, an adaptive chaotic opposition initialization strategy is incorporated to improve the distribution quality and diversity of the initial population. Second, a dynamic elite-guided sine evolution strategy is designed to reduce the dependence on a single best individual and improve the coordination between global exploration and local exploitation. Third, an Economic Investment Strategy is introduced to adaptively allocate search efforts according to the optimization potential of individuals. To verify the effectiveness of MIGoldSA, extensive experiments are conducted on the IEEE CEC2017 and CEC2022 benchmark suites and compared with nine advanced optimization algorithms. The results show that MIGoldSA obtains the best or tied-best mean fitness values on 60 out of 84 benchmark cases, accounting for 71.43% of all test cases. In the Wilcoxon signed-rank test, MIGoldSA achieves 662 wins, 57 ties, and 37 losses among 756 pairwise comparisons, corresponding to an overall win rate of 87.57% and a non-inferiority rate of 95.11%. In addition, the Friedman mean ranks of MIGoldSA are 1.47, 2.00, 3.98, and 4.17 under the four benchmark settings, which are reduced by 85.26%, 79.94%, 45.25%, and 42.32%, respectively, compared with the original GoldSA. Furthermore, the proposed algorithm is applied to cloud task scheduling problems under different task scales. The experimental results show that MIGoldSA maintains competitive time-cost performance and achieves clear reductions in load cost, price cost, and comprehensive scheduling cost. Compared with the original GoldSA, the normalized comprehensive scheduling cost is reduced by approximately 9–14% in small-scale scenarios and approximately 18–21% in large-scale scenarios. Meanwhile, the normalized load cost and price cost are reduced by about 18–25% and 10–18%, respectively, and the time cost shows an approximately 8–12% reduction in large-scale scheduling scenarios. These quantitative results demonstrate that MIGoldSA can improve the optimization accuracy, statistical robustness, and overall scheduling cost efficiency of the original GoldSA on most tested problems.

Check for updates

Academic Editor: Petr Stodola

Received: 28 May 2026
Revised: 27 June 2026
Accepted: 29 June 2026
Published: 7 July 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

Keywords: cloud task scheduling; global optimization; golden sine optimization algorithm; CEC2017; CEC2022; metaheuristic algorithms

MSC: 68W50

1. Introduction

With the rapid development of cloud computing [1], the Internet of Things (IoT) [2], artificial intelligence [3], mobile communication [4], and big data technologies [5], massive computational tasks are continuously generated in modern network environments. Various intelligent applications, such as smart healthcare, intelligent transportation, industrial automation, online social platforms, and scientific data analysis, require increasingly high computing efficiency, storage capacity, and service quality. However, traditional local computing paradigms are often insufficient for processing large-scale and computation-intensive tasks due to their limited computing and storage resources. As a result, cloud computing has become an effective paradigm for providing scalable, flexible, and on-demand computing services. By employing virtualization technology and distributed resource management mechanisms, cloud computing platforms can dynamically allocate computing resources according to user requirements, thereby improving computational efficiency and reducing operational costs [6,7].

Among the key issues in cloud computing systems, cloud task scheduling is one of the most critical and challenging optimization problems. Its main purpose is to assign heterogeneous tasks to appropriate virtual machines or computing resources while optimizing multiple objectives, such as execution time, energy consumption, transmission cost, load balancing, and resource utilization. Nevertheless, due to the dynamic, nonlinear, and highly complex characteristics of cloud environments, cloud task scheduling is generally regarded as an NP-hard problem [8]. As the number of tasks and virtual machines increases, the search space expands rapidly, making it difficult for traditional deterministic methods to obtain high-quality scheduling solutions within an acceptable computational time. Therefore, developing efficient and robust optimization algorithms for cloud task scheduling remains an important research topic.

Metaheuristic optimization algorithms have attracted extensive attention due to their strong global search capability, flexible problem adaptability, and limited dependence on mathematical properties of optimization problems. However, existing metaheuristic algorithms still face several common challenges, including insufficient population diversity, premature convergence, poor adaptability in high-dimensional search spaces, and difficulty in maintaining a proper balance between exploration and exploitation [9,10]. According to the No Free Lunch theorem, no single optimization algorithm can perform best for all optimization problems. Therefore, improving existing algorithms and designing more effective search mechanisms are still necessary for complex global optimization and practical engineering problems.

The Golden Sine Algorithm (GoldSA) is a metaheuristic optimization algorithm inspired by the golden ratio and the periodic characteristics of the sine function [11]. Owing to its simple structure, few control parameters, and easy implementation, GoldSA has been applied to various optimization problems. However, the original GoldSA still has several limitations when dealing with high-dimensional and complex multimodal problems. First, the random initialization mechanism may lead to insufficient population diversity and uneven solution distribution. Second, the sine-guided position update mechanism mainly depends on the current best individual, which may cause population aggregation and increase the risk of premature convergence. Third, the original GoldSA lacks an adaptive

mechanism to dynamically coordinate exploration and exploitation according to the population state. These limitations may reduce its optimization accuracy and robustness in complex search spaces.

To address the above limitations, this paper proposes a Multi-strategy Improved Golden Sine Optimization Algorithm (MIGoldSA) for global optimization and cloud task scheduling problems. It should be noted that the proposed method is not intended to introduce a completely new optimization paradigm, but to refine the search behavior of the original GoldSA through three targeted modifications. These modifications are designed from different aspects of the optimization process, including population initialization, elite-guided evolution, and adaptive search resource allocation.

First, an adaptive chaotic opposition initialization strategy is incorporated into GoldSA. This strategy combines the ergodicity of chaotic mapping with the search-space expansion capability of opposition-based learning. Since both chaotic initialization and opposition-based learning have been widely used in metaheuristic optimization, the contribution of this strategy lies mainly in their adaptive integration into the initialization stage of GoldSA. Its purpose is to improve the distribution quality of the initial population and provide more diverse candidate solutions before the iterative search begins.

Second, a dynamic elite-guided sine evolution strategy is designed to modify the original sine-based position updating mechanism of GoldSA. Different from the original GoldSA, which mainly relies on the current best solution as the search target, this strategy constructs a weighted elite guidance center using multiple high-quality individuals. In addition, two time-varying coefficients are introduced to adjust the influence of elite guidance and individual historical positions during different search stages. Therefore, this strategy represents a mechanism-level modification of the GoldSA updating rule, aiming to reduce the dependence on a single best individual and improve the coordination between exploration and exploitation.

Third, an Economic Investment Strategy is introduced as an adaptive search intensity allocation mechanism. Inspired by the idea that resources are allocated according to potential returns in economic systems, this strategy evaluates the relative optimization potential of each individual based on its fitness value. Individuals with higher potential are assigned stronger exploitation guidance toward the current best solution, whereas individuals with lower potential are encouraged to perform more exploration-oriented updates through differential information from randomly selected individuals. The innovation of this strategy lies in introducing a fitness-dependent allocation rule into the GoldSA search process, rather than proposing an independent economic optimization model. In this way, the strategy provides a simple and adaptive mechanism for balancing local refinement and population diversity maintenance.

Overall, the proposed MIGoldSA is a multi-strategy enhanced version of GoldSA. The adaptive chaotic opposition initialization strategy mainly serves as a combinatorial improvement for population initialization, while the dynamic elite-guided sine evolution strategy and the Economic Investment Strategy further modify the search updating process of GoldSA. Through these complementary modifications, MIGoldSA aims to improve the population diversity, convergence stability, and search adaptability of the original GoldSA in complex optimization problems, without substantially increasing the algorithmic complexity.

The main contributions of this paper are summarized as follows:

1. A multi-strategy improved version of the Golden Sine Algorithm, named MIGoldSA, is proposed for global optimization and cloud task scheduling problems. The proposed algorithm improves the original GoldSA from the aspects of population initialization, elite-guided evolution, and adaptive search intensity allocation.

2. An adaptive chaotic opposition initialization strategy is integrated into GoldSA. This strategy combines chaotic mapping and opposition-based learning to improve the initial population distribution and enhance early-stage search diversity.
3. A dynamic elite-guided sine evolution strategy is developed by replacing the single best-solution guidance in the original GoldSA with a weighted multi-elite guidance center. This mechanism modifies the original sine-based updating rule and helps reduce excessive dependence on a single best individual.
4. An Economic Investment Strategy is introduced to adaptively adjust the search intensity of different individuals according to their relative fitness quality. This strategy provides a fitness-dependent mechanism for coordinating exploitation-oriented and exploration-oriented updates.
5. The effectiveness of MIGoldSA is evaluated on the IEEE CEC2017 and CEC2022 benchmark suites and further verified on cloud task scheduling problems. The experimental results show that the proposed modifications can improve the optimization performance of GoldSA on most tested problems.

The remainder of this paper is organized as follows. Section 2 reviews the related works on cloud task scheduling, metaheuristic optimization algorithms, and GoldSA-based methods. Section 3 introduces the original Golden Sine Algorithm. Section 4 presents the proposed MIGoldSA algorithm and its three improvement strategies. Section 5 reports the experimental results on the IEEE CEC2017 and CEC2022 benchmark suites. Section 6 applies MIGoldSA to cloud task scheduling problems and analyzes the corresponding experimental results. Finally, Section 7 concludes this paper and discusses future research directions.

2. Related Work

2.1. Cloud Task Scheduling Optimization

Cloud task scheduling has received considerable attention in recent years because it directly affects the execution efficiency, resource utilization, load balance, and service quality of cloud computing systems. Traditional optimization methods, such as linear programming [12], dynamic programming [13], and heuristic rule-based scheduling algorithms [14], can obtain satisfactory solutions for small-scale or simplified scheduling problems. However, these methods usually have limited scalability and adaptability when dealing with large-scale, nonlinear, heterogeneous, and dynamically changing cloud environments. To overcome these limitations, metaheuristic optimization algorithms have been widely introduced into cloud task scheduling problems. For example, Faten A. Saif et al. proposed a Multi-Objective Gray Wolf Optimizer (MGWO) to minimize latency and energy consumption under Quality of Service (QoS) requirements. The simulation results demonstrated that MGWO could effectively reduce latency and energy consumption compared with existing methods [15]. Neetu Sharma et al. developed an improved Ant Colony Optimization algorithm based on a neural network mechanism to optimize task allocation and resource organization. The results showed that the proposed LBACO algorithm achieved better scheduling performance and faster optimization speed [16]. Cui et al. constructed a multi-objective cloud task scheduling model by considering execution time, execution cost, and virtual machine load balancing, and proposed a scheduling algorithm based on evolutionary multifactor optimization. Their results indicated that knowledge transfer among subpopulations could accelerate convergence and improve the optimization performance of multiple objectives [17].

Although these studies have improved cloud task scheduling performance from different perspectives, the scheduling problem remains challenging due to the high dimensionality, complex constraints, and strong coupling among scheduling objectives. Therefore,

more robust and adaptive optimization algorithms are still required to obtain high-quality scheduling solutions under different task scales.

2.2. Metaheuristic Optimization Algorithms

Metaheuristic algorithms have been widely used in global optimization and engineering applications because they do not require strict mathematical assumptions and can effectively search complex solution spaces. In recent years, many improved metaheuristic algorithms have been proposed to enhance convergence accuracy, population diversity, and robustness. For instance, Qin et al. proposed an Enhanced Red-Tailed Hawk algorithm based on a multi-elite strategy and chaotic mapping for cloud task scheduling. The proposed ERTH algorithm showed faster convergence speed and higher stability in minimizing task completion time and system load cost [18]. Liu et al. proposed an improved Ant Colony Optimization variant, named IHMACO, for global path planning problems. By incorporating several new mechanisms, IHMACO achieved superior performance compared with multiple existing path planning methods [19]. Shen et al. introduced a population evolution strategy into the Whale Optimization Algorithm and proposed MEWOA for solving the CEC2019 benchmark functions and engineering design problems. The experimental results demonstrated its strong competitiveness and solution accuracy [20].

Despite their successful applications, many metaheuristic algorithms still suffer from common drawbacks, such as insufficient population diversity, premature convergence, and imbalance between exploration and exploitation. These problems become more serious when solving high-dimensional, multimodal, hybrid, and composition optimization problems. Therefore, designing effective improvement strategies to maintain population diversity, strengthen elite guidance, and adaptively regulate search behaviors is essential for improving optimization performance.

2.3. GoldSA and Its Improved Variants

The Golden Sine Algorithm (GoldSA), proposed by Tanyildizi et al. [11], is a mathematical metaheuristic algorithm inspired by the golden ratio and the periodic oscillation of the sine function. By combining sine-based position updating with the golden section mechanism, GoldSA can guide search agents toward promising regions and gradually narrow the search interval. Due to its simple structure, few parameters, and easy implementation, GoldSA has been used as an effective optimization component in several hybrid and improved algorithms.

For example, Liu et al. proposed a hybrid optimization algorithm combining the Arithmetic Optimization Algorithm and GoldSA. The population was divided into two subgroups and optimized using different search mechanisms. The experimental results on industrial engineering design problems demonstrated that the hybrid algorithm achieved better optimization accuracy and convergence speed than the original AOA, GoldSA, and other comparison algorithms [21]. Li et al. introduced the Runge–Kutta method and the Golden Sine strategy into the Arithmetic Optimization Algorithm to improve its convergence speed and global search capability. The results showed significant improvements in convergence speed, convergence accuracy, and stability compared with the standard AOA [22].

These studies indicate that the GoldSA mechanism has good potential for improving optimization performance. However, the original GoldSA still has some limitations, including insufficient diversity in the initialization stage, strong dependence on the current best solution, and lack of adaptive search regulation. Therefore, this paper further improves GoldSA by introducing an adaptive chaotic opposition initialization strategy, a dynamic elite-guided sine evolution strategy, and an Economic Investment Strategy. Compared with

existing GoldSA-related methods, the proposed MIGoldSA focuses not only on improving convergence accuracy but also on strengthening population diversity maintenance and adaptive exploration–exploitation coordination, making it more suitable for complex global optimization and cloud task scheduling problems.

3. Golden Sine Algorithm (GoldSA)

Golden Sine Algorithm (GoldSA) is a population-based metaheuristic optimization algorithm inspired by mathematical mechanisms. Different from traditional swarm intelligence algorithms inspired by biological behaviors or physical phenomena, GoldSA is developed based on the periodic characteristics of the sine function and the search mechanism of the golden section theory. By combining the oscillatory search behavior of the sine function with the interval contraction capability of the golden section method, GoldSA can effectively balance global exploration and local exploitation during the optimization process. Owing to its simple structure, few control parameters, and fast convergence characteristics, GoldSA has attracted increasing attention in solving continuous optimization problems.

3.1. Basic Principle of GoldSA

The inspiration of GoldSA mainly originates from the periodic variation property of the sine function. As a classical trigonometric function, the sine function exhibits periodic oscillation behavior within the interval $[-1,\ 1]$. Such oscillatory characteristics are highly similar to the dynamic search behavior of metaheuristic algorithms in the solution space. Therefore, GoldSA utilizes the sine function to guide search agents to move periodically around the target solution, thereby enhancing the diversity of population search trajectories and improving the ability to escape local optima.

Meanwhile, in order to further improve the search efficiency, GoldSA introduces the golden section mechanism into the optimization process. The golden section method is a classical mathematical optimization technique that gradually narrows the search interval according to the golden ratio, enabling the search process to focus more efficiently on promising regions. By integrating the sine oscillation mechanism and the golden section strategy, GoldSA achieves dynamic adjustment of the search range during iterations, which significantly enhances convergence performance and solution accuracy.

3.2. Population Initialization

Similar to most population-based optimization algorithms, GoldSA first randomly initializes the population within the feasible search space. Suppose the population size is N and the problem dimension is D. The position of the i-th search agent in the j-th dimension is represented by $V(i,j)$. The initialization process can be expressed as follows:

$$V(i,j) = rand \times \left(ub_j - lb_j\right) + lb_j \tag{1}$$

where $rand$ is a uniformly distributed random number within $[0,1]$; ub_j and lb_j denote the upper and lower bounds of the j-th dimension, respectively; $V(i,j)$ represents the position value of the i-th individual in the j-th dimension.

This initialization strategy enables the search agents to be evenly distributed throughout the search space, thereby improving the population diversity during the early optimization stage and enhancing the global exploration capability of the algorithm.

3.3. Position Updating Mechanism

The core component of GoldSA is the position updating strategy based on the sine function and golden section coefficients. During each iteration, the search agents dy-

namically update their positions around the current target solution according to the following equation:

$$V(i,j) = V(i,j) \times |\sin(r_1)| - r_2 \times \sin(r_1) \times |x_1 \times D(j) - x_2 \times V(i,j)| \quad (2)$$

where $V(i,j)$ denotes the current position of the search agent; $D(j)$ represents the position of the current best solution in the j-th dimension; $r_1 \in [0,\ 2\pi]$ is a random parameter controlling the periodic variation in the sine function; $r_2 \in [0,\ \pi]$ is a random step-size adjustment parameter; x_1 and x_2 are the golden section coefficients.

In this mechanism, the periodic oscillation property of the sine function enables the search agents to perform nonlinear movements within the search space. When the value of $|\sin(r_1)|$ is large, the search step becomes relatively large, which enhances global exploration. Conversely, when $|\sin(r_1)|$ is small, the movement amplitude decreases, and the algorithm tends to focus on local exploitation. Meanwhile, the target solution $D(j)$ provides directional guidance for the population, allowing the search agents to gradually converge toward promising regions.

Therefore, the position updating mechanism of GoldSA establishes a dynamic balance between exploration and exploitation by integrating periodic oscillatory search behavior with elite-guided search.

3.4. Golden Section Search Mechanism

The golden section mechanism is one of the most important components of GoldSA, and its primary objective is to gradually shrink the search interval to improve optimization efficiency. The golden ratio can be defined as follows:

$$\tau = \frac{\sqrt{5}-1}{2} \approx 0.618 \quad (3)$$

Assume that the current search interval is $[a, b]$. Then, two golden section points can be calculated as:

$$x_1 = a(1-\tau) + b\tau \quad (4)$$

$$x_2 = a\tau + b(1-\tau) \quad (5)$$

where a and b denote the lower and upper boundaries of the search interval, respectively; τ is the golden ratio coefficient; x_1 and x_2 are the two golden section search points. In GoldSA, the initial values are generally set as:

$$a = -\pi, b = \pi \quad (6)$$

During the iterative process, the search interval is dynamically adjusted according to the fitness relationship between the current individual and the target solution. Specifically, if the newly generated solution is better than the current target solution, the corresponding interval boundary is updated to further narrow the search region. Consequently, the search process gradually focuses on more promising areas, thereby accelerating convergence and improving solution precision.

The incorporation of the golden section mechanism allows GoldSA to avoid excessive random exploration and enhances the local exploitation capability of the algorithm.

3.5. Optimization Procedure of GoldSA

The overall optimization process of GoldSA can be summarized as follows:

1. Randomly initialize the population within the feasible search space;
2. Evaluate the fitness values of all search agents;

3. Determine the current global best solution as the target solution;
4. Update the positions of search agents using the sine-based updating equation and golden section coefficients;
5. Recalculate the fitness values and update the global best solution;
6. Repeat the iterative process until the termination criterion is satisfied;
7. Output the final global optimal solution.

Through the collaborative interaction between the sine oscillation mechanism and the golden section search strategy, GoldSA exhibits relatively strong optimization capability in solving continuous optimization problems. The GoldSA execution flowchart is shown in Figure 1.

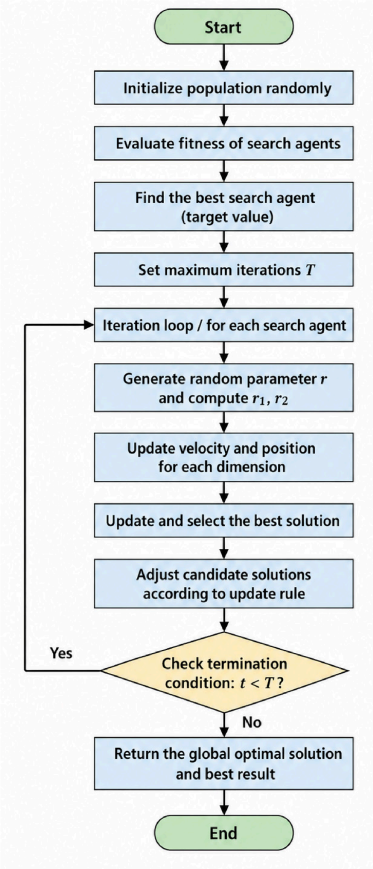


Figure 1. GoldSA Execution Flowchart.

3.6. Advantages and Limitations of GoldSA

Compared with conventional metaheuristic algorithms, GoldSA possesses several advantages, including simple structure, fewer control parameters, fast convergence speed, and easy implementation. Furthermore, the sine-based oscillatory search mechanism improves population diversity, while the golden section strategy enhances local exploitation efficiency.

However, the original GoldSA still suffers from several shortcomings when dealing with complex and high-dimensional optimization problems. First, since the population is initialized by a purely random mechanism and the search process gradually contracts around promising regions, the population diversity may decrease rapidly in the later optimization stage, which increases the risk of premature convergence [9,10]. Second, according to the position updating mechanism of GoldSA, the search direction is mainly guided by the current target solution, which may make the algorithm prone to being trapped in local optima when solving complex multimodal problems [11,21]. Third, although the

sine-based search mechanism and golden section strategy contribute to exploration and exploitation, the original GoldSA lacks an adaptive mechanism to dynamically adjust the search behavior according to the population state, which may lead to an insufficient balance between global exploration and local exploitation in high-dimensional optimization tasks [22]. Therefore, further improvement strategies are necessary to enhance the global optimization performance, population diversity maintenance ability, and local exploitation capability of GoldSA.

4. Proposed MIGoldSA

To overcome the shortcomings of the original Golden Sine Algorithm (GoldSA), such as insufficient population diversity, weak balance between exploration and exploitation, and susceptibility to premature convergence in complex multimodal optimization problems, this paper proposes a Multi-Strategy Improved Golden Sine Algorithm (MIGoldSA). Specifically, three novel enhancement strategies are introduced, including an adaptive chaotic opposition initialization strategy, a dynamic elite-guided sine evolution strategy, and an Economic Investment Strategy. The proposed strategies collaboratively improve the global exploration capability, local exploitation efficiency, and adaptive resource allocation ability of the original GoldSA, thereby significantly enhancing its optimization performance in high-dimensional and complex search spaces. In particular, the Economic Investment Strategy dynamically allocates search efforts according to the optimization potential of individuals, enabling high-quality individuals to receive stronger exploitation guidance while maintaining sufficient population diversity through exploration-oriented updates for inferior individuals.

4.1. Adaptive Chaotic Opposition Initialization Strategy

In the original GoldSA, the initial population is generated purely by uniform random distribution. Although such a mechanism is simple and easy to implement, it often suffers from uneven population distribution and insufficient diversity, especially in high-dimensional optimization problems. Consequently, the algorithm may fail to adequately explore promising regions during the early search stage.

To address this issue, an adaptive chaotic opposition initialization strategy is proposed. First, chaotic mapping is introduced to improve the randomness and ergodicity of the initial population. Then, opposition-based learning is employed to further expand the search coverage of the population [23].

The chaotic sequence is generated using the Tent chaotic map as follows:

$$z_{i+1} = \begin{cases} \frac{z_i}{\mu}, & z_i < \mu \\ \frac{1-z_i}{1-\mu}, & z_i \geq \mu \end{cases} \tag{7}$$

where z_i denotes the chaotic variable of the i-th iteration; $\mu \in (0,\ 1)$ is the control parameter of the Tent map. Using the generated chaotic sequence, the initial population can be constructed as:

$$X_{i,j} = lb_j + z_i \times \left(ub_j - lb_j\right) \tag{8}$$

where $X_{i,j}$ represents the position of the i-th individual in the j-th dimension; lb_j and ub_j denote the lower and upper bounds of the j-th dimension, respectively.

Subsequently, opposition-based learning is employed to generate opposite candidate solutions:

$$\hat{X}_{i,j} = lb_j + ub_j - X_{i,j} \tag{9}$$

where $\hat{X}_{i,j}$ denotes the opposite solution of $X_{i,j}$. Finally, the better individuals are selected from the combined population according to fitness values to construct the initial population of MIGoldSA.

By combining chaotic mapping and opposition-based learning, the proposed strategy significantly enhances the diversity and distribution uniformity of the initial population, thereby improving the global exploration capability of the algorithm during the early optimization stage. Figure 2 illustrates a schematic diagram of this strategy.

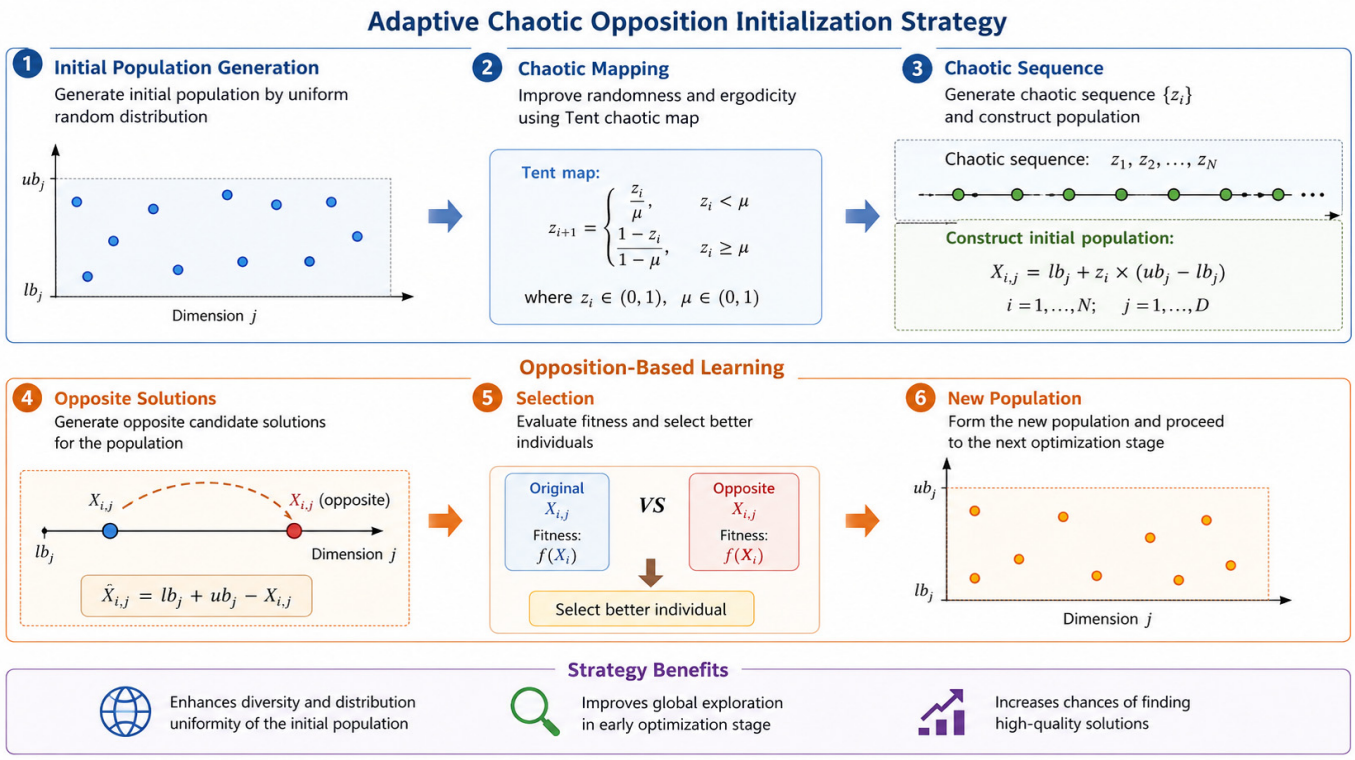


Figure 2. Schematic Diagram of the Adaptive Chaotic Opposition Initialization Strategy.

4.2. Dynamic Elite-Guided Sine Evolution Strategy

In the original GoldSA, the position updating process excessively relies on the current global best solution. Although this mechanism accelerates convergence, it also increases the risk of premature convergence and population aggregation around local optima.

To alleviate this issue, a dynamic elite-guided sine evolution strategy is proposed. Instead of using only the global best individual, multiple elite individuals are utilized collaboratively to guide the population evolution process.

First, the elite guidance center is constructed as:

$$E = \sum_{k=1}^{K} \omega_k X_k \tag{10}$$

where E denotes the elite guidance center; K is the number of elite individuals; X_k represents the k-th elite individual; ω_k denotes the adaptive weight of the k-th elite individual. The adaptive weights are calculated according to fitness values:

$$\omega_k = \frac{1/(f_k + \varepsilon)}{\sum_{m=1}^{K} 1/(f_m + \varepsilon)} \tag{11}$$

where f_k denotes the fitness value of the k-th elite individual; ε is a very small constant used to avoid division by zero. Then, the original GoldSA position updating equation is modified as:

$$X_{i,j}^{t+1} = X_{i,j}^{t} \cdot |\sin(r_1)| - r_2 \sin(r_1) \cdot |c_1 E_j - c_2 X_{i,j}^{t}| \tag{12}$$

where $X_{i,j}^{t}$ and $X_{i,j}^{t+1}$ denote the positions before and after updating, respectively; E_j represents the j-th dimension of the elite guidance center; $r_1 \in [0, 2\pi]$ and $r_2 \in [0, \pi]$ are random parameters; c_1 and c_2 are adaptive control coefficients. The adaptive coefficients are dynamically adjusted as:

$$c_1 = 2\left(1 - \frac{t}{T}\right) \tag{13}$$

$$c_2 = \frac{t}{T} \tag{14}$$

where t denotes the current iteration number; T denotes the maximum number of iterations.

At the early optimization stage, larger c_1 values encourage global exploration around elite regions. As iterations proceed, larger c_2 values gradually enhance local exploitation capability.

Therefore, the proposed strategy effectively improves the balance between exploration and exploitation while reducing the dependence on a single best solution. Figure 3 illustrates a schematic diagram of this strategy.

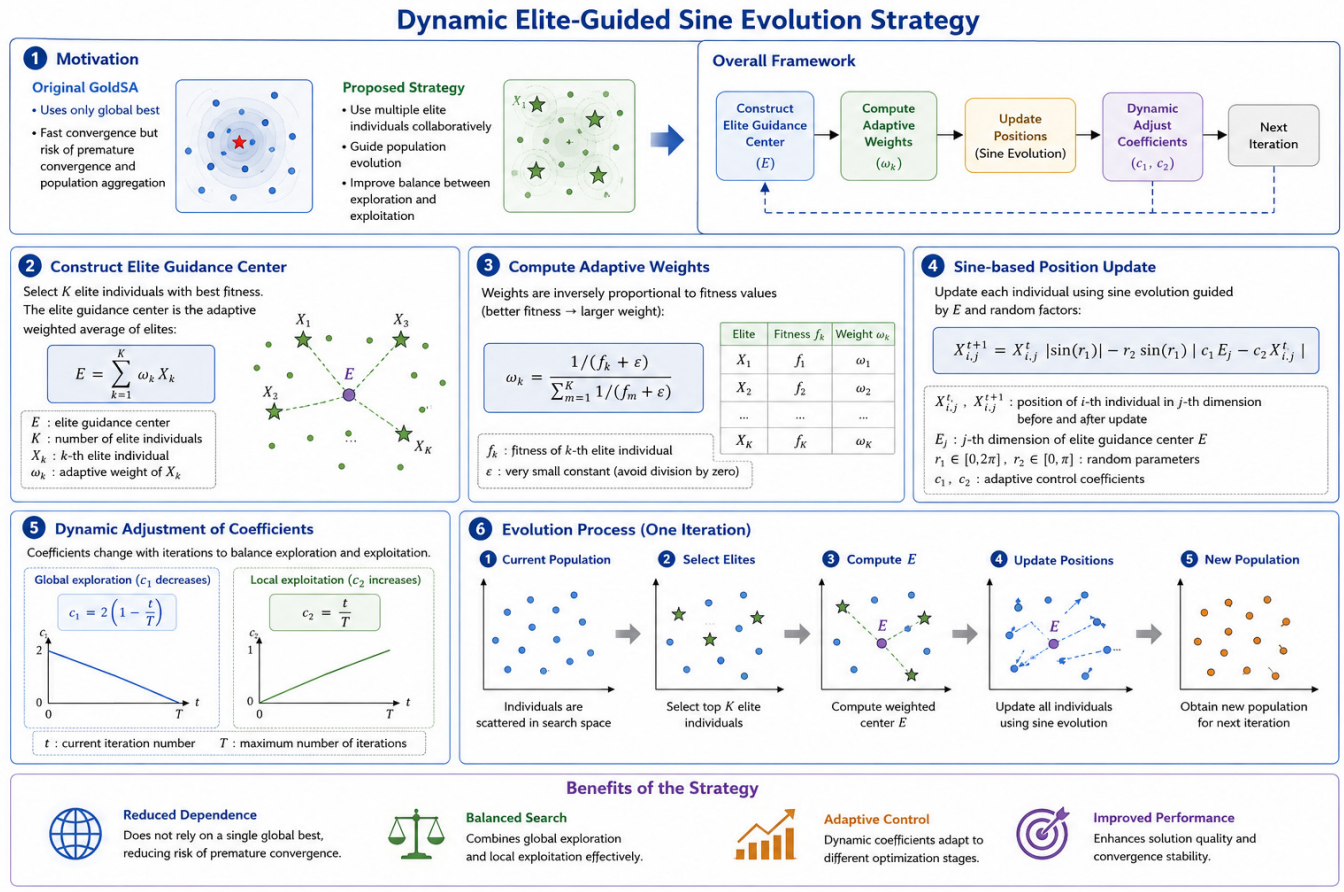


Figure 3. Schematic Diagram of Dynamic Elite-Guided Sine Evolution Strategy.

4.3. Economic Investment Strategy

To further enhance the search efficiency of MIGoldSA and improve its ability to allocate search resources adaptively, an Economic Investment Strategy is introduced. The main

idea of this strategy is inspired by the investment decision-making process in economic systems. In real investment activities, more resources are usually allocated to assets with higher potential returns, while fewer resources are assigned to low-quality or high-risk assets. Similarly, in the optimization process, individuals with better fitness values can be regarded as high-return search assets, while inferior individuals are considered low-return assets. Therefore, different update intensities are assigned to individuals according to their fitness quality, so that the algorithm can adaptively balance exploration and exploitation.

First, the investment potential of the i-th individual is defined according to its fitness value:

$$I_i = \frac{f_{worst} - f_i + \varepsilon}{f_{worst} - f_{best} + \varepsilon} \tag{15}$$

where I_i denotes the investment potential of the i-th individual, f_i represents the fitness value of the i-th individual, f_{best} and f_{worst} are the best and worst fitness values in the current population, respectively, and ε is a small constant used to avoid division by zero. For minimization problems, a larger I_i indicates that the individual has better optimization potential and should receive stronger exploitation guidance. Then, the investment coefficient is calculated as follows:

$$\lambda_i = \lambda_{min} + (\lambda_{max} - \lambda_{min}) I_i \tag{16}$$

where λ_i is the adaptive investment coefficient of the i-th individual, and λ_{min} and λ_{max} denote the minimum and maximum investment levels, respectively.

Based on the investment coefficient, the position of each individual is updated by:

$$X_i^{t+1} = X_i^t + \lambda_i r_1 \left(X_{best}^t - X_i^t\right) + (1 - \lambda_i) r_2 \left(X_{r1}^t - X_{r2}^t\right) \tag{17}$$

where X_i^t represents the position of the i-th individual at iteration t, X_{best}^t denotes the current global best solution, X_{r1}^t and X_{r2}^t are two randomly selected individuals from the population, and r_1 and r_2 are random numbers in $[0,\ 1]$.

In this update mechanism, individuals with higher investment potential are guided more strongly by the current best solution, thereby accelerating convergence and strengthening local exploitation. Conversely, individuals with lower investment potential rely more on the difference between randomly selected individuals, which helps maintain population diversity and enhances global exploration. Therefore, the proposed Economic Investment Strategy enables MIGoldSA to dynamically allocate search resources according to individual quality, improving both convergence accuracy and robustness in complex optimization problems. Figure 4 illustrates a schematic diagram of this strategy.

4.4. Computational Complexity Analysis of MIGoldSA

Assume that the population size is N, the problem dimension is D, and the maximum iteration number is T. In MIGoldSA, the adaptive chaotic opposition initialization strategy introduces additional opposition-based solution generation operations, whose computational complexity is $O(ND)$. The dynamic elite-guided sine evolution strategy mainly involves elite individual weighting, sine-guided position updating, and adaptive search operations, and its computational complexity also remains $O(ND)$. Furthermore, the proposed Economic Investment Strategy dynamically calculates the investment potential and adaptive investment coefficient for each individual according to its fitness quality, followed by investment-guided position updates. The computational cost of these operations is likewise $O(ND)$.

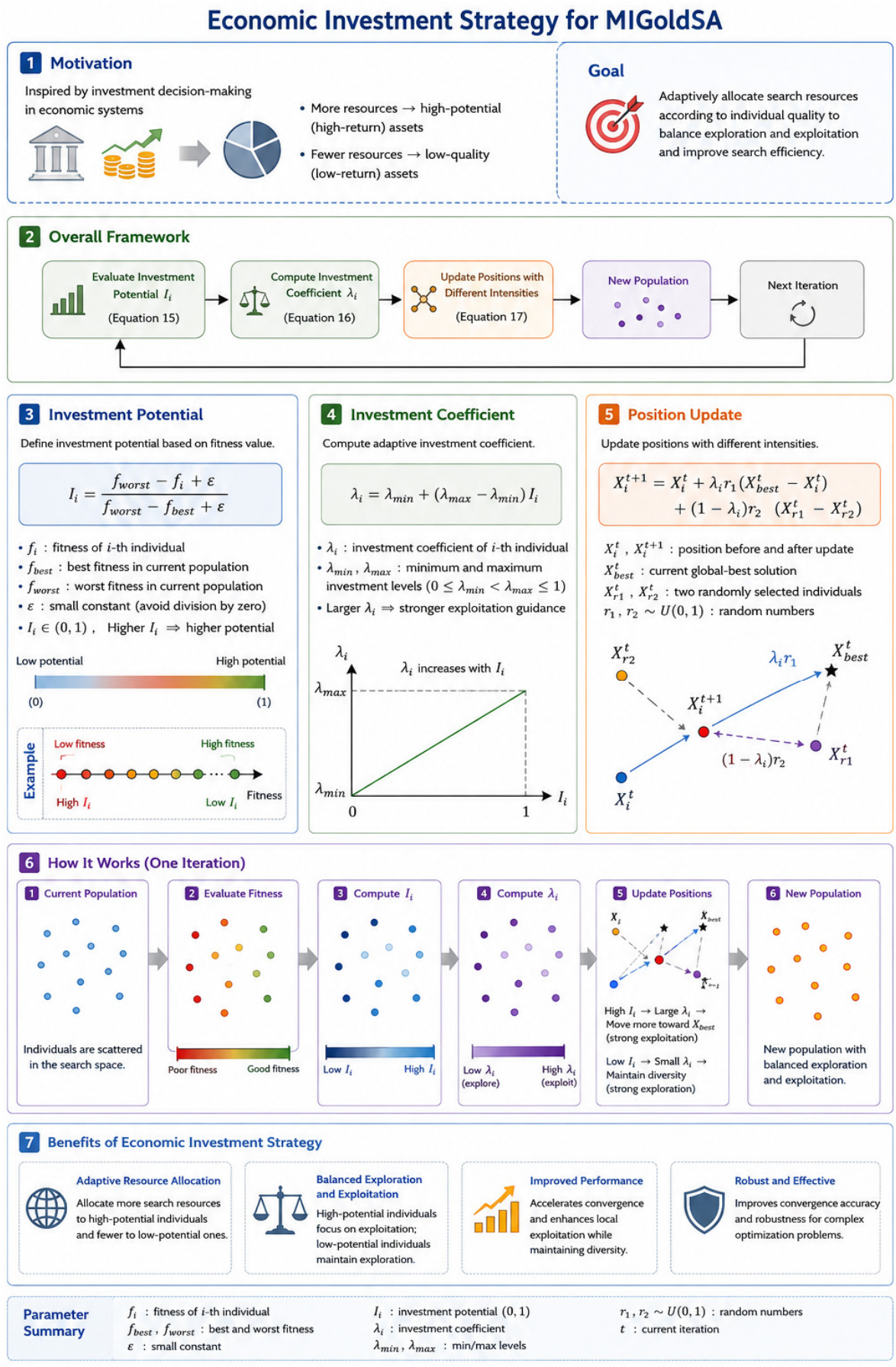


Figure 4. Schematic Diagram of Economic Investment Strategy.

Therefore, the overall computational complexity of MIGoldSA can still be expressed as $O(T \times N \times D)$. which is consistent with the original GoldSA in terms of complexity order. Consequently, the proposed improvement strategies do not significantly increase the computational burden of the algorithm while substantially enhancing its global exploration capability, local exploitation efficiency, and optimization accuracy in complex search spaces. Figure 5 illustrates the flowchart of the MIGoldSA. The global convergence analysis of MIGoldSA proposed in this paper is presented in Appendix A.

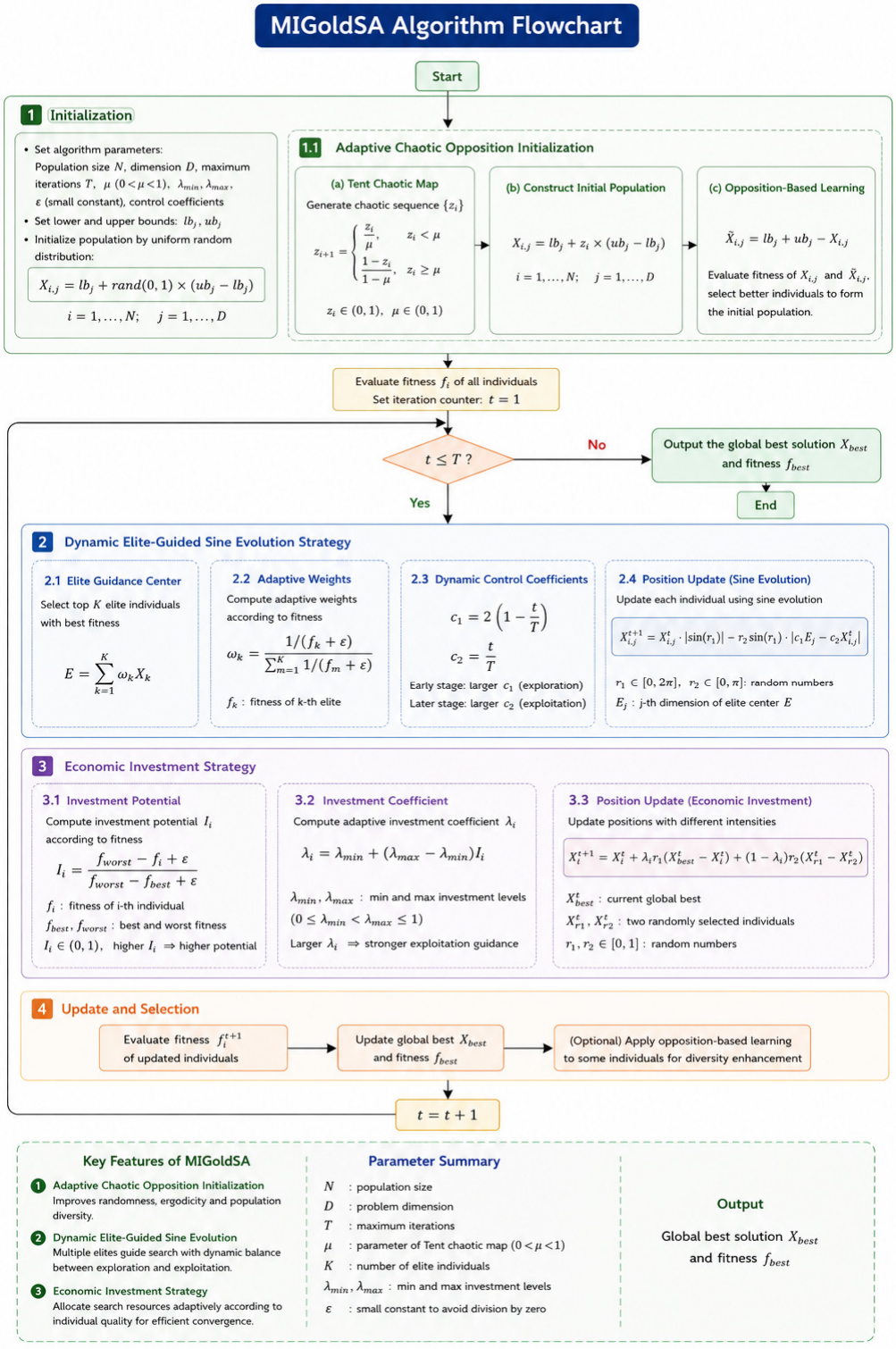


Figure 5. MIGoldSA Flowchart.

5. Global Optimization Experiment Analysis

This section aims to comprehensively evaluate the optimization performance of the proposed MIGoldSA through extensive experiments conducted on the IEEE CEC2017 and IEEE CEC2022 benchmark test suites. First, a brief introduction to the characteristics of the benchmark functions is provided, followed by detailed descriptions of the comparison algorithms and their parameter settings. Subsequently, MIGoldSA is compared with nine advanced state-of-the-art optimization algorithms to verify its effectiveness and competitiveness in solving complex optimization problems. Finally, statistical analyses are carried out to further validate the superiority and robustness of the proposed MIGoldSA.

To ensure the fairness and consistency of the experimental results, the population size of all comparison algorithms was uniformly set to 50, while the maximum number of iterations was fixed at 1000. Moreover, each algorithm was independently executed 30 times to reduce the influence of randomness on the experimental results. All experiments were implemented in MATLAB 2023a and conducted on a computer equipped with an Intel Core i7-10700F CPU operating at 2.90 GHz and 8 GB RAM.

5.1. Benchmark Test Functions

To comprehensively evaluate the optimization capability of the proposed MIGoldSA algorithm, the IEEE CEC2017 [24] and IEEE CEC2022 [25] benchmark test suites are employed in this study. These benchmark suites are widely recognized as standard evaluation platforms for metaheuristic optimization algorithms and have been extensively used to assess the convergence accuracy, robustness, exploration capability, and exploitation performance of various swarm intelligence and evolutionary optimization methods.

The IEEE CEC2017 benchmark suite consists of a diverse set of complex optimization problems, including unimodal functions, multimodal functions, hybrid functions, and composition functions. Among them, unimodal functions are mainly used to evaluate the convergence speed and local exploitation capability of optimization algorithms, while multimodal functions contain numerous local optima and are therefore suitable for testing the global exploration ability and local optimum avoidance capability of algorithms. Furthermore, hybrid and composition functions combine multiple search characteristics and complex landscape structures, significantly increasing optimization difficulty and providing a more challenging evaluation environment for metaheuristic algorithms. Consequently, the CEC2017 benchmark suite has become one of the most commonly used benchmark platforms for evaluating the overall performance of optimization algorithms.

Compared with CEC2017, the IEEE CEC2022 benchmark suite introduces more complicated landscape characteristics and stronger variable interactions, making the optimization problems more challenging. The CEC2022 benchmark functions contain highly nonlinear, nonseparable, rotated, and hybrid search landscapes, which can better simulate practical complex optimization scenarios. In particular, the benchmark suite places greater emphasis on evaluating the balance between global exploration and local exploitation, as well as the population diversity maintenance capability of optimization algorithms in high-dimensional search spaces. Therefore, CEC2022 is considered a more rigorous benchmark suite for testing the robustness and adaptability of advanced metaheuristic algorithms.

Since the CEC2017 and CEC2022 benchmark suites contain a wide variety of optimization problems with different characteristics and difficulty levels, they can comprehensively reflect the optimization performance of metaheuristic algorithms under different search environments. Therefore, in this paper, these two benchmark suites are selected to systematically evaluate the performance of the proposed MIGoldSA algorithm in terms of convergence accuracy, convergence behavior, robustness, and global optimization capability.

5.2. Competitor Algorithms and Parameters Setting

In this section, the proposed MIGoldSA algorithm is compared with nine advanced optimization algorithms to comprehensively evaluate its optimization performance, convergence capability, and robustness in solving complex optimization problems. To conduct a comprehensive comparison, we selected algorithms based on group behavior, including Velocity pausing particle swarm optimization (VPPSO), Self-Learning Particle Swarm Optimizer (SLPSO), Artificial Lemming Algorithm (ALA), Traffic jam optimizer (TJO), Gekko Japonicus Algorithm (GJA), Newton Downhill Optimizer (NDO), Zebra Optimization Algorithm (ZOA), Stochastic social learning optimization (SSLO), and Golden Sine Algorithm (GoldSA). Table 1 summarizes the parameter settings of these algorithms for easier reading.

Table 1. Parameter Configurations of the Compared Algorithms.

Algorithms	Parameter Name	Parameter Value	Reference
VPPSO	w_Max, w_Min	0.9, 0.1	[26]
SLPSO	$alpha,\ beta$	0.5, 0.01	[27]
ALA	$beta$	1.5	[28]
TJO	$a_{min}, a_{max}, c_{min}, c_{max}$	0, 2, 0, 2	[29]
GJA	$beta_{start}, beta_{end}$	1.2, 0.3	[30]
NDO	u	0.5	[31]
ZOA	R	0.1	[32]
SSLO	lp	0.25	[33]
GoldSA	--	--	--

5.3. Performance Evaluation on the CEC2017 Benchmark Functions

In this section, the optimization performance of MIGoldSA is evaluated on the 30-dimensional and 50-dimensional CEC2017 benchmark functions and compared with nine advanced state-of-the-art optimization algorithms. The corresponding experimental results are presented in Tables 2 and 3, where "Mean" and "Std" denote the average value and standard deviation obtained from 30 independent runs, respectively. To more intuitively demonstrate the convergence behavior of different algorithms during the optimization process, the convergence curves of the ten algorithms are illustrated in Figure 6. In addition, to further analyze the robustness and stability of the algorithms over multiple independent runs, the boxplot distributions of the experimental results obtained from 30 runs are provided in Figure 7.

Table 2. Performance Results of Different Algorithms on the CEC2017 Benchmark Functions (Dim = 30).

ID	Metric	VPPSO	SLPSO	ALA	TJO	GJA	NDO	ZOA	SSLO	GoldSA	MIGoldSA
F1	mean	5.6061×10^{3}	3.0584×10^{8}	4.1623×10^{3}	4.0307×10^{3}	1.2171×10^{6}	5.8024×10^{3}	7.1922×10^{9}	1.1349×10^{7}	4.9825×10^{10}	2.9933×10^{3}
	std	5.7048×10^{3}	7.9067×10^{8}	4.3947×10^{3}	1.6318×10^{3}	1.7857×10^{5}	6.1536×10^{3}	2.4590×10^{9}	2.8393×10^{6}	7.6063×10^{9}	3.5328×10^{3}
F2	mean	2.1408×10^{18}	8.5678×10^{32}	3.1085×10^{19}	4.3324×10^{10}	5.3089×10^{12}	1.9547×10^{12}	1.1541×10^{36}	6.4319×10^{22}	2.1390×10^{46}	5.1535×10^{13}
	std	4.4034×10^{18}	4.2457×10^{33}	1.5379×10^{20}	1.8724×10^{11}	8.7509×10^{12}	6.4894×10^{12}	6.2504×10^{36}	2.3616×10^{23}	8.0198×10^{46}	2.3968×10^{14}
F3	mean	2.1670×10^{4}	9.8081×10^{4}	3.0081×10^{3}	2.0448×10^{3}	3.2292×10^{3}	5.4786×10^{4}	3.2420×10^{4}	1.0788×10^{5}	9.2359×10^{4}	1.9762×10^{4}
	std	5.6431×10^{3}	4.1063×10^{4}	1.1818×10^{3}	1.4056×10^{3}	2.0422×10^{3}	1.5549×10^{4}	7.3980×10^{3}	1.6664×10^{4}	2.4340×10^{3}	9.3119×10^{3}
F4	mean	4.9605×10^{2}	6.1041×10^{2}	5.0096×10^{2}	4.9765×10^{2}	5.0660×10^{2}	4.8660×10^{2}	1.3751×10^{3}	5.3323×10^{2}	9.0482×10^{3}	4.8757×10^{2}
	std	2.0380×10^{1}	2.5148×10^{2}	1.8771×10^{1}	2.6855×10^{1}	2.0427×10^{1}	2.8902×10^{1}	7.7820×10^{2}	1.2181×10^{1}	2.4863×10^{3}	1.6386×10^{0}
F5	mean	6.3731×10^{2}	5.5701×10^{2}	5.9733×10^{2}	7.4575×10^{2}	5.6804×10^{2}	5.8423×10^{2}	6.9333×10^{2}	6.0340×10^{2}	9.3442×10^{2}	5.2388×10^{2}
	std	2.8690×10^{1}	2.1067×10^{1}	2.7262×10^{1}	4.4449×10^{1}	1.6974×10^{1}	2.1953×10^{1}	3.2358×10^{1}	8.4801×10^{0}	2.4820×10^{1}	6.3039×10^{0}

Table 2. *Cont.*

ID	Metric	VPPSO	SLPSO	ALA	TJO	GJA	NDO	ZOA	SSLO	GoldSA	MIGoldSA
F6	mean	6.2877×10^{2}	6.0529×10^{2}	6.0527×10^{2}	6.5813×10^{2}	6.0607×10^{2}	6.0272×10^{2}	6.5036×10^{2}	6.0280×10^{2}	6.8799×10^{2}	6.0004×10^{2}
	std	9.1736×10^{0}	8.2491×10^{0}	4.2717×10^{0}	7.9661×10^{0}	1.8025×10^{0}	3.8490×10^{0}	6.8779×10^{0}	2.7269×10^{-1}	9.8214×10^{0}	1.9071×10^{-2}
F7	mean	8.8396×10^{2}	7.8064×10^{2}	8.4753×10^{2}	9.8414×10^{2}	7.9954×10^{2}	8.2196×10^{2}	1.1091×10^{3}	8.9038×10^{2}	1.4386×10^{3}	7.5798×10^{2}
	std	4.9493×10^{1}	2.1036×10^{1}	3.3351×10^{1}	7.8460×10^{1}	1.6442×10^{1}	2.5332×10^{1}	7.4799×10^{1}	1.3115×10^{1}	6.7717×10^{1}	8.7240×10^{0}
F8	mean	9.1109×10^{2}	8.4408×10^{2}	8.8544×10^{2}	9.9733×10^{2}	8.6479×10^{2}	8.6965×10^{2}	9.4439×10^{2}	9.0383×10^{2}	1.1384×10^{3}	8.2611×10^{2}
	std	2.2520×10^{1}	1.0012×10^{1}	2.2353×10^{1}	4.6485×10^{1}	1.5273×10^{1}	1.7646×10^{1}	2.6259×10^{1}	1.2034×10^{1}	2.6881×10^{1}	8.1345×10^{0}
F9	mean	2.5537×10^{3}	1.1421×10^{3}	1.4448×10^{3}	4.3698×10^{3}	9.6901×10^{2}	1.2430×10^{3}	4.2596×10^{3}	1.5848×10^{3}	1.1103×10^{4}	9.0149×10^{2}
	std	8.6171×10^{2}	3.7469×10^{2}	3.7349×10^{2}	1.2651×10^{3}	5.3714×10^{1}	4.1076×10^{2}	5.4761×10^{2}	1.7647×10^{2}	1.3944×10^{3}	1.4919×10^{0}
F10	mean	4.7595×10^{3}	4.3888×10^{3}	4.9936×10^{3}	5.3917×10^{3}	4.0478×10^{3}	5.0036×10^{3}	4.7713×10^{3}	4.9198×10^{3}	8.6941×10^{3}	4.8201×10^{3}
	std	7.9814×10^{2}	7.4998×10^{2}	6.6029×10^{2}	8.7530×10^{2}	5.5442×10^{2}	1.1636×10^{3}	3.9007×10^{2}	2.2802×10^{2}	5.3163×10^{2}	5.9851×10^{2}
F11	mean	1.2764×10^{3}	1.8272×10^{3}	1.1871×10^{3}	1.2621×10^{3}	1.2244×10^{3}	1.1634×10^{3}	1.8085×10^{3}	1.6500×10^{3}	1.0910×10^{4}	1.1565×10^{3}
	std	5.2171×10^{1}	6.0557×10^{2}	4.2947×10^{1}	4.2785×10^{1}	3.9374×10^{1}	3.4997×10^{1}	5.5036×10^{2}	1.3296×10^{2}	2.7711×10^{3}	2.7790×10^{1}
F12	mean	1.3739×10^{7}	2.8675×10^{7}	4.7922×10^{5}	9.0241×10^{6}	1.9600×10^{6}	1.1673×10^{5}	7.4378×10^{8}	5.0787×10^{6}	8.6804×10^{9}	7.8820×10^{4}
	std	1.0482×10^{7}	8.3195×10^{7}	5.7902×10^{5}	6.2874×10^{6}	1.1426×10^{6}	7.7218×10^{4}	8.3637×10^{8}	2.3600×10^{6}	3.6630×10^{9}	6.3747×10^{4}
F13	mean	1.1906×10^{5}	1.0254×10^{5}	2.0484×10^{4}	4.5177×10^{4}	1.0063×10^{5}	1.1482×10^{4}	9.3011×10^{7}	1.7380×10^{5}	1.4188×10^{9}	1.6508×10^{3}
	std	5.2164×10^{4}	2.7953×10^{5}	2.0412×10^{4}	2.3226×10^{4}	2.9576×10^{4}	1.3064×10^{4}	2.4598×10^{8}	1.9099×10^{5}	1.4244×10^{9}	2.5572×10^{2}
F14	mean	2.8476×10^{4}	3.2049×10^{5}	1.5283×10^{3}	1.6603×10^{4}	1.2220×10^{4}	1.9470×10^{4}	3.9716×10^{5}	1.1189×10^{5}	6.7249×10^{6}	1.4305×10^{3}
	std	2.4059×10^{4}	3.5264×10^{5}	2.3343×10^{1}	1.2922×10^{4}	1.0693×10^{4}	1.1702×10^{4}	5.3971×10^{5}	8.5009×10^{4}	7.9286×10^{6}	3.1517×10^{0}
F15	mean	4.8067×10^{4}	3.4363×10^{4}	2.3243×10^{3}	2.9819×10^{4}	3.2674×10^{4}	8.4693×10^{3}	7.8497×10^{5}	3.3502×10^{4}	3.5520×10^{8}	1.5363×10^{3}
	std	3.0120×10^{4}	3.0440×10^{4}	2.1190×10^{2}	1.0625×10^{4}	1.2187×10^{4}	7.7855×10^{3}	2.1476×10^{6}	1.8773×10^{4}	3.6685×10^{8}	2.1658×10^{1}
F16	mean	2.7639×10^{3}	2.4032×10^{3}	2.4035×10^{3}	3.2193×10^{3}	2.4434×10^{3}	2.4680×10^{3}	2.9610×10^{3}	2.5586×10^{3}	5.5168×10^{3}	2.0045×10^{3}
	std	2.0699×10^{2}	3.2810×10^{2}	2.5458×10^{2}	4.2898×10^{2}	2.1754×10^{2}	2.3692×10^{2}	2.2427×10^{2}	1.5483×10^{2}	7.6331×10^{2}	2.3343×10^{2}
F17	mean	2.0986×10^{3}	2.0610×10^{3}	2.0212×10^{3}	2.4458×10^{3}	1.9568×10^{3}	2.0172×10^{3}	2.2330×10^{3}	1.9869×10^{3}	3.6017×10^{3}	1.8605×10^{3}
	std	1.6576×10^{2}	2.3086×10^{2}	1.1338×10^{2}	2.2307×10^{2}	1.0095×10^{2}	1.7158×10^{2}	1.9502×10^{2}	8.9920×10^{1}	1.8692×10^{3}	1.2525×10^{2}
F18	mean	4.4936×10^{5}	1.4006×10^{6}	4.7404×10^{3}	2.1782×10^{5}	1.6637×10^{5}	2.5756×10^{5}	1.3009×10^{6}	5.1862×10^{5}	7.5914×10^{7}	5.7526×10^{3}
	std	3.4356×10^{5}	1.4592×10^{6}	1.7083×10^{3}	1.4345×10^{5}	8.2392×10^{4}	2.3382×10^{5}	1.8634×10^{6}	3.1270×10^{5}	8.5157×10^{7}	4.8761×10^{3}
F19	mean	1.0757×10^{6}	6.6542×10^{4}	5.7864×10^{3}	1.4132×10^{6}	4.4169×10^{4}	1.2468×10^{4}	1.3958×10^{6}	6.2529×10^{4}	2.6268×10^{8}	1.9207×10^{3}
	std	6.5349×10^{5}	1.3989×10^{5}	1.1327×10^{4}	7.8904×10^{5}	2.6240×10^{4}	1.2829×10^{4}	1.3631×10^{6}	5.9720×10^{4}	3.2274×10^{8}	1.5518×10^{1}
F20	mean	2.4645×10^{3}	2.4949×10^{3}	2.3952×10^{3}	2.6243×10^{3}	2.3042×10^{3}	2.3450×10^{3}	2.3955×10^{3}	2.3464×10^{3}	3.0700×10^{3}	2.2868×10^{3}
	std	1.5926×10^{2}	2.2963×10^{2}	1.5144×10^{2}	2.0998×10^{2}	1.6022×10^{2}	1.8498×10^{2}	9.8230×10^{1}	8.5064×10^{1}	1.7653×10^{2}	1.9731×10^{2}
F21	mean	2.4195×10^{3}	2.3511×10^{3}	2.3859×10^{3}	2.5717×10^{3}	2.3627×10^{3}	2.3716×10^{3}	2.4905×10^{3}	2.3982×10^{3}	2.7025×10^{3}	2.3267×10^{3}
	std	3.7287×10^{1}	1.7826×10^{1}	2.2305×10^{1}	5.0602×10^{1}	1.7637×10^{1}	2.3480×10^{1}	2.9361×10^{1}	1.6661×10^{1}	5.0231×10^{1}	8.3942×10^{0}
F22	mean	2.9965×10^{3}	2.7779×10^{3}	5.3957×10^{3}	5.1420×10^{3}	2.3942×10^{3}	2.3011×10^{3}	4.8980×10^{3}	2.3253×10^{3}	9.3416×10^{3}	5.9002×10^{3}
	std	1.4291×10^{3}	9.7942×10^{2}	1.8425×10^{3}	2.3956×10^{3}	4.4956×10^{2}	1.8090×10^{0}	1.3283×10^{3}	1.0322×10^{1}	8.4324×10^{2}	1.1783×10^{3}
F23	mean	2.7852×10^{3}	2.7331×10^{3}	2.7492×10^{3}	3.4112×10^{3}	2.7152×10^{3}	2.7351×10^{3}	3.1677×10^{3}	2.7450×10^{3}	3.4031×10^{3}	2.6765×10^{3}
	std	3.1753×10^{1}	4.5352×10^{1}	2.8857×10^{1}	1.4168×10^{2}	1.5288×10^{1}	3.1982×10^{1}	1.0866×10^{2}	9.6770×10^{0}	1.2608×10^{2}	1.0017×10^{1}
F24	mean	2.9356×10^{3}	2.9074×10^{3}	2.9279×10^{3}	3.3766×10^{3}	2.8878×10^{3}	2.9028×10^{3}	3.3883×10^{3}	2.9130×10^{3}	3.5514×10^{3}	2.8483×10^{3}
	std	3.2441×10^{1}	6.2834×10^{1}	2.8388×10^{1}	1.6881×10^{2}	2.7397×10^{1}	2.6681×10^{1}	9.5601×10^{1}	1.2823×10^{1}	1.0339×10^{2}	8.2906×10^{0}
F25	mean	2.9111×10^{3}	2.9650×10^{3}	2.8907×10^{3}	2.9357×10^{3}	2.9059×10^{3}	2.8980×10^{3}	3.1211×10^{3}	2.9236×10^{3}	4.2415×10^{3}	2.8872×10^{3}
	std	1.7995×10^{1}	1.0251×10^{2}	7.5374×10^{0}	2.0185×10^{1}	1.6037×10^{1}	1.8026×10^{1}	1.0604×10^{2}	8.7503×10^{0}	3.4196×10^{2}	4.0246×10^{-1}
F26	mean	4.7094×10^{3}	4.4757×10^{3}	4.8162×10^{3}	6.4796×10^{3}	3.7163×10^{3}	4.5070×10^{3}	7.6289×10^{3}	4.7162×10^{3}	1.0472×10^{4}	3.8429×10^{3}
	std	1.2369×10^{3}	6.3194×10^{2}	3.3384×10^{2}	2.6704×10^{3}	9.8336×10^{2}	6.4443×10^{2}	1.0577×10^{3}	1.5324×10^{2}	1.1402×10^{3}	1.0995×10^{2}
F27	mean	3.2808×10^{3}	3.2563×10^{3}	3.2307×10^{3}	3.7237×10^{3}	3.2440×10^{3}	3.2396×10^{3}	3.8395×10^{3}	3.2308×10^{3}	4.0595×10^{3}	3.2043×10^{3}
	std	3.4384×10^{1}	3.5145×10^{1}	2.8549×10^{1}	3.3135×10^{2}	1.9389×10^{1}	1.4198×10^{1}	1.7370×10^{2}	6.4798×10^{0}	3.4760×10^{2}	9.2996×10^{0}
F28	mean	3.2687×10^{3}	3.4311×10^{3}	3.5557×10^{3}	3.2403×10^{3}	3.2382×10^{3}	3.2048×10^{3}	3.7632×10^{3}	3.2901×10^{3}	6.0767×10^{3}	3.2367×10^{3}
	std	2.7479×10^{1}	2.0428×10^{2}	9.4432×10^{2}	2.5314×10^{1}	2.2867×10^{1}	3.2434×10^{1}	3.3502×10^{2}	1.3687×10^{1}	6.7896×10^{2}	4.7735×10^{1}
F29	mean	4.1081×10^{3}	3.9116×10^{3}	3.7770×10^{3}	4.8048×10^{3}	3.6041×10^{3}	3.6274×10^{3}	4.7634×10^{3}	3.7080×10^{3}	7.7647×10^{3}	3.5727×10^{3}
	std	2.7611×10^{2}	2.8905×10^{2}	1.8143×10^{2}	4.1294×10^{2}	1.6193×10^{2}	1.7664×10^{2}	4.1356×10^{2}	8.6705×10^{1}	2.7939×10^{3}	1.8747×10^{2}
F30	mean	4.4610×10^{6}	1.4711×10^{6}	2.3170×10^{4}	3.8146×10^{6}	1.7501×10^{5}	1.0122×10^{4}	1.8107×10^{7}	7.7789×10^{4}	7.9614×10^{8}	6.0790×10^{3}
	std	2.3970×10^{6}	2.7782×10^{6}	1.0039×10^{4}	2.1240×10^{6}	6.6981×10^{4}	3.4759×10^{3}	1.3741×10^{7}	3.7377×10^{4}	6.3939×10^{8}	9.8611×10^{2}

Table 3. Performance Results of Different Algorithms on the CEC2017 Benchmark Functions (Dim = 50).

ID	Metric	VPPSO	SLPSO	ALA	TJO	GJA	NDO	ZOA	SSLO	GoldSA	MIGoldSA
F1	mean	4.2045×10^{3}	1.8688×10^{9}	1.2938×10^{6}	4.8990×10^{4}	4.3135×10^{6}	4.9588×10^{3}	2.3851×10^{10}	7.0912×10^{8}	9.7513×10^{10}	3.9849×10^{6}
	std	6.1414×10^{3}	1.8361×10^{9}	1.1236×10^{6}	1.2356×10^{4}	5.8348×10^{5}	4.5754×10^{3}	7.4182×10^{9}	9.4557×10^{7}	8.3235×10^{9}	2.1131×10^{6}
F2	mean	1.0861×10^{45}	3.8262×10^{62}	7.8306×10^{41}	4.5199×10^{28}	2.8427×10^{30}	4.8393×10^{35}	5.5462×10^{63}	7.8473×10^{48}	6.4632×10^{83}	7.1148×10^{36}
	std	3.6612×10^{45}	1.5001×10^{63}	4.0409×10^{42}	2.2735×10^{29}	1.2343×10^{31}	2.1883×10^{36}	2.1765×10^{64}	2.0849×10^{49}	3.2517×10^{84}	3.8056×10^{37}
F3	mean	8.2376×10^{4}	2.1241×10^{5}	4.5880×10^{4}	4.3012×10^{4}	5.0587×10^{4}	1.6580×10^{5}	1.0833×10^{5}	2.3247×10^{5}	2.2062×10^{5}	1.3729×10^{5}
	std	1.4826×10^{4}	6.4426×10^{4}	1.1753×10^{4}	1.5581×10^{4}	1.0137×10^{4}	2.4104×10^{4}	1.2924×10^{4}	3.7843×10^{4}	3.1772×10^{4}	2.8063×10^{4}
F4	mean	6.1643×10^{2}	1.0377×10^{3}	6.1585×10^{2}	6.1835×10^{2}	5.8563×10^{2}	5.5357×10^{2}	3.9438×10^{3}	8.0294×10^{2}	3.1857×10^{4}	5.9692×10^{2}
	std	4.5741×10^{1}	1.3570×10^{3}	4.8704×10^{1}	4.7246×10^{1}	3.6517×10^{1}	4.5613×10^{1}	1.3210×10^{3}	3.1110×10^{1}	5.9314×10^{3}	3.2511×10^{1}
F5	mean	7.6291×10^{2}	6.2409×10^{2}	6.8289×10^{2}	8.7660×10^{2}	6.5296×10^{2}	6.7212×10^{2}	8.5267×10^{2}	7.7265×10^{2}	1.1843×10^{3}	5.6870×10^{2}
	std	4.9755×10^{1}	4.3582×10^{1}	4.0495×10^{1}	6.3495×10^{1}	3.1103×10^{1}	2.8249×10^{1}	4.1588×10^{1}	1.9195×10^{1}	3.0839×10^{1}	1.1956×10^{1}
F6	mean	6.4331×10^{2}	6.0922×10^{2}	6.1931×10^{2}	6.6582×10^{2}	6.1288×10^{2}	6.0726×10^{2}	6.6239×10^{2}	6.0931×10^{2}	7.0097×10^{2}	6.0104×10^{2}
	std	6.0944×10^{0}	1.0972×10^{1}	6.3907×10^{0}	7.9134×10^{0}	3.5329×10^{0}	4.5835×10^{0}	4.7693×10^{0}	5.8353×10^{-1}	5.1110×10^{0}	2.9045×10^{-1}
F7	mean	1.1062×10^{3}	9.1473×10^{2}	1.0498×10^{3}	1.3738×10^{3}	9.3759×10^{2}	9.9638×10^{2}	1.5927×10^{3}	1.2041×10^{3}	2.0482×10^{3}	8.4530×10^{2}
	std	7.7925×10^{1}	6.1786×10^{1}	7.7756×10^{1}	1.1714×10^{2}	2.4082×10^{1}	6.8876×10^{1}	8.1290×10^{1}	2.3449×10^{1}	5.9770×10^{1}	1.8963×10^{1}
F8	mean	1.0460×10^{3}	9.1668×10^{2}	9.9349×10^{2}	1.2009×10^{3}	9.3704×10^{2}	9.7754×10^{2}	1.1693×10^{3}	1.0665×10^{3}	1.4773×10^{3}	8.7012×10^{2}
	std	4.3670×10^{1}	4.4117×10^{1}	4.0216×10^{1}	5.9112×10^{1}	2.5014×10^{1}	3.4867×10^{1}	4.2544×10^{1}	1.9459×10^{1}	4.0284×10^{1}	1.5166×10^{1}
F9	mean	8.3531×10^{3}	1.8816×10^{3}	4.5213×10^{3}	1.3494×10^{4}	1.9894×10^{3}	4.6483×10^{3}	1.5159×10^{4}	6.0798×10^{3}	3.6253×10^{4}	9.3806×10^{2}
	std	2.5786×10^{3}	1.0862×10^{3}	1.9566×10^{3}	4.3403×10^{3}	5.9807×10^{2}	3.1208×10^{3}	2.6377×10^{3}	9.4729×10^{2}	3.0653×10^{3}	2.3511×10^{1}
F10	mean	7.6403×10^{3}	7.5664×10^{3}	9.4452×10^{3}	8.5772×10^{3}	6.3367×10^{3}	8.8869×10^{3}	8.7140×10^{3}	8.8005×10^{3}	1.4523×10^{4}	9.1122×10^{3}
	std	1.0761×10^{3}	1.1287×10^{3}	1.2769×10^{3}	7.4429×10^{2}	9.6502×10^{2}	1.7948×10^{3}	7.1677×10^{2}	4.3286×10^{2}	7.6653×10^{2}	1.0523×10^{3}
F11	mean	1.5050×10^{3}	7.5870×10^{3}	1.3224×10^{3}	1.4116×10^{3}	1.3218×10^{3}	1.2753×10^{3}	3.7634×10^{3}	5.5490×10^{3}	1.9058×10^{4}	1.2373×10^{3}
	std	1.1610×10^{2}	5.4615×10^{3}	6.2463×10^{1}	6.8995×10^{1}	5.0505×10^{1}	1.1937×10^{2}	1.4552×10^{3}	1.3218×10^{3}	4.5627×10^{3}	3.4570×10^{1}
F12	mean	4.9124×10^{7}	1.6407×10^{8}	8.0985×10^{6}	3.8673×10^{7}	2.4837×10^{7}	2.4547×10^{6}	7.7347×10^{9}	1.6110×10^{8}	5.8566×10^{10}	2.3050×10^{6}
	std	3.4731×10^{7}	2.2722×10^{8}	4.4290×10^{6}	1.4931×10^{7}	1.5684×10^{7}	1.9458×10^{6}	4.2561×10^{9}	3.2191×10^{7}	1.1844×10^{10}	1.7385×10^{6}
F13	mean	1.1307×10^{5}	3.7811×10^{7}	2.9849×10^{4}	5.7481×10^{4}	3.8589×10^{5}	8.1345×10^{3}	7.8842×10^{8}	3.2849×10^{6}	2.9186×10^{10}	1.6384×10^{4}
	std	7.8157×10^{4}	1.3918×10^{8}	2.6399×10^{4}	2.5284×10^{4}	7.3762×10^{4}	8.0952×10^{3}	1.3100×10^{9}	2.6438×10^{6}	1.3530×10^{10}	7.3246×10^{3}
F14	mean	2.7145×10^{5}	2.3723×10^{6}	2.1064×10^{3}	9.2427×10^{4}	7.0227×10^{4}	8.4542×10^{4}	1.3328×10^{6}	1.0041×10^{6}	5.9339×10^{7}	1.6094×10^{3}
	std	2.3359×10^{5}	2.6745×10^{6}	6.2843×10^{2}	4.9972×10^{4}	4.9273×10^{4}	4.9770×10^{4}	1.2755×10^{6}	5.3453×10^{5}	5.2241×10^{7}	6.1529×10^{1}
F15	mean	3.2460×10^{4}	5.6947×10^{4}	1.9940×10^{4}	3.5078×10^{4}	1.1631×10^{5}	9.3566×10^{3}	1.5408×10^{8}	2.7496×10^{5}	5.3285×10^{9}	2.4152×10^{3}
	std	1.0845×10^{4}	8.4493×10^{4}	1.0371×10^{4}	2.8940×10^{4}	3.1673×10^{4}	5.6892×10^{3}	3.4708×10^{8}	3.3153×10^{5}	2.6793×10^{9}	2.5595×10^{2}
F16	mean	3.4217×10^{3}	3.2651×10^{3}	3.5518×10^{3}	4.0970×10^{3}	2.9612×10^{3}	3.2887×10^{3}	3.9243×10^{3}	3.7294×10^{3}	8.4224×10^{3}	2.6607×10^{3}
	std	4.8200×10^{2}	5.2920×10^{2}	4.2398×10^{2}	6.4794×10^{2}	4.3678×10^{2}	4.6518×10^{2}	5.3428×10^{2}	1.8365×10^{2}	1.5691×10^{3}	3.9953×10^{2}
F17	mean	3.3094×10^{3}	3.0083×10^{3}	3.1540×10^{3}	3.6271×10^{3}	2.9712×10^{3}	2.9262×10^{3}	3.6269×10^{3}	3.2308×10^{3}	7.5213×10^{3}	2.2355×10^{3}
	std	3.4878×10^{2}	3.1702×10^{2}	2.9416×10^{2}	4.5986×10^{2}	3.0963×10^{2}	4.1354×10^{2}	3.7759×10^{2}	1.5611×10^{2}	2.5432×10^{3}	2.4859×10^{2}
F18	mean	2.2454×10^{6}	1.3524×10^{7}	7.9121×10^{4}	1.2186×10^{6}	1.4909×10^{6}	1.3034×10^{6}	1.0447×10^{7}	5.7319×10^{6}	1.8741×10^{8}	7.3547×10^{4}
	std	1.7642×10^{6}	3.0760×10^{7}	3.9014×10^{4}	5.7951×10^{5}	7.6771×10^{5}	8.8956×10^{5}	1.2223×10^{7}	2.8231×10^{6}	9.5853×10^{7}	3.3534×10^{4}
F19	mean	7.5591×10^{5}	2.5697×10^{5}	1.5177×10^{4}	1.7763×10^{6}	1.0131×10^{5}	1.7692×10^{4}	7.2366×10^{6}	9.2403×10^{4}	1.4812×10^{9}	2.1019×10^{3}
	std	6.7157×10^{5}	7.5343×10^{5}	1.5102×10^{4}	1.5223×10^{6}	3.6759×10^{4}	1.2692×10^{4}	1.3159×10^{7}	4.7760×10^{4}	9.7645×10^{8}	7.1947×10^{1}
F20	mean	3.0977×10^{3}	2.9093×10^{3}	3.1877×10^{3}	3.5576×10^{3}	2.7033×10^{3}	3.1214×10^{3}	2.8497×10^{3}	3.1911×10^{3}	4.1303×10^{3}	2.6913×10^{3}
	std	3.5757×10^{2}	3.0462×10^{2}	2.8118×10^{2}	3.0895×10^{2}	2.6595×10^{2}	4.9475×10^{2}	1.8977×10^{2}	1.4617×10^{2}	3.1682×10^{2}	3.2473×10^{2}
F21	mean	2.5657×10^{3}	2.4203×10^{3}	2.4681×10^{3}	2.8562×10^{3}	2.4412×10^{3}	2.4565×10^{3}	2.7201×10^{3}	2.5596×10^{3}	3.1430×10^{3}	2.3770×10^{3}
	std	4.9311×10^{1}	3.8835×10^{1}	2.7464×10^{1}	8.9976×10^{1}	2.7099×10^{1}	3.8085×10^{1}	5.2781×10^{1}	1.7623×10^{1}	7.4584×10^{1}	1.5524×10^{1}
F22	mean	9.2477×10^{3}	8.6038×10^{3}	1.1337×10^{4}	1.0759×10^{4}	6.8848×10^{3}	9.4578×10^{3}	1.0732×10^{4}	9.6054×10^{3}	1.6564×10^{4}	1.0790×10^{4}
	std	1.6491×10^{3}	1.3703×10^{3}	1.1525×10^{3}	8.5380×10^{2}	2.6643×10^{3}	3.1239×10^{3}	7.1835×10^{2}	2.2610×10^{3}	6.2346×10^{2}	1.3090×10^{3}
F23	mean	3.0323×10^{3}	2.9419×10^{3}	2.9489×10^{3}	4.1634×10^{3}	2.8948×10^{3}	2.9229×10^{3}	3.8086×10^{3}	3.0075×10^{3}	4.2639×10^{3}	2.7975×10^{3}
	std	6.7370×10^{1}	1.1486×10^{2}	4.7123×10^{1}	2.0214×10^{2}	3.0457×10^{1}	4.4311×10^{1}	1.4121×10^{2}	1.9362×10^{1}	2.1007×10^{2}	2.0971×10^{1}

Table 3. *Cont.*

ID	Metric	VPPSO	SLPSO	ALA	TJO	GJA	NDO	ZOA	SSLO	GoldSA	MIGoldSA
F24	mean	3.1784×10^{3}	3.1201×10^{3}	3.1108×10^{3}	4.1466×10^{3}	3.0660×10^{3}	3.0838×10^{3}	4.0831×10^{3}	3.1575×10^{3}	4.4775×10^{3}	2.9648×10^{3}
	std	6.2865×10^{1}	1.3732×10^{2}	6.1773×10^{1}	3.8431×10^{2}	4.2955×10^{1}	4.8696×10^{1}	1.3815×10^{2}	2.4879×10^{1}	2.0636×10^{2}	1.6394×10^{1}
F25	mean	3.1511×10^{3}	3.4950×10^{3}	3.0894×10^{3}	3.1082×10^{3}	3.1138×10^{3}	3.0735×10^{3}	4.8980×10^{3}	3.2914×10^{3}	1.2921×10^{4}	3.0459×10^{3}
	std	4.2629×10^{1}	5.1496×10^{2}	3.1576×10^{1}	1.7801×10^{1}	3.0817×10^{1}	2.8124×10^{1}	6.4458×10^{2}	3.6773×10^{1}	1.5125×10^{3}	2.3470×10^{1}
F26	mean	6.6383×10^{3}	6.0891×10^{3}	6.0094×10^{3}	1.1611×10^{4}	5.0408×10^{3}	6.0960×10^{3}	1.2762×10^{4}	6.6782×10^{3}	1.7174×10^{4}	4.4236×10^{3}
	std	2.3982×10^{3}	1.2140×10^{3}	5.0308×10^{2}	2.8025×10^{3}	2.6174×10^{3}	7.9529×10^{2}	9.9059×10^{2}	1.4859×10^{2}	1.0056×10^{3}	1.4485×10^{2}
F27	mean	3.6670×10^{3}	3.6485×10^{3}	3.4557×10^{3}	5.7171×10^{3}	3.4768×10^{3}	3.4899×10^{3}	5.4455×10^{3}	3.5098×10^{3}	6.6401×10^{3}	3.3477×10^{3}
	std	1.4770×10^{2}	1.9470×10^{2}	1.2015×10^{2}	7.7327×10^{2}	1.0335×10^{2}	1.1012×10^{2}	4.0085×10^{2}	3.0803×10^{1}	9.0630×10^{2}	7.0323×10^{1}
F28	mean	3.4443×10^{3}	4.6069×10^{3}	4.1021×10^{3}	3.3600×10^{3}	3.3884×10^{3}	3.3559×10^{3}	5.6640×10^{3}	3.5907×10^{3}	1.1807×10^{4}	4.2406×10^{3}
	std	6.9941×10^{1}	7.3630×10^{2}	1.7479×10^{3}	4.1023×10^{1}	3.7515×10^{1}	3.9262×10^{1}	6.2658×10^{2}	5.2339×10^{1}	1.1547×10^{3}	1.4461×10^{3}
F29	mean	5.1993×10^{3}	4.8520×10^{3}	4.4996×10^{3}	6.6427×10^{3}	4.1603×10^{3}	4.0839×10^{3}	8.1252×10^{3}	4.3724×10^{3}	9.4026×10^{4}	3.6607×10^{3}
	std	3.9444×10^{2}	4.5561×10^{2}	3.2452×10^{2}	5.4717×10^{2}	3.1224×10^{2}	2.9229×10^{2}	1.4485×10^{3}	1.5847×10^{2}	2.7531×10^{5}	2.1093×10^{2}
F30	mean	5.5415×10^{7}	6.7170×10^{7}	2.9075×10^{6}	8.6089×10^{7}	2.8467×10^{6}	1.1772×10^{6}	2.8070×10^{8}	8.0198×10^{6}	3.7273×10^{9}	1.0572×10^{6}
	std	1.6594×10^{7}	1.3369×10^{8}	1.6712×10^{6}	6.4260×10^{6}	4.3195×10^{5}	2.5855×10^{5}	2.4998×10^{8}	2.0065×10^{6}	1.6131×10^{9}	1.7910×10^{5}

Figure 6 shows that MIGoldSA generally exhibits faster convergence and better final optimization accuracy on the CEC2017 benchmark functions. Compared with the other algorithms, MIGoldSA can continuously reduce the objective value during the iterative process and obtain lower convergence curves on most tested functions. This indicates that the proposed strategies effectively improve the search efficiency of GoldSA and enhance its ability to escape local optima. In particular, MIGoldSA maintains stable convergence trends in both 30-dimensional and 50-dimensional cases, demonstrating good robustness when dealing with higher-dimensional optimization problems. Overall, the convergence results verify that MIGoldSA achieves a better balance between global exploration and local exploitation.

Tables 2 and 3 present the performance comparison results of different algorithms on the CEC2017 benchmark functions under 30-dimensional and 50-dimensional settings, respectively. It can be observed that MIGoldSA achieves highly competitive optimization performance on most benchmark functions, especially on complex multimodal, hybrid, and composition functions. Compared with the other algorithms, MIGoldSA generally obtains lower mean values and smaller standard deviations on many test functions, indicating that the proposed algorithm possesses better convergence accuracy, stronger robustness, and more stable optimization capability. In particular, on several difficult functions such as F12–F19 and F23–F30, MIGoldSA demonstrates superior search performance and effectively avoids premature convergence. Furthermore, when the problem dimension increases from 30 to 50, the optimization difficulty of the benchmark functions becomes significantly higher. Nevertheless, MIGoldSA still maintains relatively stable optimization performance and exhibits strong adaptability in high-dimensional search spaces. In contrast, some comparison algorithms experience obvious performance degradation, larger fluctuations, or premature stagnation as the dimensionality increases. This demonstrates that the proposed adaptive chaotic opposition initialization strategy, dynamic elite-guided sine evolution strategy, and Economic Investment Strategy effectively enhance population diversity, strengthen the balance between exploration and exploitation, and improve the global optimization capability of the original GoldSA. Overall, the experimental results further verify the effectiveness, robustness, and superiority of MIGoldSA in solving complex high-dimensional optimization problems.

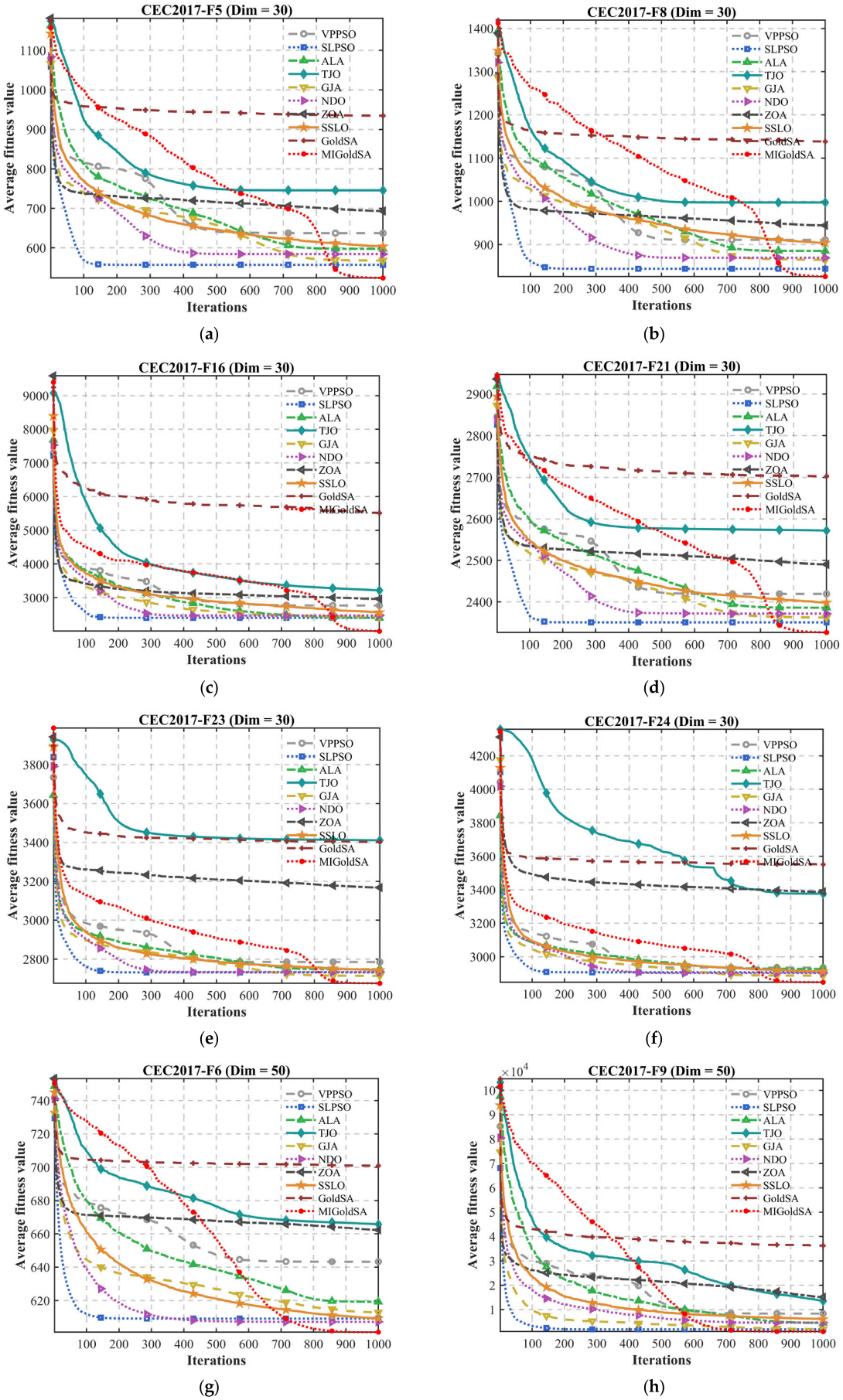


Figure 6. *Cont.*

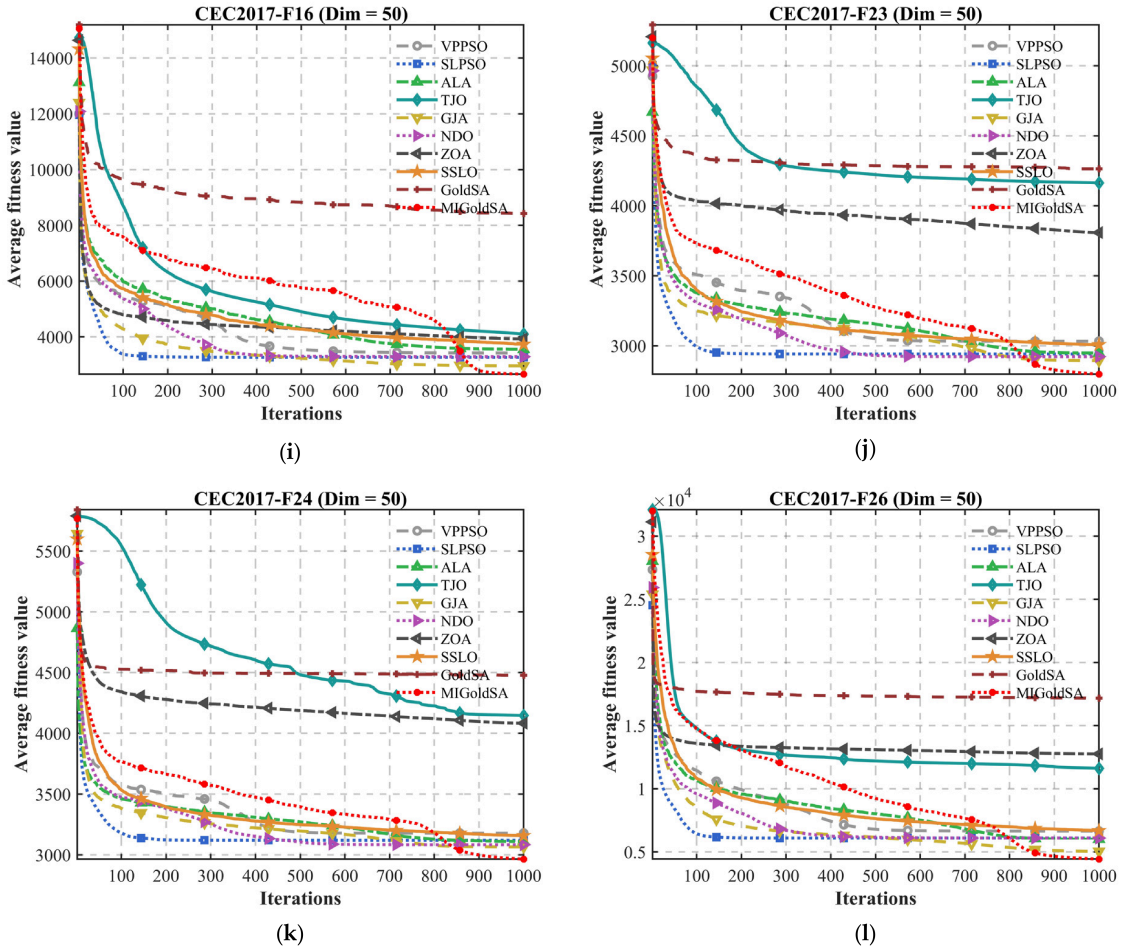


Figure 6. Convergence Behavior of Different Algorithms on the CEC2017 Benchmark Functions.

Figure 7 presents the boxplot analysis results of different algorithms on the CEC2017 benchmark functions. It can be observed that MIGoldSA generally exhibits narrower box ranges and fewer outliers on most benchmark functions compared with the other algorithms. This indicates that the proposed algorithm possesses better stability and stronger robustness during repeated independent runs. In addition, the median values of MIGoldSA are lower than those of most comparison algorithms, further demonstrating its superior optimization accuracy and convergence capability.

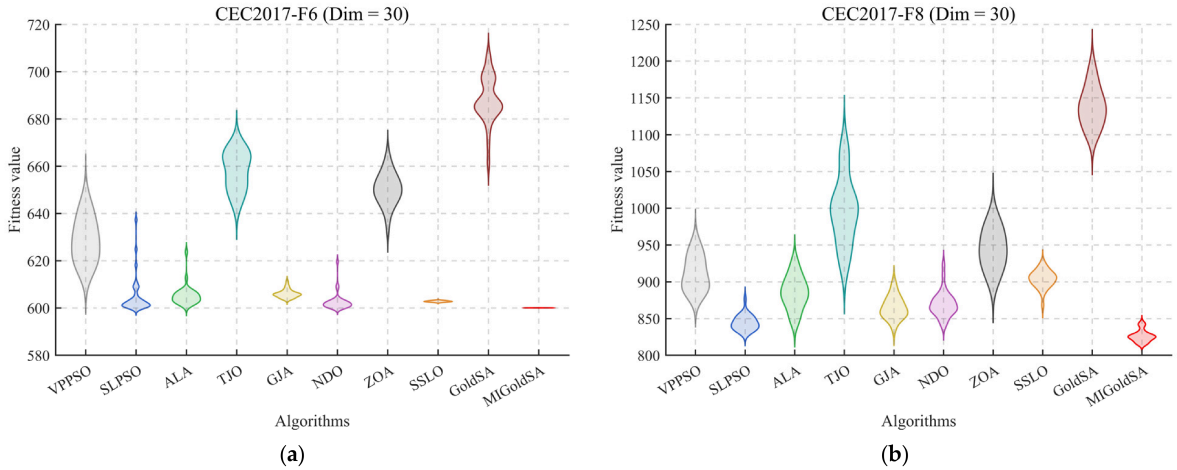


Figure 7. *Cont.*

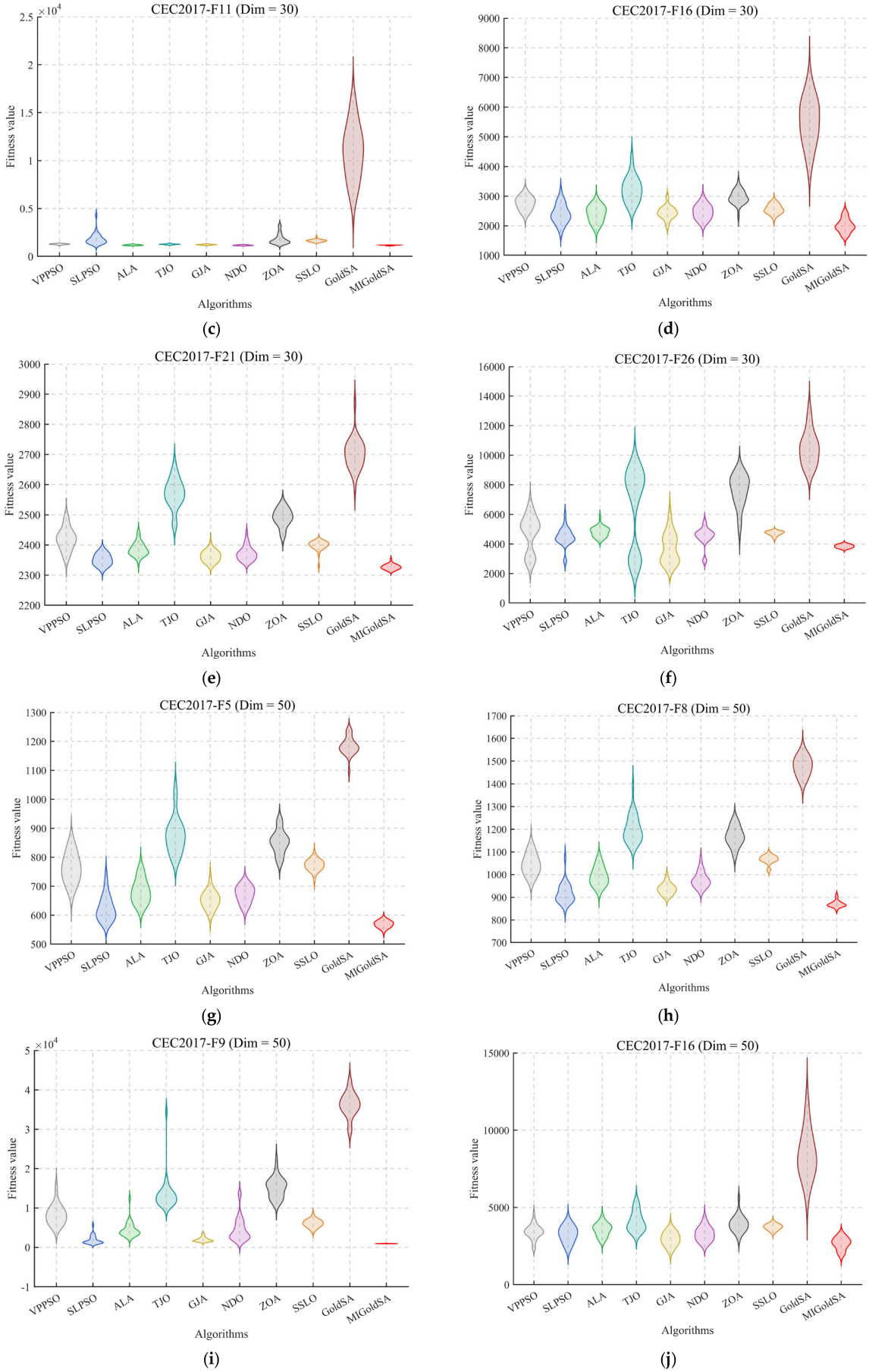


Figure 7. *Cont.*

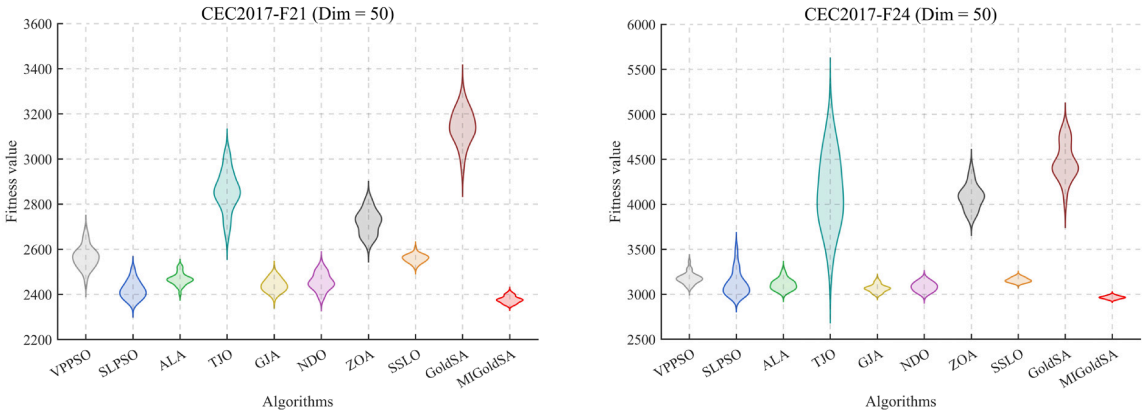


Figure 7. Boxplot Analysis of Different Algorithms on the CEC2017 Benchmark Functions.

Moreover, the distribution of MIGoldSA results is more concentrated, especially on complex multimodal and hybrid functions, indicating that the proposed strategies effectively maintain population diversity and reduce the influence of random fluctuations during the optimization process. In contrast, several comparison algorithms exhibit wider box distributions and larger fluctuations, suggesting weaker stability and a higher risk of falling into local optima. Overall, the boxplot analysis further verifies that MIGoldSA achieves a better balance between exploration and exploitation and demonstrates excellent reliability and robustness in solving complex optimization problems.

5.4. Performance Evaluation on the CEC2022 Benchmark Functions

In this section, the optimization performance of MIGoldSA is evaluated on the 10-dimensional and 20-dimensional CEC2022 benchmark functions and compared with nine advanced state-of-the-art optimization algorithms. The corresponding experimental results are presented in Tables 4 and 5, where "mean" and "std" denote the average value and standard deviation obtained from 30 independent runs, respectively. To more intuitively demonstrate the convergence behavior of different algorithms during the optimization process, the convergence curves of the ten algorithms are illustrated in Figure 8. In addition, to further analyze the robustness and stability of the algorithms over multiple independent runs, the boxplot distributions of the experimental results obtained from 30 runs are provided in Figure 9.

Table 4. Performance Results of Different Algorithms on the CEC2022 Benchmark Functions (Dim = 10).

ID	Metric	VPPSO	SLPSO	ALA	TJO	GJA	NDO	ZOA	SSLO	GoldSA	MIGoldSA
F1	mean	3.0311×10^{2}	1.6379×10^{3}	3.0000×10^{2}	3.0000×10^{2}	3.0011×10^{2}	3.0000×10^{2}	5.7637×10^{2}	1.7815×10^{3}	9.3224×10^{3}	3.0000×10^{2}
	std	1.1016×10^{1}	1.7302×10^{3}	7.9859×10^{-12}	5.1806×10^{-5}	2.5974×10^{-2}	8.0593×10^{-8}	4.6768×10^{2}	5.2664×10^{2}	1.1650×10^{3}	4.3522×10^{-14}
F2	mean	4.1193×10^{2}	4.1497×10^{2}	4.0616×10^{2}	4.1579×10^{2}	4.0436×10^{2}	4.0702×10^{2}	4.4156×10^{2}	4.0304×10^{2}	7.5937×10^{2}	4.0596×10^{2}
	std	2.0561×10^{1}	2.0176×10^{1}	2.7224×10^{0}	2.7838×10^{1}	1.3002×10^{1}	1.7609×10^{1}	3.0925×10^{1}	2.3528×10^{0}	1.9709×10^{2}	2.4562×10^{0}
F3	mean	6.0347×10^{2}	6.0010×10^{2}	6.0000×10^{2}	6.1853×10^{2}	6.0024×10^{2}	6.0001×10^{2}	6.1735×10^{2}	6.0000×10^{2}	6.5250×10^{2}	6.0000×10^{2}
	std	2.7596×10^{0}	4.7483×10^{-1}	2.0187×10^{-4}	9.4373×10^{0}	7.4875×10^{-2}	1.1142×10^{-2}	5.9866×10^{0}	1.4971×10^{-4}	1.0688×10^{1}	2.6257×10^{-6}
F4	mean	8.1715×10^{2}	8.0957×10^{2}	8.1304×10^{2}	8.3207×10^{2}	8.0975×10^{2}	8.1569×10^{2}	8.1265×10^{2}	8.0725×10^{2}	8.5130×10^{2}	8.0902×10^{2}
	std	7.7188×10^{0}	5.7424×10^{0}	4.8250×10^{0}	1.1508×10^{1}	3.8922×10^{0}	7.0677×10^{0}	4.6101×10^{0}	2.1447×10^{0}	1.0500×10^{1}	4.4183×10^{0}
F5	mean	9.0148×10^{2}	9.0032×10^{2}	9.0003×10^{2}	9.0501×10^{2}	9.0003×10^{2}	9.0122×10^{2}	1.0066×10^{3}	9.0037×10^{2}	1.6447×10^{3}	9.0000×10^{2}
	std	1.7069×10^{0}	1.4408×10^{0}	8.6397×10^{-2}	1.8693×10^{1}	6.7915×10^{-3}	1.9852×10^{0}	5.9061×10^{1}	2.6782×10^{-1}	2.4222×10^{2}	0.0000×10^{0}

Table 4. *Cont.*

ID	Metric	VPPSO	SLPSO	ALA	TJO	GJA	NDO	ZOA	SSLO	GoldSA	MIGoldSA
F6	mean	3.9008×10^{3}	3.8765×10^{3}	1.8079×10^{3}	3.1663×10^{3}	2.9363×10^{3}	3.2961×10^{3}	3.0928×10^{3}	2.1859×10^{3}	1.4848×10^{6}	1.8006×10^{3}
	std	2.1373×10^{3}	1.9762×10^{3}	4.9050×10^{0}	1.8996×10^{3}	8.8467×10^{2}	1.5680×10^{3}	1.1583×10^{3}	2.9812×10^{2}	2.6772×10^{6}	3.6540×10^{-1}
F7	mean	2.0325×10^{3}	2.0217×10^{3}	2.0191×10^{3}	2.0471×10^{3}	2.0151×10^{3}	2.0130×10^{3}	2.0376×10^{3}	2.0045×10^{3}	2.1022×10^{3}	2.0186×10^{3}
	std	1.0518×10^{1}	1.0516×10^{1}	5.7893×10^{0}	1.2570×10^{1}	8.8910×10^{0}	1.0985×10^{1}	1.5682×10^{1}	1.7807×10^{0}	2.6887×10^{1}	6.9678×10^{0}
F8	mean	2.2248×10^{3}	2.2231×10^{3}	2.2103×10^{3}	2.2316×10^{3}	2.2189×10^{3}	2.2202×10^{3}	2.2255×10^{3}	2.2163×10^{3}	2.2431×10^{3}	2.2043×10^{3}
	std	2.2555×10^{0}	4.7370×10^{0}	7.9226×10^{0}	2.1695×10^{1}	7.5047×10^{0}	2.3211×10^{0}	2.5116×10^{0}	4.5334×10^{0}	1.7796×10^{1}	6.3002×10^{0}
F9	mean	2.5342×10^{3}	2.5488×10^{3}	2.5293×10^{3}	2.5294×10^{3}	2.5293×10^{3}	2.5293×10^{3}	2.5976×10^{3}	2.5293×10^{3}	2.6944×10^{3}	2.5293×10^{3}
	std	2.6815×10^{1}	3.2772×10^{1}	0.0000×10^{0}	7.0042×10^{-2}	1.5515×10^{-3}	4.8510×10^{-13}	3.9184×10^{1}	6.5552×10^{-10}	4.3995×10^{1}	0.0000×10^{0}
F10	mean	2.5192×10^{3}	2.5566×10^{3}	2.5152×10^{3}	2.6105×10^{3}	2.5365×10^{3}	2.5005×10^{3}	2.5614×10^{3}	2.5004×10^{3}	2.6334×10^{3}	2.5003×10^{3}
	std	4.2555×10^{1}	5.7139×10^{1}	3.8629×10^{1}	1.7573×10^{2}	5.2077×10^{1}	2.1825×10^{-1}	6.1882×10^{1}	6.3435×10^{-2}	1.0886×10^{2}	7.6450×10^{-2}
F11	mean	2.6637×10^{3}	2.7546×10^{3}	2.6467×10^{3}	2.6633×10^{3}	2.6475×10^{3}	2.6401×10^{3}	2.7968×10^{3}	2.6158×10^{3}	3.3428×10^{3}	2.6056×10^{3}
	std	1.3073×10^{2}	1.1443×10^{2}	1.2243×10^{2}	1.2994×10^{2}	9.8233×10^{1}	7.8256×10^{1}	1.6128×10^{2}	4.5967×10^{1}	5.7381×10^{2}	3.0424×10^{1}
F12	mean	2.8627×10^{3}	2.8695×10^{3}	2.8618×10^{3}	2.9090×10^{3}	2.8652×10^{3}	2.8671×10^{3}	2.9210×10^{3}	2.8603×10^{3}	2.9873×10^{3}	2.8614×10^{3}
	std	1.6788×10^{0}	7.8862×10^{0}	1.2062×10^{0}	4.5513×10^{1}	1.8434×10^{0}	1.9288×10^{0}	3.5013×10^{1}	1.1258×10^{0}	8.5252×10^{1}	9.6572×10^{-1}

Table 5. Performance Results of Different Algorithms on the CEC2022 Benchmark Functions (Dim = 20).

ID	Metric	VPPSO	SLPSO	ALA	TJO	GJA	NDO	ZOA	SSLO	GoldSA	MIGoldSA
F1	mean	1.2036×10^{3}	1.6623×10^{4}	3.1899×10^{2}	3.0008×10^{2}	3.0109×10^{2}	4.6532×10^{3}	1.0108×10^{4}	2.4522×10^{4}	6.8513×10^{4}	5.5101×10^{2}
	std	5.6140×10^{2}	8.7158×10^{3}	1.7189×10^{1}	4.1381×10^{-2}	3.3516×10^{-1}	1.9855×10^{3}	3.2506×10^{3}	4.9087×10^{3}	3.3182×10^{4}	2.4836×10^{2}
F2	mean	4.5714×10^{2}	4.6607×10^{2}	4.4493×10^{2}	4.5501×10^{2}	4.5490×10^{2}	4.5069×10^{2}	5.9822×10^{2}	4.4699×10^{2}	2.1348×10^{3}	4.4867×10^{2}
	std	1.9810×10^{1}	2.7371×10^{1}	1.7189×10^{1}	1.0703×10^{1}	1.0205×10^{1}	1.6028×10^{1}	8.3503×10^{1}	5.9223×10^{0}	7.0225×10^{2}	1.2782×10^{0}
F3	mean	6.1837×10^{2}	6.0070×10^{2}	6.0038×10^{2}	6.5371×10^{2}	6.0235×10^{2}	6.0045×10^{2}	6.3933×10^{2}	6.0049×10^{2}	6.8101×10^{2}	6.0000×10^{2}
	std	7.1862×10^{0}	1.4787×10^{0}	3.4262×10^{-1}	8.7089×10^{0}	1.0791×10^{0}	5.9772×10^{-1}	6.3455×10^{0}	9.8776×10^{-2}	7.0252×10^{0}	5.3748×10^{-4}
F4	mean	8.5642×10^{2}	8.2523×10^{2}	8.5050×10^{2}	9.0881×10^{2}	8.3463×10^{2}	8.4476×10^{2}	8.5808×10^{2}	8.4281×10^{2}	9.6426×10^{2}	8.1457×10^{2}
	std	1.4230×10^{1}	1.3399×10^{1}	1.6598×10^{1}	2.3888×10^{1}	1.3820×10^{1}	1.3699×10^{1}	9.0571×10^{0}	7.8719×10^{0}	1.6226×10^{1}	5.1481×10^{0}
F5	mean	1.2665×10^{3}	9.1421×10^{2}	9.3264×10^{2}	1.8323×10^{3}	9.1107×10^{2}	9.3416×10^{2}	1.7481×10^{3}	1.0238×10^{3}	3.5671×10^{3}	9.0003×10^{2}
	std	2.6886×10^{2}	2.6562×10^{1}	2.7365×10^{1}	4.0346×10^{2}	8.2394×10^{0}	5.2462×10^{1}	2.2159×10^{2}	3.6551×10^{1}	4.4631×10^{2}	8.6974×10^{-2}
F6	mean	4.6218×10^{3}	4.9753×10^{5}	4.7260×10^{3}	4.2688×10^{3}	1.9557×10^{4}	4.9942×10^{3}	5.9108×10^{6}	1.6788×10^{5}	1.2056×10^{9}	3.0903×10^{3}
	std	4.4845×10^{3}	2.7048×10^{6}	4.8277×10^{3}	3.1398×10^{3}	6.6385×10^{3}	3.5737×10^{3}	1.1603×10^{7}	1.4528×10^{5}	9.8680×10^{8}	3.3180×10^{3}
F7	mean	2.0853×10^{3}	2.0573×10^{3}	2.0445×10^{3}	2.1413×10^{3}	2.0482×10^{3}	2.0429×10^{3}	2.0932×10^{3}	2.0493×10^{3}	2.2395×10^{3}	2.0418×10^{3}
	std	2.9922×10^{1}	3.3434×10^{1}	1.3220×10^{1}	4.6224×10^{1}	2.6424×10^{1}	1.5974×10^{1}	1.8890×10^{1}	5.6464×10^{0}	7.1950×10^{1}	2.9682×10^{1}
F8	mean	2.2414×10^{3}	2.2599×10^{3}	2.2283×10^{3}	2.2894×10^{3}	2.2498×10^{3}	2.2312×10^{3}	2.2681×10^{3}	2.2253×10^{3}	2.2836×10^{3}	2.2233×10^{3}
	std	3.1805×10^{1}	5.6252×10^{1}	2.8517×10^{0}	6.2113×10^{1}	4.9450×10^{1}	3.0238×10^{1}	5.6584×10^{1}	8.2642×10^{-1}	6.5000×10^{1}	1.6282×10^{0}
F9	mean	2.4848×10^{3}	2.4996×10^{3}	2.4808×10^{3}	2.4814×10^{3}	2.4809×10^{3}	2.4836×10^{3}	2.5928×10^{3}	2.4813×10^{3}	2.8989×10^{3}	2.4808×10^{3}
	std	5.5609×10^{0}	1.8568×10^{1}	2.6990×10^{-5}	2.4082×10^{-1}	1.1687×10^{-1}	1.5184×10^{1}	6.4890×10^{1}	2.1255×10^{-1}	1.4171×10^{2}	1.3306×10^{-10}
F10	mean	2.7133×10^{3}	3.1868×10^{3}	3.4900×10^{3}	4.0505×10^{3}	2.5769×10^{3}	2.5183×10^{3}	3.1201×10^{3}	2.5175×10^{3}	6.1448×10^{3}	3.0886×10^{3}
	std	5.1874×10^{2}	6.6961×10^{2}	7.8448×10^{2}	1.1292×10^{3}	9.1312×10^{1}	4.5866×10^{1}	8.7543×10^{2}	4.4009×10^{1}	1.0547×10^{3}	6.3513×10^{2}
F11	mean	2.9200×10^{3}	3.1222×10^{3}	2.9300×10^{3}	2.9135×10^{3}	2.9084×10^{3}	2.9167×10^{3}	4.5822×10^{3}	2.9241×10^{3}	8.3361×10^{3}	2.9400×10^{3}
	std	7.6111×10^{1}	2.6319×10^{2}	7.9445×10^{1}	9.7237×10^{1}	8.2993×10^{1}	7.4664×10^{1}	9.7125×10^{2}	5.8468×10^{1}	7.5116×10^{2}	4.9827×10^{1}
F12	mean	2.9756×10^{3}	2.9795×10^{3}	2.9501×10^{3}	3.2932×10^{3}	2.9620×10^{3}	2.9650×10^{3}	3.3281×10^{3}	2.9452×10^{3}	3.4043×10^{3}	2.9402×10^{3}
	std	3.1575×10^{1}	3.2493×10^{1}	9.4587×10^{0}	1.9876×10^{2}	1.6775×10^{1}	1.5845×10^{1}	9.8401×10^{1}	2.8539×10^{0}	2.7877×10^{2}	4.3842×10^{0}

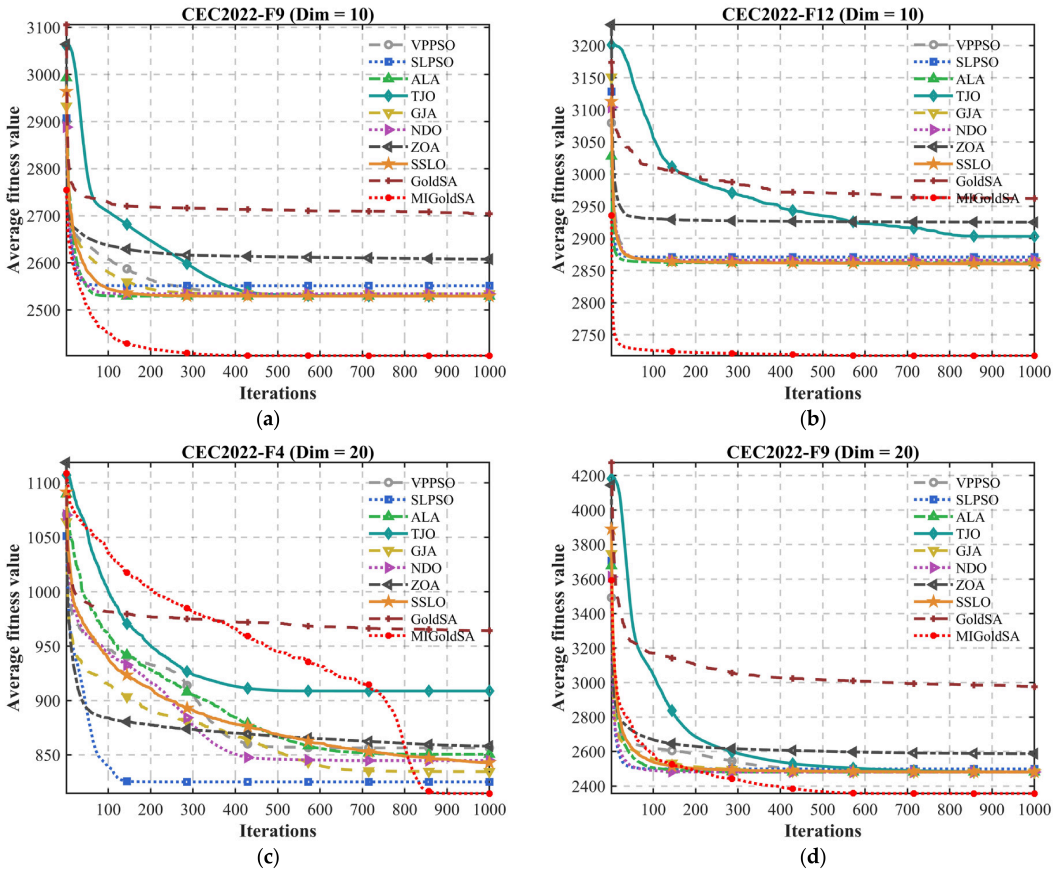


Figure 8. Convergence Behavior of Different Algorithms on the CEC2022 Benchmark Functions.

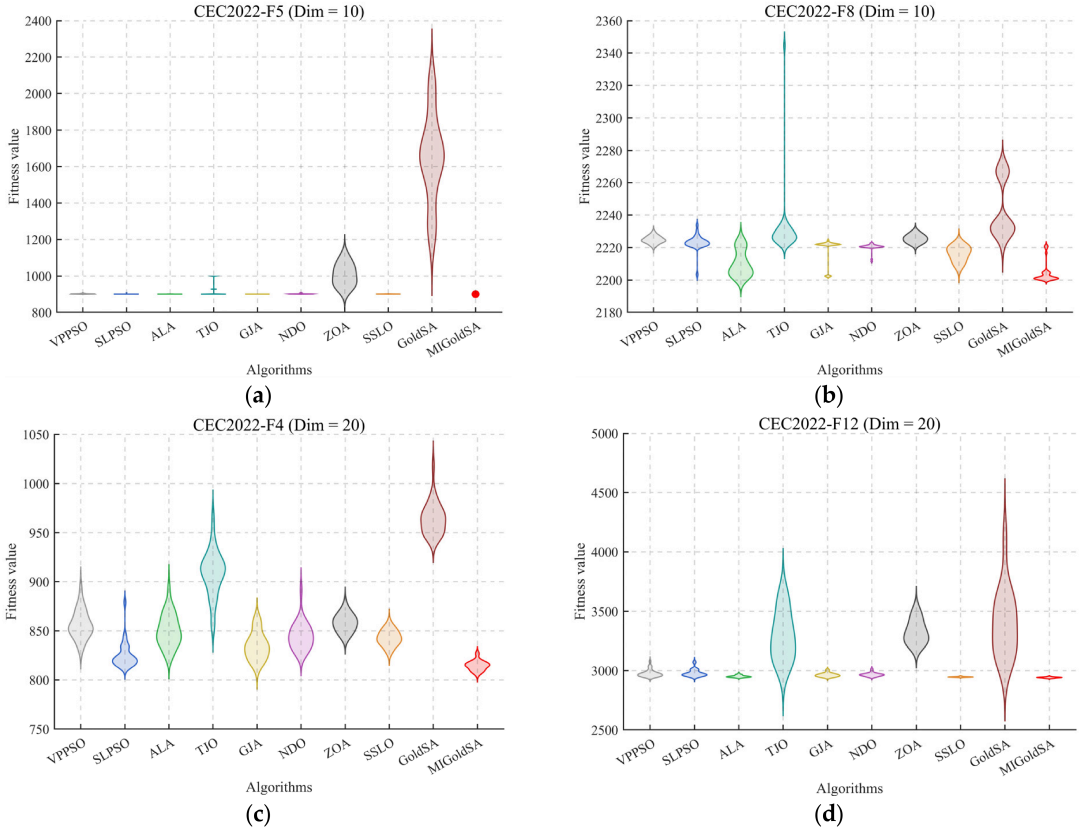


Figure 9. Boxplot Analysis of Different Algorithms on the CEC2022 Benchmark Functions.

Figure 8 illustrates the convergence behavior of different algorithms on the CEC2022 benchmark functions. It can be observed that MIGoldSA demonstrates faster convergence speed and better optimization accuracy on most benchmark functions compared with the other algorithms. During the early optimization stage, MIGoldSA can rapidly reduce the objective values, indicating that the proposed initialization and exploration strategies effectively improve the global search capability of the algorithm. Meanwhile, in the later optimization stage, MIGoldSA still maintains continuous convergence trends and achieves lower final fitness values, demonstrating strong local exploitation capability and high convergence precision. In addition, MIGoldSA exhibits relatively stable convergence performance on complex multimodal and composition functions in the CEC2022 test suite. Compared with several comparison algorithms that suffer from premature convergence or stagnation during iterations, MIGoldSA can effectively avoid falling into local optima and maintain superior search capability throughout the optimization process. This indicates that the proposed adaptive chaotic opposition initialization strategy, dynamic elite-guided sine evolution strategy, and Economic Investment Strategy collaboratively enhance the balance between exploration and exploitation. Overall, the convergence results on the CEC2022 benchmark functions further verify the effectiveness, robustness, and optimization superiority of the proposed MIGoldSA algorithm in solving complex high-dimensional optimization problems.

Tables 4 and 5 present the performance comparison results of different algorithms on the CEC2022 benchmark functions under 10-dimensional and 20-dimensional settings, respectively. It can be observed that MIGoldSA achieves highly competitive optimization results on most benchmark functions and generally obtains lower mean values and smaller standard deviations compared with the other algorithms. These results indicate that the proposed algorithm possesses strong optimization capability, good convergence accuracy, and stable search performance on the CEC2022 benchmark suite. In particular, on several complex multimodal and hybrid functions, MIGoldSA demonstrates superior ability to avoid local optima and maintain stable convergence behavior during repeated independent runs. Furthermore, when the problem dimension increases from 10 to 20, the optimization difficulty of the benchmark functions becomes more challenging. Nevertheless, MIGoldSA still maintains relatively stable optimization performance and exhibits strong adaptability in higher-dimensional search spaces. Compared with several comparison algorithms that experience obvious performance degradation or larger fluctuations as dimensionality increases, MIGoldSA continues to achieve competitive optimization results on most benchmark functions. This demonstrates that the proposed adaptive chaotic opposition initialization strategy, dynamic elite-guided sine evolution strategy, and Economic Investment Strategy effectively enhance population diversity, strengthen the balance between exploration and exploitation, and improve the robustness of the original GoldSA. Overall, the experimental results on the CEC2022 benchmark functions further verify the effectiveness and superiority of MIGoldSA in solving complex high-dimensional optimization problems.

Figure 9 presents the boxplot analysis results of different algorithms on the CEC2022 benchmark functions. It can be observed that MIGoldSA generally achieves narrower box distributions and lower median values on most benchmark functions compared with the other algorithms. This indicates that the proposed algorithm possesses stronger stability, higher optimization accuracy, and better robustness during repeated independent runs. In particular, the relatively concentrated distribution of MIGoldSA demonstrates that the algorithm can consistently obtain high-quality solutions with smaller performance fluctuations. Furthermore, compared with several comparison algorithms that exhibit wider box ranges and more outliers, MIGoldSA maintains more stable optimization behavior on complex multimodal and composition functions. This suggests that the proposed

strategies effectively improve population diversity and reduce the probability of premature convergence during the optimization process. As the complexity of the CEC2022 benchmark functions increases, MIGoldSA still maintains good search consistency and convergence reliability, further verifying its superior capability in balancing global exploration and local exploitation. Overall, the boxplot analysis on the CEC2022 benchmark functions further confirms the effectiveness and robustness of the proposed MIGoldSA algorithm in solving complex optimization problems.

5.5. Statistical Analysis

To further investigate whether the performance differences among the compared algorithms are statistically significant, a comprehensive statistical analysis of MIGoldSA is conducted in this section. Specifically, two widely used nonparametric statistical methods, namely the Wilcoxon Signed-Rank Test and the Friedman Mean Rank Test, are employed to evaluate the superiority and robustness of the proposed algorithm. The detailed analyses are presented as follows:

5.5.1. Wilcoxon Signed Rank Test

The Wilcoxon Signed-Rank Test is a widely used nonparametric statistical method for evaluating the performance differences between optimization algorithms. Compared with traditional parametric statistical tests, the Wilcoxon Signed-Rank Test does not require the experimental data to satisfy normal distribution assumptions, making it particularly suitable for analyzing the stochastic results generated by metaheuristic algorithms. Due to the randomness and uncertainty inherent in heuristic optimization processes, this test has been extensively adopted in the literature to determine whether the performance differences among optimization algorithms are statistically significant. Therefore, in this subsection, the Wilcoxon Signed-Rank Test is employed to statistically evaluate the optimization performance of MIGoldSA and further verify the effectiveness and superiority of the proposed algorithm [34]. The experimental results are shown below.

Figure 10 presents the Wilcoxon Signed-Rank Test results of MIGoldSA compared with the other state-of-the-art algorithms on the benchmark functions. It can be observed that MIGoldSA achieves a larger number of wins against most comparison algorithms on both the CEC2017 and CEC2022 benchmark suites. This indicates that the optimization performance improvements achieved by MIGoldSA are statistically significant rather than being caused by random fluctuations. In particular, MIGoldSA demonstrates clear advantages over several traditional and recently proposed optimization algorithms on complex multimodal and hybrid benchmark functions. Furthermore, the relatively small numbers of losses and ties further verify the robustness and effectiveness of the proposed algorithm. The statistical results demonstrate that the proposed adaptive chaotic opposition initialization strategy, dynamic elite-guided sine evolution strategy, and Economic Investment Strategy effectively improve the search capability and convergence stability of the original GoldSA. Overall, the Wilcoxon Signed-Rank Test results confirm that MIGoldSA possesses superior optimization performance and strong statistical reliability in solving complex optimization problems.

5.5.2. Friedman Mean Rank Test

The Friedman Mean Rank Test is a widely used nonparametric statistical analysis method for evaluating the overall performance differences among multiple optimization algorithms across multiple benchmark problems. Unlike traditional parametric statistical methods, the Friedman Mean Rank Test does not require the experimental results to satisfy normal distribution assumptions, making it particularly suitable for analyzing the stochastic optimization results produced by metaheuristic algorithms. By calculating the average

ranking of each algorithm on different benchmark functions, the Friedman Mean Rank Test can comprehensively reflect the relative optimization performance and robustness of competing algorithms. Due to its effectiveness and reliability in statistical performance evaluation, this test has been extensively adopted in the field of swarm intelligence and evolutionary optimization. Therefore, in this subsection, the Friedman Mean Rank Test is employed to further evaluate the optimization performance of the proposed MIGoldSA algorithm and verify its superiority compared with other state-of-the-art optimization algorithms [35]. The experimental results are shown below.

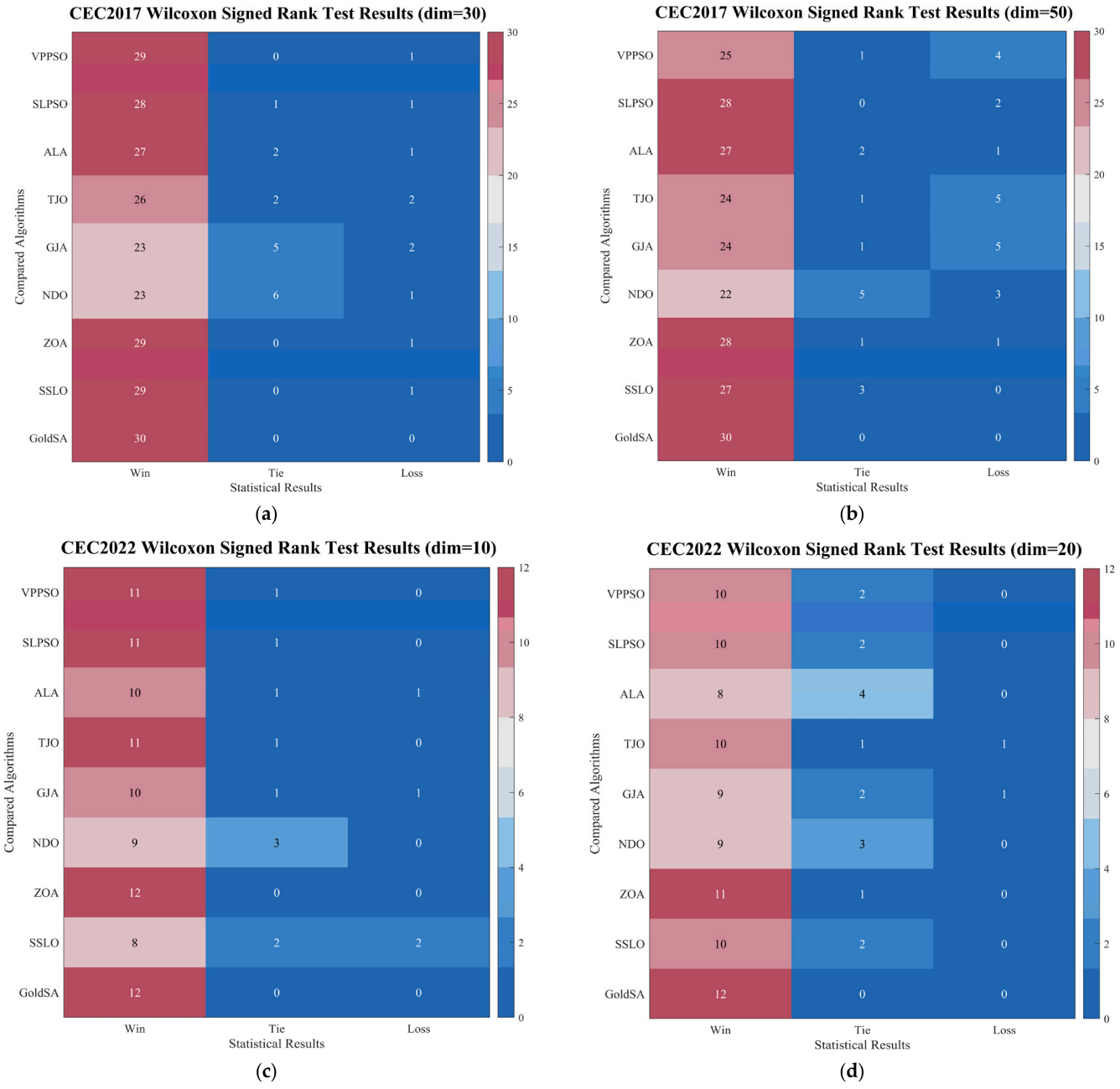


Figure 10. Wilcoxon Signed-Rank Test Results.

Figure 11 presents the Friedman Mean Rank Test results of different algorithms on the CEC2017 and CEC2022 benchmark functions. It can be observed that MIGoldSA achieves the best or highly competitive average ranking among all comparison algorithms, indicating that the proposed algorithm demonstrates superior overall optimization performance across different benchmark functions and dimensional settings. Compared with the other

algorithms, MIGoldSA maintains more stable ranking performance on both simple and complex optimization problems, which further verifies its strong robustness and adaptability in various search environments.

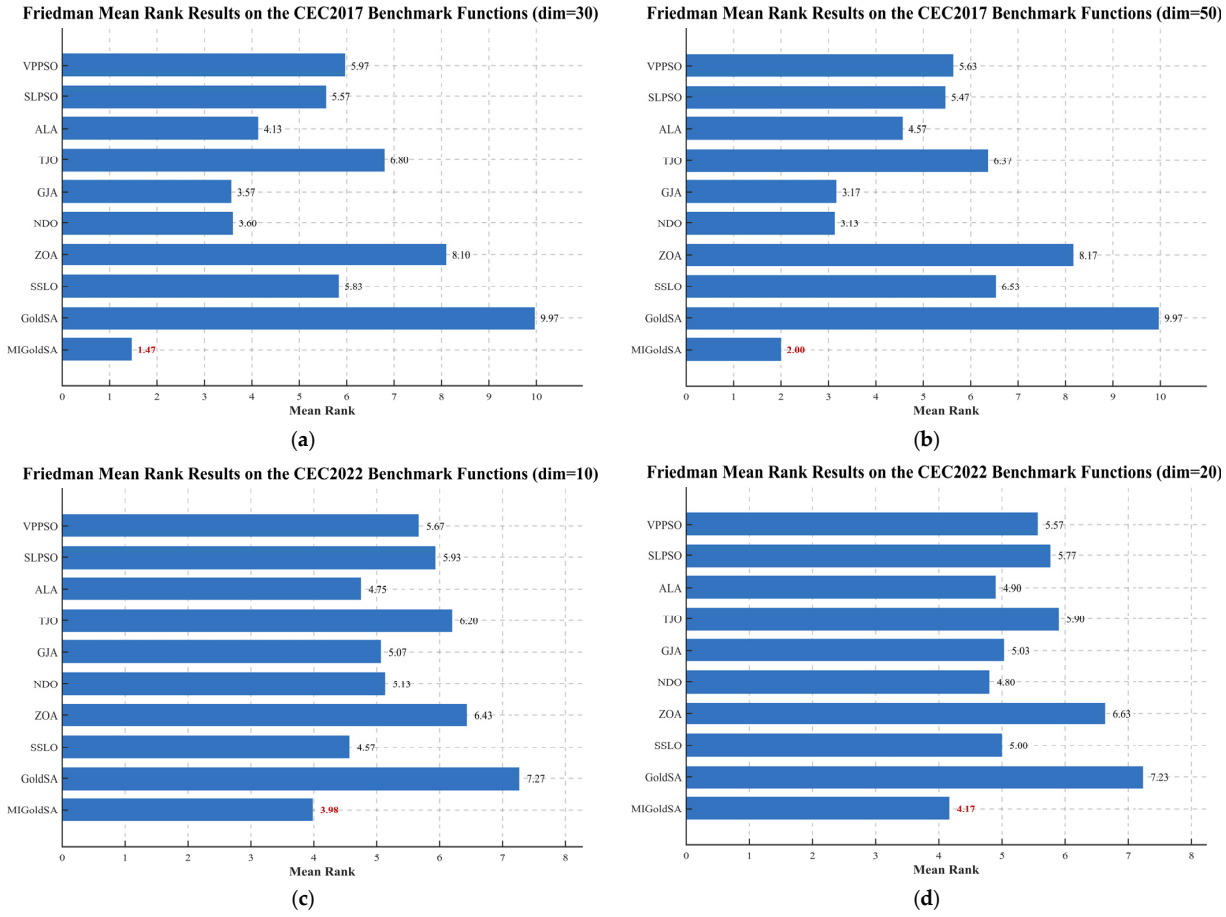


Figure 11. Friedman Mean Rank Test Results.

In addition, the Friedman ranking results demonstrate that MIGoldSA achieves a better balance between global exploration and local exploitation throughout the optimization process. Benefiting from the adaptive chaotic opposition initialization strategy, dynamic elite-guided sine evolution strategy, and Economic Investment Strategy, the proposed algorithm can effectively improve population diversity, accelerate convergence speed, and reduce the probability of premature convergence. Overall, the Friedman Mean Rank Test results further confirm the effectiveness, stability, and superiority of MIGoldSA in solving complex high-dimensional optimization problems.

5.6. Ablation Experiment Study

Ablation experiments are essential for evaluating the effectiveness and contribution of each improvement strategy in an enhanced metaheuristic algorithm. Although the overall performance of an improved algorithm can be verified through comparisons with other algorithms, it is still necessary to further investigate whether each proposed mechanism plays a positive role in improving optimization accuracy, convergence speed, population diversity, and robustness. By removing or independently combining different strategies, the contribution of each component to the final optimization performance can be clearly analyzed, thereby demonstrating that the performance improvement is not obtained by chance but results from the coordinated effect of the proposed mechanisms. Therefore, in this subsection, ablation experiments are conducted on the proposed MIGoldSA to

verify the effectiveness of its key improvement strategies and to further analyze their individual and combined impacts on algorithm performance. The experimental results are shown below.

Figure 12 presents the Friedman mean rank results of the ablation experiments on the CEC2017 and CEC2022 benchmark functions. It can be observed that all single-strategy variants achieve lower mean ranks than the original GoldSA, indicating that each proposed strategy has a positive contribution to the improvement of optimization performance. On the CEC2017 benchmark functions, the original GoldSA obtains the worst mean rank of 4.97 under both 30-dimensional and 50-dimensional settings. After introducing the adaptive chaotic opposition initialization strategy, dynamic elite-guided sine evolution strategy, and Economic Investment Strategy separately, the mean ranks are reduced to 2.43, 2.50, and 3.80 for Dim = 30, and 2.40, 2.33, and 3.93 for Dim = 50, respectively. These results show that ACOIS and DEGSES provide more evident improvements on CEC2017, while EIS also contributes to performance enhancement compared with the original GoldSA. For the CEC2022 benchmark functions, the mean rank of GoldSA is 5.00 in both Dim = 10 and Dim = 20 cases, whereas the single-strategy variants obtain better ranks. In particular, DEGSES achieves the best performance among the single-strategy variants, with mean ranks of 2.70 and 2.30 under Dim = 10 and Dim = 20, respectively, suggesting that the dynamic multi-elite guidance mechanism is especially helpful for improving the search behavior on more complex benchmark functions. More importantly, the complete MIGoldSA consistently achieves the best mean rank in all four experimental settings, with values of 1.30, 1.37, 1.50, and 1.10, respectively. Compared with the original GoldSA, the mean rank of MIGoldSA is reduced by approximately 73.84%, 72.43%, 70.00%, and 78.00%, respectively. These results demonstrate that the performance improvement of MIGoldSA is not only obtained from a single strategy, but also from the complementary cooperation among the three strategies. Therefore, the ablation experiment results verify the effectiveness of ACOIS, DEGSES, and EIS, and further confirm that their integration can enhance the overall optimization capability and robustness of the original GoldSA.

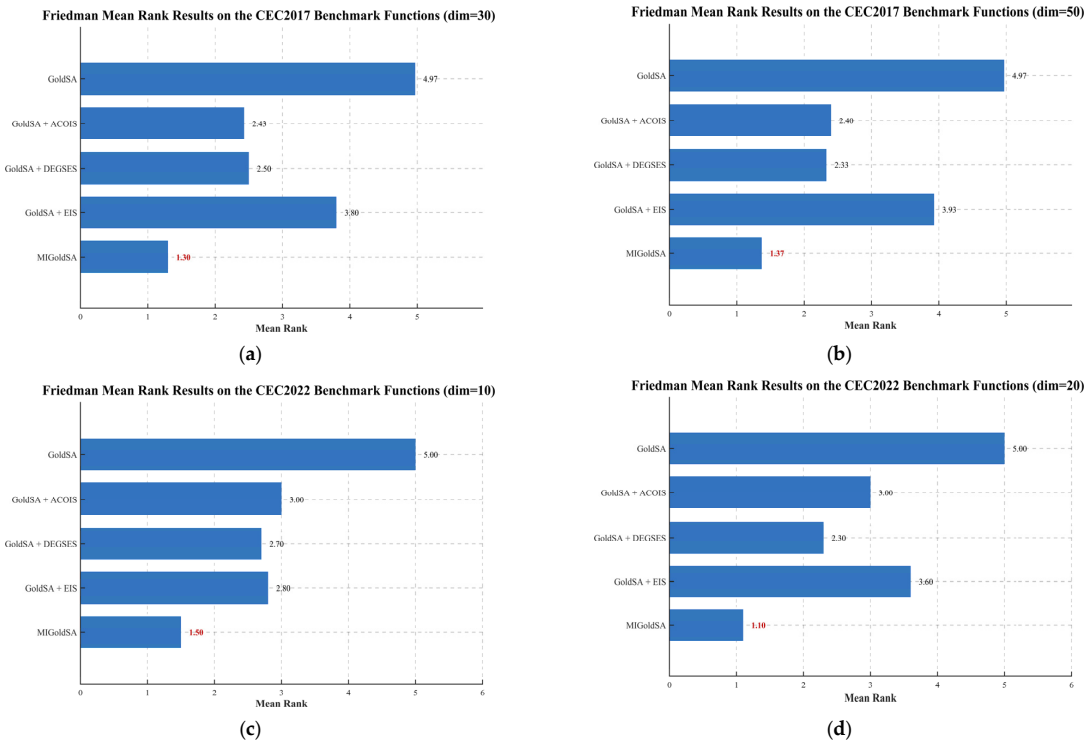


Figure 12. Analysis of ablation experiment results.

6. MIGoldSA for Cloud Task Scheduling

To further verify the effectiveness and practical applicability of the proposed MIGoldSA algorithm, this section applies MIGoldSA to the cloud computing task scheduling problem. Cloud task scheduling is a typical large-scale combinatorial optimization problem, whose objective is to reasonably allocate computational tasks to different virtual machines in order to optimize overall system performance. In practical cloud computing environments, task scheduling usually involves multiple conflicting optimization objectives, such as execution time, load balancing, and execution cost. Therefore, a multi-objective cloud task scheduling model is constructed in this section, and MIGoldSA is employed to solve the corresponding optimization problem [17,18,36,37]. The detailed modeling process is described as follows.

6.1. Cloud Computing Task Scheduling Model

Cloud computing task scheduling is one of the core technologies in cloud resource management systems. Its fundamental purpose is to efficiently allocate large-scale computational tasks to available virtual machine resources according to specific optimization objectives, thereby improving resource utilization, reducing execution overhead, and enhancing system service quality. In cloud computing environments, efficient task scheduling can effectively alleviate resource congestion, improve system throughput, and optimize the overall operational efficiency of cloud platforms.

Assume that the cloud computing environment contains M virtual machines and N computational tasks. The virtual machine set and task set can be respectively defined as:

$$VMs = \{M_1, M_2, \cdots, M_M\} \tag{18}$$

$$Tasks = \{T_1, T_2, \cdots, T_N\} \tag{19}$$

where VMs represents the set of virtual machine resource nodes; $Tasks$ denotes the set of cloud computing tasks; M and N denote the numbers of virtual machines and tasks, respectively.

In the cloud task scheduling problem, each task needs to be assigned to one virtual machine for execution. The scheduling relationship between tasks and virtual machines can be represented by the following allocation matrix:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1M} \\ a_{21} & a_{22} & \cdots & a_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NM} \end{bmatrix} \tag{20}$$

where $a_{ij} = \begin{cases} 1, & \text{if task } i \text{ is assigned to virtual machine } j \\ 0, & \text{otherwise} \end{cases}$. The resource attributes of each virtual machine mainly include computational capability, storage capacity, and communication bandwidth. Therefore, the resource characteristics of the j-th virtual machine can be expressed as:

$$VM_j = (E_{n,j}, S_{n,j}, C_{n,j}) \tag{21}$$

where $E_{n,j}$ denotes the computational processing capability of the j-th virtual machine; $S_{n,j}$ represents the storage resource capacity; $C_{n,j}$ denotes the communication bandwidth resource.

Similarly, the resource requirements of the i-th task can be represented as:

$$Task_i = (E_{t,i}, S_{t,i}, C_{t,i}) \tag{22}$$

where $E_{t,i}$ denotes the computational requirement of task i; $S_{t,i}$ represents the storage requirement; $C_{t,i}$ denotes the bandwidth requirement.

6.2. Objective Function Design

In practical cloud computing environments, cloud task scheduling generally involves multiple optimization objectives. To comprehensively evaluate scheduling performance, this paper considers three important optimization objectives, including execution time cost, load balancing cost, and execution price cost.

6.2.1. Time Cost

Execution time is one of the most important indicators in cloud computing task scheduling, which directly affects system response speed and user Quality of Service (QoS). Lower execution time indicates that tasks can be completed more efficiently, thereby improving overall system throughput and computational efficiency. Therefore, the time cost model can be defined as:

$$Time = \sum_{i=1}^{N} \sum_{j=1}^{M} a_{ij} \frac{E_{t,i}}{E_{n,j}} \tag{23}$$

where $E_{t,i}$ represents the computational demand of task i; $E_{n,j}$ denotes the computational capability of virtual machine j.

6.2.2. Load Balancing Cost

Load balancing is another critical factor in cloud task scheduling. Reasonable load distribution can effectively avoid excessive concentration of tasks on specific resource nodes, thereby preventing system congestion and performance degradation.

The load balancing cost can be expressed as:

$$Load = \sum_{i=1}^{N} \sum_{j=1}^{M} a_{ij} \frac{S_{t,i}}{S_{n,j}} \tag{24}$$

where $S_{t,i}$ denotes the storage requirement of task i; $S_{n,j}$ represents the storage capacity of virtual machine j.

A smaller load cost indicates better resource allocation balance and higher resource utilization efficiency.

6.2.3. Price Cost

In cloud computing systems, execution cost is also an important optimization objective. Lower scheduling cost can reduce operational expenses while improving economic efficiency and service competitiveness.

The price cost model can be formulated as:

$$Price = \sum_{i=1}^{N} \sum_{j=1}^{M} a_{ij} \frac{E_{t,i}}{E_{n,j}} \times \frac{C_{t,i}}{C_{n,j}} \times P \tag{25}$$

where $C_{t,i}$ denotes the bandwidth requirement of task i; $C_{n,j}$ represents the communication bandwidth of virtual machine j. P denotes the unit execution price coefficient.

6.3. Objective Function Normalization

Since the values of different objective functions vary significantly in magnitude, normalization processing is required before constructing the comprehensive fitness function. The normalized execution time objective can be expressed as:

$$Exetime = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} a_{ij}\frac{E_{t,i}/E_{n,j}}{\underset{i,j}{max}\{E_{t,i}/E_{n,j}\}} \tag{26}$$

The normalized load balancing objective can be represented as:

$$Vmload = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} a_{ij}\frac{S_{t,i}/S_{n,j}}{\underset{i,j}{max}\{S_{t,i}/S_{n,j}\}} \tag{27}$$

Similarly, the normalized execution cost objective is defined as:

$$Execost = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} a_{ij}\frac{(PE_{t,i}C_{t,i})/(E_{n,j}C_{n,j})}{\underset{i,j}{max}\{(PE_{t,i}C_{t,i})/(E_{n,j}C_{n,j})\}} \tag{28}$$

6.4. Comprehensive Fitness Function

To comprehensively optimize the cloud task scheduling problem, the above three objectives are integrated into a unified weighted fitness function:

$$F(i) = \omega_1 Exetime(i) + \omega_2 Vmload(i) + \omega_3 Execost(i) \tag{29}$$

where ω_1, ω_2, and ω_3 represent the weighting coefficients of execution time, load balancing, and execution cost, respectively; $\omega_1 + \omega_2 + \omega_3 = 1$.

In this paper, to ensure a balanced optimization among different objectives, the three weighting coefficients are uniformly set as:

$$\omega_1 = \omega_2 = \omega_3 = \frac{1}{3} \tag{30}$$

Therefore, minimizing the objective function $F(i)$ corresponds to obtaining the optimal cloud task scheduling scheme.

6.5. Analysis of Experimental Results

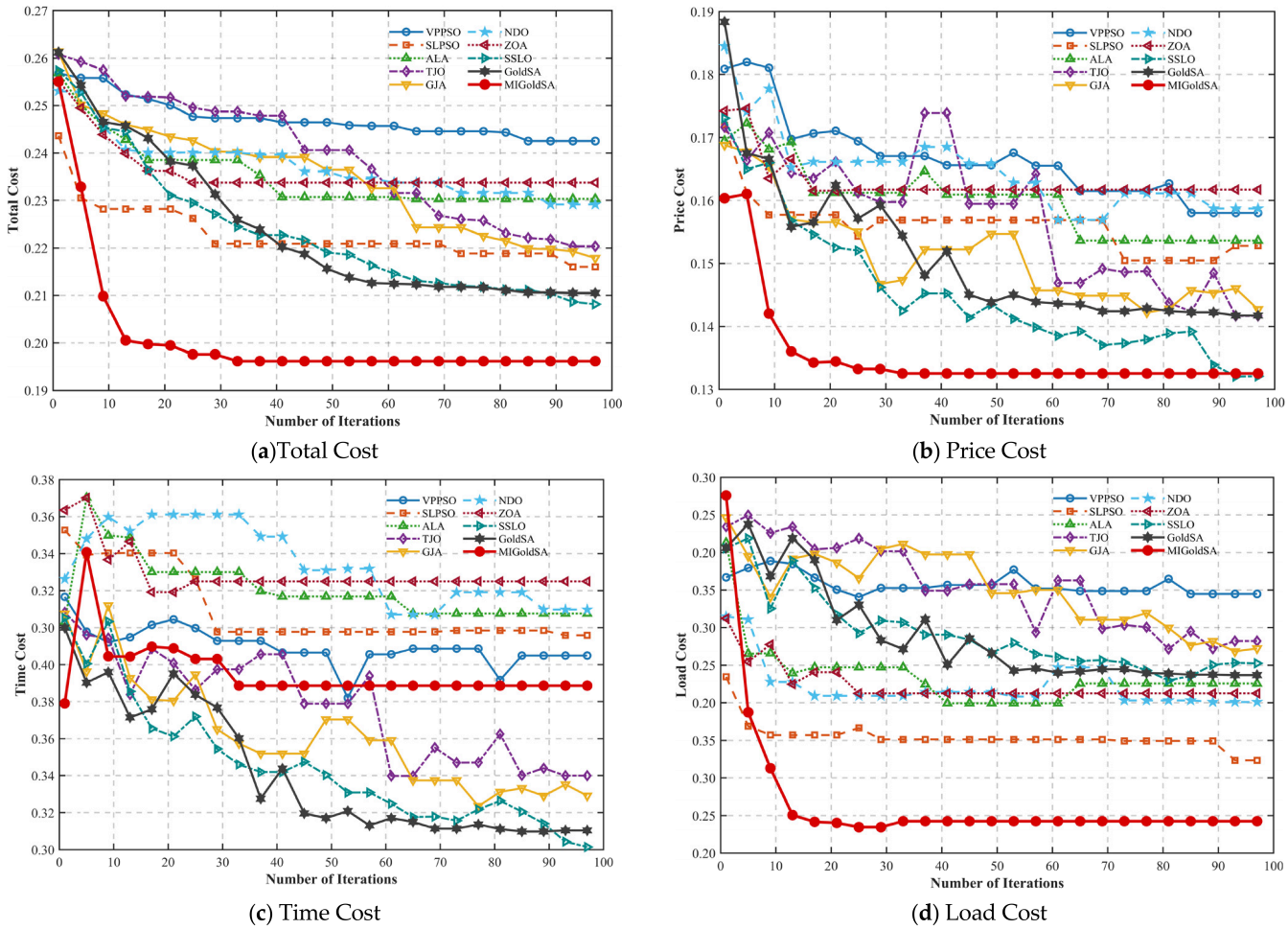
In this section, the performance of the proposed MIGoldSA algorithm is comprehensively evaluated through a series of cloud task scheduling simulation experiments and compared with nine state-of-the-art optimization algorithms. To objectively assess the scheduling effectiveness of different algorithms, three important evaluation metrics are considered, including time cost, load cost, and price cost. Specifically, the time cost reflects the overall task execution efficiency of the cloud system, the load cost measures the balance degree of resource allocation among virtual machines, and the price cost represents the economic expenditure during task processing. By integrating these three components, the comprehensive scheduling cost is finally obtained and used as the primary indicator for evaluating the overall optimization performance of the algorithms. The detailed experimental results and corresponding analyses are presented as follows:

6.5.1. Experimental Analysis on Small-Scale Cloud Task Scheduling

In this section, a comprehensive evaluation of the proposed algorithm is conducted on small-scale cloud task scheduling problems. Specifically, the convergence behavior

of different algorithms is analyzed, and the corresponding scheduling costs under dynamically varying task numbers are compared in detail. Through these experiments, the effectiveness and robustness of MIGoldSA in handling small-scale scheduling scenarios can be thoroughly verified. The detailed experimental results and analyses are presented as follows:

Convergence Behavior Analysis: In this section, the convergence behavior of different algorithms is comprehensively analyzed. To ensure the fairness and consistency of the experiments, the number of tasks was fixed at 100, while the maximum number of iterations was uniformly set to 100 for all comparison algorithms. Under the same experimental conditions, the convergence performance of the proposed MIGoldSA was compared with that of the other nine state-of-the-art optimization algorithms. By analyzing the convergence curves, the optimization efficiency, convergence speed, and search stability of different algorithms can be effectively evaluated. The corresponding experimental results are illustrated in Figure 13.



(**a**)Total Cost (**b**) Price Cost

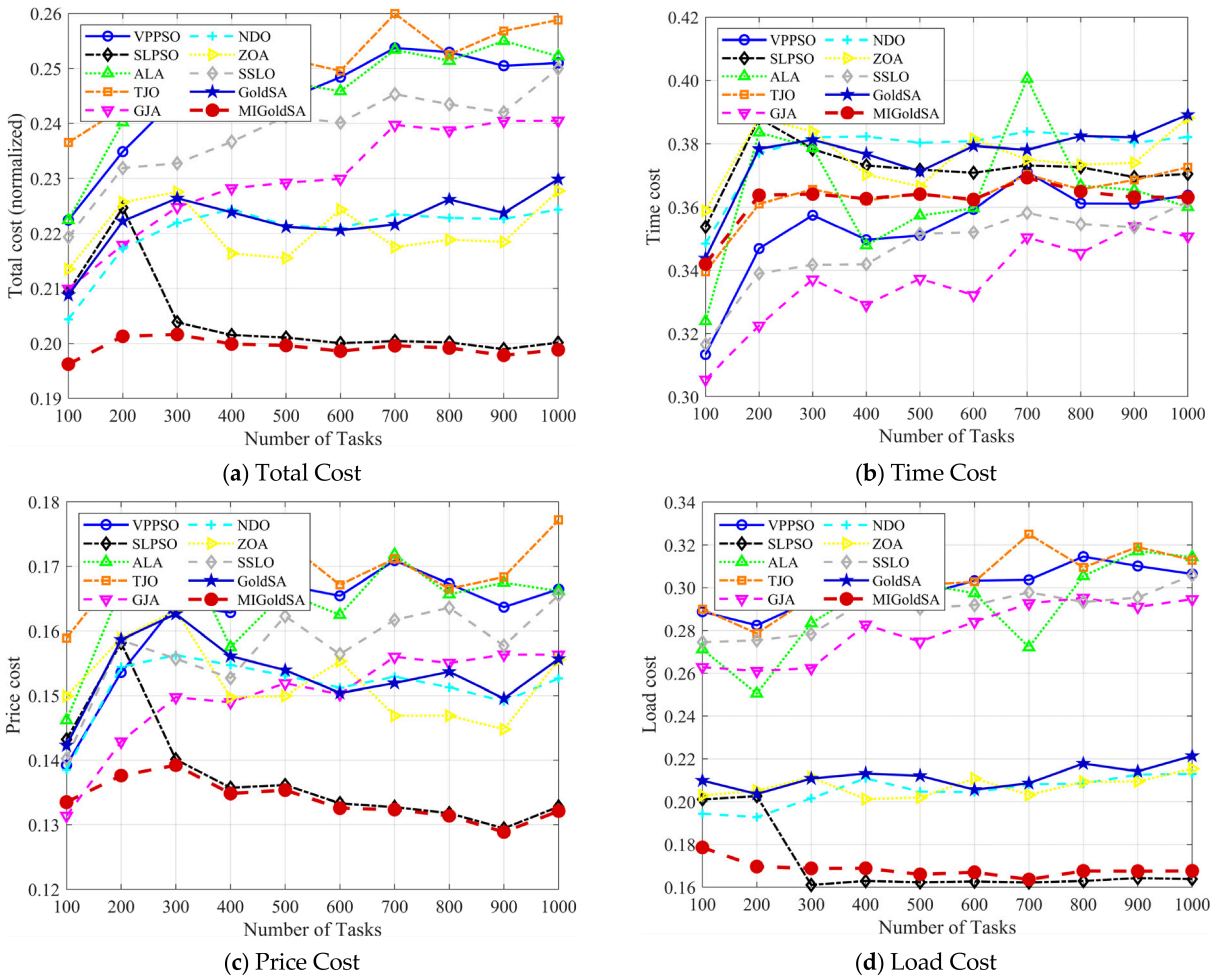
(**c**) Time Cost (**d**) Load Cost

Figure 13. Convergence Curves of Different Algorithms on Small-Scale Cloud Task Scheduling Problems.

Figure 13 illustrates the convergence curves of different algorithms on small-scale cloud task scheduling problems. It can be observed that MIGoldSA demonstrates faster convergence speed and lower scheduling cost compared with the other algorithms throughout the iterative process. In the early optimization stage, MIGoldSA can rapidly reduce the objective value, indicating that the proposed strategies effectively enhance the global exploration capability and improve the search efficiency of the algorithm. Meanwhile, during the later optimization stage, MIGoldSA continues to maintain stable convergence behavior

and achieves lower final scheduling costs, demonstrating strong local exploitation capability and high convergence accuracy. Furthermore, compared with several comparison algorithms that exhibit premature convergence or stagnation during iterations, MIGoldSA maintains better optimization stability and stronger search capability. This indicates that the adaptive chaotic opposition initialization strategy, dynamic elite-guided sine evolution strategy, and Economic Investment Strategy collaboratively improve population diversity and strengthen the balance between exploration and exploitation. Overall, the convergence results further verify that MIGoldSA possesses excellent optimization capability and robustness in solving small-scale cloud task scheduling problems.

Cost Comparison Under Varying Dynamic Task Loads: In this section, extensive experiments were conducted under dynamically varying task loads to further evaluate the scheduling performance of different algorithms. Specifically, the number of cloud computing tasks was gradually increased from 100 to 1000 in order to analyze the scalability and robustness of each algorithm under different scheduling scenarios. Figure 14 presents the variation trends of four important evaluation metrics for all comparison algorithms as the task load continuously increases. Through these experiments, the effectiveness and adaptability of MIGoldSA in handling dynamic cloud task scheduling problems can be comprehensively assessed.



(**a**) Total Cost (**b**) Time Cost

(**c**) Price Cost (**d**) Load Cost

Figure 14. Performance Comparison of Different Algorithms under Varying Small-Scale Task Loads.

Figure 14 presents the performance comparison results of different algorithms under varying small-scale task loads. It can be observed that as the number of cloud computing tasks gradually increases, the scheduling costs of all algorithms show an upward trend

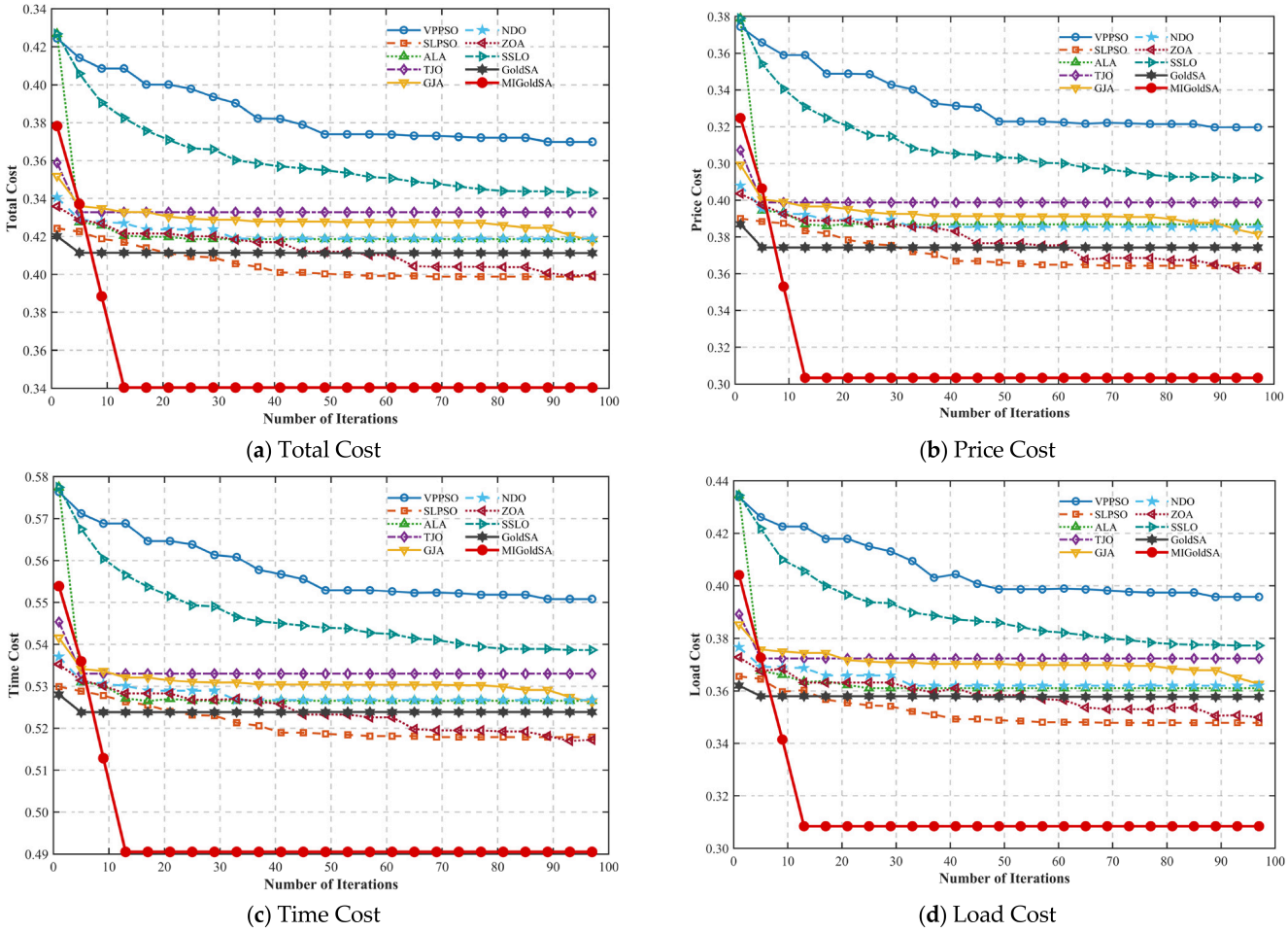
due to the increasing complexity of the scheduling problem. However, compared with the other algorithms, MIGoldSA consistently maintains lower comprehensive scheduling costs under different task loads, demonstrating superior optimization capability and stronger adaptability in dynamic cloud scheduling environments. Furthermore, MIGoldSA exhibits more stable performance trends in terms of time cost, load cost, and price cost as the task scale increases. Compared with several comparison algorithms that experience obvious performance degradation or larger fluctuations under heavier task loads, MIGoldSA can effectively maintain better scheduling efficiency and resource allocation balance. This indicates that the proposed adaptive chaotic opposition initialization strategy, dynamic elite-guided sine evolution strategy, and Economic Investment Strategy effectively improve the global search capability and convergence stability of the algorithm. Overall, the experimental results further verify that MIGoldSA possesses excellent robustness, scalability, and optimization performance in handling dynamic small-scale cloud task scheduling problems.

6.5.2. Experimental Analysis on Large-Scale Cloud Task Scheduling

In this section, we conduct a comprehensive evaluation of the proposed algorithm in the context of large-scale cloud task scheduling. Specifically, we analyze the convergence characteristics of various algorithms and provide a detailed comparison of their respective scheduling costs under scenarios involving dynamically changing task loads. Through these experiments, we aim to fully validate the effectiveness and robustness of MIGoldSA when handling scheduling scenarios of varying scales. The detailed experimental results and analysis are presented below:

Analysis of Convergence Characteristics: In this section, we perform a comprehensive analysis of the convergence characteristics of the various algorithms under consideration. To ensure experimental fairness and consistency, we fixed the number of tasks at 1000 and uniformly set the maximum number of iterations for all comparative algorithms to 100. Under these identical experimental conditions, we compared the convergence performance of the proposed MIGoldSA algorithm against nine other state-of-the-art optimization algorithms. By analyzing the convergence curves, we are able to effectively assess the optimization efficiency, convergence speed, and search stability of the different algorithms. The corresponding experimental results are illustrated in Figure 15.

Figure 15 illustrates the convergence curves of different algorithms on large-scale cloud task scheduling problems. It can be observed that as the scale of cloud tasks increases, the optimization difficulty of the scheduling problem becomes significantly higher. Nevertheless, MIGoldSA still maintains faster convergence speed and lower scheduling costs compared with the other algorithms throughout the iterative process. In the early optimization stage, MIGoldSA can effectively explore promising search regions and rapidly reduce the objective value, indicating that the proposed strategies significantly enhance the global exploration capability of the algorithm in large-scale search spaces. Moreover, during the later optimization stage, MIGoldSA continues to exhibit stable convergence behavior and achieves lower final scheduling costs, demonstrating strong local exploitation capability and good convergence precision. In contrast, several comparison algorithms show obvious stagnation phenomena or premature convergence under large-scale task scheduling scenarios, resulting in poorer optimization performance. This further demonstrates that the adaptive chaotic opposition initialization strategy, dynamic elite-guided sine evolution strategy, and Economic Investment Strategy effectively improve population diversity and strengthen the balance between exploration and exploitation. Overall, the convergence results verify that MIGoldSA possesses excellent scalability, robustness, and optimization capability in solving complex large-scale cloud task scheduling problems.



(**a**) Total Cost (**b**) Price Cost

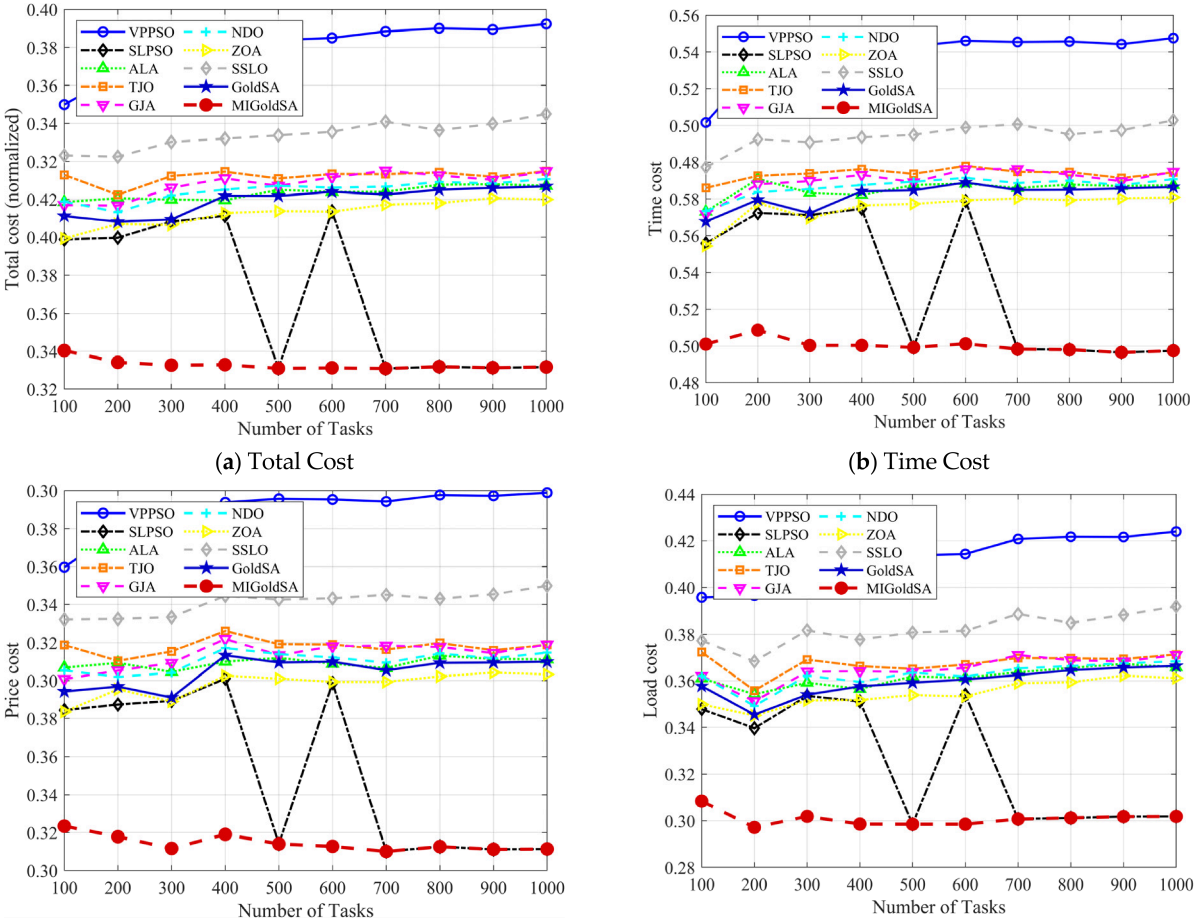
(**c**) Time Cost (**d**) Load Cost

Figure 15. Convergence Curves of Different Algorithms on large-Scale Cloud Task Scheduling Problems.

Cost Comparison under Varying Dynamic Task Loads: In this section, we conducted extensive experiments under dynamically changing task loads to further evaluate the scheduling performance of various algorithms. Specifically, we progressively increased the number of cloud computing tasks from 1000 to 10,000 to analyze the scalability and robustness of each algorithm across different scheduling scenarios. Figure 16 illustrates the trends observed across four key evaluation metrics for all compared algorithms as the task load continuously increased. Through these experiments, we are able to comprehensively assess the effectiveness and adaptability of MIGoldSA in addressing dynamic cloud task scheduling problems.

Figure 16 presents the performance comparison results of different algorithms under varying large-scale task loads. It can be observed that as the number of cloud computing tasks continuously increases, the scheduling complexity and computational burden of all algorithms become significantly higher. Consequently, the comprehensive scheduling costs of different algorithms gradually increase with the growth of task loads. However, compared with the other algorithms, MIGoldSA consistently maintains lower scheduling costs and exhibits superior optimization performance under different large-scale scheduling scenarios. Furthermore, MIGoldSA demonstrates more stable variation trends in terms of time cost, load cost, and price cost as the task scale increases. In contrast, several comparison algorithms experience obvious performance degradation, unstable fluctuations, or reduced scheduling efficiency under heavy task loads. Benefiting from the adaptive chaotic opposition initialization strategy, dynamic elite-guided sine evolution strategy, and Economic Investment Strategy, MIGoldSA can effectively improve population diversity,

strengthen global exploration capability, and maintain strong local exploitation performance in complex large-scale search spaces. Overall, the experimental results further verify that MIGoldSA possesses excellent scalability, robustness, and optimization capability in handling dynamic large-scale cloud task scheduling problems.



(**a**) Total Cost (**b**) Time Cost

(**c**) Price Cost (**d**) Load Cost

Figure 16. Performance Comparison of Different Algorithms under Varying large-Scale Task Loads.

7. Conclusions

In this paper, a Multi-strategy Improved Golden Sine Optimization Algorithm, named MIGoldSA, is proposed for global optimization and cloud task scheduling problems. The proposed method is developed as an enhanced version of the original GoldSA rather than a completely new optimization paradigm. Three targeted strategies are incorporated into the original GoldSA framework. The adaptive chaotic opposition initialization strategy improves the diversity and distribution quality of the initial population. The dynamic elite-guided sine evolution strategy replaces the single best-solution guidance with a weighted multi-elite guidance mechanism, thereby reducing excessive dependence on one individual. The Economic Investment Strategy adaptively allocates search intensity according to individual optimization potential, enabling the algorithm to coordinate exploitation-oriented and exploration-oriented updates more effectively.

To evaluate the optimization performance of MIGoldSA, comprehensive experiments were conducted on the IEEE CEC2017 and CEC2022 benchmark test suites. The experimental results show that MIGoldSA achieves the best or tied-best mean fitness values on 60 out of 84 benchmark cases, accounting for 71.43% of all test cases. This indicates that the proposed modifications can effectively improve the solution accuracy of the original GoldSA on most benchmark functions. In addition, the Wilcoxon signed-rank test further

confirms the statistical reliability of MIGoldSA. Among 756 pairwise comparisons with the competing algorithms, MIGoldSA obtains 662 wins, 57 ties, and 37 losses, corresponding to an overall win rate of 87.57% and a non-inferiority rate of 95.11%. The Friedman mean-rank results also show that MIGoldSA obtains the best overall ranks under the four benchmark settings, with mean ranks of 1.47, 2.00, 3.98, and 4.17. Compared with the original GoldSA, these ranks are reduced by 85.26%, 79.94%, 45.25%, and 42.32%, respectively. These quantitative statistical results verify the effectiveness, stability, and robustness of MIGoldSA in solving complex global optimization problems.

Furthermore, MIGoldSA was applied to cloud task scheduling problems under both small-scale and large-scale task scenarios. The scheduling experiments considered time cost, load cost, price cost, and comprehensive scheduling cost as evaluation metrics. The results show that MIGoldSA can obtain a better overall cost trade-off than most comparison algorithms. In small-scale scheduling scenarios, MIGoldSA maintains the normalized comprehensive scheduling cost at approximately 0.20 and reduces it by about 9–14% compared with the original GoldSA. In large-scale scheduling scenarios, MIGoldSA maintains the normalized comprehensive scheduling cost at approximately 0.33–0.34, corresponding to an approximate reduction of 18–21% compared with the original GoldSA. In terms of individual scheduling metrics, MIGoldSA reduces the normalized load cost by about 18–25% and the normalized price cost by about 10–18%. For time cost, MIGoldSA shows competitive performance in small-scale scenarios and achieves an approximate reduction of 8–12% in large-scale scenarios compared with the original GoldSA. These results indicate that MIGoldSA has good scalability and cost-efficiency in cloud task scheduling applications, especially when the task scale increases.

Although MIGoldSA achieves promising results, there are still several limitations that should be further studied. First, the current improvement strategies are mainly evaluated on continuous benchmark functions and simulated cloud task scheduling scenarios, and their performance in more complex real-world dynamic cloud environments requires further validation. Second, although the proposed strategies do not change the overall computational complexity order of GoldSA, they still introduce additional operations, such as opposition-based initialization, elite weight calculation, and investment coefficient updating. Therefore, future work will further investigate lightweight implementations of MIGoldSA and extend it to multi-objective scheduling, real-time cloud-edge collaborative scheduling, and other practical engineering optimization problems.

Author Contributions: Conceptualization, Z.Z. and H.Z.; methodology, Z.Z. and H.Z.; software, Z.Z. and H.Z.; validation, Z.Z. and H.Z.; formal analysis, Z.Z. and H.Z.; investigation, Z.Z. and H.Z.; resources, Z.Z. and H.Z.; data curation, Z.Z. and H.Z.; writing—original draft preparation, Z.Z. and H.Z.; writing—review and editing, Z.Z. and H.Z.; visualization, Z.Z. and H.Z.; supervision, Z.Z. and H.Z.; funding acquisition, Z.Z. and H.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

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

Appendix A. Global Convergence Analysis of MIGoldSA

Convergence is an important theoretical property for evaluating the reliability of a metaheuristic optimization algorithm. Although metaheuristic algorithms are usually stochastic and do not guarantee that the global optimum can be reached within a finite number of iterations, their global convergence behavior can be analyzed from a probabilistic perspective. In this subsection, the global convergence of MIGoldSA is analyzed based

on Markov stochastic process theory and the global convergence theorem of random search algorithms.

Assume that the feasible search space of the optimization problem is denoted as $\Omega \subseteq \mathbb{R}^D$, where D is the problem dimension. For a bounded optimization problem, $\Omega = [lb, ub]^D$. Let $X_i(t) \in \Omega$ represent the position of the i-th search agent at iteration t, where $i = 1, 2, \ldots, N$, and N is the population size. The population state of MIGoldSA at iteration t can be defined as:

$$S(t) = X_1(t), X_2(t), \ldots, X_N(t), X^*(t), \tag{A1}$$

where $X^*(t)$ denotes the best-so-far solution retained by the algorithm up to iteration t. The corresponding population state space is defined as:

$$\mathcal{S} = S(t) \mid X_i(t) \in \Omega, i = 1, 2, \ldots, N. \tag{A2}$$

Since the search space is bounded and the algorithm is implemented under finite numerical precision, the population state space $\mathcal{S}$ can be regarded as a finite or countable state space in practical computation.

In MIGoldSA, the next population state $S(t+1)$ is generated according to the current population state $S(t)$, the current fitness values, and several random variables used in the adaptive chaotic opposition initialization strategy, the dynamic elite-guided sine evolution strategy, and the Economic Investment Strategy. Specifically, the elite guidance center, adaptive weights, investment potential, and investment coefficients are all calculated from the current population and its fitness distribution. The random parameters used in the sine-based update and differential exploration are independently generated during the current iteration. Therefore, once $S(t)$ is given, the distribution of $S(t+1)$ is independent of all previous states $S(0), S(1), \ldots, S(t-1)$. This satisfies the Markov property, namely

$$P(S(t+1) \mid S(t), S(t-1), \ldots, S(0)) = P(S(t+1) \mid S(t)). \tag{A3}$$

Thus, the population evolution process of MIGoldSA can be regarded as a Markov stochastic process on the population state space $\mathcal{S}$.

For a minimization problem, let f^* denote the global optimal fitness value, and let X_g be a global optimal solution satisfying $f(X_g) = f^*$. The global optimal individual state set is defined as:

$$G = X \in \Omega \mid f(X) = f^*. \tag{A4}$$

Accordingly, the optimal population state set is defined as:

$$H = S \in \mathcal{S} \mid \exists X_i \in S, ; X_i \in G; \text{or}; X^* \in G. \tag{A5}$$

That is, if a population state contains at least one global optimal solution or its best-so-far solution is globally optimal, then this population state belongs to H.

Theorem A1. *The optimal population state set H of MIGoldSA is a closed set.*

Proof. In MIGoldSA, the best-so-far solution is updated according to the greedy preservation rule. For a minimization problem, the update rule can be expressed as

$$X^{(t+1)} = \begin{cases} X_i(t+1), \text{if } f(X_i(t+1)) < f(X(t)), \\ X^*(t), \text{ otherwise.} \end{cases} \tag{A6}$$

Therefore, the best fitness value satisfies:

$$f\left(X^{(t+1)}\right) \leq f\left(X^{(t)}\right). \tag{A7}$$

This indicates that the best-so-far solution will not be replaced by an inferior solution during the iterative process. If $S(t) \in H$, then the population already contains a global optimal solution or the best-so-far solution $X^{(t)}$ is globally optimal. Since $X^{(t)}$ is retained in the subsequent iteration, $X^*(t+1)$ will still be globally optimal. Therefore,

$$P(S(t+1) \notin H \mid S(t) \in H) = 0. \tag{A8}$$

Thus, once the population state enters H, it cannot leave H. Hence, H is a closed set. □

Theorem A2. *MIGoldSA has global convergence in probability.*

Proof. Let H_ε denote the ε-optimal population state set, which is defined as:

$$H_\varepsilon = S \in \mathcal{S} \mid \exists X_i \in S; f(X_i) \leq f^* \varepsilon; \text{or } f(\mathrm{X}) \leq f^* + \varepsilon \tag{A9}$$

where $\varepsilon > 0$ is an arbitrarily small positive constant. Similar to Theorem 1, due to the best-so-far preservation rule, H_ε is also a closed set. Once the population reaches an ε-optimal solution, the best-so-far solution will preserve this solution or a better one in all subsequent iterations.

During the search process of MIGoldSA, the adaptive chaotic opposition initialization strategy increases the probability that the initial population covers different regions of the feasible search space. In the iterative stage, the dynamic elite-guided sine evolution strategy generates candidate solutions through stochastic sine-based movements around the weighted elite guidance center. Meanwhile, the Economic Investment Strategy introduces fitness-dependent exploitation and differential exploration components. For individuals with lower investment potential, the differential term constructed from randomly selected individuals provides additional exploration ability. Since the random variables used in these update mechanisms are generated within continuous intervals, for any measurable subset $B \subseteq \Omega$ with positive measure, there exists a nonzero probability that at least one candidate solution generated by MIGoldSA enters B, provided that the search process has not yet reached H_ε.

Let p_t denote the probability that the population enters H_ε at iteration t. Under the above stochastic search condition, $p_t > 0$. Therefore, the probability that MIGoldSA does not enter H_ε after T iterations can be written as:

$$P(S(T) \notin H_\varepsilon) \leq \prod_{t=0}^{T-1} (1 - p_t) \tag{A10}$$

If the stochastic search process satisfies the standard global convergence condition:

$$\sum_{t=0}^{\infty} p_t = \infty \tag{A11}$$

then

$$\lim_{T \to \infty} \prod_{t=0}^{T-1} (1 - p_t) = 0. \tag{A12}$$

Thus,

$$\lim_{T \to \infty} P(S(T) \in H_\varepsilon) = 1. \tag{A13}$$

This means that, as the number of iterations tends to infinity, the population state of MIGoldSA converges to the ε-optimal population state set with probability 1. Since ε can be arbitrarily small, the best-so-far solution $X^*(t)$ converges to the global optimal solution in probability, namely

$$\lim_{t\to\infty} P(|f(X(t)-f|\leq\varepsilon)=1. \tag{A14}$$

Therefore, MIGoldSA is globally convergent in probability.

The above analysis shows that MIGoldSA satisfies the basic conditions of stochastic global convergence. On the one hand, the stochastic search mechanisms introduced by chaotic opposition initialization, sine-based elite guidance, and investment-guided differential exploration provide a nonzero probability of reaching promising regions in the feasible search space. On the other hand, the best-so-far preservation mechanism ensures that once an optimal or near-optimal solution is obtained, it will not be lost in subsequent iterations. Therefore, with sufficient iterations, MIGoldSA can converge to the global optimal solution or an arbitrarily close neighborhood of the global optimum with probability 1. □

References

1. Sandhu, R.; Faiz, M.; Kaur, H.; Srivastava, A.; Narayan, V. Enhancement in Performance of Cloud Computing Task Scheduling Using Optimization Strategies. *Clust. Comput.* **2024**, *27*, 6265–6288. [CrossRef]
2. Tang, Q.; Xie, R.; Yu, F.R.; Chen, T.; Zhang, R.; Huang, T.; Liu, Y. Distributed Task Scheduling in Serverless Edge Computing Networks for the Internet of Things: A Learning Approach. *IEEE Internet Things J.* **2022**, *9*, 19634–19648. [CrossRef]
3. Alla, V.R.S.P.; Medikondu, N.R.; Parige, L.S.; Satyanarayana, K.; Kankhva, V.S.; Dhaliwal, N.; Saxena, A.K. Optimizing Task Scheduling in Cloud Computing: A Hybrid Artificial Intelligence Approach. *Cogent Eng.* **2024**, *11*, 2328355. [CrossRef]
4. Liu, S.; Yu, Y.; Lian, X.; Feng, Y.; She, C.; Yeoh, P.L.; Guo, L.; Vucetic, B.; Li, Y. Dependent Task Scheduling and Offloading for Minimizing Deadline Violation Ratio in Mobile Edge Computing Networks. *IEEE J. Sel. Areas Commun.* **2023**, *41*, 538–554. [CrossRef]
5. Pal, S.; Jhanjhi, N.; Abdulbaqi, A.S.; Akila, D.; Alsubaei, F.S.; Almazroi, A.A. An Intelligent Task Scheduling Model for Hybrid Internet of Things and Cloud Environment for Big Data Applications. *Sustainability* **2023**, *15*, 5104. [CrossRef]
6. Prity, F.S.; Gazi, M.H.; Uddin, K.A. A Review of Task Scheduling in Cloud Computing Based on Nature-Inspired Optimization Algorithm. *Clust. Comput.* **2023**, *26*, 3037–3067. [CrossRef]
7. Devi, N.; Dalal, S.; Solanki, K.; Dalal, S.; Lilhore, U.K.; Simaiya, S.; Nuristani, N. A Systematic Literature Review for Load Balancing and Task Scheduling Techniques in Cloud Computing. *Artif. Intell. Rev.* **2024**, *57*, 276. [CrossRef]
8. Abraham, O.L.; Ngadi, M.A.B.; Sharif, J.B.M.; Sidik, M.K.M. Task Scheduling in Cloud Environment–Techniques, Applications, and Tools: A Systematic Literature Review. *IEEE Access* **2024**, *12*, 138252–138279. [CrossRef]
9. Sharma, P.; Raju, S. Metaheuristic Optimization Algorithms: A Comprehensive Overview and Classification of Benchmark Test Functions: P. Sharma, S. Raju. *Soft Comput.* **2024**, *28*, 3123–3186. [CrossRef]
10. Benaissa, B.; Kobayashi, M.; Al Ali, M.; Khatir, T.; Elmeliani, M.E.A.E. Metaheuristic Optimization Algorithms: An Overview. *HCMCOU J. Sci.-Adv. Comput. Struct.* **2024**, *14*, 33–61. [CrossRef]
11. Tanyildizi, E.; Demir, G. Golden Sine Algorithm: A Novel Math-Inspired Algorithm. *Adv. Electr. Comput. Eng.* **2017**, *17*, 71–78. [CrossRef]
12. Najafizadeh, A.; Salajegheh, A.; Rahmani, A.M.; Sahafi, A. Multi-Objective Task Scheduling in Cloud-Fog Computing Using Goal Programming Approach. *Clust. Comput.* **2022**, *25*, 141–165. [CrossRef]
13. Banerjee, P.; Roy, S.; Sinha, A.; Hassan, M.M.; Burje, S.; Agrawal, A.; Bairagi, A.K.; Alshathri, S.; El-Shafai, W. MTD-DHJS: Makespan-Optimized Task Scheduling Algorithm for Cloud Computing with Dynamic Computational Time Prediction. *IEEE Access* **2023**, *11*, 105578–105618. [CrossRef]
14. Barut, C.; Yildirim, G.; Tatar, Y. An Intelligent and Interpretable Rule-Based Metaheuristic Approach to Task Scheduling in Cloud Systems. *Knowl.-Based Syst.* **2024**, *284*, 111241. [CrossRef]
15. Saif, F.A.; Latip, R.; Hanapi, Z.M.; Shafinah, K. Multi-Objective Grey Wolf Optimizer Algorithm for Task Scheduling in Cloud-Fog Computing. *IEEE Access* **2023**, *11*, 20635–20646. [CrossRef]
16. Sharma, N.; Sonal; Garg, P. Ant Colony Based Optimization Model for QoS-Based Task Scheduling in Cloud Computing Environment. *Meas. Sens.* **2022**, *24*, 100531. [CrossRef]
17. Cui, Z.; Zhao, T.; Wu, L.; Qin, A.K.; Li, J. Multi-Objective Cloud Task Scheduling Optimization Based on Evolutionary Multi-Factor Algorithm. *IEEE Trans. Cloud Comput.* **2023**, *11*, 3685–3699. [CrossRef]

18. Qin, X.; Li, S.; Tong, J.; Xie, C.; Zhang, X.; Wu, F.; Xie, Q.; Ling, Y.; Lin, G. ERTH Scheduler: Enhanced Red-Tailed Hawk Algorithm for Multi-Cost Optimization in Cloud Task Scheduling. *Artif. Intell. Rev.* **2024**, *57*, 328. [CrossRef]
19. Liu, C.; Wu, L.; Xiao, W.; Li, G.; Xu, D.; Guo, J.; Li, W. An Improved Heuristic Mechanism Ant Colony Optimization Algorithm for Solving Path Planning. *Knowl.-Based Syst.* **2023**, *271*, 110540. [CrossRef]
20. Shen, Y.; Zhang, C.; Soleimanian Gharehchopogh, F.; Mirjalili, S. An Improved Whale Optimization Algorithm Based on Multi-Population Evolution for Global Optimization and Engineering Design Problems. *Expert Syst. Appl.* **2023**, *215*, 119269. [CrossRef]
21. Liu, Q.; Li, N.; Jia, H.; Qi, Q.; Abualigah, L.; Liu, Y. A Hybrid Arithmetic Optimization and Golden Sine Algorithm for Solving Industrial Engineering Design Problems. *Mathematics* **2022**, *10*, 1567. [CrossRef]
22. Li, M.; Liu, Z.; Song, H. An Improved Algorithm Optimization Algorithm Based on RungeKutta and Golden Sine Strategy. *Expert Syst. Appl.* **2024**, *247*, 123262. [CrossRef]
23. Ding, S.; Shi, F.; Erkan, U.; Toktas, A.; Li, Q.; Wang, C.; Gao, S.; Mou, J. Design of a Three-Dimensional Logistic Map and Its Application to Seafood Image Encryption. *J. Supercomput.* **2026**, *82*, 225.
24. Wu, G.; Mallipeddi, R.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for the CEC 2017 Competition on Constrained Real-Parameter Optimization*; National University of Defense Technology: Changsha, China; Kyungpook National University: Daegu, Republic of Korea; Nanyang Technological University: Singapore, 2017; Volume 9, p. 2017.
25. Ahrari, A.; Elsayed, S.; Sarker, R.; Essam, D.; Coello, C.A.C. Problem Definition and Evaluation Criteria for the CEC'2022 Competition on Dynamic Multimodal Optimization. In Proceedings of the IEEE World Congress on Computational Intelligence (IEEE WCCI 2022), Padua, Italy, 18–23 July 2022; pp. 18–23.
26. Shami, T.M.; Mirjalili, S.; Al-Eryani, Y.; Daoudi, K.; Izadi, S.; Abualigah, L. Velocity Pausing Particle Swarm Optimization: A Novel Variant for Global Optimization. *Neural Comput. Appl.* **2023**, *35*, 9193–9223. [CrossRef]
27. Li, C.; Yang, S.; Nguyen, T.T. A Self-Learning Particle Swarm Optimizer for Global Optimization Problems. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **2011**, *42*, 627–646. [CrossRef]
28. Xiao, Y.; Cui, H.; Khurma, R.A.; Castillo, P.A. Artificial Lemming Algorithm: A Novel Bionic Meta-Heuristic Technique for Solving Real-World Engineering Optimization Problems. *Artif. Intell. Rev.* **2025**, *58*, 84.
29. Wang, J.; Shang, Z. Traffic Jam Optimizer: A Novel Swarm-Based Metaheuristic Algorithm for Solving Global Optimization Problems. *Appl. Math. Model.* **2025**, *150*, 116410. [CrossRef]
30. Zhang, K.; Zhao, H.; Li, X.; Fu, C.; Jin, J. Gekko Japonicus Algorithm: A Novel Nature-Inspired Algorithm for Engineering Problems and Path Planning. *J. Bionic Eng.* **2026**, *23*, 431–471. [CrossRef]
31. Xiao, W.; Ouyang, K.; Lian, J.J.; Gu, S.; Liu, Y.; Wei, D.; Sha, X.; Wang, J.; Fu, S.; Qiu, M. Newton Downhill Optimizer for Global Optimization. *Res. Sq.* **2025**. [CrossRef] [PubMed]
32. Trojovská, E.; Dehghani, M.; Trojovský, P. Zebra Optimization Algorithm: A New Bio-Inspired Optimization Algorithm for Solving Optimization Algorithm. *IEEE Access* **2022**, *10*, 49445–49473. [CrossRef]
33. Ye, J.; Sunat, K.; Chiewchanwattana, S. Stochastic Social Learning Optimization: Combining Social Learning and Bucket Theory for Efficient Optimization. *Knowl.-Based Syst.* **2026**, *341*, 115767. [CrossRef]
34. Woolson, R.F. Wilcoxon signed-rank test. In *Wiley Encyclopedia of Clinical Trials*; Wiley: Hoboken, NJ, USA, 2007; pp. 1–3.
35. Zimmerman, D.W.; Zumbo, B.D. Relative Power of the Wilcoxon Test, the Friedman Test, and Repeated-Measures ANOVA on Ranks. *J. Exp. Educ.* **1993**, *62*, 75–86. [CrossRef]
36. Zhu, M.; Li, J.; Yang, X. A Hybrid SAO and RIME Optimizer for Global Optimization and Cloud Task Scheduling. *Biomimetics* **2025**, *10*, 690. [CrossRef] [PubMed]
37. Xu, S.; Zhang, W. An Improved DOA for Global Optimization and Cloud Task Scheduling. *Symmetry* **2025**, *17*, 1670. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.