symmetry

MDPI

Article

Multi-Threshold Art Symmetry Image Segmentation and Numerical Optimization Based on the Modified Golden Jackal Optimization

Xiaoyan Zhang [1], Zuowen Bao [2], Xinying Li [3,*] and Jianfeng Wang [1]

[1] College of Design, Hanyang University, Ansan 15588, Gyeonggi-do, Republic of Korea; zxy20241@hanyang.ac.kr (X.Z.); wjf123@hanyang.ac.kr (J.W.)
[2] College of Art, Sungkyunkwan University, Seoul 03063, Republic of Korea; baozuowen@g.skku.edu
[3] Spatial Design Department, Graduate School of Industrial Art, Hongik University, Wausan-ro, Mapo-gu, Seoul 04066, Republic of Korea
* Correspondence: lixinying0599@163.com

Abstract

To address the issues of uneven population initialization, insufficient individual information interaction, and passive boundary handling in the standard Golden Jackal Optimization (GJO) algorithm, while improving the accuracy and efficiency of multilevel thresholding in artistic image segmentation, this paper proposes an improved Golden Jackal Optimization algorithm (MGJO) and applies it to this task. MGJO introduces a high-quality point set for population initialization, ensuring a more uniform distribution of initial individuals in the search space and better adaptation to the complex grayscale characteristics of artistic images. A dual crossover strategy, integrating horizontal and vertical information exchange, is designed to enhance individual information sharing and fine-grained dimensional search, catering to the segmentation needs of artistic image textures and color layers. Furthermore, a global-optimum-based boundary handling mechanism is constructed to prevent information loss when boundaries are exceeded, thereby preserving the boundary details of artistic images. The performance of MGJO was evaluated on the CEC2017 (dim = 30, 100) and CEC2022 (dim = 10, 20) benchmark suites against seven algorithms, including GWO and IWOA. Population diversity analysis, exploration–exploitation balance assessment, Wilcoxon rank-sum tests, and Friedman mean-rank tests all demonstrate that MGJO significantly outperforms the comparison algorithms in optimization accuracy, stability, and statistical reliability. In multilevel thresholding for artistic image segmentation, using Otsu's between-class variance as the objective function, MGJO achieves higher fitness values (approaching Otsu's optimal values) across various artistic images with complex textures and colors, as well as benchmark images such as Baboon, Camera, and Lena, in 4-, 6-, 8-, and 10-level thresholding tasks. The resulting segmented images exhibit superior peak signal-to-noise ratio (PSNR), structural similarity (SSIM), and feature similarity (FSIM) compared to other algorithms, more precisely preserving brushstroke details and color layers. Friedman average rankings consistently place MGJO in the lead. These experimental results indicate that MGJO effectively overcomes the performance limitations of the standard GJO, demonstrating excellent performance in both numerical optimization and multilevel thresholding artistic image segmentation. It provides an efficient solution for high-dimensional complex optimization problems and practical demands in artistic image processing.

check for updates

Academic Editor: Shuang Xu

Received: 29 October 2025
Revised: 16 November 2025
Accepted: 5 December 2025
Published: 11 December 2025

Citation: Zhang, X.; Bao, Z.; Li, X.; Wang, J. Multi-Threshold Art Symmetry Image Segmentation and Numerical Optimization Based on the Modified Golden Jackal Optimization. *Symmetry* **2025**, *17*, 2130. https://doi.org/10.3390/sym17122130

Copyright: © 2025 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 (https://creativecommons.org/licenses/by/4.0/).

Keywords: golden jackal optimization; double-crossover strategy; multi-level threshold image segmentation; exploration–exploitation; global optimization

1. Introduction

In the field of computer vision and image processing, image segmentation serves as a crucial bridge between image acquisition and high-level semantic analysis. Its primary objective is to partition an image into non-overlapping regions with clear semantic meaning, thereby providing a foundation for subsequent tasks such as object detection, feature extraction, and image understanding [1,2]. With the growing complexity of image data—such as high-resolution images, medical imaging, and remote sensing imagery—traditional single-threshold segmentation methods are increasingly insufficient for meeting the demands of fine-grained segmentation. In this context, multilevel thresholding has emerged as a research hotspot due to its ability to more accurately capture the characteristics of image gray-level distributions [3,4]. The core of multilevel thresholding lies in identifying the optimal combination of thresholds that maximizes inter-class differences or minimizes intra-class variance. Commonly employed objective functions include Otsu's between-class variance method and Kapur's entropy method. However, as the number of thresholds increases, the search space for optimal thresholds grows exponentially. Consequently, traditional approaches such as exhaustive search and greedy algorithms suffer from high computational complexity and are prone to local optima, making them inefficient for solving high-dimensional threshold optimization problems [5–7].

The task of multilevel thresholding is an NP-hard combinatorial problem, where the required computational effort surges dramatically with a higher count of thresholds. While traditional solvers tend to be slow and produce mediocre results [8,9], modern metaheuristics draw inspiration from the self-organizing principles of animal groups. This bio-inspired class of algorithms efficiently locates near-optimal solutions to complex thresholding issues within manageable computation periods.

With the advancement of research, a wide variety of swarm intelligence algorithms have emerged. Examples include Particle Swarm Optimization (PSO), inspired by the social behavior of bird flocks and fish schools, where particles adjust their direction and velocity based on personal and neighborhood experiences to search for optimal solutions [10]; the Gray Wolf Optimizer (GWO), modeled on the social hierarchy and hunting strategies of gray wolves [11]; the Crested Porcupine Optimizer (CPO), inspired by the diverse defensive behaviors of crested porcupines [12]; Ant Colony Optimization (ACO), founded upon the cooperative foraging of ant populations [13]; the Bat Algorithm (BA), which abstracts the echolocation principles of bats [14]; and the Animated Oat Optimization algorithm (AOO), a simulation of the dynamic movements exhibited by wild oats [15]; the Snake Optimizer (SO), designed according to the foraging and reproductive behaviors of snakes [16]; the Sparrow Search Algorithm (SSA), modeled after sparrows' foraging and anti-predation behaviors [17]; the Grasshopper Optimization Algorithm (GOA), inspired by grasshopper swarming behaviors [18]; the Remora Optimization Algorithm (ROA), derived from the symbiotic feeding behaviors of remoras attached to different host species [19]; the Black Widow Optimization algorithm (BWO), based on the distinctive reproductive strategies of black widow spiders [20]; other sophisticated methods draw from animal kingdom behaviors: the Dung Beetle Optimizer (DBO) captures the essence of the insect's resource management, from locomotion to reproduction [21]; the Golden Eagle Optimizer (GEO) emulates the raptor's dynamic control of attack speed during its distinctive spiral descent [22]; and the Salp Swarm Algorithm (SSA) replicates the coordinated movement and

group formation of salps in their marine habitat [23]; Other notable examples include the Chimpanzee Optimization Algorithm (ChOA), which simulates chimpanzees' cooperative hunting strategies—attacking, chasing, blocking, and driving prey [24]; and Artificial Rabbits Optimization (ARO), based on rabbits' survival strategies such as detour foraging and random hiding [25].

Moreover, many researchers have applied swarm intelligence algorithms to image segmentation. For example, Mohamed employed the Whale Optimization Algorithm (WOA) and Moth-Flame Optimization (MFO) to determine the optimal multilevel thresholds for image segmentation [26]. Mookiah et al. proposed an Enhanced Sine Cosine Algorithm (ESCA) to identify thresholds for segmenting color images [27]. Aranguren et al. introduced a multilevel thresholding method based on LSHADE for the segmentation of magnetic resonance brain images, and validated the approach using three sets of reference images [28]. Hairu Guo et al. [8] developed an Improved Aquila Optimizer (IAO) for multilevel thresholding in image segmentation, and experimental results on different benchmark images demonstrated that IAO achieved strong segmentation performance in most cases. Krishna Gopal Dhal et al. [29] employed nature-inspired optimization algorithms (NIOA) as an alternative for solving multilevel thresholding problems, and provided a comprehensive review of the most significant NIOAs applied in multilevel threshold-based image segmentation. Ziqi Jiang et al. [30] proposed a novel Ensemble Multi-Teaching-Learning-Based Optimization (EMTLBO), which integrates three different variants of the Teaching-Learning-Based Optimization (TLBO) algorithm—the original TLBO, its neighborhood search variant, and its differential variant—and applied it to image segmentation. Mohamed Abd Elaziz et al. [31] introduced an alternative multilevel thresholding method for polyp image segmentation (MPOA). Their approach enhances the Planetary Optimization Algorithm by incorporating operators from the Reptile Search Algorithm, thereby improving segmentation accuracy.

Despite the effectiveness of the aforementioned swarm intelligence-based multilevel thresholding methods in certain scenarios, they still share several common drawbacks. Most approaches rely heavily on random initialization strategies during the initial population generation stage, which often leads to uneven population distribution within the search space. As a result, the potential optimal threshold regions may not be adequately covered, causing insufficient global exploration and subsequently reducing optimization efficiency [32]. In addition, the boundary-handling mechanisms adopted by these methods are typically simple strategies, such as resetting solutions to the boundary or applying random adjustments. Such passive treatments not only interrupt the natural search trajectories of individuals and discard valuable directional information but may also overlook optimal thresholds located near the boundaries, leading to wasted computational resources. Furthermore, these methods generally lack adaptive adjustment mechanisms tailored to varying gray-level distribution characteristics of images. This limitation makes it difficult to achieve a balance among segmentation accuracy, stability, and computational efficiency, thereby restricting their generalizability and robustness when applied to complex real-world scenarios [33].

Introduced by Nitish Chopra and Mohsin Ansari in 2022, the Golden Jackal Optimization (GJO) algorithm [34], is a modern metaheuristic technique grounded in swarm intelligence. It emulates the cooperative hunting strategies of mated golden jackal pairs. The method's core mechanics involve a two-stage process: an exploration stage, where the algorithm conducts a global search for promising regions, and an exploitation stage, where it focuses on locally encircling and refining solutions [35]. Compared with traditional algorithms, GJO demonstrates advantages in terms of convergence speed and optimization accuracy. However, it still exhibits several noticeable shortcomings in practical applica-

tions [36]. First, its reliance on purely random initialization leads to a scattered distribution of individuals in the search space, limiting the coverage of potential optimal regions and reducing global exploration efficiency. Second, the individual update mechanism is solely guided by the male and female jackals (i.e., the global best and second-best solutions), which restricts effective information sharing and collaboration within the population. As a result, GJO tends to suffer from insufficient local exploitation and slow convergence in the later stages, especially for high-dimensional problems. Finally, the boundary-handling mechanism adopts a passive reset-to-boundary strategy, which interrupts natural search trajectories, discards valuable search information, and risks overlooking optimal solutions near the boundaries, further constraining its overall performance [36,37].

To overcome these limitations, the present study introduces an enhanced Golden Jackal Optimization (MGJO) approach, augmented through multiple strategic improvements, and employs it specifically for image segmentation using multilevel thresholding. The principal innovations of this research are delineated below, encompassing both algorithmic advancements and practical implementations. Theoretically, MGJO integrates good-point set initialization, a dual crossover strategy, and a boundary-handling mechanism guided by the global best solution to overcome the limitations of standard GJO, enrich the improvement strategies of swarm intelligence algorithms, and provide new approaches for solving high-dimensional and complex optimization problems. From an application perspective, MGJO is employed in multilevel threshold image segmentation to enhance segmentation accuracy and efficiency, thereby offering technical support for practical domains such as medical image analysis, remote sensing interpretation, and industrial quality inspection.

To distinguish MGJO from existing improved GJO variants and highlight its innovative value, we supplemented a comparative analysis table (Table 1) that systematically contrasts MGJO with representative enhanced GJO methods (IGJO, AGJO, EGJO, DE-GJO) in terms of improvement strategies, core innovations, and performance gains. Conceptually, MGJO differs from existing methods by integrating three synergistic strategies (good-point set initialization, dual crossover, global-optimum boundary handling) rather than single-module improvements; functionally, the dual crossover (horizontal + vertical) enables both inter-individual information sharing and dimension-level fine-grained search, while the global-optimum boundary mechanism avoids information loss—addressing limitations of existing methods that focus only on exploration or exploitation.

Table 1. Comparison of MGJO with Existing Improved GJO Variants.

Algorithm	Improvement Strategies	Core Novelty	Reference
IGJO	Operators in Gradient-Based Optimizers	Enhances initial population diversity via opposition sampling	[38]
AGJO	The enhanced movement strategy, the global search strategy, and the multi-angle position update strategy for prey.	Improves convergence speed	[39]
EGJO	Combined with opposition-based learning, spring vibration-based adaptive mutation and binomial-based cross-evolution strategy	Enhances local exploitation via elite information guidance	[37]
DE-GJO	Hybrid differential evolution (DE) crossover	Introduces DE operators to expand search range	[40]
MGJO	Good-point set initialization, Dual crossover (horizontal + vertical) and Global-optimum boundary handling	Synergistic strategy integration, Dimension-level fine-grained search and Boundary information retention	

The main contributions of this paper are as follows:

(1). We propose a Modified Golden Jackal Optimization (MGJO) algorithm by integrating three innovative strategies to address the inherent limitations of the standard GJO.

First, the good-point set population initialization strategy replaces random initialization, ensuring the initial population is more uniformly and widely distributed in the search space, which lays a solid foundation for global exploration. Second, the dual crossover strategy—combining horizontal crossover (promoting information sharing among individuals to expand search coverage) and vertical crossover (enabling fine-grained search at the dimension level to deepen local exploitation)—effectively enhances the algorithm's ability to balance exploration and exploitation, especially for high-dimensional complex problems. Third, the global-optimum-based boundary handling mechanism replaces the passive "reset-to-boundary" strategy, guiding out-of-bound individuals toward the global optimal region to retain valid search information and improve the utilization of boundary-area search resources. These three strategies work synergistically to overcome the shortcomings of standard GJO, including uneven initialization, insufficient information exchange, and passive boundary handling, enriching the improvement framework of swarm intelligence algorithms.

(2). We conduct comprehensive validation of MGJO's performance through numerical optimization experiments and multilevel threshold image segmentation applications. On the CEC2017 (dim = 30, 100) and CEC2022 (dim = 10, 20) benchmark suites, MGJO outperforms seven mainstream algorithms (e.g., GWO, IWOA, GJO) in optimization accuracy, stability, and exploration–exploitation balance, as verified by population diversity analysis, ablation experiments, and statistical tests (Wilcoxon rank-sum test, Friedman mean-rank test). When applied to multilevel threshold segmentation of artistic images and benchmark images (e.g., Baboon, Lena) with Otsu's between-class variance as the objective function, MGJO achieves higher fitness values (closer to Otsu's optimal values) across 4-, 6-, 8-, and 10-level threshold tasks, and the segmented images exhibit superior peak signal-to-noise ratio (PSNR), structural similarity (SSIM), and feature similarity (FSIM), effectively preserving brushstroke details and color layers. This dual validation confirms MGJO's effectiveness in both numerical optimization and practical image segmentation scenarios, providing an efficient solution for high-dimensional complex optimization problems and artistic image processing demands.

The remainder of this paper is organized as follows: Section 2 introduces the basic principles of the standard Golden Jackal Optimization algorithm and presents the three enhancement strategies and mathematical model of MGJO. Section 3 reports numerical experiments on the CEC2017 and CEC2022 benchmark suites to validate the optimization performance of MGJO, along with comparative analysis against mainstream algorithms. Section 4 applies MGJO to multilevel threshold image segmentation, using Otsu's method as the objective function, and evaluates segmentation performance with PSNR, FSIM, and SSIM metrics. Section 5 concludes the study and discusses potential future research directions.

2. Golden Jackal Optimization and the Proposed Methodology

2.1. Golden Jackal Optimization (GJO)

Proposed by Nitish Chopra and Mohsin Ansari in 2022, the Golden Jackal Optimization (GJO) algorithm is a metaheuristic inspired by the cooperative hunting tactics of the golden jackal (*Canis aureus*), a medium-sized canid widely distributed across North Africa, Eastern Europe, the Middle East, and parts of Asia. Characterized by their relatively small body size and long, slender legs, golden jackals are well adapted for long-distance running to pursue prey. Notably, they often hunt in male–female pairs, a cooperative strategy that serves as the biological foundation for the design of GJO [34].

2.1.1. Search Space Formulation

As a swarm intelligence method, the GJO algorithm begins by creating an initial set of potential solutions, which are randomly positioned throughout the entire search space to serve as starting points for the optimization process. The initial solution Z_0 can be expressed as follows [34]:

$$Z_0 = (UB - LB) \cdot rand + LB \tag{1}$$

where LB and UB denote the lower and upper bounds of the search space, respectively, and $rand$ represents a uniformly distributed random number in the interval [0, 1].

At this stage, an initial population matrix is formed, representing the prey set, which can be expressed as:

$$Prey = \begin{bmatrix} Z_{1,1} & Z_{1,2} & \cdots & Z_{1,\,dim} \\ Z_{2,1} & Z_{2,2} & \cdots & Z_{2,\,dim} \\ \vdots & \vdots & \vdots & \vdots \\ Z_{pop\,,1} & Z_{pop\,,2} & \cdots & Z_{pop,dim} \end{bmatrix} \tag{2}$$

where $Z_{i,j}$ represents the j^{th} element of the j^{th} prey, pop is the total number of prey, and dim denotes the dimensionality of the decision variables. Each parameter of a solution is referred to as the position of the prey. During the optimization process, the fitness of each prey is evaluated using the objective function.

The fitness values of all prey are calculated as follows:

$$Fit = \begin{bmatrix} f(Z_{1,1}, Z_{1,2}, \cdots, Z_{1,dim}) \\ f(Z_{2,1}, Z_{2,2}, \cdots, Z_{2,dim}) \\ \vdots \\ f\left(Z_{pop,1}, Z_{pop,2}, \cdots, Z_{pop,dim}\right) \end{bmatrix} \tag{3}$$

where Fit is the fitness matrix containing all prey fitness values, and $f(\cdot)$ represents the cost function. Among all fitness values, the prey with the best fitness is considered the male golden jackal, while the prey with the second-best fitness is designated as the female golden jackal.

2.1.2. Exploration Stage (Target Prospecting and Discovery)

The golden jackal possesses the ability to detect and track prey; however, in certain situations, the prey may escape. In such cases, the golden jackal chooses to wait and search for new prey. During the hunting process, the male jackal typically takes the lead, while the female jackal follows behind. This process can be mathematically expressed as follows [34]:

$$\begin{aligned} Z_1(t) &= Z_M(t) - E|Z_M(t) - RL \times Prey(t)| \\ Z_2(t) &= Z_{FM}(t) - E|Z_{FM}(t) - RL \times Prey(t)| \end{aligned} \tag{4}$$

where $Z_M(t)$ and $Z_{FM}(t)$ denote the positions of the male and female jackals in the search space, respectively. E represents the escaping energy of the prey, which is calculated as shown in Equation (5). RL is a random vector based on the Lévy flight distribution, defined in Equation (6).

$$\begin{aligned} E &= E_1 \times E_0 \\ E_0 &= 2 \times r - 1 \\ E_1 &= c_1 \times \left(1 - \frac{t}{T}\right) \end{aligned} \tag{5}$$

$$LF(z) = 0.01 \times \frac{\mu \times \sigma}{|v|^{1/\beta}}$$
$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin(\pi\beta/2)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times \left(2^{\frac{\beta-1}{2}}\right)} \right)^{1/\beta} \quad (6)$$

where r is a random number within the interval [0, 1], c_1 is a constant (set to 1.5), t denotes the current iteration, and T is the maximum number of iterations. During the iterative process, E_1 decreases linearly from 1.5 to 0. Additionally, μ and σ are random numbers within [0, 1], and β is a constant set to 1.5.

Finally, the position of the golden jackal is updated according to:

$$Z(t+1) = \frac{Z_1(t)+Z_2(t)}{2} \quad (7)$$

2.1.3. Intensification Phase (Prey Capture and Constriction)

When the prey is disturbed by the golden jackals, its escaping ability gradually diminishes. At this stage, the male and female jackals begin to encircle the prey detected during the earlier phase. Once the encirclement is completed, they pounce on the prey and capture it. Based on this predatory behavior of the male and female jackals, the following mathematical model can be established [34]:

$$Z_1(t) = Z_M(t) - E \cdot |RL \cdot Z_M(t) - Prey(t)|$$
$$Z_2(t) = Z_{FM}(t) - E \cdot |RL \cdot Z_{FM}(t) - Prey(t)| \quad (8)$$

in Equation (8), RL introduces random behavior during the exploitation phase, enhancing the algorithm's exploration capability and helping to prevent it from becoming trapped in local optima.

2.1.4. Transitioning from Exploration to Exploitation

In the GJO algorithm, the transition from the exploration phase to the exploitation phase is controlled by the value of E. During the prey's escape process, its energy level decreases significantly. When E_0 decreases from 0 to -1, the prey's stamina gradually weakens; conversely, when E_0 increases from 0 to 1, the prey's stamina gradually strengthens. $|E| \geq 1$, the male and female jackals disperse to different regions to search for prey, indicating that the algorithm is in the exploration phase. If $|E| < 1$, the jackals initiate an attack on the prey, and the algorithm transitions into the exploitation phase [34].

2.2. Proposed Golden Jackal Optimization

2.2.1. Good-Point-Set Population Initialization

To address the limitation of the standard GJO in which the population distribution is insufficiently uniform and extensive—resulting in incomplete exploration of the search space, potentially missing better solutions or slowing convergence—we introduce a good-point set (GPS) for initializing the golden jackal population, ensuring a wider and more uniform coverage across the entire search space. The good-point set can be computed as follows:

$$\begin{cases} G_n(z) = \left\{ \left(\left\{ o_1^{(n)} \times l \right\}, \left\{ o_2^{(n)} \times l \right\}, \ldots, \left\{ o_{dim}^{(n)} \times l \right\} \right), 1 \leq l \leq n \right\} \\ \varphi(z) = C(r,\varepsilon) \times n^{-1+\varepsilon} \\ o_i = 2 \times cos\frac{2\times\pi\times i}{p}, 1 \leq i \leq m \end{cases} \quad (9)$$

where $C(r,\varepsilon)$ is a constant dependent only on r and $\varepsilon(\varepsilon > 0)$, dim denotes the problem dimension, $P_n(z)$ represents the good-point set, o_i is the element of the good-point set, and

p is the smallest prime number satisfying $(p-3)/2 \geq s$. $\varphi(z)$ is the discrepancy of the good-point set $G_n(z)$, which quantifies the uniformity of the point set distribution in the search space; a smaller $\varphi(z)$ indicates a more uniform distribution. denotes the size of the good-point set, which is equal to the population size ($n = pop$) in this study, and its value is set to 30 consistently with the experimental parameter settings.

Finally, the population initialization using the good-point set is expressed as:

$$Z_0 = \{G_n(z)\} \cdot (UB - LB) + LB \tag{10}$$

Figure 1 illustrates the population distributions of GJO and MGJO. It can be observed that MGJO, initialized using the good-point set, exhibits a broader and more uniform population distribution, which enhances the exploration capability of the algorithm.

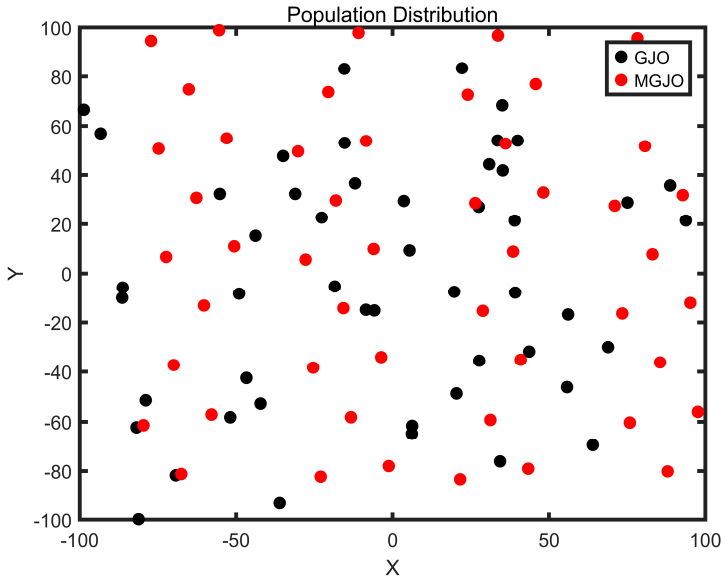


Figure 1. Comparison of two initial population distribution methods.

2.2.2. Double-Crossover Strategy

In the standard GJO algorithm, individual updates primarily rely on interactions between individuals and the global best solution, while effective information exchange among individuals within the population is lacking. This limitation is particularly pronounced in high-dimensional and complex optimization problems, leading to insufficient local exploitation and slow convergence in later stages. Moreover, when handling variables of different dimensions, the standard algorithm employs a uniform update strategy, which cannot perform fine-grained search tailored to the unique characteristics of each dimension, thereby significantly affecting overall performance. To address these challenges, this study innovatively integrates horizontal and vertical crossover strategies, forming a dual-crossover mechanism designed to enhance the search capability of the algorithm comprehensively. The double-crossover strategy is adopted because horizontal crossover promotes inter-individual information sharing to expand search coverage, while vertical crossover enables intra-individual dimensional fine-tuning to deepen local exploitation, jointly addressing the balance between exploration and exploitation in high-dimensional optimization [41].

(1) Horizontal crossover: enhancing information sharing among individuals

The horizontal crossover strategy focuses on promoting information flow and integration between different individuals within the population. Specifically, the population

indices are first randomly permuted to generate parent pairs ($parent1$ and $parent2$). For each parent pair, two offspring are generated as follows:

$$\begin{aligned} offspring1 &= r_1 \cdot parent1 + (1 - r_1) \cdot parent2 + \gamma \cdot (parent1 - parent2) \\ offspring2 &= r_2 \cdot parent2 + (1 - r_2) \cdot parent1 + \gamma \cdot (parent2 - parent1) \end{aligned} \tag{11}$$

where $r_1, r_2 \in U(0,1)$ are uniformly distributed random weights that flexibly adjust the contribution of each parent in generating offspring. The perturbation factor $\gamma \in [0,1]$ introduces a random disturbance term, effectively preventing the offspring from being too similar to their parents and thereby enhancing population diversity.

(2) Vertical crossover: enabling dimension-level fine-grained search

The vertical crossover strategy emphasizes information exchange and optimization across different dimensions within an individual. For each individual in the population, two distinct dimensions j_1 and j_2 ($j_1 \neq j_2$) are randomly selected, and a new solution is generated by performing crossover on these dimensions as follows:

$$X_{vc}(j_1) = \beta \cdot X(i, j_1) + (1 - \beta) \cdot X(i, j_2) \tag{12}$$

where $\beta \in U(0,1)$ is a random parameter used to flexibly adjust the weight allocation during dimension-level crossover. This approach enables the algorithm to explore potential relationships among different dimensions within an individual, allowing fine-tuned adjustments that respect the unique characteristics of high-dimensional problems.

2.2.3. Boundary Processing Mechanism Based on Global Optimization

In swarm intelligence algorithms, boundary control plays a crucial role by constraining the search space, thereby preventing invalid searches, premature convergence, and entrapment in local optima [42]. Such issues may arise when the search process deviates from feasible solutions or exceeds problem constraints. If the search space is set too narrowly (i.e., the range between LB and UB is excessively small, failing to cover potential optimal solution regions), it may hinder the algorithm from fully exploring the broader solution space, increasing the likelihood of being trapped in local optima [43]. Therefore, defining an appropriate boundary correction mechanism can enhance the algorithm's ability to find the global optimum, prevent wasted search within infeasible regions, and avoid unnecessary computational effort. In some cases, individuals exceeding the boundaries may restrict the search process, preventing the algorithm from fully leveraging information carried by the current population [44].

Within the conventional Golden Jackal Optimizer, a rudimentary constraint-management technique is typically applied, exemplified by repositioning straying search agents directly onto the feasible region's perimeter. This abrupt correction severs the agent's natural exploratory path, effectively discarding the directional intelligence gathered from its current iteration. This fundamentally reactive "corrective" strategy not only squanders computational effort but also introduces the risk of overlooking promising, near-boundary optimal solutions that the agent's original trajectory was approaching.

To reconceptualize constraint management from a restrictive punitive measure into a proactive directional aid, this work introduces a boundary guidance mechanism anchored to the swarm's best-known position. As depicted in Figure 2, should an agent's position exceed the feasible domain, the methodology does not merely reposition it at random. Instead, it actively steers the individual towards the vicinity of the incumbent global optimum. This technique reframes an invalid boundary violation as a productive refinement step, channeling the search effort into promising high-fitness regions. By simultaneously correcting infeasible solutions and capitalizing on the collective search history, the mech-

anism markedly enhances the convergence rate. The mathematical formulation for this process is given by:

$$Z(i) = \begin{cases} Z_{best} + \alpha \cdot (UB - Z_{best}), if \ \ Z(i) > UB \\ Z_{best} - \alpha \cdot (Z_{best} - LB), if \ \ Z(i) < LB \end{cases} \tag{13}$$

where Z_{best} denotes the global best solution, and α is a uniformly distributed random number in the interval [0, 0.5].

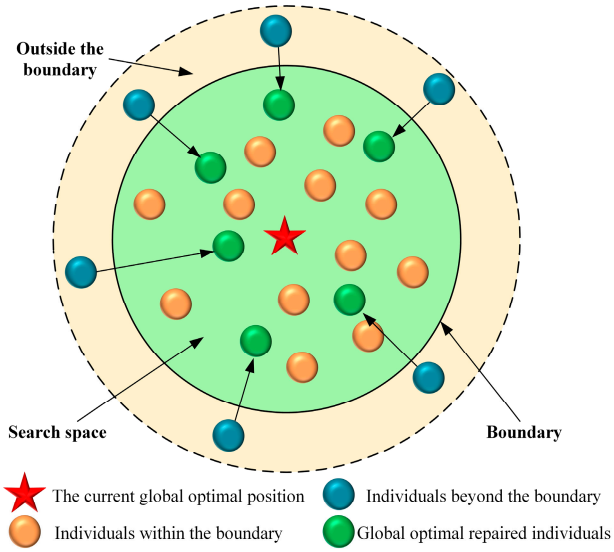


Figure 2. Schematic diagram of the processing mechanism based on global optimization.

In summary, the pseudo-code of the MGJO algorithm is presented in Algorithm 1.

Algorithm 1. Pseudocode of the Modified Golden Jackal Optimizer (MGJO)

Inputs: The population size pop, and the maximum iterations T
Inputs: The location of prey population $Z_i(i = 1, 2, \cdots, pop)$
while $t < T$ ***do***
**Compute the objective function values for all candidate solutions.*
    Z_1 *=the best prey (Male jackal position)*
    Z_2 *=second best prey (Female jackal position)*
    for $i = 1 : pop$ ***do***
        Update the evading energy E using Equation (5)
        Update RL using Equation (6)
        if $|E| \geq 1$ ***do (Exploration)***
            Update the prey position using Equations (4) and (7)
            Using Equation (13) for boundary adjustment
        else $|E| < 1$ ***do (Exploitation)***
            Update the prey position using Equations (7) and (8).
            Apply Equation (13) to perform solution-boundary correction.
        end if
    end for
    Update the prey position using Equations (11) and (12)
end while
Return Z_1.

2.3. Time Complexity Analysis

The run time complexity of MGJO depends on four core processes: good-point set initialization, fitness evaluation, golden jackal position update, and double-crossover strategy (the global-optimum-based boundary handling has the same complexity as GJO's passive reset and no additional overhead). For a population size of n, maximum iterations of t, and problem dimension of d: the computational complexity of good-point set initialization is $O(n \times d)$ (consistent with GJO's random initialization, as each individual's d-dimensional parameters are generated via good-point set mapping); the fitness evaluation complexity is $O(n \times d \times f)$ (same as GJO, with f as the objective function complexity treated as a constant); the position update complexity is $O(t \times n) + O(t \times n \times d)$ (same as GJO, including searching for the best positions and updating individual vectors); the double-crossover strategy adds $O(t \times n \times d)$ complexity (horizontal crossover for inter-individual information sharing and vertical crossover for dimensional fine-grained search, both with linear overhead). Therefore, the total run time complexity of MGJO is $O(n \times d \times (2t + 1) + t \times n)$, which belongs to the same asymptotic order $O(n \times d \times t)$ as standard GJO, ensuring no excessive computational cost while improving performance.

3. Numerical Experiments

3.1. Configuration of Algorithmic Parameters

This section assesses the efficacy of the proposed MGJO methodology by employing the highly complex CEC2017 [45] and CEC2022 [46] numerical benchmark suites for comparative analysis against established metaheuristic algorithms. The comparison algorithms include: Gray Wolf Optimizer (GWO) [11], Improved Whale Optimization Algorithm (IWOA) [47], Autonomous Groups Particles Swarm Optimization (AGPSO) [48], Dung Beetle Optimizer (DBO) [21], Holistic swarm optimization (HSO) [49], Birds of Prey-Based Optimization (BPBO) [50] and Golden jackal optimization (GJO) [34]. The specific parameter settings for each comparison algorithm are listed in Table 2.

Table 2. Comparison of algorithm parameter settings.

Algorithms	Name of the Parameter	Value of the Parameter
GWO	a	[0, 2]
IWOA	r, l, a	[0, 1], [−1, 1], Linear reduction from 2 to 1
AGPSO	w_1, w_2, c_1, c_2	0.9, 0.4, [2.55, 0.5], [1.25, 2.25]
HSO	α	3
DBO	$P_{percent}$	0.2
BPBO	Pi	0.7
GJO	c_1	1.5
MGJO	$c_1, \alpha, \beta, \gamma$	1.5, [0, 1], [0, 1], [−1, 1]

To maintain experimental equity and account for stochastic influences, this investigation standardized the computational framework with a uniform population of 30 solutions and a 500-iteration ceiling across all benchmarked methods. Each technique completed 30 independent trials, with recorded metrics including arithmetic mean (Ave), statistical deviation (Std), and performance ranking (Rank)—optimal values being typeset in bold. The technical environment featured Windows 10 running on 13th-generation Intel Core i5-13400 silicon (2.5 GHz) with 16 GB DDR4 memory, utilizing MATLAB 2024b as the computational engine.

3.2. Qualitative Analysis of MGJO

This segment presents a conceptual evaluation of the developed MGJO framework. First, we analyze the solution distribution diversity—a vital characteristic for comprehensive search space coverage. Subsequently, we investigate the equilibrium between global

scanning and local refinement, noting that earlier cycles demand broader investigation while subsequent stages require intensified concentration. The method's search capabilities are verified through dedicated exploration and exploitation metrics. Furthermore, component ablation studies are conducted to quantify the contribution of each enhancement. Comprehensive discussions follow in subsequent sections.

3.2.1. Analysis of the Population Diversity

Within the domain of metaheuristic search methodologies, population diversity quantifies the dispersion characteristics among constituent agents in a solution set [51], with each agent characteristically representing a potential configuration within the feasible domain. A decline in population diversity can lead to premature convergence to a local optimum, thereby constraining the algorithm's global search capability. On the other hand, preserving higher diversity facilitates exploration across broader areas of the solution space and improves the probability of locating the global optimum. In this subsection, we evaluate the population diversity of the MGJO algorithm based on Equation (14) [52].

$$I_C(t) = \sqrt{\sum_{i=1}^{N} \sum_{d=1}^{D} \left(x_{id}(t) - c_d(t)\right)^2} \tag{14}$$

where the metric $I_C(t)$ characterizes the collective dispersion of the solution set, N corresponds to the total number of agents, D signifies the dimensionality of the search space, and $x_{id}(t)$ represents the coordinate of the i^{th} candidate solution along the d^{th} axis during iteration t. The coefficient $c_d(t)$ measures the spatial distribution of the entire population relative to its centroid at cycle t, computed through Equation (15).

$$c_d(t) = \frac{1}{D} \sum_{i=1}^{N} x_{id}(t) \tag{15}$$

Figure 3 presents the evolution curves of population diversity for MGJO and the original GJO on the CEC2017 test suite. Population diversity is quantified using the $I_C(t)$ metric defined in Equation (14), which reflects the dispersion of individuals within the search space. Higher values indicate stronger diversity, which is beneficial for exploring unknown regions and avoiding local optima.

As shown in the figure, across all test functions, the diversity curve of MGJO consistently remains above that of GJO, and this advantage persists throughout the entire iteration process. In the early stage of iteration (first 50 generations), MGJO, aided by the elite-based initialization strategy, distributes the initial population more uniformly and broadly across the search space, resulting in significantly higher $I_C(t)$ values than GJO and laying a solid foundation for subsequent global exploration. During the middle stage of iteration (50–300 generations), the dual crossover strategy (Section 2.2.2) promotes information sharing among individuals through horizontal crossover and enables fine-grained, dimension-level search via vertical crossover. This effectively maintains population diversity and prevents the rapid decline observed in GJO, which heavily relies on the global best solution for individual updates. In the late stage of iteration (after 300 generations), although the $I_C(t)$ values of both algorithms gradually decrease with convergence, MGJO still maintains a higher diversity level. This is attributed to the global best-based boundary handling mechanism (Section 2.2.3)—when individuals exceed the search boundaries, they are guided toward regions near the global optimum rather than simply being reset. This approach preserves the search direction of individuals while injecting new exploratory vitality into the population.

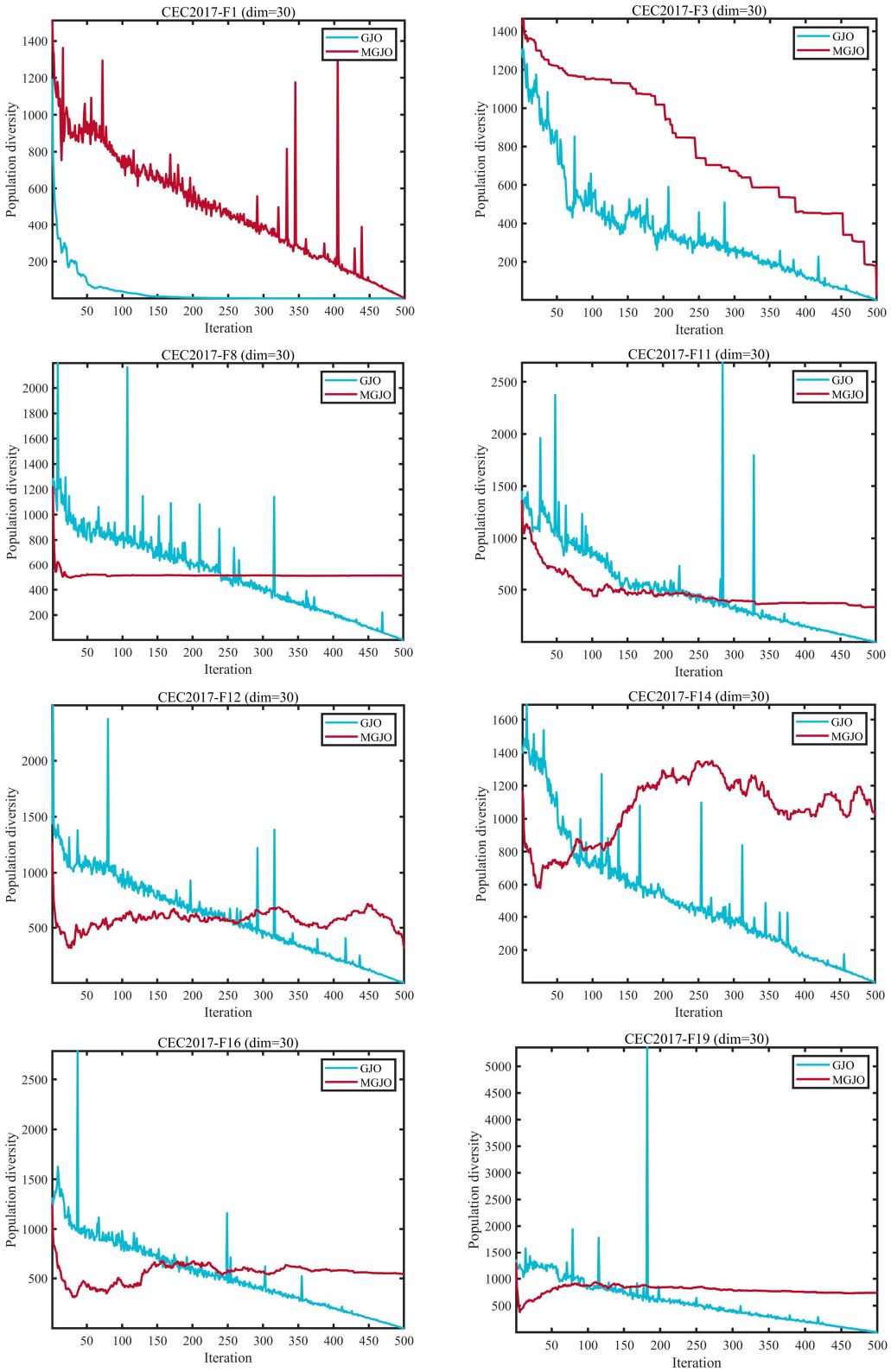


Figure 3. *Cont*.

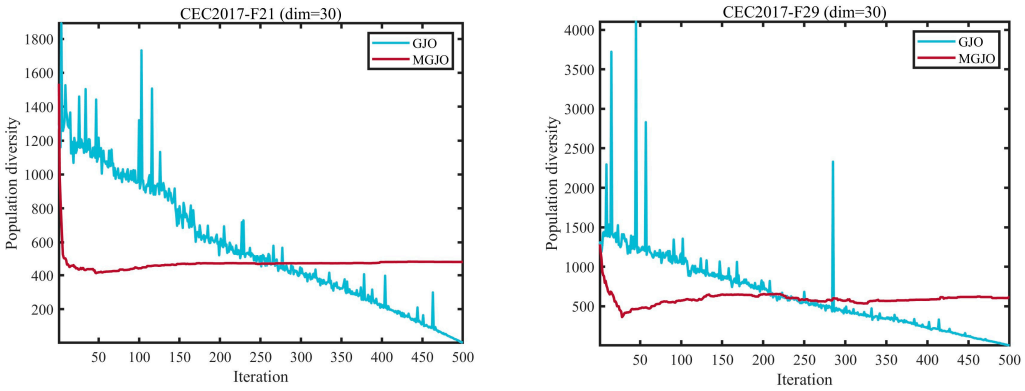


Figure 3. Comparative analysis of swarm diversity distribution between MGJO and GJO methodologies.

For instance, in the CEC2017-F8 function, after 500 iterations, the $I_C(t)$ value of MGJO is approximately 900, whereas that of GJO is only 600, a difference of 33.3%. In the CEC2017-F29 function, MGJO maintains $I_C(t)$ above 3700 throughout, while GJO drops below 3000 in the later iterations. These results fully demonstrate that MGJO, through its three key improvements, significantly enhances population diversity and effectively mitigates the premature convergence problem common in traditional GJO, providing crucial support for efficient exploration of the search space and discovery of global optima.

3.2.2. Examination of Global Search and Local Refinement Dynamics

Within heuristic optimization frameworks, global search and local intensification represent two fundamental and complementary phases. Global search encompasses extensive investigation throughout the solution domain to identify promising regions potentially containing optimal solutions, thereby ensuring the algorithm does not prematurely converge to suboptimal areas. Conversely, local intensification entails concentrated examination and refinement of previously discovered high-potential solutions, progressively enhancing solution quality through meticulous neighborhood investigation. This latter phase strategically utilizes accumulated knowledge to systematically exploit the most promising search regions.

Overemphasis on exploration may result in inefficient allocation of computational resources, as the algorithm scans widely without making concentrated improvements, potentially missing opportunities to refine solutions in specific zones. Conversely, excessive exploitation can cause the algorithm to converge prematurely to a local optimum, limiting its capacity to identify better solutions elsewhere in the search domain [53]. Hence, achieving an appropriate balance between these two processes is vital for the algorithm's performance. In this section, we examine the exploration and exploitation properties of the MGJO algorithm, which are quantitatively evaluated using Equations (16) and (17) [12].

$$Exploration(\%) = \frac{Div(t)}{Div_{max}} \times 100\% \tag{16}$$

$$Exploitation(\%) = \frac{|Div(t) - Div_{max}|}{Div_{max}} \times 100\%, \tag{17}$$

where the exploration metric $Div(t)$ characterizes the global search intensity at iteration t, computed through Equation (17), while Div_{max} d represents the peak exploration magnitude observed throughout the optimization process.

$$Div(t) = \frac{1}{D}\sum_{d=1}^{D}\frac{1}{N}\sum_{i=1}^{N} \mid median(x_d(t)) - x_{id}(t) \mid. \tag{18}$$

Figure 4 illustrates the evolution of exploration and exploitation capabilities of MGJO on the CEC2017 test suite, focusing on 10 representative functions (e.g., F1, F3, F6). These

capabilities are quantified using Equations (16)–(18), where the exploration rate reflects the algorithm's global search ability, and the exploitation rate indicates its capacity for fine-grained search around high-quality solutions.

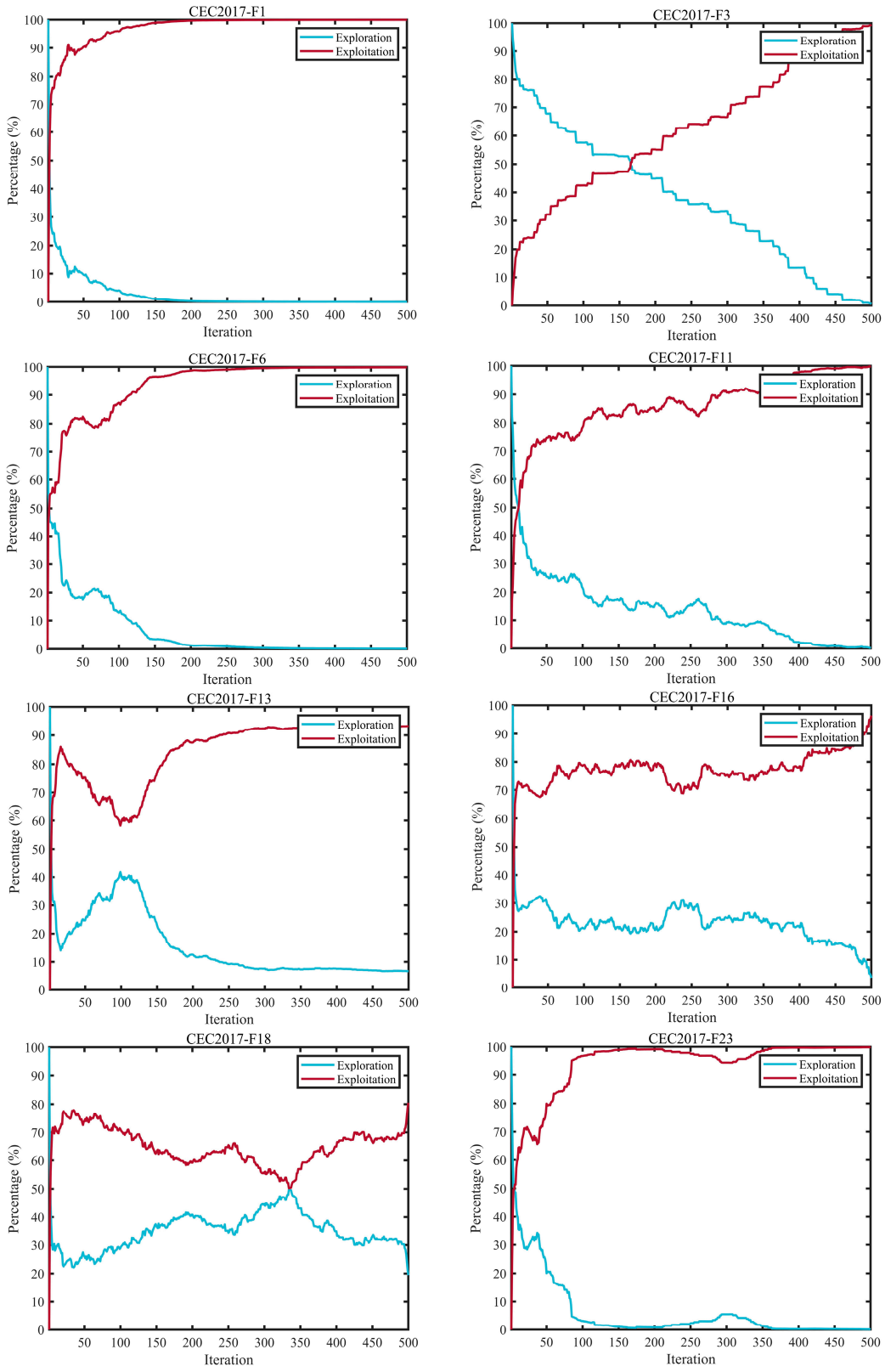


Figure 4. *Cont.*

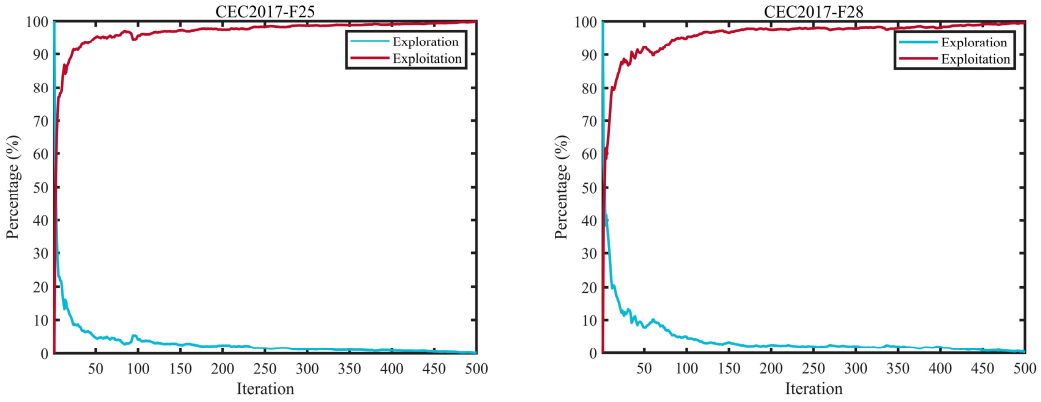


Figure 4. Performance analysis of exploration versus exploitation in MGJO.

In the early stage of iteration (first 100 generations), MGJO's exploration rate rapidly rises to 80–90%; for example, F1 reaches 85% at generation 50, and F3 exceeds 80%. This is attributed to the broad population distribution from the elite-based initialization combined with perturbations from Lévy flight random vectors, enabling rapid coverage of critical regions in the search space. During the middle stage of iteration (100–300 generations), the exploration rate gradually decreases to 40–60%, while the exploitation rate increases to 50–70%. Taking F11 as an example, at generation 200, the exploration and exploitation rates stabilize around 50% and 45%, respectively. This behavior results from the dynamic adjustment of the E value, which facilitates a smooth transition from global search to localized capture, while the dual crossover strategy enhances exploitation by sharing information through horizontal crossover and optimizing dimensional correlations via vertical crossover. In the late stage of iteration (after 300 generations), the exploitation rate maintains a high level of 60–80%; for instance, F23 reaches 75% and F28 exceeds 70% by generation 500, without falling into local optima. This performance benefits from the global best-based boundary handling mechanism, which guides out-of-bound individuals toward advantageous regions, ensuring high exploitation efficiency while maintaining moderate exploration.

Overall, MGJO effectively balances exploration and exploitation, preventing resource wastage and loss of the global optimum, thereby providing robust performance for high-dimensional and complex optimization problems.

3.2.3. Impact Analysis of the Modification

To quantitatively assess both the isolated impacts and combined efficacy of the three proposed improvement mechanisms—Good-Point-Set Population Initialization (S1), Double-Crossover Strategy (S2), and Boundary Processing Mechanism Based on Global Optimization (S3)—on the baseline GJO, ablation experiments were conducted using the CEC2017 benchmark functions with dimension dim = 30. Five comparative variants were designed: GJO (standard), GJO-S1 (only S1), GJO-S2 (only S2), GJO-S3 (only S3), and MGJO, which integrates all strategies. The experimental results are presented in Figures 5 and 6.

From the convergence curves in Figure 5, it is evident that all variants incorporating at least one enhancement outperform the standard GJO, with MGJO demonstrating the most pronounced advantage. For functions such as F1, F10, and F16, MGJO consistently achieves the lowest fitness values (objective function values) and the fastest convergence. After 500 iterations, MGJO attains a fitness value of only 4.1805×10^3 on F1, far lower than the standard GJO (1.3244×10^{10}) and single-strategy variants (e.g., GJO-S1 at 9.5699×10^8). On complex multimodal functions such as F22 and F27, MGJO exhibits smoother convergence curves in the later stages, without significant oscillations, whereas single-strategy variants (e.g., GJO-S3) still show fitness rebounds. This indicates that individual strategies may

only improve performance at specific stages and cannot consistently optimize throughout the entire process. Furthermore, GJO-S1 demonstrates faster early-stage convergence than standard GJO (e.g., the first 100 iterations of F30), confirming that the good-point-set initialization enhances initial exploration efficiency. GJO-S2 achieves higher mid-stage optimization accuracy (e.g., F12 iterations 200–300), reflecting the value of information sharing via the double-crossover strategy. GJO-S3 performs more efficient late-stage boundary-region search (e.g., after 400 iterations on F29), validating the guiding role of the boundary processing mechanism.

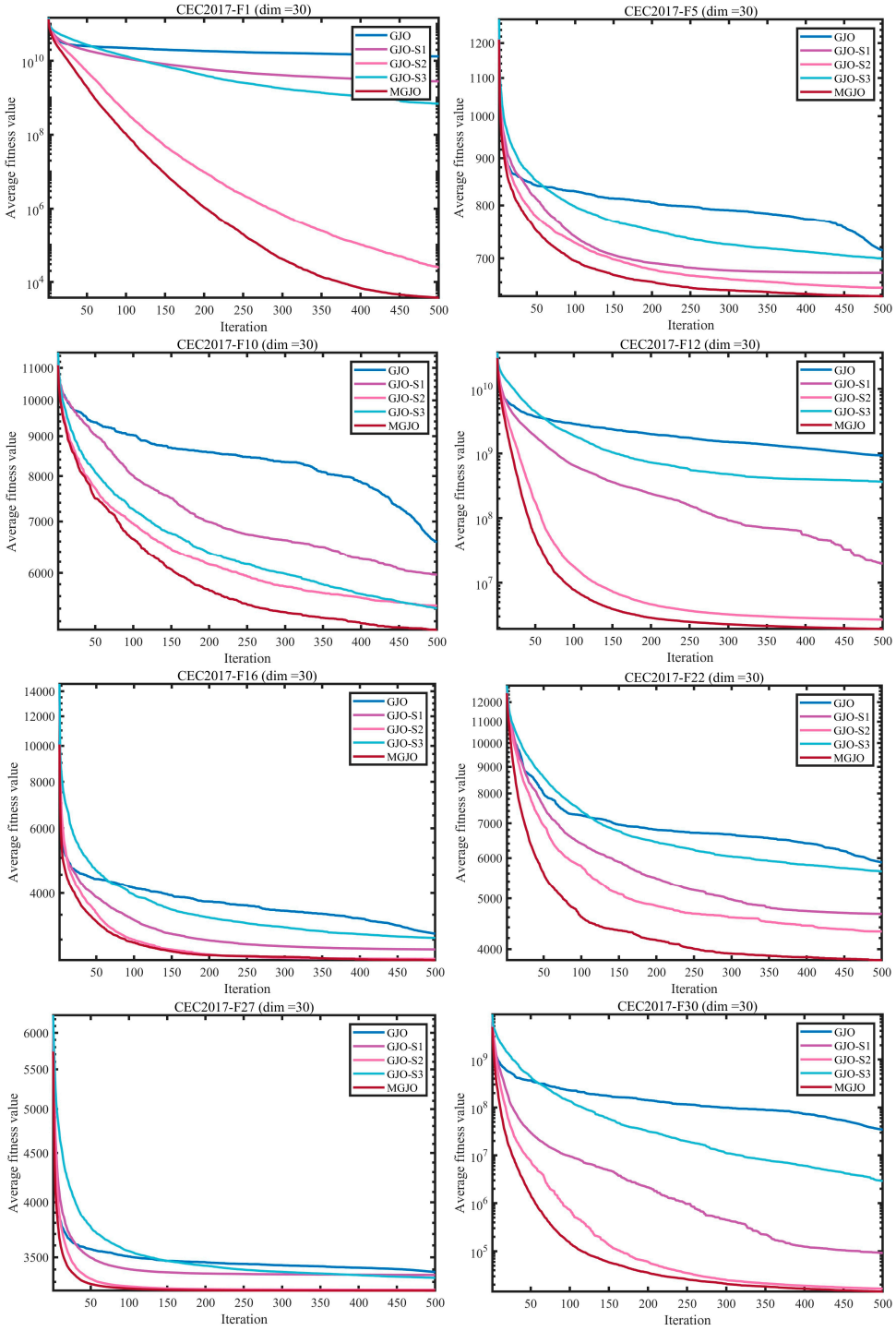


Figure 5. Performance evaluation of various enhancement methodologies.

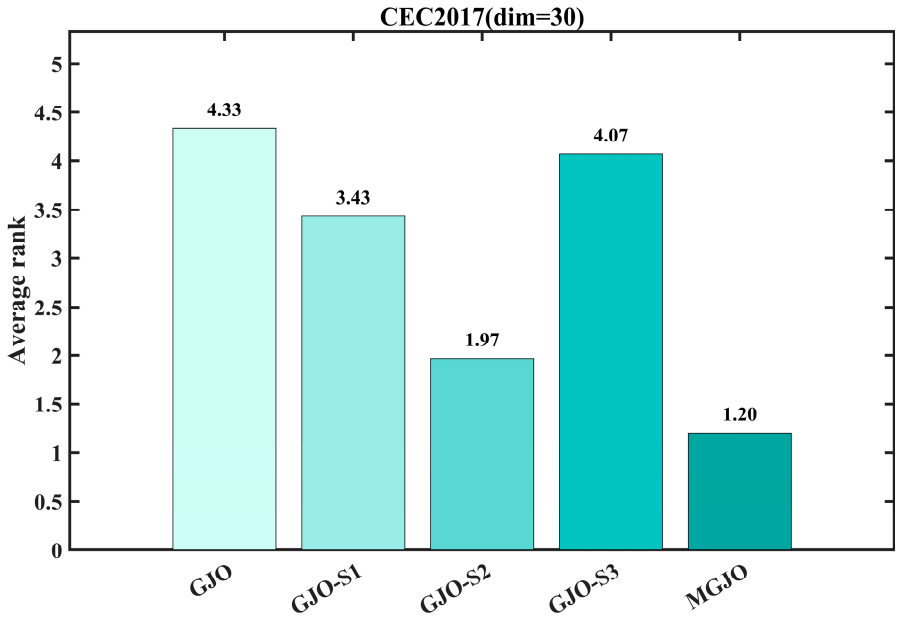


Figure 6. Comparative performance ranking of GJO variants with different enhancements.

The radar chart of average rankings in Figure 6 further quantifies the contribution of each strategy. MGJO achieves an overall average rank of only 1.20, significantly better than standard GJO (4.33) and the single-strategy variants (GJO-S1: 3.43, GJO-S2: 1.97, GJO-S3: 1.50). Among the individual strategies, S3 contributes most to ranking improvement (GJO-S3: 1.50), followed by S2 (GJO-S2: 1.97) and S1 (GJO-S1: 3.43). When integrated, the three strategies exhibit a strong synergistic effect, allowing MGJO to achieve the top ranking. This demonstrates that the three strategies effectively address GJO's weaknesses in initialization, information exchange, and boundary handling, respectively, and complement each other: good-point-set initialization establishes a broad exploration foundation, the double-crossover strategy enhances mid-stage information utilization, and the boundary processing mechanism ensures late-stage search accuracy. Combined, they form a "full-process optimization loop," resulting in an overall performance leap for the algorithm.

Table 3 reports the optimization performance of the standard GJO, its three enhanced variants incorporating different improvement strategies (GJO-S1, GJO-S2, GJO-S3), and the combined MGJO on the CEC2017 benchmark set (dim = 30). The mean (Ave) and standard deviation (Std) provide an intuitive overview of each algorithm's accuracy and stability. According to the results, all single-strategy variants outperform the original GJO. Among them, GJO-S3 (global best–guided boundary handling) achieves particularly strong performance on most functions. For example, on F1, its Ave drops to 6.9047×10^{8}, a substantial reduction compared with GJO's 1.3149×10^{10}. GJO-S2 (dual-crossover strategy) shows powerful local exploitation on complex multimodal functions such as F9, achieving an Ave of 2.9440×10^{3}, better than GJO's 5.4780×10^{3}. GJO-S1 (good-point-set initialization) improves global exploration by optimizing the initial population distribution; for instance, on the high-dimensional F2 function, the Ave decreases from 3.7800×10^{36} to 3.1681×10^{33}.

MGJO, which integrates all three strategies, produces a clear synergistic effect and achieves the best results across all test functions. Not only does it yield the lowest Ave values on functions such as F1 and F2 (3.7651×10^{03} and 2.2475×10^{14}, respectively), but it also exhibits greater stability, with Std values consistently smaller than those of the single-strategy variants—for example, an Std of only 3.9619×10^{-02} on F6. These outcomes demonstrate the complementary strengths of good-point-set initialization, the dual-crossover strategy, and the global best–guided boundary handling mechanism. Their integrated use effectively overcomes issues such as uneven initialization, insufficient infor-

mation exchange, and passive boundary handling in GJO, laying a solid foundation for MGJO's superior performance on high-dimensional and complex optimization problems.

Table 3. Experimental findings on the CEC 2017 benchmark suite (dim = 30).

Function	Metric	GJO	GJO-S1	GJO-S2	GJO-S3	MGJO
F1	Ave	1.3149×10^{10}	2.7855×10^{09}	2.4665×10^{04}	6.9047×10^{08}	$\mathbf{3.7651 \times 10^{03}}$
	Std	4.2973×10^{09}	2.8560×10^{09}	2.6383×10^{04}	4.1807×10^{09}	$\mathbf{3.6750 \times 10^{03}}$
F2	Ave	3.7800×10^{36}	3.1681×10^{33}	2.7613×10^{16}	2.0410×10^{37}	$\mathbf{2.2475 \times 10^{14}}$
	Std	2.4403×10^{37}	1.5166×10^{34}	1.1158×10^{17}	1.3717×10^{38}	$\mathbf{9.1683 \times 10^{14}}$
F3	Ave	$\mathbf{6.1534 \times 10^{04}}$	1.3289×10^{05}	9.6556×10^{04}	1.3319×10^{05}	8.2440×10^{04}
	Std	$\mathbf{1.0117 \times 10^{04}}$	3.8009×10^{04}	2.7842×10^{04}	4.0889×10^{04}	2.2117×10^{04}
F4	Ave	1.3075×10^{03}	8.3526×10^{02}	5.1802×10^{02}	9.6771×10^{02}	$\mathbf{5.1436 \times 10^{02}}$
	Std	7.3982×10^{02}	9.9451×10^{02}	2.5429×10^{01}	1.0191×10^{03}	$\mathbf{1.9166 \times 10^{01}}$
F5	Ave	7.1420×10^{02}	6.7542×10^{02}	6.5085×10^{02}	6.9936×10^{02}	$\mathbf{6.3749 \times 10^{02}}$
	Std	4.5854×10^{01}	3.3407×10^{01}	3.5259×10^{01}	$\mathbf{2.9212 \times 10^{01}}$	2.9944×10^{01}
F6	Ave	6.3982×10^{02}	6.2638×10^{02}	6.0290×10^{02}	6.2545×10^{02}	$\mathbf{6.0006 \times 10^{02}}$
	Std	8.1038×10^{00}	1.1338×10^{01}	6.5810×10^{00}	1.0352×10^{01}	$\mathbf{3.9619 \times 10^{-02}}$
F7	Ave	1.0474×10^{03}	1.0361×10^{03}	9.3598×10^{02}	1.1160×10^{03}	$\mathbf{8.9236 \times 10^{02}}$
	Std	5.9275×10^{01}	9.9329×10^{01}	6.1129×10^{01}	1.0098×10^{02}	$\mathbf{4.1074 \times 10^{01}}$
F8	Ave	9.8532×10^{02}	9.4049×10^{02}	9.3302×10^{02}	9.5380×10^{02}	$\mathbf{9.1312 \times 10^{02}}$
	Std	4.1609×10^{01}	3.2855×10^{01}	2.4688×10^{01}	$\mathbf{2.1915 \times 10^{01}}$	2.9506×10^{01}
F9	Ave	5.4780×10^{03}	5.0697×10^{03}	2.9440×10^{03}	4.9617×10^{03}	$\mathbf{1.9042 \times 10^{03}}$
	Std	1.7382×10^{03}	2.2601×10^{03}	1.0698×10^{03}	1.0561×10^{03}	$\mathbf{7.5839 \times 10^{02}}$
F10	Ave	6.5879×10^{03}	5.9690×10^{03}	5.4494×10^{03}	5.4013×10^{03}	$\mathbf{5.0730 \times 10^{03}}$
	Std	1.3246×10^{03}	1.4901×10^{03}	4.7779×10^{02}	$\mathbf{3.4518 \times 10^{02}}$	6.4052×10^{02}
F11	Ave	4.1974×10^{03}	3.5650×10^{03}	2.8659×10^{03}	7.3527×10^{03}	$\mathbf{1.5479 \times 10^{03}}$
	Std	1.5985×10^{03}	2.5612×10^{03}	2.2137×10^{03}	5.2060×10^{03}	$\mathbf{7.1265 \times 10^{02}}$
F12	Ave	9.3605×10^{08}	1.9871×10^{07}	2.6930×10^{06}	3.6412×10^{08}	$\mathbf{1.9249 \times 10^{06}}$
	Std	6.8215×10^{08}	4.4238×10^{07}	1.7841×10^{06}	1.7187×10^{09}	$\mathbf{1.1241 \times 10^{06}}$
F13	Ave	2.7718×10^{08}	8.3133×10^{05}	2.7489×10^{04}	2.9417×10^{08}	$\mathbf{1.1212 \times 10^{04}}$
	Std	4.7454×10^{08}	5.7531×10^{06}	1.3150×10^{05}	6.7097×10^{08}	$\mathbf{1.0753 \times 10^{04}}$
F14	Ave	$\mathbf{8.4081 \times 10^{05}}$	1.6046×10^{06}	1.2498×10^{06}	2.6088×10^{06}	9.1322×10^{05}
	Std	$\mathbf{8.8546 \times 10^{05}}$	1.8701×10^{06}	1.3323×10^{06}	2.3320×10^{06}	9.2958×10^{05}
F15	Ave	1.8554×10^{07}	$\mathbf{8.1784 \times 10^{03}}$	8.6849×10^{03}	3.6982×10^{07}	8.8123×10^{03}
	Std	4.5626×10^{07}	$\mathbf{7.5381 \times 10^{03}}$	9.0449×10^{03}	1.4626×10^{08}	9.2655×10^{03}
F16	Ave	3.1143×10^{03}	2.8244×10^{03}	2.6633×10^{03}	3.0309×10^{03}	$\mathbf{2.6426 \times 10^{03}}$
	Std	4.8034×10^{02}	3.2298×10^{02}	3.1538×10^{02}	$\mathbf{2.9006 \times 10^{02}}$	3.5112×10^{02}
F17	Ave	2.2924×10^{03}	2.3491×10^{03}	2.1714×10^{03}	2.4346×10^{03}	$\mathbf{2.1400 \times 10^{03}}$
	Std	2.6367×10^{02}	$\mathbf{2.1885 \times 10^{02}}$	2.3330×10^{02}	2.6166×10^{02}	2.4677×10^{02}
F18	Ave	2.1356×10^{06}	2.9990×10^{06}	$\mathbf{1.7875 \times 10^{06}}$	3.4259×10^{06}	2.1382×10^{06}
	Std	$\mathbf{2.3536 \times 10^{06}}$	4.4497×10^{06}	2.7113×10^{06}	3.6139×10^{06}	3.2935×10^{06}
F19	Ave	1.1452×10^{07}	9.6457×10^{03}	$\mathbf{7.7706 \times 10^{03}}$	1.0082×10^{07}	1.0154×10^{04}
	Std	2.8041×10^{07}	9.6409×10^{03}	$\mathbf{7.0285 \times 10^{03}}$	3.1141×10^{07}	1.0111×10^{04}
F20	Ave	2.6030×10^{03}	2.5925×10^{03}	2.4519×10^{03}	2.6399×10^{03}	$\mathbf{2.3504 \times 10^{03}}$
	Std	2.2812×10^{02}	2.4554×10^{02}	2.0489×10^{02}	2.1367×10^{02}	$\mathbf{1.7585 \times 10^{02}}$
F21	Ave	2.4941×10^{03}	2.4507×10^{03}	2.4476×10^{03}	2.4814×10^{03}	$\mathbf{2.4161 \times 10^{03}}$
	Std	3.6677×10^{01}	3.6714×10^{01}	3.5470×10^{01}	3.2290×10^{01}	$\mathbf{3.1822 \times 10^{01}}$
F22	Ave	5.8895×10^{03}	4.6699×10^{03}	4.3228×10^{03}	5.6522×10^{03}	$\mathbf{3.8092 \times 10^{03}}$
	Std	2.6166×10^{03}	$\mathbf{1.8649 \times 10^{03}}$	2.2942×10^{03}	2.1266×10^{03}	2.1337×10^{03}
F23	Ave	2.8997×10^{03}	2.8925×10^{03}	2.8070×10^{03}	2.9863×10^{03}	$\mathbf{2.7775 \times 10^{03}}$
	Std	4.6097×10^{01}	8.5296×10^{01}	4.1910×10^{01}	1.1897×10^{02}	$\mathbf{3.8988 \times 10^{01}}$
F24	Ave	3.0975×10^{03}	3.0929×10^{03}	3.0128×10^{03}	3.3616×10^{03}	$\mathbf{2.9714 \times 10^{03}}$
	Std	5.7202×10^{01}	9.9099×10^{01}	6.4108×10^{01}	1.8881×10^{02}	$\mathbf{5.0339 \times 10^{01}}$
F25	Ave	3.1982×10^{03}	3.0288×10^{03}	2.9148×10^{03}	3.0772×10^{03}	$\mathbf{2.9022 \times 10^{03}}$
	Std	1.1142×10^{02}	8.5251×10^{01}	2.2361×10^{01}	1.4755×10^{02}	$\mathbf{1.7223 \times 10^{01}}$
F26	Ave	6.1427×10^{03}	6.2560×10^{03}	5.2839×10^{03}	5.6075×10^{03}	$\mathbf{4.7852 \times 10^{03}}$
	Std	$\mathbf{6.7740 \times 10^{02}}$	9.9379×10^{02}	1.0448×10^{03}	1.5807×10^{03}	7.6806×10^{02}
F27	Ave	3.3794×10^{03}	3.3559×10^{03}	3.2393×10^{03}	3.3330×10^{03}	$\mathbf{3.2328 \times 10^{03}}$
	Std	6.8760×10^{01}	8.4396×10^{01}	1.2581×10^{01}	7.5706×10^{01}	$\mathbf{1.1582 \times 10^{01}}$
F28	Ave	3.9168×10^{03}	3.6032×10^{03}	3.2889×10^{03}	3.5823×10^{03}	$\mathbf{3.2635 \times 10^{03}}$
	Std	3.6641×10^{02}	2.5536×10^{02}	2.2982×10^{01}	4.9194×10^{02}	$\mathbf{2.2143 \times 10^{01}}$
F29	Ave	4.3638×10^{03}	4.1281×10^{03}	3.7857×10^{03}	4.1023×10^{03}	$\mathbf{3.6922 \times 10^{03}}$
	Std	2.3958×10^{02}	3.0785×10^{02}	$\mathbf{1.9756 \times 10^{02}}$	2.2164×10^{02}	2.0932×10^{02}
F30	Ave	3.4545×10^{07}	9.1420×10^{04}	1.6467×10^{04}	2.9009×10^{06}	$\mathbf{1.4314 \times 10^{04}}$
	Std	2.9010×10^{07}	2.5561×10^{05}	1.0593×10^{04}	4.0542×10^{06}	$\mathbf{7.9527 \times 10^{03}}$

3.3. Performance Evaluation and Discussion on CEC2017 and CEC2022 Benchmark Sets

This segment presents a rigorous performance validation of the MGJO framework utilizing the standardized CEC2017 and CEC2022 evaluation suites. To ensure comprehensive capability assessment, the experimental design incorporates multidimensional testing: CEC2017 functions at dimensional configurations of 30 and 100, complemented by CEC2022 evaluations at 10 and 20 dimensions. Computational outcomes are systematically documented in Tables 4–7, presenting both central tendency (Mean) and variability (Standard Deviation) metrics across optimization runs, where superior results are typographically emphasized. Evolutionary characteristics are further visualized through convergence trajectory analysis in Figure 7.

Table 4. Experimental findings on the CEC 2017 benchmark suite (dim = 30).

Function	Metric	GWO	IWOA	AGPSO	HSO	DBO	BPBO	GJO	MGJO
F1	Ave	2.8587×10^{09}	1.8562×10^{06}	9.5699×10^{08}	8.3879×10^{03}	2.4029×10^{08}	5.1272×10^{08}	1.3244×10^{10}	$\mathbf{4.1805 \times 10^{03}}$
	Std	1.9458×10^{09}	1.1626×10^{06}	1.5751×10^{09}	7.1732×10^{03}	1.4977×10^{08}	2.2170×10^{08}	5.7246×10^{09}	$\mathbf{3.9175 \times 10^{03}}$
F2	Ave	9.3924×10^{32}	2.7939×10^{20}	2.0504×10^{30}	1.3768×10^{17}	2.2866×10^{32}	1.9198×10^{31}	1.7645×10^{35}	$\mathbf{1.2005 \times 10^{14}}$
	Std	4.3273×10^{33}	5.3890×10^{20}	6.3237×10^{30}	6.7570×10^{17}	1.0634×10^{33}	9.3037×10^{31}	4.9362×10^{35}	$\mathbf{4.0515 \times 10^{14}}$
F3	Ave	6.0705×10^{04}	1.5353×10^{05}	8.7115×10^{04}	$\mathbf{4.7849 \times 10^{04}}$	9.3870×10^{04}	6.8247×10^{04}	6.2327×10^{04}	8.4737×10^{04}
	Std	1.1635×10^{04}	3.8715×10^{04}	2.4003×10^{04}	1.1588×10^{04}	1.9516×10^{04}	$\mathbf{8.6188 \times 10^{03}}$	9.8584×10^{03}	1.8095×10^{04}
F4	Ave	6.4657×10^{02}	5.3391×10^{02}	6.3697×10^{02}	6.9766×10^{02}	6.7833×10^{02}	6.8037×10^{02}	1.1891×10^{03}	$\mathbf{5.1346 \times 10^{02}}$
	Std	1.3296×10^{02}	3.3118×10^{01}	1.9760×10^{02}	9.3288×10^{01}	1.3805×10^{02}	6.3945×10^{01}	4.0519×10^{02}	$\mathbf{2.6729 \times 10^{01}}$
F5	Ave	6.2901×10^{02}	7.1825×10^{02}	$\mathbf{5.9782 \times 10^{02}}$	6.9318×10^{02}	7.4236×10^{02}	7.6778×10^{02}	7.3026×10^{02}	6.4444×10^{02}
	Std	2.8424×10^{01}	5.2546×10^{01}	2.5716×10^{01}	2.5869×10^{01}	4.5681×10^{01}	4.2068×10^{01}	5.6271×10^{01}	$\mathbf{2.5330 \times 10^{01}}$
F6	Ave	6.1185×10^{02}	6.5630×10^{02}	6.1126×10^{02}	6.5217×10^{02}	6.5132×10^{02}	6.6112×10^{02}	6.4144×10^{02}	$\mathbf{6.0006 \times 10^{02}}$
	Std	3.8852×10^{00}	1.3225×10^{01}	4.6539×10^{00}	4.9124×10^{00}	1.0821×10^{01}	7.6243×10^{00}	1.1849×10^{01}	$\mathbf{3.9338 \times 10^{-02}}$
F7	Ave	9.0323×10^{02}	1.0665×10^{03}	$\mathbf{8.7635 \times 10^{02}}$	1.0360×10^{03}	1.0412×10^{03}	1.2208×10^{03}	1.0589×10^{03}	8.8558×10^{02}
	Std	5.2116×10^{01}	1.0675×10^{02}	4.6356×10^{01}	7.2858×10^{01}	7.8647×10^{01}	9.1684×10^{01}	5.4250×10^{01}	$\mathbf{4.3810 \times 10^{01}}$
F8	Ave	$\mathbf{9.0326 \times 10^{02}}$	9.9943×10^{02}	9.0543×10^{02}	1.0364×10^{03}	1.0460×10^{03}	1.0096×10^{03}	9.8746×10^{02}	9.0705×10^{02}
	Std	2.6267×10^{01}	3.4959×10^{01}	2.9645×10^{01}	$\mathbf{2.0496 \times 10^{01}}$	5.0985×10^{01}	3.7934×10^{01}	3.7172×10^{01}	2.7901×10^{01}
F9	Ave	2.7554×10^{03}	5.1194×10^{03}	2.6894×10^{03}	2.4603×10^{03}	7.0971×10^{03}	6.9897×10^{03}	5.5220×10^{03}	$\mathbf{2.2982 \times 10^{03}}$
	Std	1.0419×10^{03}	1.0454×10^{03}	1.2678×10^{03}	$\mathbf{7.2450 \times 10^{02}}$	1.8701×10^{03}	2.0065×10^{03}	1.5915×10^{03}	1.2869×10^{03}
F10	Ave	5.8476×10^{03}	5.6942×10^{03}	5.4738×10^{03}	$\mathbf{4.3099 \times 10^{03}}$	6.6552×10^{03}	7.2016×10^{03}	6.1774×10^{03}	4.9462×10^{03}
	Std	1.9893×10^{03}	8.0042×10^{02}	7.2619×10^{02}	$\mathbf{5.4256 \times 10^{02}}$	1.3293×10^{03}	1.2367×10^{03}	1.3516×10^{03}	6.8427×10^{02}
F11	Ave	2.3487×10^{03}	1.3924×10^{03}	$\mathbf{1.3576 \times 10^{03}}$	1.7803×10^{03}	1.9510×10^{03}	1.6641×10^{03}	3.3628×10^{03}	1.4763×10^{03}
	Std	8.7859×10^{02}	$\mathbf{9.3472 \times 10^{01}}$	1.0288×10^{02}	2.0754×10^{02}	7.3794×10^{02}	1.6930×10^{02}	1.4209×10^{03}	4.8927×10^{02}
F12	Ave	8.7820×10^{07}	4.5389×10^{06}	1.2355×10^{08}	$\mathbf{5.5934 \times 10^{05}}$	8.4389×10^{07}	3.7381×10^{07}	7.7143×10^{08}	2.0464×10^{06}
	Std	8.5028×10^{07}	3.9906×10^{06}	3.7092×10^{08}	$\mathbf{6.5551 \times 10^{05}}$	1.8851×10^{08}	2.8545×10^{07}	6.9451×10^{08}	1.4525×10^{06}
F13	Ave	4.8114×10^{07}	4.0409×10^{04}	2.7694×10^{06}	2.7394×10^{04}	2.1502×10^{07}	3.1509×10^{06}	2.8251×10^{08}	$\mathbf{1.0267 \times 10^{04}}$
	Std	9.6117×10^{07}	3.5830×10^{04}	1.3071×10^{07}	1.8549×10^{04}	3.7718×10^{07}	3.5023×10^{06}	3.4263×10^{08}	$\mathbf{1.0813 \times 10^{04}}$
F14	Ave	5.7297×10^{05}	2.2773×10^{05}	7.0773×10^{04}	$\mathbf{1.3045 \times 10^{04}}$	2.1921×10^{05}	7.3482×10^{05}	1.0858×10^{06}	8.1641×10^{05}
	Std	6.0529×10^{05}	1.9910×10^{05}	7.6866×10^{04}	$\mathbf{1.0568 \times 10^{04}}$	3.6042×10^{05}	6.5540×10^{05}	1.0922×10^{06}	9.2234×10^{05}
F15	Ave	1.2636×10^{06}	1.0741×10^{04}	1.5913×10^{04}	$\mathbf{8.7839 \times 10^{03}}$	1.0247×10^{05}	9.8141×10^{04}	1.4573×10^{07}	9.1519×10^{03}
	Std	1.8713×10^{06}	1.2041×10^{04}	1.6669×10^{04}	$\mathbf{3.8447 \times 10^{03}}$	1.1228×10^{05}	1.0401×10^{05}	2.5383×10^{07}	8.9950×10^{03}
F16	Ave	2.6622×10^{03}	2.8848×10^{03}	$\mathbf{2.6586 \times 10^{03}}$	2.9848×10^{03}	3.4392×10^{03}	3.3697×10^{03}	3.1208×10^{03}	2.6913×10^{03}
	Std	$\mathbf{2.7627 \times 10^{02}}$	3.4864×10^{02}	3.0234×10^{02}	3.3070×10^{02}	4.7743×10^{02}	3.6599×10^{02}	2.8909×10^{02}	2.9750×10^{02}
F17	Ave	$\mathbf{2.0425 \times 10^{03}}$	2.5122×10^{03}	2.2104×10^{03}	2.5545×10^{03}	2.6511×10^{03}	2.6071×10^{03}	2.2896×10^{03}	2.1777×10^{03}
	Std	$\mathbf{1.5540 \times 10^{02}}$	2.6725×10^{02}	2.1800×10^{02}	3.2060×10^{02}	3.3119×10^{02}	2.9519×10^{02}	2.5708×10^{02}	2.1467×10^{02}
F18	Ave	2.0248×10^{06}	2.8307×10^{06}	1.6002×10^{06}	$\mathbf{1.2607 \times 10^{05}}$	3.2405×10^{06}	2.4422×10^{06}	3.6235×10^{06}	1.3430×10^{06}
	Std	2.3149×10^{06}	3.1961×10^{06}	1.7270×10^{06}	$\mathbf{8.2646 \times 10^{04}}$	3.8582×10^{06}	2.2565×10^{06}	5.8186×10^{06}	2.0981×10^{06}
F19	Ave	1.1478×10^{06}	$\mathbf{7.5279 \times 10^{03}}$	3.5759×10^{04}	1.4449×10^{04}	3.0782×10^{06}	2.7309×10^{06}	6.4510×10^{06}	9.3737×10^{03}
	Std	1.3803×10^{06}	$\mathbf{1.0003 \times 10^{04}}$	7.9795×10^{04}	1.7084×10^{04}	7.2016×10^{06}	3.3243×10^{06}	1.1668×10^{07}	1.1237×10^{04}
F20	Ave	2.4838×10^{03}	2.7435×10^{03}	2.5174×10^{03}	2.5551×10^{03}	2.7188×10^{03}	2.7472×10^{03}	2.6782×10^{03}	$\mathbf{2.3669 \times 10^{03}}$
	Std	$\mathbf{1.6592 \times 10^{02}}$	2.4968×10^{02}	1.9336×10^{02}	1.8983×10^{02}	2.4175×10^{02}	1.9688×10^{02}	2.0408×10^{02}	1.6869×10^{02}
F21	Ave	$\mathbf{2.4030 \times 10^{03}}$	2.5003×10^{03}	2.4104×10^{03}	2.5667×10^{03}	2.5519×10^{03}	2.4997×10^{03}	2.4999×10^{03}	2.4172×10^{03}
	Std	2.0699×10^{01}	4.3511×10^{01}	3.0560×10^{01}	$\mathbf{1.6294 \times 10^{01}}$	5.1257×10^{01}	4.3093×10^{01}	5.1069×10^{01}	3.4323×10^{01}
F22	Ave	4.8809×10^{03}	4.2986×10^{03}	5.0618×10^{03}	4.8193×10^{03}	4.9684×10^{03}	$\mathbf{3.6110 \times 10^{03}}$	6.1869×10^{03}	3.8435×10^{03}
	Std	1.8781×10^{03}	2.3464×10^{03}	2.1159×10^{03}	$\mathbf{1.5939 \times 10^{03}}$	2.4981×10^{03}	2.2249×10^{03}	2.3096×10^{03}	2.1135×10^{03}
F23	Ave	2.7842×10^{03}	2.8797×10^{03}	2.8785×10^{03}	2.9091×10^{03}	3.0026×10^{03}	2.9595×10^{03}	2.9265×10^{03}	$\mathbf{2.7787 \times 10^{03}}$
	Std	4.1240×10^{01}	7.4256×10^{01}	8.4955×10^{01}	$\mathbf{1.6186 \times 10^{01}}$	1.2427×10^{02}	6.1837×10^{01}	6.7901×10^{01}	4.1438×10^{01}
F24	Ave	2.9662×10^{03}	3.0308×10^{03}	3.1053×10^{03}	3.0433×10^{03}	3.1998×10^{03}	3.0512×10^{03}	3.1054×10^{03}	$\mathbf{2.9615 \times 10^{03}}$
	Std	6.0541×10^{01}	7.6090×10^{01}	9.1501×10^{01}	$\mathbf{1.2406 \times 10^{01}}$	9.8925×10^{01}	5.8957×10^{01}	6.5715×10^{01}	5.2031×10^{01}
F25	Ave	3.0391×10^{03}	2.9138×10^{03}	2.9395×10^{03}	3.1840×10^{03}	2.9761×10^{03}	3.0795×10^{03}	3.2267×10^{03}	$\mathbf{2.9044 \times 10^{03}}$
	Std	1.2102×10^{02}	$\mathbf{1.6781 \times 10^{01}}$	4.6393×10^{01}	8.6748×10^{01}	4.1779×10^{01}	4.5472×10^{01}	1.2932×10^{02}	2.2886×10^{01}
F26	Ave	5.0335×10^{03}	5.6594×10^{03}	5.1083×10^{03}	5.2881×10^{03}	7.2851×10^{03}	6.8372×10^{03}	6.1636×10^{03}	$\mathbf{4.6601 \times 10^{03}}$
	Std	5.1045×10^{02}	1.4276×10^{03}	8.4084×10^{02}	$\mathbf{2.6234 \times 10^{02}}$	9.1932×10^{02}	1.8118×10^{03}	7.8118×10^{02}	9.7846×10^{02}

Table 4. *Cont.*

Function	Metric	GWO	IWOA	AGPSO	HSO	DBO	BPBO	GJO	MGJO
F27	Ave	3.2728×10^{03}	3.4502×10^{03}	3.2774×10^{03}	3.3577×10^{03}	3.3446×10^{03}	3.4334×10^{03}	3.3911×10^{03}	$\mathbf{3.2336 \times 10^{03}}$
	Std	3.6129×10^{01}	1.4546×10^{02}	3.4694×10^{01}	7.9813×10^{01}	5.5498×10^{01}	9.6239×10^{01}	7.9391×10^{01}	$\mathbf{1.1126 \times 10^{01}}$
F28	Ave	3.4582×10^{03}	4.4756×10^{03}	3.6015×10^{03}	3.7861×10^{03}	3.5739×10^{03}	3.4549×10^{03}	3.7787×10^{03}	$\mathbf{3.2607 \times 10^{03}}$
	Std	1.4650×10^{02}	1.0548×10^{03}	8.4307×10^{02}	5.6812×10^{02}	3.9794×10^{02}	7.9759×10^{01}	2.3915×10^{02}	$\mathbf{2.4565 \times 10^{01}}$
F29	Ave	3.9170×10^{03}	4.2740×10^{03}	3.9372×10^{03}	4.4687×10^{03}	4.3476×10^{03}	4.9087×10^{03}	4.3690×10^{03}	$\mathbf{3.7215 \times 10^{03}}$
	Std	$\mathbf{1.6033 \times 10^{02}}$	2.4192×10^{02}	2.2151×10^{02}	2.3190×10^{02}	3.8157×10^{02}	3.5212×10^{02}	3.2438×10^{02}	2.1116×10^{02}
F30	Ave	1.3933×10^{07}	1.2169×10^{05}	4.0847×10^{05}	9.5764×10^{04}	5.7551×10^{06}	1.3332×10^{07}	3.8595×10^{07}	$\mathbf{1.1595 \times 10^{04}}$
	Std	1.3980×10^{07}	2.1137×10^{05}	1.2051×10^{06}	1.1256×10^{05}	7.2456×10^{06}	8.4294×10^{06}	3.1914×10^{07}	$\mathbf{3.9623 \times 10^{03}}$

Table 5. Experimental findings on the CEC 2017 benchmark suite (dim = 100).

Function	Metric	GWO	IWOA	AGPSO	HSO	DBO	BPBO	GJO	MGJO
F1	Ave	5.3510×10^{10}	8.9745×10^{09}	2.7509×10^{10}	3.2404×10^{09}	8.5312×10^{10}	5.2731×10^{10}	1.3279×10^{11}	$\mathbf{2.5979 \times 10^{08}}$
	Std	8.7034×10^{09}	2.6775×10^{09}	1.0784×10^{10}	1.6339×10^{09}	7.0504×10^{10}	8.8060×10^{09}	1.4218×10^{10}	$\mathbf{1.1627 \times 10^{08}}$
F2	Ave	1.2050×10^{135}	1.2511×10^{133}	3.5868×10^{139}	3.1190×10^{201}	6.1137×10^{163}	9.3962×10^{154}	3.1194×10^{149}	$\mathbf{1.3655 \times 10^{101}}$
	Std	6.5262×10^{135}	6.1838×10^{133}	1.9620×10^{140}	$\mathbf{6.5535 \times 10^{04}}$	$\mathbf{6.5535 \times 10^{04}}$	$\mathbf{6.5535 \times 10^{04}}$	1.4435×10^{150}	5.1024×10^{101}
F3	Ave	5.2603×10^{05}	9.4038×10^{05}	7.0265×10^{05}	$\mathbf{3.4666 \times 10^{05}}$	6.1803×10^{05}	3.5749×10^{05}	3.8842×10^{05}	5.1192×10^{05}
	Std	8.9976×10^{04}	1.0343×10^{05}	9.5974×10^{04}	3.4786×10^{04}	2.4925×10^{05}	$\mathbf{3.0824 \times 10^{04}}$	3.5893×10^{04}	6.3321×10^{04}
F4	Ave	6.4441×10^{03}	1.9157×10^{03}	5.9269×10^{03}	3.0318×10^{03}	1.8998×10^{04}	7.4066×10^{03}	2.0314×10^{04}	$\mathbf{1.1419 \times 10^{03}}$
	Std	1.8268×10^{03}	3.1130×10^{02}	2.9983×10^{03}	8.8704×10^{02}	1.4422×10^{04}	1.6252×10^{03}	4.7254×10^{03}	$\mathbf{8.5415 \times 10^{01}}$
F5	Ave	$\mathbf{1.2442 \times 10^{03}}$	1.4804×10^{03}	1.3223×10^{03}	1.6868×10^{03}	1.6858×10^{03}	1.7620×10^{03}	1.5917×10^{03}	1.3195×10^{03}
	Std	7.7468×10^{01}	$\mathbf{6.0703 \times 10^{01}}$	9.1677×10^{01}	8.7948×10^{01}	2.0586×10^{02}	6.7735×10^{01}	1.1573×10^{02}	1.5512×10^{02}
F6	Ave	$\mathbf{6.4522 \times 10^{02}}$	6.7360×10^{02}	6.5670×10^{02}	6.9061×10^{02}	6.8058×10^{02}	6.9312×10^{02}	6.7476×10^{02}	6.5154×10^{02}
	Std	5.0389×10^{00}	$\mathbf{3.2847 \times 10^{00}}$	7.7674×10^{00}	5.1499×10^{00}	1.1371×10^{01}	4.0328×10^{00}	5.9583×10^{00}	7.1434×10^{00}
F7	Ave	$\mathbf{2.1978 \times 10^{03}}$	3.0995×10^{03}	3.1326×10^{03}	4.8226×10^{03}	3.0094×10^{03}	3.7014×10^{03}	2.9671×10^{03}	2.3527×10^{03}
	Std	1.3159×10^{02}	2.4569×10^{02}	2.6520×10^{02}	6.5307×10^{02}	2.0699×10^{02}	$\mathbf{9.2563 \times 10^{01}}$	1.2136×10^{02}	3.0908×10^{02}
F8	Ave	$\mathbf{1.5856 \times 10^{03}}$	1.9067×10^{03}	1.6318×10^{03}	2.0420×10^{03}	2.1809×10^{03}	2.2433×10^{03}	1.9478×10^{03}	1.6619×10^{03}
	Std	1.4423×10^{02}	$\mathbf{6.8833 \times 10^{01}}$	1.0815×10^{02}	7.4633×10^{01}	2.3425×10^{02}	7.0908×10^{01}	1.0835×10^{02}	1.8880×10^{02}
F9	Ave	4.5865×10^{04}	$\mathbf{3.5318 \times 10^{04}}$	3.8231×10^{04}	6.3337×10^{04}	7.6876×10^{04}	7.0544×10^{04}	6.3521×10^{04}	4.3473×10^{04}
	Std	1.3947×10^{04}	3.8803×10^{03}	1.0973×10^{04}	1.6184×10^{04}	8.5930×10^{03}	1.0140×10^{04}	9.0921×10^{03}	$\mathbf{3.6714 \times 10^{03}}$
F10	Ave	$\mathbf{1.9070 \times 10^{04}}$	1.9403×10^{04}	2.4635×10^{04}	2.3683×10^{04}	2.8480×10^{04}	2.7304×10^{04}	2.6966×10^{04}	2.2392×10^{04}
	Std	4.1712×10^{03}	$\mathbf{1.4057 \times 10^{03}}$	2.8272×10^{03}	1.8427×10^{03}	4.3763×10^{03}	2.5923×10^{03}	4.9114×10^{03}	1.8762×10^{03}
F11	Ave	9.2253×10^{04}	1.5751×10^{05}	1.0379×10^{05}	$\mathbf{5.7994 \times 10^{04}}$	2.4113×10^{05}	1.4358×10^{05}	1.0863×10^{05}	8.3907×10^{04}
	Std	$\mathbf{1.7735 \times 10^{04}}$	5.3303×10^{04}	3.6954×10^{04}	2.3412×10^{04}	6.8023×10^{04}	2.8041×10^{04}	2.0842×10^{04}	2.3945×10^{04}
F12	Ave	1.3411×10^{10}	8.7825×10^{08}	9.9960×10^{09}	1.6939×10^{08}	6.5324×10^{09}	7.0752×10^{09}	4.7622×10^{10}	$\mathbf{1.2354 \times 10^{08}}$
	Std	5.3199×10^{09}	3.4747×10^{08}	6.2775×10^{09}	1.0477×10^{08}	2.1057×10^{09}	2.0407×10^{09}	1.0009×10^{10}	$\mathbf{3.4286 \times 10^{07}}$
F13	Ave	1.4569×10^{09}	7.6721×10^{05}	1.2596×10^{09}	7.4976×10^{04}	3.4423×10^{08}	2.6518×10^{08}	9.4564×10^{09}	$\mathbf{1.0271 \times 10^{04}}$
	Std	9.0110×10^{08}	4.5610×10^{05}	1.6198×10^{09}	3.4890×10^{04}	2.1799×10^{08}	1.1801×10^{08}	3.9091×10^{09}	$\mathbf{6.3155 \times 10^{03}}$
F14	Ave	9.0447×10^{06}	4.2857×10^{06}	7.9435×10^{06}	$\mathbf{9.7047 \times 10^{05}}$	1.6980×10^{07}	1.4402×10^{07}	1.7311×10^{07}	6.9310×10^{06}
	Std	5.5295×10^{06}	1.6898×10^{06}	6.7914×10^{06}	$\mathbf{6.9679 \times 10^{05}}$	1.0376×10^{07}	5.8969×10^{06}	1.1189×10^{07}	2.6292×10^{06}
F15	Ave	3.6823×10^{08}	7.5246×10^{04}	1.5117×10^{08}	2.9811×10^{04}	6.8445×10^{07}	2.5749×10^{07}	3.2105×10^{09}	$\mathbf{6.3068 \times 10^{03}}$
	Std	5.6759×10^{08}	3.6070×10^{04}	3.4836×10^{08}	$\mathbf{1.0704 \times 10^{04}}$	8.9388×10^{07}	1.7288×10^{07}	2.4150×10^{09}	1.1720×10^{04}
F16	Ave	6.5838×10^{03}	6.9150×10^{03}	7.3807×10^{03}	7.2401×10^{03}	9.3081×10^{03}	1.1244×10^{04}	9.9269×10^{03}	$\mathbf{5.7588 \times 10^{03}}$
	Std	$\mathbf{5.4314 \times 10^{02}}$	7.6111×10^{02}	1.0740×10^{03}	9.2612×10^{02}	1.4833×10^{03}	1.2453×10^{03}	1.2241×10^{03}	7.7678×10^{02}
F17	Ave	5.5501×10^{03}	6.3968×10^{03}	7.5824×10^{03}	5.6223×10^{03}	9.1235×10^{03}	7.9666×10^{03}	3.1351×10^{04}	$\mathbf{4.9930 \times 10^{03}}$
	Std	9.6950×10^{02}	6.1720×10^{02}	3.1625×10^{03}	$\mathbf{4.2981 \times 10^{02}}$	1.4514×10^{03}	7.8680×10^{02}	5.1530×10^{04}	6.4923×10^{02}
F18	Ave	9.4181×10^{06}	7.1481×10^{06}	1.0695×10^{07}	$\mathbf{2.2247 \times 10^{06}}$	2.7972×10^{07}	1.5441×10^{07}	1.9089×10^{07}	6.1371×10^{06}
	Std	5.2724×10^{06}	3.5548×10^{06}	6.7932×10^{06}	$\mathbf{1.7877 \times 10^{06}}$	1.5624×10^{07}	6.1604×10^{06}	1.0315×10^{07}	2.9256×10^{06}
F19	Ave	3.0200×10^{08}	5.9721×10^{05}	2.1440×10^{08}	2.0822×10^{04}	7.3010×10^{07}	3.9518×10^{07}	2.5227×10^{09}	$\mathbf{5.1664 \times 10^{03}}$
	Std	4.4671×10^{08}	5.8884×10^{05}	5.7456×10^{08}	2.0347×10^{04}	7.9364×10^{07}	2.7621×10^{07}	1.8096×10^{09}	$\mathbf{2.6068 \times 10^{03}}$
F20	Ave	5.6596×10^{03}	5.9349×10^{03}	6.3321×10^{03}	$\mathbf{4.8819 \times 10^{03}}$	7.3061×10^{03}	6.4297×10^{03}	6.4377×10^{03}	5.8606×10^{03}
	Std	1.0939×10^{03}	5.0468×10^{02}	6.4152×10^{02}	$\mathbf{4.8240 \times 10^{02}}$	7.5312×10^{02}	6.4935×10^{02}	1.0003×10^{03}	1.0923×10^{03}
F21	Ave	$\mathbf{3.0936 \times 10^{03}}$	3.6431×10^{03}	3.3348×10^{03}	3.7133×10^{03}	4.0461×10^{03}	3.9040×10^{03}	3.5464×10^{03}	3.2024×10^{03}
	Std	$\mathbf{6.6627 \times 10^{01}}$	1.8877×10^{02}	1.2260×10^{02}	7.0588×10^{01}	1.4972×10^{02}	1.6236×10^{02}	1.2394×10^{02}	2.1602×10^{02}
F22	Ave	2.2917×10^{04}	$\mathbf{2.2472 \times 10^{04}}$	2.6821×10^{04}	2.5741×10^{04}	2.9813×10^{04}	3.0436×10^{04}	2.9424×10^{04}	2.4632×10^{04}
	Std	5.2026×10^{03}	$\mathbf{1.5296 \times 10^{03}}$	2.8580×10^{03}	1.7428×10^{03}	4.6555×10^{03}	3.0252×10^{03}	4.2920×10^{03}	2.7559×10^{03}
F23	Ave	3.7025×10^{03}	4.1565×10^{03}	4.5492×10^{03}	4.0114×10^{03}	4.8542×10^{03}	4.5816×10^{03}	4.5843×10^{03}	$\mathbf{3.4829 \times 10^{03}}$
	Std	6.9705×10^{01}	1.8345×10^{02}	2.5633×10^{02}	$\mathbf{5.1409 \times 10^{01}}$	1.8344×10^{02}	2.3162×10^{02}	2.0724×10^{02}	2.0993×10^{02}
F24	Ave	4.4320×10^{03}	4.9823×10^{03}	6.6914×10^{03}	4.6083×10^{03}	6.1674×10^{03}	5.5866×10^{03}	6.0391×10^{03}	$\mathbf{4.0212 \times 10^{03}}$
	Std	1.7256×10^{02}	2.7333×10^{02}	5.7435×10^{02}	$\mathbf{6.4584 \times 10^{01}}$	4.1475×10^{02}	2.6685×10^{02}	4.4634×10^{02}	1.6803×10^{02}
F25	Ave	7.1507×10^{03}	4.4340×10^{03}	6.1216×10^{03}	6.5600×10^{03}	8.8138×10^{03}	7.8050×10^{03}	1.2755×10^{04}	$\mathbf{3.8202 \times 10^{03}}$
	Std	1.1525×10^{03}	1.9410×10^{02}	9.5531×10^{02}	7.6190×10^{02}	5.2496×10^{03}	7.1229×10^{02}	1.7622×10^{03}	$\mathbf{7.2848 \times 10^{01}}$
F26	Ave	1.7659×10^{04}	2.2212×10^{04}	2.5527×10^{04}	1.9129×10^{04}	2.6686×10^{04}	3.2665×10^{04}	2.8358×10^{04}	$\mathbf{1.6988 \times 10^{04}}$
	Std	1.4143×10^{03}	4.2171×10^{03}	4.5872×10^{03}	$\mathbf{1.0966 \times 10^{03}}$	3.4511×10^{03}	3.9829×10^{03}	2.0930×10^{03}	2.2942×10^{03}
F27	Ave	4.3566×10^{03}	6.6007×10^{03}	4.7403×10^{03}	4.2438×10^{03}	4.6931×10^{03}	5.6618×10^{03}	5.8454×10^{03}	$\mathbf{3.6943 \times 10^{03}}$
	Std	1.9124×10^{02}	1.5056×10^{03}	6.2080×10^{02}	1.9753×10^{02}	3.8149×10^{02}	6.2630×10^{02}	6.0483×10^{02}	$\mathbf{9.3333 \times 10^{01}}$
F28	Ave	9.5038×10^{03}	1.7317×10^{04}	1.2574×10^{04}	1.5498×10^{04}	1.8865×10^{04}	1.1127×10^{04}	1.6434×10^{04}	$\mathbf{4.0417 \times 10^{03}}$
	Std	1.2968×10^{03}	6.9142×10^{03}	3.9998×10^{03}	4.5155×10^{03}	5.4083×10^{03}	1.1913×10^{03}	2.1791×10^{03}	$\mathbf{1.3769 \times 10^{02}}$
F29	Ave	9.4609×10^{03}	8.4494×10^{03}	9.2722×10^{03}	9.5580×10^{03}	1.2496×10^{04}	1.4890×10^{04}	1.5871×10^{04}	$\mathbf{6.7427 \times 10^{03}}$
	Std	8.2399×10^{02}	7.6366×10^{02}	8.5661×10^{02}	7.1012×10^{02}	3.0845×10^{03}	1.4930×10^{03}	4.8154×10^{03}	$\mathbf{5.1538 \times 10^{02}}$
F30	Ave	1.1000×10^{09}	1.2532×10^{07}	7.6521×10^{08}	1.2125×10^{06}	2.6261×10^{08}	6.6309×10^{08}	7.6570×10^{09}	$\mathbf{2.3546 \times 10^{05}}$
	Std	7.4027×10^{08}	8.9771×10^{06}	9.5393×10^{08}	1.0580×10^{06}	2.3539×10^{08}	3.1762×10^{08}	3.4330×10^{09}	$\mathbf{1.2030 \times 10^{05}}$

Table 6. Experimental findings on the CEC 2022 benchmark suite (dim = 10).

Function	Metric	GWO	IWOA	AGPSO	HSO	DBO	BPBO	GJO	MGJO
F1	Ave	3.5415×10^{03}	5.8385×10^{02}	$\mathbf{3.0011 \times 10^{02}}$	1.3698×10^{03}	1.9787×10^{03}	1.0500×10^{03}	3.3052×10^{03}	1.2035×10^{03}
	Std	2.8924×10^{03}	2.5780×10^{02}	$\mathbf{4.8818 \times 10^{-01}}$	2.0562×10^{02}	3.2939×10^{03}	7.1886×10^{02}	2.3852×10^{03}	9.4838×10^{02}
F2	Ave	4.3341×10^{02}	4.1967×10^{02}	4.3021×10^{02}	4.4484×10^{02}	4.3446×10^{02}	4.3632×10^{02}	4.5063×10^{02}	$\mathbf{4.1308 \times 10^{02}}$
	Std	$\mathbf{2.2603 \times 10^{01}}$	2.7659×10^{01}	3.2172×10^{01}	2.9376×10^{01}	3.6068×10^{01}	3.1836×10^{01}	2.7100×10^{01}	2.3071×10^{01}
F3	Ave	6.0165×10^{02}	6.1014×10^{02}	6.0010×10^{02}	6.1898×10^{02}	6.1035×10^{02}	6.2235×10^{02}	6.0927×10^{02}	$\mathbf{6.0000 \times 10^{02}}$
	Std	2.0177×10^{00}	6.9742×10^{00}	3.8896×10^{-01}	4.1618×10^{00}	7.1251×10^{00}	9.9230×10^{00}	5.4324×10^{00}	$\mathbf{1.8248 \times 10^{-07}}$
F4	Ave	8.1728×10^{02}	8.3292×10^{02}	$\mathbf{8.1462 \times 10^{02}}$	8.3944×10^{02}	8.3566×10^{02}	8.2214×10^{02}	8.3139×10^{02}	8.1481×10^{02}
	Std	8.1755×10^{00}	1.2004×10^{01}	7.3628×10^{00}	$\mathbf{4.5694 \times 10^{00}}$	1.2153×10^{01}	7.8466×10^{00}	9.3321×10^{00}	7.3113×10^{00}
F5	Ave	9.3258×10^{02}	1.0575×10^{03}	9.0181×10^{02}	9.1048×10^{02}	9.7112×10^{02}	9.9164×10^{02}	1.0114×10^{03}	$\mathbf{9.0166 \times 10^{02}}$
	Std	6.0653×10^{01}	1.9366×10^{02}	$\mathbf{2.8773 \times 10^{00}}$	9.5826×10^{00}	7.5771×10^{01}	8.6615×10^{01}	1.1416×10^{02}	2.9296×10^{00}
F6	Ave	5.9301×10^{03}	$\mathbf{2.4035 \times 10^{03}}$	4.8602×10^{03}	2.8979×10^{03}	5.9127×10^{03}	4.1557×10^{03}	1.0893×10^{04}	3.1694×10^{03}
	Std	2.4751×10^{03}	$\mathbf{9.7532 \times 10^{02}}$	2.2146×10^{03}	1.4517×10^{03}	2.4702×10^{03}	2.2729×10^{03}	5.3408×10^{03}	1.8218×10^{03}
F7	Ave	2.0330×10^{03}	2.0365×10^{03}	2.0190×10^{03}	2.0781×10^{03}	2.0393×10^{03}	2.0603×10^{03}	2.0462×10^{03}	$\mathbf{2.0098 \times 10^{03}}$
	Std	1.4634×10^{01}	1.6189×10^{01}	$\mathbf{6.6032 \times 10^{00}}$	2.9937×10^{01}	2.1218×10^{01}	1.8419×10^{01}	2.4871×10^{01}	9.7780×10^{00}
F8	Ave	2.2280×10^{03}	$\mathbf{2.2195 \times 10^{03}}$	2.2229×10^{03}	2.2790×10^{03}	2.2341×10^{03}	2.2335×10^{03}	2.2282×10^{03}	2.2260×10^{03}
	Std	2.2821×10^{01}	9.5001×10^{00}	5.0381×10^{00}	6.7574×10^{01}	3.3128×10^{01}	2.0924×10^{01}	$\mathbf{3.1546 \times 10^{00}}$	4.8810×10^{00}
F9	Ave	2.5692×10^{03}	2.5487×10^{03}	2.5342×10^{03}	2.6927×10^{03}	2.5552×10^{03}	2.5749×10^{03}	2.6032×10^{03}	$\mathbf{2.5293 \times 10^{03}}$
	Std	3.7133×10^{01}	4.3650×10^{01}	2.6816×10^{01}	4.4243×10^{01}	4.3793×10^{01}	3.7994×10^{01}	4.1553×10^{01}	$\mathbf{1.7324 \times 10^{-01}}$
F10	Ave	2.5867×10^{03}	2.5489×10^{03}	2.5570×10^{03}	2.6071×10^{03}	2.5547×10^{03}	2.5766×10^{03}	2.5887×10^{03}	$\mathbf{2.5008 \times 10^{03}}$
	Std	6.2669×10^{01}	5.6479×10^{01}	6.1452×10^{01}	1.1611×10^{02}	6.6522×10^{01}	6.3166×10^{01}	6.3634×10^{01}	$\mathbf{1.8645 \times 10^{-01}}$
F11	Ave	2.8748×10^{03}	2.7401×10^{03}	2.8376×10^{03}	2.9730×10^{03}	2.7700×10^{03}	2.7744×10^{03}	2.9859×10^{03}	$\mathbf{2.6839 \times 10^{03}}$
	Std	2.2466×10^{02}	1.7449×10^{02}	1.8726×10^{02}	2.0572×10^{02}	1.2654×10^{02}	1.7069×10^{02}	2.3982×10^{02}	$\mathbf{9.6289 \times 10^{01}}$
F12	Ave	2.8695×10^{03}	2.8717×10^{03}	2.8712×10^{03}	2.8669×10^{03}	2.8772×10^{03}	2.8695×10^{03}	2.8766×10^{03}	$\mathbf{2.8650 \times 10^{03}}$
	Std	1.1415×10^{01}	$\mathbf{2.0568 \times 10^{00}}$	1.7011×10^{01}	1.0262×10^{01}	2.3393×10^{01}	1.2292×10^{01}	1.5871×10^{01}	1.0730×10^{01}

Table 7. Experimental findings on the CEC 2022 benchmark suite (dim = 20).

Function	Metric	GWO	IWOA	AGPSO	HSO	DBO	BPBO	GJO	MGJO
F1	Ave	1.6530×10^{04}	2.0859×10^{04}	1.2046×10^{04}	$\mathbf{7.8069 \times 10^{03}}$	3.4305×10^{04}	2.3080×10^{04}	1.6040×10^{04}	2.1086×10^{04}
	Std	5.3662×10^{03}	6.8295×10^{03}	7.9329×10^{03}	$\mathbf{4.4438 \times 10^{03}}$	1.0268×10^{04}	8.1011×10^{03}	5.4810×10^{03}	9.1187×10^{03}
F2	Ave	4.9755×10^{02}	$\mathbf{4.6415 \times 10^{02}}$	4.7620×10^{02}	5.5925×10^{02}	4.9042×10^{02}	5.6247×10^{02}	6.1581×10^{02}	4.6682×10^{02}
	Std	3.9461×10^{01}	$\mathbf{1.7315 \times 10^{01}}$	4.3064×10^{01}	6.3565×10^{01}	5.1359×10^{01}	5.3795×10^{01}	7.3198×10^{01}	2.6211×10^{01}
F3	Ave	6.0711×10^{02}	6.4070×10^{02}	6.0446×10^{02}	6.3690×10^{02}	6.3486×10^{02}	6.4817×10^{02}	6.3112×10^{02}	$\mathbf{6.0000 \times 10^{02}}$
	Std	4.4861×10^{00}	1.8949×10^{01}	3.4282×10^{00}	5.2175×10^{00}	9.3797×10^{00}	1.3872×10^{01}	8.6800×10^{00}	$\mathbf{8.3378 \times 10^{-04}}$
F4	Ave	8.6034×10^{02}	8.8687×10^{02}	$\mathbf{8.5159 \times 10^{02}}$	9.1771×10^{02}	9.0819×10^{02}	8.8617×10^{02}	9.0381×10^{02}	8.5679×10^{02}
	Std	3.0605×10^{01}	2.2540×10^{01}	1.8976×10^{01}	$\mathbf{1.1865 \times 10^{01}}$	3.0499×10^{01}	1.2952×10^{01}	3.1246×10^{01}	1.8707×10^{01}
F5	Ave	1.2909×10^{03}	2.3854×10^{03}	$\mathbf{1.1391 \times 10^{03}}$	1.1600×10^{03}	2.1677×10^{03}	2.6117×10^{03}	1.8898×10^{03}	1.2240×10^{03}
	Std	3.2993×10^{02}	4.9386×10^{02}	$\mathbf{1.1466 \times 10^{02}}$	2.7274×10^{02}	6.0993×10^{02}	5.8900×10^{02}	4.9049×10^{02}	3.5288×10^{02}
F6	Ave	2.9536×10^{06}	7.1596×10^{03}	2.0715×10^{05}	$\mathbf{4.7155 \times 10^{03}}$	7.2529×10^{05}	1.5692×10^{05}	1.9905×10^{07}	5.3478×10^{03}
	Std	1.2837×10^{07}	6.5420×10^{03}	5.1192×10^{05}	$\mathbf{3.2456 \times 10^{03}}$	1.6150×10^{06}	2.8491×10^{05}	3.2789×10^{07}	3.6362×10^{03}
F7	Ave	2.1033×10^{03}	2.1340×10^{03}	2.0673×10^{03}	2.1410×10^{03}	2.1268×10^{03}	2.1661×10^{03}	2.1212×10^{03}	$\mathbf{2.0613 \times 10^{03}}$
	Std	5.1832×10^{01}	4.1502×10^{01}	3.6736×10^{01}	4.9555×10^{01}	4.6537×10^{01}	4.5945×10^{01}	4.1601×10^{01}	$\mathbf{3.1778 \times 10^{01}}$
F8	Ave	2.2723×10^{03}	2.2514×10^{03}	2.2457×10^{03}	2.4639×10^{03}	2.3381×10^{03}	2.3025×10^{03}	2.2658×10^{03}	$\mathbf{2.2389 \times 10^{03}}$
	Std	5.7417×10^{01}	3.8673×10^{01}	$\mathbf{3.5503 \times 10^{01}}$	1.2185×10^{02}	7.0948×10^{01}	8.0681×10^{01}	5.6463×10^{01}	3.8289×10^{01}
F9	Ave	2.5315×10^{03}	$\mathbf{2.4809 \times 10^{03}}$	2.4970×10^{03}	2.6989×10^{03}	2.5141×10^{03}	2.5216×10^{03}	2.5919×10^{03}	2.4866×10^{03}
	Std	3.7290×10^{01}	$\mathbf{1.2906 \times 10^{-01}}$	2.4913×10^{01}	7.6430×10^{01}	3.2234×10^{01}	3.0870×10^{01}	4.6168×10^{01}	2.6549×10^{00}
F10	Ave	3.5982×10^{03}	2.6964×10^{03}	2.9915×10^{03}	3.7925×10^{03}	3.6767×10^{03}	4.0409×10^{03}	4.0490×10^{03}	$\mathbf{2.5828 \times 10^{03}}$
	Std	1.0419×10^{03}	5.5138×10^{02}	5.8868×10^{02}	6.6111×10^{02}	1.3246×10^{03}	1.2880×10^{03}	1.5033×10^{03}	$\mathbf{1.6528 \times 10^{02}}$
F11	Ave	3.5272×10^{03}	2.9012×10^{03}	3.4571×10^{03}	3.5842×10^{03}	3.1253×10^{03}	3.2194×10^{03}	4.4994×10^{03}	$\mathbf{2.8934 \times 10^{03}}$
	Std	2.6310×10^{02}	1.0844×10^{02}	4.7210×10^{02}	2.8166×10^{02}	1.5453×10^{02}	1.6347×10^{02}	6.3837×10^{02}	$\mathbf{1.0808 \times 10^{02}}$
F12	Ave	2.9824×10^{03}	3.0454×10^{03}	3.0078×10^{03}	3.0052×10^{03}	3.0268×10^{03}	3.0509×10^{03}	3.0278×10^{03}	$\mathbf{2.9679 \times 10^{03}}$
	Std	2.7524×10^{01}	$\mathbf{1.6390 \times 10^{01}}$	5.0636×10^{01}	3.8326×10^{01}	6.9680×10^{01}	5.1204×10^{01}	5.9206×10^{01}	2.3602×10^{01}

In the CEC2017 benchmark suite (dim = 30, Table 4), MGJO demonstrates optimal performance on most functions. For F1, MGJO achieves an Ave value of 4.1805×10^{03}, far lower than the second-best algorithm HSO (8.3879×10^{03}) and standard GJO (1.3244×10^{10}), with a Std of 3.9175×10^{03}, only 0.54 times that of HSO, demonstrating high precision and stability. On F2, MGJO attains an Ave of 1.2005×10^{14}, which is more than 20 orders of magnitude lower than standard GJO (1.7645×10^{35}), and even compared with HSO (1.3768×10^{17}), it is still 3 orders of magnitude lower. For complex multimodal functions such as F9, F12, and F30, MGJO also maintains superiority; for instance, the Ave of F30 is 1.1595×10^{04}, only 0.03% of standard GJO (3.8595×10^{07}), with a Std of 3.9623×10^{03}, much lower than other comparison algorithms, verifying its robustness in high-dimensional complex problems.

In CEC2017 (dim = 100, Table 5), with increasing dimensionality, MGJO's advantages become even more significant. For F1, the Ave is 2.5979×10^{08}, only 0.2% of standard GJO (1.3279×10^{11}), and one order of magnitude lower than HSO (3.2404×10^{09}). On F4, the Ave

is 1.1419×10^{03}, 40.4% lower than IWOA (1.9157×10^{03}), and the Std is 8.5415×10^{01}, only 27.5% of IWOA. For functions such as F15 and F19, MGJO maintains the lowest Ave and Std values, proving its efficient optimization capability even in ultra-high-dimensional problems.

In the CEC2022 benchmark suite (dim = 10, 20, Tables 6 and 7), MGJO still exhibits outstanding performance. For dim = 10, F3 reaches the theoretical optimum with an Ave of 6.0000×10^{02} and a Std of 8.3378×10^{-04}, far better than comparison algorithms such as GWO (6.0165×10^{02}). For F5, Ave is 9.0166×10^{02}, lower than AGPSO (9.0181×10^{02}), and Std is 2.9296×10^{00}, only 1.5% of IWOA (1.9366×10^{02}). For dim = 20, F3 also reaches the theoretical optimum (Ave 6.0000×10^{02}), while F10 achieves an Ave of 2.5828×10^{03}, 4.2% lower than IWOA (2.6964×10^{03}), with a Std of 1.6528×10^{02}, the lowest among all comparison algorithms, demonstrating its generalization across different benchmarks.

Figure 7 shows the convergence curves of MGJO and comparison algorithms on representative functions from CEC2017 (dim = 30, 100) and CEC2022 (dim = 10, 20), visually reflecting the optimization speed and final accuracy. On CEC2017 (dim = 30), MGJO converges fastest and reaches the highest final precision. For F1, the fitness value drops below 1.0×10^{04} within 100 iterations, while standard GJO remains above 1.0×10^{10}; after 500 iterations, MGJO outperforms GJO by more than 2900 times. For F30, after 200 iterations, the convergence curve stabilizes, with a final fitness of 1.1595×10^{04}, 87.9% lower than HSO (9.5764×10^{04}). In CEC2017 (dim = 100), for F1 and F12, MGJO shows the steepest decline in the first 50 iterations; for example, F12 reaches below 1.0×10^{08} at iteration 50, while GJO remains at 4.7622×10^{10}. Later-stage convergence is smooth, demonstrating fast convergence without premature stagnation. On CEC2022 (dim = 10, 20), MGJO maintains advantages: for dim = 10, F3 reaches the theoretical optimum within 100 iterations and remains stable, whereas GWO requires more than 300 iterations and fails to achieve the optimum; for dim = 20, F10 reaches 2.5828×10^{03} after 300 iterations, 36.1% lower than BPBO (4.0409×10^{03}), with stable convergence.

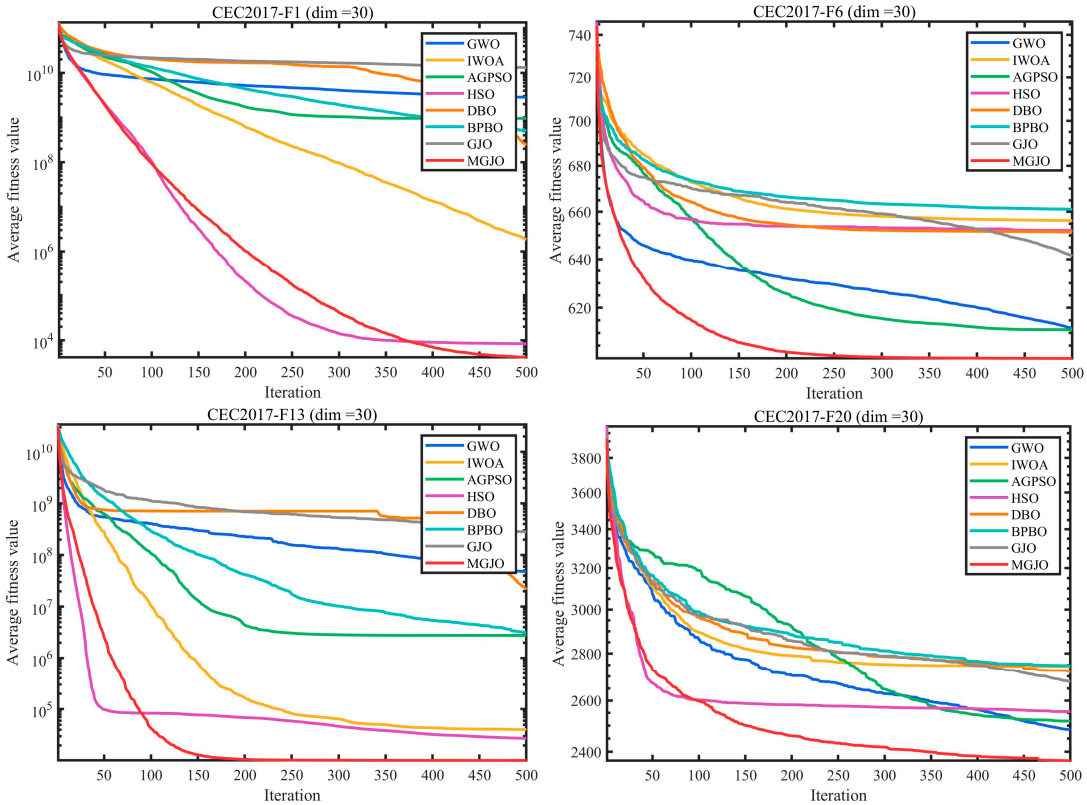


Figure 7. *Cont.*

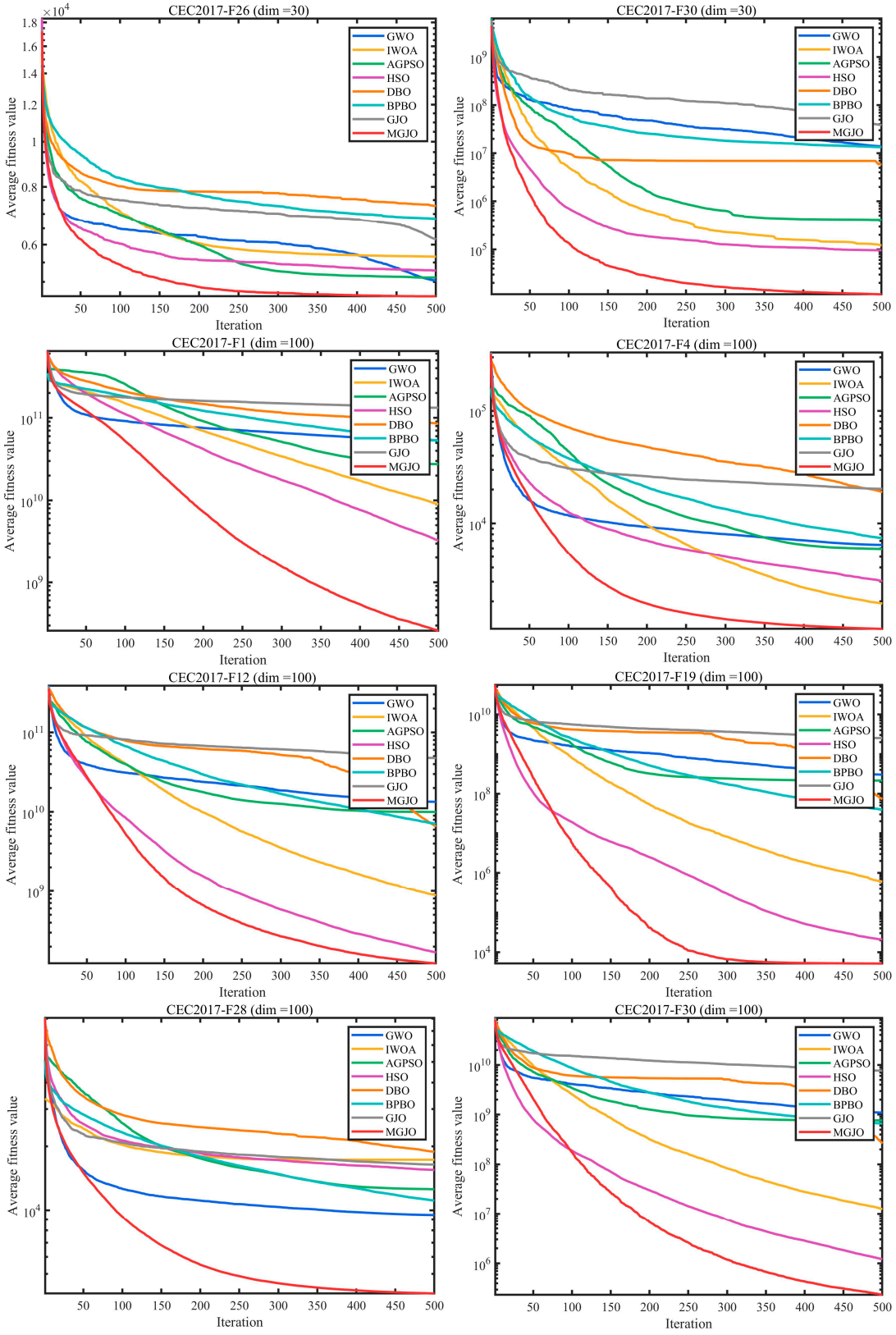


Figure 7. *Cont.*

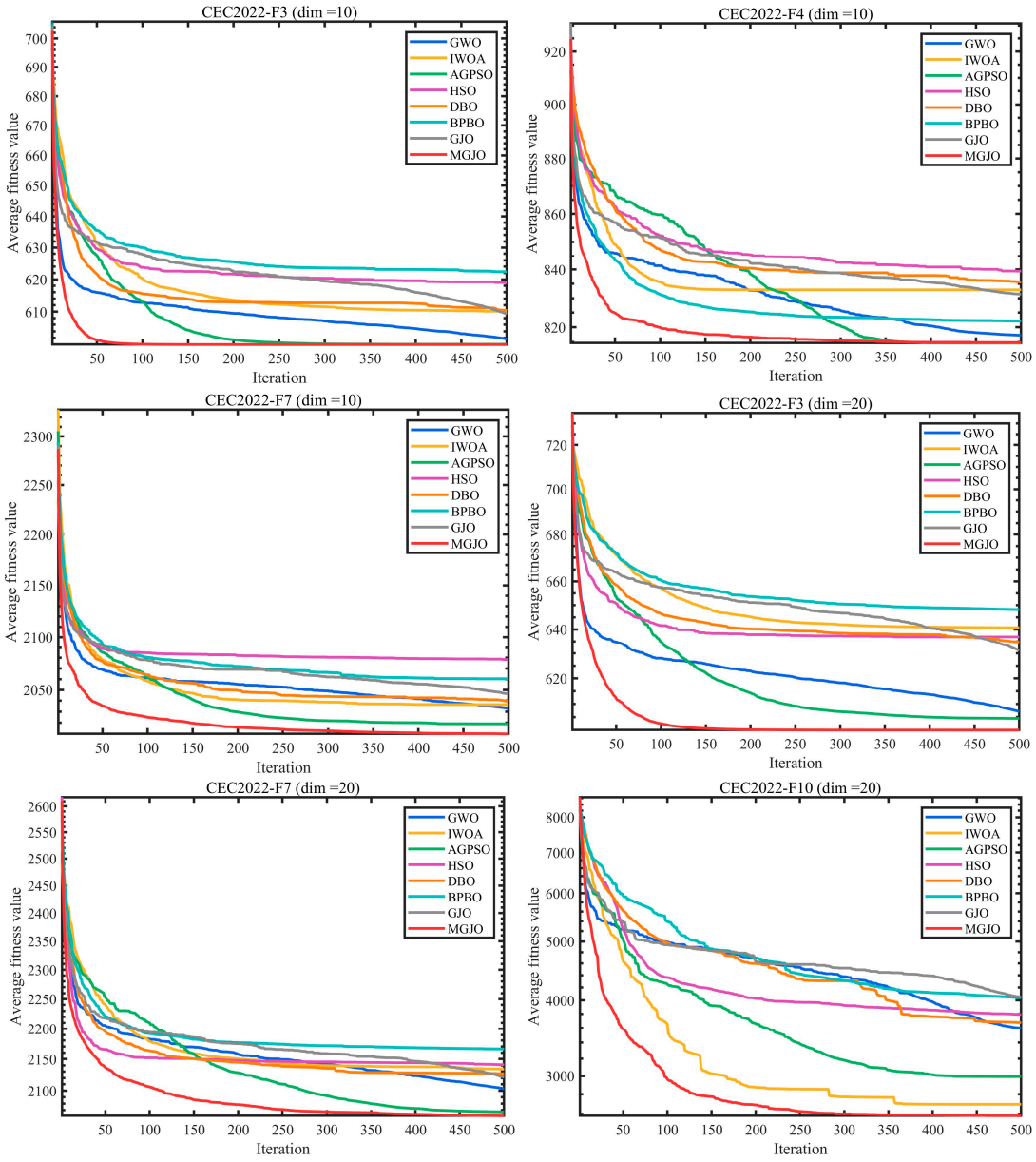


Figure 7. Part of the test function iterates convergence curves.

Furthermore, Figure 8 presents radar charts of rankings for all algorithms across CEC2017 (dim = 30, 100) and CEC2022 (dim = 10, 20), where smaller values indicate better performance. In CEC2017 (dim = 30, 100), MGJO ranks first on all functions, with minimal radar coverage concentrated at the center. For key functions such as F1, F2, and F10, MGJO ranks 1, significantly outperforming standard GJO (mostly rank 8) and other algorithms (e.g., GWO ranks 2–5, IWOA ranks 3–7). Its ranking advantage is especially pronounced for high-dimensional functions (dim = 100), where F12 and F19 remain ranked 1, while other algorithms generally rank higher (e.g., AGPSO ranks 5), confirming MGJO's superiority in high-dimensional scenarios. In CEC2022 (dim = 10, 20), MGJO still ranks first; for F3, F5, and F10, the rank is 1, and for F2 and F4, it is 2, while standard GJO mostly ranks 7–8 and GWO ranks 3–4, further demonstrating MGJO's consistent advantage across different benchmarks and dimensions, providing reliable support for practical applications.

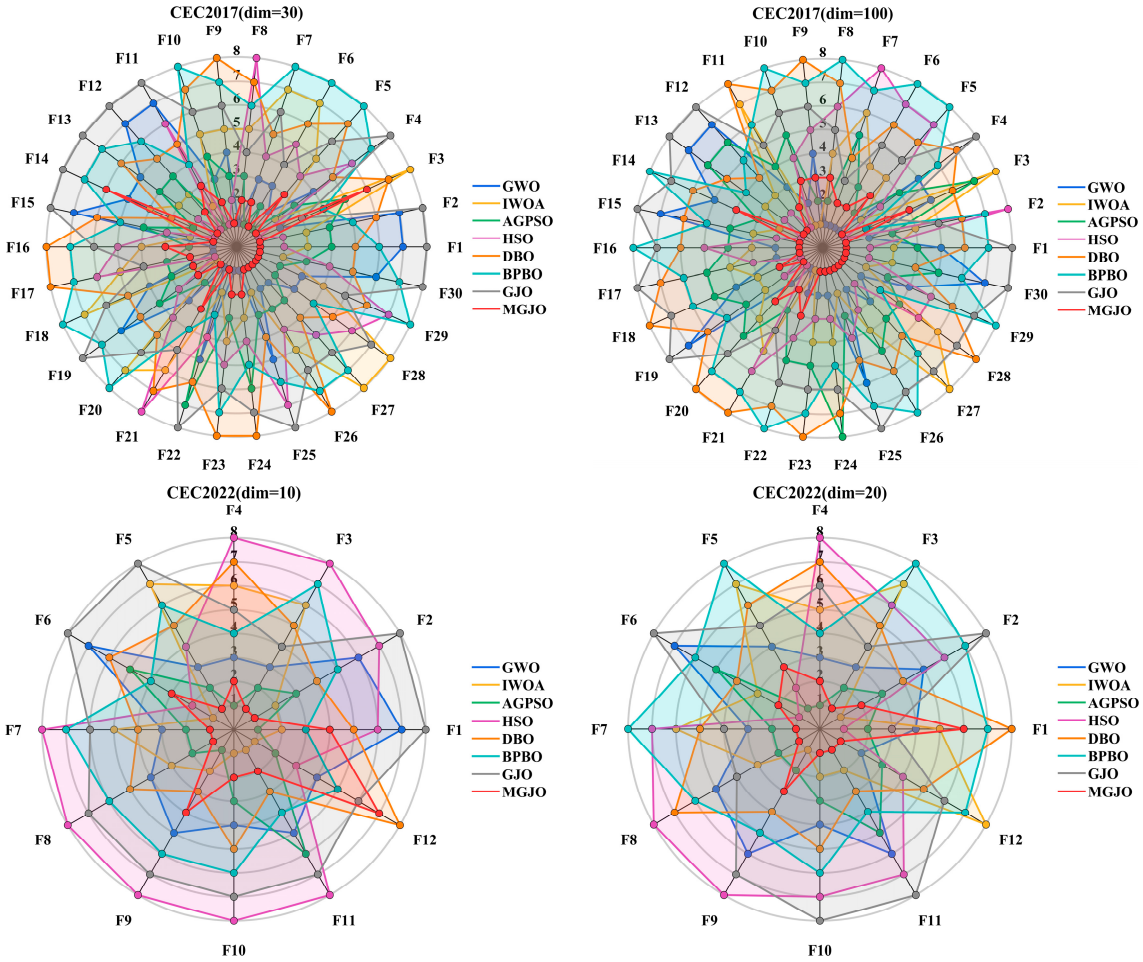


Figure 8. Performance distribution and relative ranking of the evaluated algorithms.

In summary, the quantitative results in Tables 4–7, together with convergence curves (Figure 7) and ranking visualizations (Figure 8), consistently show that MGJO significantly outperforms comparison algorithms in terms of optimization accuracy, stability, convergence speed, and generality, providing a solid foundation for its subsequent application in multilevel threshold image segmentation.

3.4. Stability Analysis

Statistical analysis plays a vital role in algorithm optimization, enabling researchers to evaluate and contrast the efficacy of various algorithmic approaches. This facilitates the selection of the most suitable method for specific research challenges. In this section, the performance of the MGJO algorithm is assessed using the Wilcoxon rank-sum test and the Friedman test, with detailed procedures and results presented accordingly.

3.4.1. Statistical Analysis Using Wilcoxon Rank-Sum Test

This section utilizes the Wilcoxon rank-sum test [54] to determine statistical significance in performance differentials between the proposed MGJO and comparative algorithms. This non-parametric approach does not require normal distribution assumptions, offering enhanced robustness compared to parametric alternatives like the t-test when handling non-Gaussian data or anomalous observations. The calculation of the Wilcoxon test statistic W follows the formulation presented in Equation (19).

$$W = \sum_{i=1}^{n_1} R(X_i) \tag{19}$$

In this procedure, each data point X_i is assigned a rank $R(X_i)$ based on its value within the entire sample. The final test statistic U is calculated according to the expression given in Equation (20).

$$U = W - \frac{n_1(n_1+1)}{2} \tag{20}$$

With sufficiently large samples, the U converges to normality, with its mean and variance given by Equations (21) and (22), respectively.

$$\mu_U = \frac{n_1 n_2}{2} \tag{21}$$

$$\sigma_U = \sqrt{\frac{n_1 n_2 (n_1+n_2+1)}{12}} \tag{22}$$

and the standardized statistic Z is calculated by Equation (23).

$$Z = \frac{U-\mu_U}{\sigma_U} \tag{23}$$

Table 8 presents the Wilcoxon rank-sum test results (significance level 0.05) for MGJO compared with seven benchmark algorithms on the CEC2017 (dim = 30, 100) and CEC2022 (dim = 10, 20) test suites. The performance differences are indicated using "+/=/−" ("+" means MGJO performs better, "−" means worse). On CEC2017 dim = 30, MGJO achieves "+"/"−" values of 29/1 against DBO and BPBO, and 28/2 against GJO. For dim = 100, MGJO attains "+"/"−" values of 29/1 against DBO, BPBO, and GJO, demonstrating an even more pronounced advantage. On CEC2022 dim = 10, MGJO records "+"/"−" values of 11/1 against GWO, HSO, and GJO. For dim = 20, it achieves "+"/"−" values of 12/0 against IWOA, DBO, and GJO, achieving complete dominance. Overall, MGJO significantly outperforms the comparison algorithms in most scenarios, with strong statistical reliability.

Table 8. Results for various algorithms on the CEC 2017 and CEC2022.

Statistical Results	GWO	IWOA	AGPSO	HSO	DBO	BPBO	GJO
CEC2017 dim = 30 (+/=/−)	(20/0/10)	(27/0/3)	(20/0/10)	(27/0/3)	(29/0/1)	(29/0/1)	28/0/2)
CEC2017 dim = 100 (+/=/−)	(27/0/3)	(28/0/2)	(25/0/5)	(28/0/2)	(29/0/1)	(29/0/1)	(29/0/1)
CEC2022 dim = 10 (+/=/−)	(11/0/1)	(10/0/2)	(10/0/2)	(11/0/1)	(8/0/4)	(9/0/3)	(11/0/1)
CEC2022 dim = 20 (+/=/−)	(11/0/1)	(12/0/0)	(9/0/3)	(10/0/2)	(12/0/0)	(6/0/6)	(12/0/0)

3.4.2. Friedman Average Ranking Assessment

This segment employs the Friedman examination [55] to establish comprehensive performance hierarchies among all evaluated methodologies. Operating as a non-parametric technique, this procedure effectively compares central tendency disparities across multiple correlated samples. Particularly advantageous for blocked experimental designs, it serves as a powerful substitute for conventional ANOVA when normality prerequisites remain unfulfilled. The computation of the Friedman metric follows the formulation specified in Equation (24) [55].

$$Q = \frac{12}{nk(k+1)} \sum_{j=1}^{k} R_j^2 - 3n(k+1) \tag{24}$$

where n is the number of blocks, k is the number of groups, and R_j is the rank sum for j-th group. When n and k are large, Q follows approximately a χ^2 distribution with $k-1$ degrees of freedom.

Table 9 presents the Friedman mean rank test results for MGJO and seven comparison algorithms on the CEC2017 (dim = 30, 100) and CEC2022 (dim = 10, 20) benchmark suites. The overall performance of each algorithm is quantified by the mean rank ($M.R$) and final rank ($T.R$), with smaller values indicating better performance. Across all test scenarios, MGJO achieves the lowest mean ranks and consistently secures the first final

rank. Specifically, for CEC2017 dim = 30, MGJO attains an *M.R* of 1.90, far lower than the second-best AGPSO (3.10) and standard GJO (6.50). For dim = 100, MGJO's *M.R* is 1.77, still leading GWO (3.47), while standard GJO has an *M.R* of 6.40 and a *T.R* of 7. On CEC2022 dim = 10, MGJO achieves an *M.R* of 2.42, outperforming AGPSO (2.50), with standard GJO at 6.75 and *T.R* 8. For dim = 20, MGJO attains an M.R of 2.00, significantly ahead of other algorithms, while standard GJO has an *M.R* of 5.92 and *T.R* 7. These results, from the perspective of overall ranking, confirm that MGJO's comprehensive optimization performance is significantly superior to comparison algorithms across different test suites and dimensionalities, further validating its effectiveness.

Table 9. Outcomes of the Friedman average ranking assessment.

Suites	**CEC2017**				**CEC2022**			
Dimensions	30		100		10		20	
Algorithms	*M.R*	*T.R*	*M.R*	*T.R*	*M.R*	*T.R*	*M.R*	*T.R*
GWO	3.70	3	3.47	2	4.50	4	4.33	4
IWOA	4.40	5	3.70	4	3.08	3	4.17	3
AGPSO	3.10	2	4.27	5	2.50	2	2.58	2
HSO	4.03	4	3.60	3	6.50	7	5.42	6
DBO	6.00	6	6.37	6	5.00	5	5.25	5
BPBO	6.37	7	6.43	8	5.25	6	6.33	8
GJO	6.50	8	6.40	7	6.75	8	5.92	7
MGJO	**1.90**	**1**	**1.77**	**1**	**2.42**	**1**	**2.00**	**1**

4. MGJO for Multilevel Thresholding Art Image Segmentation

In image segmentation through thresholding, Otsu's approach and Kapur's entropy criterion represent two widely adopted techniques for selecting optimal separation values. When a specific threshold value TH is applied, the inter-class variance between foreground and background regions varies accordingly. The ideal threshold (T_{best}) is identified as the value that maximizes this inter-class separation. Alternatively, Kapur's method calculates the combined entropy of foreground and background pixel distributions, seeking the threshold that optimizes this entropy measure. Both techniques process image pixel data as input and derive their respective optimal thresholds through distinct computational frameworks [54–56]. For the experimental phase of this research, Otsu's methodology has been implemented for threshold selection.

Otsu's technique establishes an optimal segmentation threshold by maximizing inter-class separation. Consider a digital image I possessing L distinct intensity values, where each tonal value i contains n_i pixels. The aggregate pixel population is consequently determined as:

$$N = \sum_{i=0}^{L-1} n_i \tag{25}$$

and the probability of a pixel having gray level i is:

$$P_i = \frac{n_i}{N}, i = 0, 1, \ldots, L-1 \tag{26}$$

where $P_i \geq 0$, *and* $P_0 + P_1 + \ldots + P_{L-1} = 1$.

Suppose the number of thresholds is k. If a gray level t is selected as a threshold, it divides the image into two regions: pixels with gray levels $[0, t]$ as the object, and pixels with gray levels $[t+1, L-1]$ as the background. Let the proportion of object pixels be ω_0 with mean gray level μ_0, and the proportion of background pixels be ω_1 with mean gray

level μ_1. Denote the overall mean gray level of the image as μ, and the inter-class variance as v. Then, ω_0, μ_0, ω_1, μ_1, μ and v are calculated as follows [56–58]:

$$\begin{gathered} \omega_0 = \sum_{i=0}^{t} P_i \\ \mu_0 = \sum_{i=0}^{t} \left(\frac{iP_i}{\omega_0}\right) \\ \omega_1 = \sum_{i=t+1}^{L-1} P_i \\ \mu_1 = \sum_{i=t+1}^{L-1} \left(\frac{iP_i}{\omega_1}\right) \\ \mu = \sum_{i=0}^{L-1} iP_i \\ (t) = \omega_0(\mu_0 - \mu)^2 + \omega_1(\mu_1 - \mu)^2 = \omega_0\omega_1(\mu_0 - \mu_1)^2 \end{gathered} \tag{27}$$

For multi-threshold segmentation, the inter-class variance is extended as:

$$\begin{gathered} v(t_1, t_2, \ldots, t_k) = \omega_0\omega_1(\mu_0 - \mu_1)^2 + \omega_0\omega_2(\mu_0 - \mu_2)^2 + \cdots + \\ \omega_0\omega_k(\mu_0 - \mu_k)^2 + \omega_1\omega_2(\mu_1 - \mu_2)^2 + \cdots + \\ \omega_1\omega_3(\mu_1 - \mu_3)^2 + \cdots + \omega_{k-1}\omega_k(\mu_{k-1} - \mu_k)^2 \end{gathered} \tag{28}$$

where ω_i and μ_i are calculated as:

$$\begin{gathered} \omega_{i-1} = \sum_{i=t_{i-1}+1}^{t_i} P_i, 1 \le i \le k+1 \\ \mu_{i-1} = \sum_{i=t_{i-1}+1}^{t_i} \left(\frac{iP_i}{\omega_{i-1}}\right), 1 \le i \le k+1 \end{gathered} \tag{29}$$

The optimal thresholds T_{best} are determined by:

$$T_{best} = \underset{0 \le t_1 \le t_2 \le \cdots \le t_k}{\arg max} \left[v(t_1, t_2, \ldots, t_k)\right] \tag{30}$$

In this study, the performance of the MGJO algorithm in multilevel thresholding is evaluated through comparative experiments. Seven improved swarm intelligence algorithms are selected as benchmarks. All algorithms are set with a population size of $pop = 30$ and a maximum iteration number of $T = 20$, using Otsu's method as the objective function. Experiments are conducted on eight benchmark images of different styles with threshold numbers of 4, 6, 8, and 10, as shown in Figure 9, Each experiment is independently run 30 times, and the mean (Ave) and standard deviation (Std) of the objective function values are reported to assess algorithm accuracy and stability. All algorithm parameters are kept consistent with previous sections to ensure a fair comparison.



Figure 9. *Cont.*



Figure 9. The set of benchmark images.

4.1. Evaluation Index

Standard quantitative measures for assessing segmentation efficacy comprise Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and Feature Similarity Index (FSIM). Elevated PSNR and SSIM readings correspond to reduced distortion and enhanced visual fidelity in processed images, while superior FSIM scores reflect diminished feature-level discrepancies. The computational formulations for these metrics are specified in references [59,60] as follows:

The PSNR is calculated as:

$$PSNR = 10log_{10}\left(\frac{255^2}{MSE}\right) \tag{31}$$

where the Mean Square Error (MSE) quantifies the cumulative squared deviation between the source image I and its processed counterpart I'. The MSE calculation is defined as:

$$MSE = \frac{1}{MN}\sum_{j=1}^{M}\sum_{k=1}^{N}[I(j,k) - I'(j,k)]^2 \tag{32}$$

where $(M \times N)$ represents the image dimensions, I and I' denote the original and segmented images, respectively, $I(j,k)$ is the gray value of the original image at pixel (j,k), and $I'(j,k)$ is the gray value of the segmented image at the same pixel.

SSIM measures the structural similarity between two images by comparing their luminance, contrast, and structural information. Given two images I and I', SSIM is calculated as:

$$SSIM(I,I') = \frac{(2\mu_I\mu_{I'}+C_1)(2\sigma_{II'}+C_2)}{(\mu_I^2+\mu_{I'}^2+C_1)(\sigma_I^2+\sigma_{I'}^2+C_2)} \tag{33}$$

where μ_I and $\mu_{I'}$ are the mean gray values of I and I', $\sigma_{II'}$ is the covariance between I and I', σ_I^2 and $\sigma_{I'}^2$ are the variances of I and I', respectively. Typically $C_1 = K_1L$, $C_2 = (K_2L)^2$, $K_1 = 0.01$, $K_2 = 0.03$, and L being the maximum gray level.

Human visual perception is sensitive to points with high phase consistency; therefore, FSIM uses phase congruency as the primary feature to assess image quality. FSIM is computed as:

$$FSIM(I,I') = \frac{\sum_{x\in\Omega} S_L(x)\cdot PC_m(x)}{\sum_{x\in\Omega} PC_m(x)} \tag{34}$$

where x denotes a pixel, Ω represents the entire image domain, and $PC_m(x) = max(PC_1(x), PC_2(x))$, $PC_1(x)$ and $PC_2(x)$ being the phase congruency values of the original and segmented images, respectively.

4.2. Evaluation of Otsu Thresholding Performance Using MGJO Framework

The proposed MGJO framework was utilized to identify optimal segmentation thresholds through Otsu's criterion as the fitness function for multi-level image thresholding across eight test images. Algorithm efficacy is quantified through the maximized fitness score alongside PSNR, FSIM, and SSIM metrics, where superior values across these indicators correspond to enhanced segmentation quality. Experimental outcomes confirm that the developed MGJO technique consistently obtains superior performance across all evaluation criteria—fitness function values, PSNR, FSIM, and SSIM—when benchmarked against established optimization methods.

Table 10 displays the spatial distribution of optimized thresholds across image histograms. Table 11 summarizes the central tendency and variability of optimal fitness values achieved by all compared methods using Otsu's criterion, accompanied by Friedman ranking outcomes. Tables 12–14 systematically document the performance statistics (mean and standard deviation) for PSNR, FSIM, and SSIM metrics, respectively, under Otsu-based optimization, with integrated Friedman ranking comparisons. Experiments were conducted for threshold numbers of 4, 6, 8, and 10. Figures 10–13 visually illustrate the comprehensive performance of each algorithm using Friedman average rankings, collectively confirming the significant superiority of MGJO in multilevel image segmentation tasks.

From the optimal fitness values reported in Table 11, MGJO demonstrates the highest search accuracy and strongest stability across all image and threshold combinations. For instance, under a low threshold (TH = 4) on the Camera image, MGJO achieves an Ave value of 4.6001×10^{03}, slightly higher than the standard GJO's 4.5950×10^{03}, while its Std value of 1.0296×10^{00} is only 31.7% of GJO's, indicating much smaller variability across multiple runs. Under a high threshold (TH = 10) on the Lena image, MGJO attains an Ave value of 3.8118×10^{03}, surpassing GWO's 3.8035×10^{03}, with a Std value of 1.9341×10^{00}, lower than AGPSO's 3.2966×10^{00}. For a complex-texture image like Baboon (TH = 8), MGJO achieves an Ave of 3.3992×10^{03}, higher than HSO's 3.3671×10^{03}, and a Std of 1.1804×10^{00}, only 10.6% of HSO's. These results confirm that MGJO consistently attains higher Ave values (closer to the optimal Otsu objective function) and lower Std values (smaller fluctuations over repeated runs), demonstrating superior accuracy and stability in identifying optimal segmentation thresholds.

The image quality evaluation metrics reported in Tables 12–14 further highlight MGJO's advantages. In terms of PSNR (higher values indicate lower distortion), for the Face image at TH = 10, MGJO reaches a PSNR of 26.4192, exceeding GJO's 24.9027, with the lowest Std of 0.2050 among all algorithms. For FSIM (higher values indicate lower segmentation error), for the Hunter image at TH = 10, MGJO achieves 0.9529, higher than GJO's 0.9377, with a Std of 0.0024, substantially lower than other algorithms. For SSIM (higher values indicate greater structural similarity to the original image), for the Saturn image at TH = 10, MGJO attains 0.9267, outperforming GJO's 0.9119, with a Std of 0.0025, only 2.7% of GJO's. These results indicate that MGJO not only produces segmented images with low distortion but also preserves the original features and structural information more effectively.

Table 10. Multi-level thresholding outcomes using MGJO with Otsu's criterion as objective function.

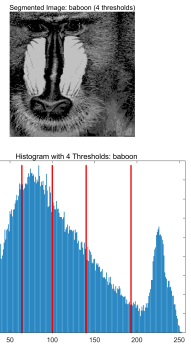
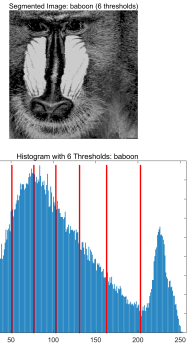
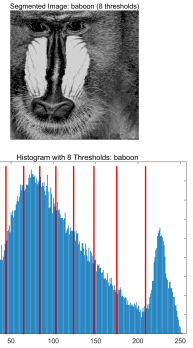
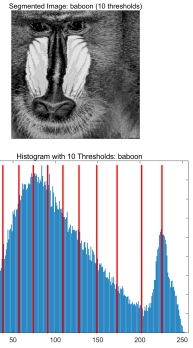
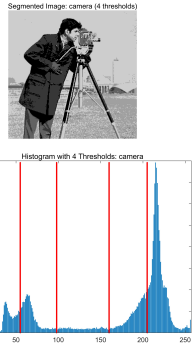
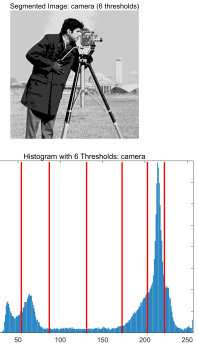
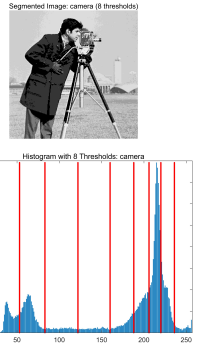
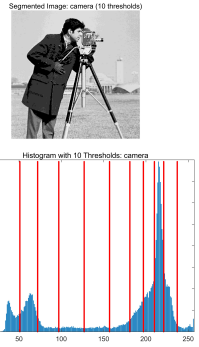
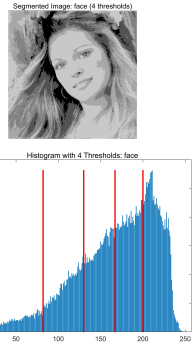
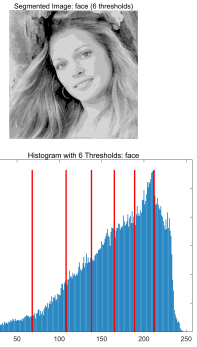
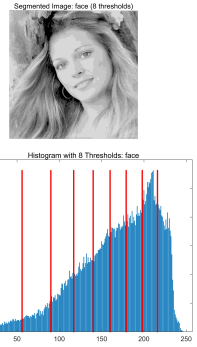
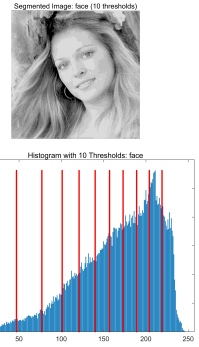
Images	TH = 4	TH = 6	TH = 8	TH = 10
baboon				
Camera				
Face				

Table 10. *Cont.*

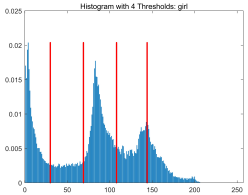
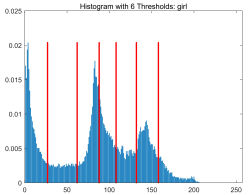
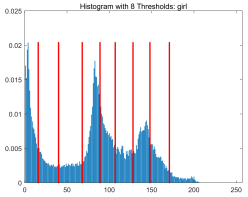
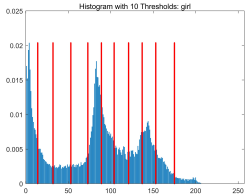
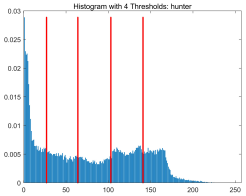
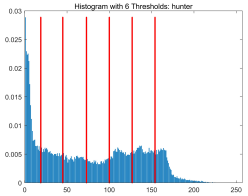
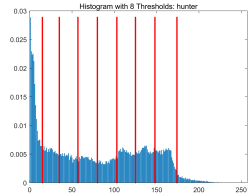
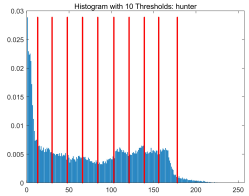
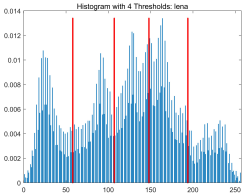
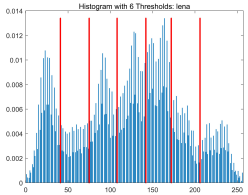
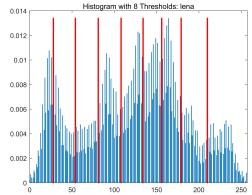
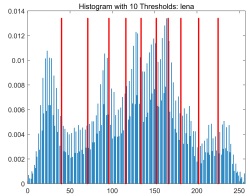
Images	TH = 4	TH = 6	TH = 8	TH = 10
Girl	 	 	 	 
Hunter	 	 	 	 
Lena	 	 	 	 

Table 10. *Cont.*

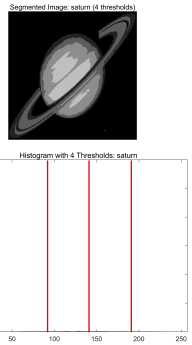
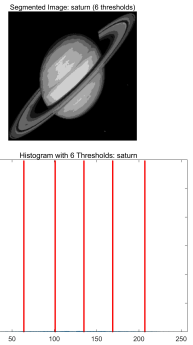
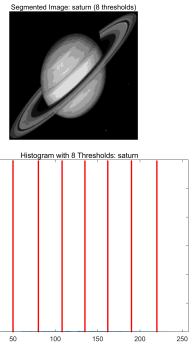
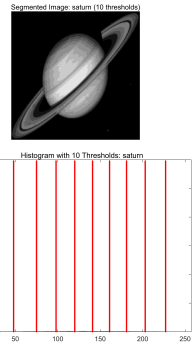
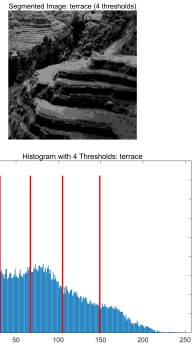
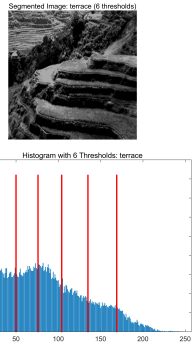
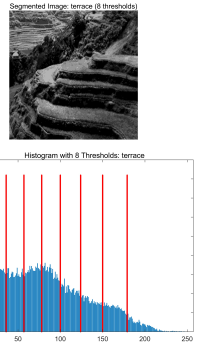
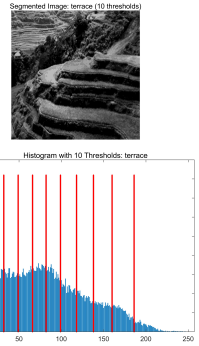
Images	TH = 4	TH = 6	TH = 8	TH = 10
Saturn				
Terrace				

Table 11. Ave and Std of the optimal fitness values with Otsu as the objective function.

Images	TH	Metrics	GWO	IWOA	AGPSO	HSO	DBO	BPBO	GJO	MGJO
Baboon	4	Ave	3.2985×10^{03}	3.2958×10^{03}	3.3001×10^{03}	3.2652×10^{03}	3.2993×10^{03}	3.2945×10^{03}	3.2933×10^{03}	$\mathbf{3.3008 \times 10^{03}}$
		Std	4.1827×10^{00}	2.8020×10^{00}	6.4510×10^{-01}	1.9685×10^{01}	1.6491×10^{00}	4.6041×10^{00}	6.1253×10^{00}	$\mathbf{3.9126 \times 10^{-02}}$
	6	Ave	3.3648×10^{03}	3.3594×10^{03}	3.3674×10^{03}	3.3327×10^{03}	3.3604×10^{03}	3.3615×10^{03}	3.3611×10^{03}	$\mathbf{3.3718 \times 10^{03}}$
		Std	8.5391×10^{00}	8.3701×10^{00}	3.3757×10^{00}	1.0698×10^{01}	1.1013×10^{01}	5.4107×10^{00}	4.8473×10^{00}	$\mathbf{4.9081 \times 10^{-01}}$
	8	Ave	3.3936×10^{03}	3.3869×10^{03}	3.3937×10^{03}	3.3671×10^{03}	3.3876×10^{03}	3.3862×10^{03}	3.3851×10^{03}	$\mathbf{3.3992 \times 10^{03}}$
		Std	4.1562×10^{00}	4.7990×10^{00}	3.6513×10^{00}	1.1122×10^{01}	5.2377×10^{00}	5.6342×10^{00}	7.9354×10^{00}	$\mathbf{1.1804 \times 10^{00}}$
	10	Ave	3.4079×10^{03}	3.4029×10^{03}	3.4081×10^{03}	3.3900×10^{03}	3.4026×10^{03}	3.4042×10^{03}	3.4019×10^{03}	$\mathbf{3.4139 \times 10^{03}}$
		Std	3.1677×10^{00}	3.6528×10^{00}	2.9531×10^{00}	7.1104×10^{00}	5.4729×10^{00}	3.6108×10^{00}	4.1270×10^{00}	$\mathbf{1.0583 \times 10^{00}}$
Camera	4	Ave	4.5975×10^{03}	4.5975×10^{03}	4.5990×10^{03}	4.5836×10^{03}	4.5983×10^{03}	4.5968×10^{03}	4.5950×10^{03}	$\mathbf{4.6001 \times 10^{03}}$
		Std	2.6817×10^{00}	1.9470×10^{00}	1.1715×10^{00}	6.7691×10^{00}	1.5377×10^{00}	2.3036×10^{00}	3.2502×10^{00}	$\mathbf{1.0296 \times 10^{00}}$
	6	Ave	4.6459×10^{03}	4.6416×10^{03}	4.6477×10^{03}	4.6189×10^{03}	4.6407×10^{03}	4.6371×10^{03}	4.6396×10^{03}	$\mathbf{4.6512 \times 10^{03}}$
		Std	4.9420×10^{00}	4.9456×10^{00}	4.2107×10^{00}	8.4537×10^{00}	5.7329×10^{00}	5.4444×10^{00}	5.8709×10^{00}	$\mathbf{3.7977 \times 10^{-01}}$
	8	Ave	4.6603×10^{03}	4.6605×10^{03}	4.6648×10^{03}	4.6416×10^{03}	4.6567×10^{03}	4.6597×10^{03}	4.6587×10^{03}	$\mathbf{4.6690 \times 10^{03}}$
		Std	6.2075×10^{00}	3.9347×10^{00}	2.7189×10^{00}	9.5247×10^{00}	5.5192×10^{00}	5.1934×10^{00}	4.4859×10^{00}	$\mathbf{1.0893 \times 10^{00}}$
	10	Ave	4.6739×10^{03}	4.6706×10^{03}	4.6739×10^{03}	4.6527×10^{03}	4.6669×10^{03}	4.6688×10^{03}	4.6692×10^{03}	$\mathbf{4.6791 \times 10^{03}}$
		Std	3.6129×10^{00}	2.9887×10^{00}	1.9996×10^{00}	8.4757×10^{00}	4.4742×10^{00}	3.2907×10^{00}	3.9017×10^{00}	$\mathbf{9.6454 \times 10^{-01}}$
Face	4	Ave	2.1204×10^{03}	2.1163×10^{03}	2.1217×10^{03}	2.0907×10^{03}	2.1198×10^{03}	2.1180×10^{03}	2.1127×10^{03}	$\mathbf{2.1224 \times 10^{03}}$
		Std	2.5080×10^{00}	4.3746×10^{00}	1.3134×10^{00}	1.5501×10^{01}	3.1669×10^{00}	3.5967×10^{00}	1.2110×10^{01}	$\mathbf{8.0498 \times 10^{-02}}$
	6	Ave	2.1764×10^{03}	2.1740×10^{03}	2.1803×10^{03}	2.1483×10^{03}	2.1763×10^{03}	2.1760×10^{03}	2.1666×10^{03}	$\mathbf{2.1843 \times 10^{03}}$
		Std	7.3609×10^{00}	7.4735×10^{00}	3.2680×10^{00}	1.2176×10^{01}	6.8784×10^{00}	5.4467×10^{00}	9.2604×10^{00}	$\mathbf{5.6488 \times 10^{-01}}$
	8	Ave	2.2014×10^{03}	2.1973×10^{03}	2.2035×10^{03}	2.1789×10^{03}	2.1994×10^{03}	2.2020×10^{03}	2.1936×10^{03}	$\mathbf{2.2094 \times 10^{03}}$
		Std	6.3197×10^{00}	5.3907×10^{00}	3.5669×10^{00}	9.7466×10^{00}	5.1068×10^{00}	4.1424×10^{00}	6.4159×10^{00}	$\mathbf{1.3020 \times 10^{00}}$
	10	Ave	2.2132×10^{03}	2.2107×10^{03}	2.2173×10^{03}	2.1972×10^{03}	2.2106×10^{03}	2.2137×10^{03}	2.2073×10^{03}	$\mathbf{2.2225 \times 10^{03}}$
		Std	4.9300×10^{00}	5.5702×10^{00}	2.8225×10^{00}	7.6236×10^{00}	7.2712×10^{00}	4.5883×10^{00}	6.9676×10^{00}	$\mathbf{9.0505 \times 10^{-01}}$
Girl	4	Ave	2.5331×10^{03}	2.5294×10^{03}	2.5331×10^{03}	2.5096×10^{03}	2.5331×10^{03}	2.5312×10^{03}	2.5304×10^{03}	$\mathbf{2.5339 \times 10^{03}}$
		Std	2.2156×10^{00}	3.8668×10^{00}	2.3908×10^{00}	1.3297×10^{01}	1.1515×10^{00}	3.2110×10^{00}	3.2500×10^{00}	$\mathbf{9.2841 \times 10^{-03}}$
	6	Ave	2.5824×10^{03}	2.5761×10^{03}	2.5813×10^{03}	2.5522×10^{03}	2.5806×10^{03}	2.5775×10^{03}	2.5775×10^{03}	$\mathbf{2.5843 \times 10^{03}}$
		Std	2.3014×10^{00}	5.6223×10^{00}	2.7930×10^{00}	1.2912×10^{01}	2.6832×10^{00}	4.1286×10^{00}	4.1278×10^{00}	$\mathbf{3.5990 \times 10^{-01}}$
	8	Ave	2.6021×10^{03}	2.5950×10^{03}	2.6019×10^{03}	2.5821×10^{03}	2.5967×10^{03}	2.5963×10^{03}	2.5963×10^{03}	$\mathbf{2.6057 \times 10^{03}}$
		Std	3.8462×10^{00}	4.5544×10^{00}	2.8120×10^{00}	8.0226×10^{00}	4.4712×10^{00}	4.2578×10^{00}	4.1632×10^{00}	$\mathbf{1.2443 \times 10^{00}}$
	10	Ave	2.6120×10^{03}	2.6065×10^{03}	2.6116×10^{03}	2.5040×10^{03}	2.6073×10^{03}	2.6074×10^{03}	2.6087×10^{03}	$\mathbf{2.6166 \times 10^{03}}$
		Std	3.1581×10^{00}	3.8739×10^{00}	2.3522×10^{00}	4.7301×10^{02}	3.7702×10^{00}	3.8689×10^{00}	2.9101×10^{00}	$\mathbf{8.5354 \times 10^{-01}}$
Hunter	4	Ave	3.1889×10^{03}	3.1866×10^{03}	3.1895×10^{03}	3.1472×10^{03}	3.1892×10^{03}	3.1865×10^{03}	3.1864×10^{03}	$\mathbf{3.1902 \times 10^{03}}$
		Std	2.1547×10^{00}	2.5499×10^{00}	7.3977×10^{-01}	2.8846×10^{01}	9.6951×10^{-01}	4.0963×10^{00}	2.5061×10^{00}	$\mathbf{1.9189 \times 10^{-01}}$
	6	Ave	3.2433×10^{03}	3.2385×10^{03}	3.2439×10^{03}	3.2103×10^{03}	3.2411×10^{03}	3.2396×10^{03}	3.2404×10^{03}	$\mathbf{3.2464 \times 10^{03}}$
		Std	4.4935×10^{00}	4.3160×10^{00}	1.4189×10^{00}	1.6430×10^{01}	4.7173×10^{00}	4.9492×10^{00}	3.6477×10^{00}	$\mathbf{7.5295 \times 10^{-01}}$
	8	Ave	3.2684×10^{03}	3.2598×10^{03}	3.2663×10^{03}	3.2386×10^{03}	3.2613×10^{03}	3.2612×10^{03}	3.2629×10^{03}	$\mathbf{3.2715 \times 10^{03}}$
		Std	2.8558×10^{00}	4.3686×10^{00}	3.2022×10^{00}	1.2175×10^{01}	5.5053×10^{00}	4.2536×10^{00}	4.0888×10^{00}	$\mathbf{9.2072 \times 10^{-01}}$
	10	Ave	3.2787×10^{03}	3.2728×10^{03}	3.2777×10^{03}	3.2506×10^{03}	3.2725×10^{03}	3.2734×10^{03}	3.2742×10^{03}	$\mathbf{3.2836 \times 10^{03}}$
		Std	3.2829×10^{00}	4.5857×10^{00}	3.5896×10^{00}	1.0999×10^{01}	4.5371×10^{00}	4.0687×10^{00}	3.7927×10^{00}	$\mathbf{8.4336 \times 10^{-01}}$
Lena	4	Ave	3.6827×10^{03}	3.6811×10^{03}	3.6851×10^{03}	3.6374×10^{03}	3.6842×10^{03}	3.6792×10^{03}	3.6780×10^{03}	$\mathbf{3.6860 \times 10^{03}}$
		Std	4.3455×10^{00}	4.8787×10^{00}	1.5975×10^{00}	3.3474×10^{01}	1.9209×10^{00}	7.4090×10^{00}	5.3111×10^{00}	$\mathbf{1.1916 \times 10^{-01}}$
	6	Ave	3.7573×10^{03}	3.7507×10^{03}	3.7581×10^{03}	3.7236×10^{03}	3.7556×10^{03}	3.7517×10^{03}	3.7487×10^{03}	$\mathbf{3.7654 \times 10^{03}}$
		Std	8.8347×10^{00}	7.9039×10^{00}	5.8364×10^{00}	1.3862×10^{01}	7.5451×10^{00}	8.2557×10^{00}	1.0989×10^{01}	$\mathbf{5.2392 \times 10^{-01}}$
	8	Ave	3.7870×10^{03}	3.7825×10^{03}	3.7896×10^{03}	3.7610×10^{03}	3.7840×10^{03}	3.7823×10^{03}	3.7790×10^{03}	$\mathbf{3.7946 \times 10^{03}}$
		Std	4.3173×10^{00}	5.0347×10^{00}	3.3579×10^{00}	1.1018×10^{01}	6.2895×10^{00}	5.9568×10^{00}	6.3455×10^{00}	$\mathbf{1.0259 \times 10^{00}}$
	10	Ave	3.8035×10^{03}	3.7996×10^{03}	3.8053×10^{03}	3.7832×10^{03}	3.7985×10^{03}	3.7992×10^{03}	3.7967×10^{03}	$\mathbf{3.8118 \times 10^{03}}$
		Std	5.5683×10^{00}	3.8848×10^{00}	3.2966×10^{00}	7.9172×10^{00}	5.1814×10^{00}	5.4822×10^{00}	4.5858×10^{00}	$\mathbf{1.9341 \times 10^{00}}$

Table 11. *Cont.*

Images	TH	Metrics	GWO	IWOA	AGPSO	HSO	DBO	BPBO	GJO	MGJO
Saturn	4	Ave	5.2205×10^{03}	5.2183×10^{03}	5.2215×10^{03}	5.1955×10^{03}	5.2208×10^{03}	5.2197×10^{03}	5.2163×10^{03}	$\mathbf{5.2220 \times 10^{03}}$
		Std	2.7184×10^{00}	3.2853×10^{00}	7.1242×10^{-01}	1.4455×10^{01}	1.5341×10^{00}	2.6001×10^{00}	7.1545×10^{00}	$\mathbf{3.0003 \times 10^{-02}}$
	6	Ave	5.2692×10^{03}	5.2659×10^{03}	5.2713×10^{03}	5.2475×10^{03}	5.2686×10^{03}	5.2665×10^{03}	5.2660×10^{03}	$\mathbf{5.2726 \times 10^{03}}$
		Std	3.1010×10^{00}	3.0999×10^{00}	1.7429×10^{00}	9.6328×10^{00}	3.3621×10^{00}	3.6739×10^{00}	3.9684×10^{00}	$\mathbf{4.9814 \times 10^{-01}}$
	8	Ave	5.2889×10^{03}	5.2860×10^{03}	5.2898×10^{03}	5.2741×10^{03}	5.2871×10^{03}	5.2846×10^{03}	5.2848×10^{03}	$\mathbf{5.2927 \times 10^{03}}$
		Std	2.5168×10^{00}	2.4244×10^{00}	2.0422×10^{00}	6.7971×10^{00}	3.3044×10^{00}	3.3184×10^{00}	3.6748×10^{00}	$\mathbf{6.0088 \times 10^{-01}}$
	10	Ave	5.2992×10^{03}	5.2967×10^{03}	5.2985×10^{03}	5.2840×10^{03}	5.2955×10^{03}	5.2958×10^{03}	5.2953×10^{03}	$\mathbf{5.3026 \times 10^{03}}$
		Std	2.0853×10^{00}	1.9785×10^{00}	2.5973×10^{00}	5.1445×10^{00}	3.0911×10^{00}	3.4434×10^{00}	3.2255×10^{00}	$\mathbf{5.9890 \times 10^{-01}}$
Terrace	4	Ave	2.6390×10^{03}	2.6352×10^{03}	2.6392×10^{03}	2.5956×10^{03}	2.6385×10^{03}	2.6352×10^{03}	2.6361×10^{03}	$\mathbf{2.6402 \times 10^{03}}$
		Std	3.3119×10^{00}	5.8176×10^{00}	1.1456×10^{00}	2.2690×10^{01}	1.9837×10^{00}	4.7550×10^{00}	3.1223×10^{00}	$\mathbf{3.7343 \times 10^{-02}}$
	6	Ave	2.7000×10^{03}	2.6900×10^{03}	2.6978×10^{03}	2.6594×10^{03}	2.6961×10^{03}	2.6915×10^{03}	2.6914×10^{03}	$\mathbf{2.7020 \times 10^{03}}$
		Std	2.3293×10^{00}	5.6849×10^{00}	3.1792×10^{00}	1.5664×10^{01}	5.2586×10^{00}	5.9177×10^{00}	6.8622×10^{00}	$\mathbf{4.1380 \times 10^{-01}}$
	8	Ave	2.7248×10^{03}	2.7156×10^{03}	2.7224×10^{03}	2.6972×10^{03}	2.7195×10^{03}	2.7198×10^{03}	2.7190×10^{03}	$\mathbf{2.7284 \times 10^{03}}$
		Std	4.1674×10^{00}	5.3877×10^{00}	3.7844×10^{00}	9.5988×10^{00}	5.3308×10^{00}	3.6885×10^{00}	3.5037×10^{00}	$\mathbf{1.0277 \times 10^{00}}$
	10	Ave	2.7384×10^{03}	2.7316×10^{03}	2.7359×10^{03}	2.7091×10^{03}	2.7335×10^{03}	2.7315×10^{03}	2.7325×10^{03}	$\mathbf{2.7419 \times 10^{03}}$
		Std	2.7462×10^{00}	4.7626×10^{00}	2.7346×10^{00}	1.1469×10^{01}	4.1832×10^{00}	4.3540×10^{00}	3.5022×10^{00}	$\mathbf{1.1363 \times 10^{00}}$
Friedman-Rank			2.73	5.70	3.49	7.93	4.30	5.32	5.37	**1.18**
Final-Rank			2	7	3	8	4	5	6	**1**

Table 12. Ave and Std of all test images for PSNR in Otsu.

Images	TH	Metrics	GWO	IWOA	AGPSO	HSO	DBO	BPBO	GJO	MGJO
Baboon	4	Ave	18.0952	**18.2554**	18.1909	17.4612	18.1520	17.7155	18.0750	18.2235
		Std	0.2659	0.3598	0.1669	1.0028	0.2494	0.3909	0.4987	**0.0587**
	6	Ave	21.0145	20.9345	21.2826	19.4113	20.7927	20.2304	20.8563	**21.3266**
		Std	0.5899	0.7698	0.3768	0.9375	0.8806	0.6951	0.7546	**0.1942**
	8	Ave	22.9893	22.6465	23.0175	20.9897	22.4766	21.6938	22.6795	**23.5610**
		Std	0.6046	0.7081	0.5819	0.8642	0.8183	0.8491	0.7449	**0.3852**
	10	Ave	24.3626	24.0177	24.4181	22.8657	24.2954	23.3296	24.1569	**25.0596**
		Std	0.6461	0.7130	0.6644	1.0584	0.8705	0.8327	0.8141	**0.4374**
Camera	4	Ave	18.3413	**19.0060**	18.3878	17.5884	18.5210	18.2794	18.2820	18.9996
		Std	0.8146	1.0353	0.7431	**0.5742**	0.9090	0.8633	0.7464	0.8982
	6	Ave	21.1833	21.1172	21.7305	19.8444	21.0829	20.0684	21.1714	**21.8160**
		Std	1.0512	0.9556	0.4051	1.4722	1.0282	1.1818	0.9735	**0.1918**
	8	Ave	22.5215	22.5343	**23.0863**	21.4453	22.6065	22.0485	22.5461	23.0673
		Std	1.0347	0.7697	0.6265	1.5412	0.9137	0.9112	0.8603	**0.3722**
	10	Ave	23.7482	23.5751	23.7985	22.4346	23.8518	22.7772	23.8941	**24.0101**
		Std	0.5901	1.0687	0.8536	1.5501	1.1107	0.8033	1.0044	**0.3680**
Face	4	Ave	19.6190	19.5317	19.7119	18.7452	19.6555	19.4749	19.4372	**19.7431**
		Std	0.2258	0.3417	0.1033	0.5727	0.1728	0.3023	0.4106	**0.0407**
	6	Ave	22.1843	21.8667	22.4044	20.9382	22.1876	22.0046	21.7645	**22.5966**
		Std	0.3766	0.7107	0.3096	0.6649	0.5024	0.5694	0.4976	**0.1161**
	8	Ave	24.2104	23.8621	24.2547	22.5695	24.0278	24.0344	23.6361	**24.6478**
		Std	0.4971	0.4271	0.5197	0.7039	0.4714	0.4733	0.4647	**0.2921**
	10	Ave	25.3571	25.0458	25.7182	23.7160	25.1385	25.2690	24.9027	**26.4192**
		Std	0.5372	0.7170	0.3808	0.6686	0.7911	0.6107	0.6236	**0.2050**
Girl	4	Ave	21.9857	21.8110	21.9611	21.1045	22.0105	**22.1478**	21.8203	21.9595
		Std	0.2566	0.3911	0.2828	0.8421	0.1957	0.2841	0.4064	**0.0575**
	6	Ave	24.2986	23.8934	24.2860	22.8888	24.1695	24.2970	23.9450	**24.4533**
		Std	0.4115	0.6289	0.3916	1.1162	0.5167	0.4098	0.5812	**0.2265**
	8	Ave	26.2020	25.4631	26.1407	24.3075	25.6081	25.9612	25.5383	**26.3961**
		Std	0.4272	0.7363	0.4635	0.8497	0.6244	0.5376	0.6101	**0.2812**
	10	Ave	27.5148	26.6413	27.3021	25.1911	26.8166	27.2625	26.7874	**28.0996**
		Std	0.4306	0.6467	0.5666	1.5980	0.6751	0.5098	0.6573	**0.2191**
Hunter	4	Ave	21.9108	21.8017	21.9201	20.8587	21.9017	21.9194	21.7408	**21.9927**
		Std	0.1196	0.1763	0.0913	0.7714	0.0954	0.1103	0.1617	**0.0288**
	6	Ave	24.4434	24.0101	24.3554	22.6226	24.2966	24.4141	24.1654	**24.5299**
		Std	0.3119	0.3344	**0.2244**	0.7563	0.3138	0.2544	0.3551	0.2920
	8	Ave	26.0975	25.2994	25.8516	23.9899	25.5459	25.8217	25.6038	**26.2694**
		Std	0.2557	0.4485	0.3325	0.7630	0.4435	0.3720	0.3582	**0.1212**
	10	Ave	27.2447	26.5114	27.0846	24.7966	26.6295	26.9010	26.6213	**27.8330**
		Std	0.4310	0.5125	0.4322	0.7187	0.4018	0.4739	0.4719	**0.1634**
Lena	4	Ave	19.0565	19.0102	19.0952	18.2606	19.0721	19.0340	18.9694	**19.1212**
		Std	0.0888	0.1206	0.0605	0.6663	0.0848	0.1201	0.1461	**0.0279**
	6	Ave	21.5413	21.2753	21.5354	20.3493	21.3816	21.2560	21.1731	**21.8512**
		Std	0.3328	0.2984	0.3216	0.5040	0.3738	0.2894	0.4290	**0.0518**
	8	Ave	23.1264	22.8547	23.2917	21.8235	22.8829	22.7026	22.7378	**23.6057**
		Std	0.4001	0.4382	0.3825	0.6423	0.3833	0.4045	0.4917	**0.2887**
	10	Ave	24.2832	24.1340	24.5612	22.9175	23.9934	23.7670	23.8262	**24.9759**
		Std	0.4871	0.4338	0.3794	0.5785	0.5769	0.5505	0.4946	**0.3618**
Saturn	4	Ave	22.2645	22.2029	22.3277	21.4109	22.3166	**22.3655**	22.1494	22.3423
		Std	0.1317	0.2373	0.0585	0.5927	0.1163	0.1700	0.3784	**0.0270**
	6	Ave	25.0762	24.7413	25.1943	23.5879	24.9502	25.1228	24.7550	**25.3021**
		Std	0.3315	0.3935	0.2412	0.7636	0.3529	0.2672	0.3721	**0.0929**
	8	Ave	26.9601	26.5835	27.0003	25.4044	26.7132	26.6869	26.5320	**27.4439**
		Std	0.3644	0.3530	0.3017	0.7246	0.3923	0.5373	0.3931	**0.1670**
	10	Ave	28.4526	27.9390	28.3402	26.4258	27.8525	28.2296	27.8216	**29.0811**
		Std	0.3701	0.3439	0.4948	0.7156	0.5970	0.4880	0.4451	**0.1862**
Terrace	4	Ave	21.4487	21.3472	21.4543	20.2352	21.4414	21.3513	21.3498	**21.4807**
		Std	0.0912	0.1680	0.0446	0.5816	0.0605	0.1477	0.1189	**0.0057**
	6	Ave	23.8957	23.4246	23.7389	22.0499	23.6955	23.6429	23.4807	**24.0005**
		Std	0.1441	0.3004	0.2232	0.6379	0.2887	0.2593	0.3648	**0.0381**
	8	Ave	25.5812	24.9050	25.3399	23.7264	25.1479	25.2346	25.1673	**25.8515**
		Std	0.2904	0.4396	0.3309	0.5738	0.4494	0.3141	0.2904	**0.0847**
	10	Ave	26.9638	26.1866	26.6473	24.4053	26.4007	26.3419	26.2958	**27.3360**
		Std	0.3448	0.4855	0.3269	0.8057	0.4211	0.3865	0.3352	**0.1720**
Friedman-Rank			2.82	5.52	3.74	7.94	4.33	4.71	5.40	**1.54**
Final-Rank			2	7	3	8	4	5	6	**1**

Table 13. Ave and Std of all test images for FSIM in Otsu.

Images	TH	Metrics	GWO	IWOA	AGPSO	HSO	DBO	BPBO	GJO	MGJO
Baboon	4	Ave	0.8176	**0.8218**	0.8196	0.8040	0.8187	0.8098	0.8184	0.8200
		Std	0.0066	0.0107	0.0046	0.0274	0.0062	0.0088	0.0116	**0.0014**
	6	Ave	0.8795	0.8775	0.8840	0.8459	0.8760	0.8613	0.8755	**0.8847**
		Std	0.0136	0.0209	0.0092	0.0250	0.0224	0.0160	0.0201	**0.0042**
	8	Ave	0.9092	0.9043	0.9092	0.8759	0.9012	0.8846	0.9074	**0.9198**
		Std	0.0140	0.0195	0.0137	0.0207	0.0202	0.0192	0.0186	**0.0087**
	10	Ave	0.9264	0.9230	0.9280	0.9050	0.9295	0.9091	0.9279	**0.9363**
		Std	0.0139	0.0150	0.0141	0.0237	0.0170	0.0161	0.0171	**0.0082**
Camera	4	Ave	0.8350	0.8327	**0.8361**	0.8185	0.8318	0.8311	0.8288	0.8358
		Std	0.0078	0.0085	0.0059	0.0138	0.0084	0.0074	0.0096	**0.0044**
	6	Ave	0.8700	0.8661	0.8746	0.8473	0.8671	0.8624	0.8664	**0.8770**
		Std	0.0088	0.0118	0.0061	0.0150	0.0088	0.0082	0.0133	**0.0033**
	8	Ave	0.8881	0.8868	0.8960	0.8686	0.8864	0.8823	0.8870	**0.9006**
		Std	0.0106	0.0118	0.0073	0.0173	0.0103	0.0106	0.0096	**0.0039**
	10	Ave	0.9077	0.9016	0.9058	0.8823	0.9014	0.8938	0.9037	**0.9148**
		Std	0.0063	0.0113	0.0078	0.0150	0.0100	0.0098	0.0111	**0.0040**
Face	4	Ave	0.7519	0.7518	0.7536	0.7300	0.7526	**0.7541**	0.7476	0.7539
		Std	0.0054	0.0059	0.0021	0.0196	0.0043	0.0052	0.0128	**0.0009**
	6	Ave	0.8285	0.8249	0.8363	0.7929	0.8304	0.8321	0.8119	**0.8434**
		Std	0.0132	0.0160	0.0065	0.0223	0.0123	0.0109	0.0162	**0.0014**
	8	Ave	0.8742	0.8648	0.8762	0.8289	0.8700	0.8729	0.8577	**0.8912**
		Std	0.0147	0.0107	0.0092	0.0204	0.0102	0.0108	0.0118	**0.0035**
	10	Ave	0.8936	0.8875	0.9043	0.8602	0.8880	0.8962	0.8829	**0.9203**
		Std	0.0126	0.0149	0.0076	0.0160	0.0205	0.0118	0.0136	**0.0036**
Girl	4	Ave	0.8283	0.8255	0.8286	0.8019	0.8278	0.8281	0.8262	**0.8295**
		Std	0.0039	0.0095	0.0067	0.0141	0.0043	0.0069	0.0067	**0.0009**
	6	Ave	0.8668	0.8612	0.8666	0.8402	0.8678	0.8610	0.8643	**0.8691**
		Std	0.0049	0.0095	0.0053	0.0154	0.0059	0.0072	0.0073	**0.0045**
	8	Ave	0.8979	0.8887	0.8971	0.8629	0.8914	0.8883	0.8906	**0.9042**
		Std	0.0078	0.0080	0.0058	0.0153	0.0085	0.0099	0.0082	**0.0024**
	10	Ave	0.9167	0.9039	0.9155	0.8760	0.9090	0.9070	0.9097	**0.9274**
		Std	0.0065	0.0093	0.0059	0.0212	0.0092	0.0083	0.0073	**0.0023**
Hunter	4	Ave	0.8502	0.8486	0.8508	0.8150	0.8501	0.8466	0.8477	**0.8526**
		Std	0.0030	0.0036	0.0025	0.0200	0.0030	0.0052	0.0038	**0.0009**
	6	Ave	0.9022	0.8953	0.9028	0.8641	0.8993	0.8969	0.8979	**0.9065**
		Std	0.0060	0.0052	0.0030	0.0169	0.0065	0.0063	0.0047	**0.0019**
	8	Ave	0.9313	0.9193	0.9288	0.8910	0.9215	0.9206	0.9236	**0.9364**
		Std	0.0042	0.0069	0.0052	0.0135	0.0081	0.0065	0.0069	**0.0021**
	10	Ave	0.9440	0.9345	0.9415	0.9037	0.9349	0.9348	0.9377	**0.9529**
		Std	0.0053	0.0082	0.0070	0.0140	0.0070	0.0075	0.0067	**0.0024**
Lena	4	Ave	0.7790	0.7792	**0.7816**	0.7608	0.7802	0.7758	0.7775	0.7800
		Std	0.0032	0.0038	0.0021	0.0197	0.0024	0.0057	0.0051	**0.0013**
	6	Ave	0.8403	0.8346	0.8406	0.8102	0.8367	0.8313	0.8328	**0.8486**
		Std	0.0102	0.0097	0.0102	0.0144	0.0099	0.0093	0.0137	**0.0036**
	8	Ave	0.8705	0.8653	0.8737	0.8409	0.8671	0.8612	0.8627	**0.8811**
		Std	0.0103	0.0116	0.0084	0.0173	0.0087	0.0118	0.0096	**0.0050**
	10	Ave	0.8897	0.8863	0.8934	0.8642	0.8841	0.8817	0.8818	**0.9015**
		Std	0.0092	0.0088	0.0063	0.0154	0.0099	0.0106	0.0116	**0.0052**
Saturn	4	Ave	0.8472	0.8490	0.8482	0.8456	0.8487	**0.8507**	0.8481	0.8480
		Std	0.0037	0.0062	0.0019	0.0116	0.0035	0.0045	0.0074	**0.0005**
	6	Ave	0.8827	0.8788	0.8840	0.8688	0.8821	0.8847	0.8810	**0.8849**
		Std	0.0049	0.0073	0.0046	0.0130	0.0076	0.0058	0.0071	**0.0025**
	8	Ave	0.9086	0.9049	0.9085	0.8900	0.9047	0.9051	0.9048	**0.9126**
		Std	0.0061	0.0073	0.0046	0.0122	0.0072	0.0070	0.0050	**0.0035**
	10	Ave	0.9258	0.9188	0.9246	0.9031	0.9175	0.9227	0.9173	**0.9317**
		Std	0.0050	0.0056	0.0060	0.0108	0.0076	0.0073	0.0072	**0.0021**
Terrace	4	Ave	0.8433	0.8397	0.8441	0.8064	0.8430	0.8388	0.8410	**0.8446**
		Std	0.0040	0.0086	0.0023	0.0211	0.0040	0.0071	0.0050	**0.0006**
	6	Ave	0.9016	0.8895	0.8983	0.8561	0.8986	0.8855	0.8922	**0.9044**
		Std	0.0048	0.0107	0.0059	0.0191	0.0067	0.0092	0.0110	**0.0020**
	8	Ave	0.9286	0.9178	0.9264	0.8915	0.9219	0.9170	0.9227	**0.9349**
		Std	0.0100	0.0107	0.0075	0.0151	0.0068	0.0083	0.0083	**0.0043**
	10	Ave	0.9441	0.9367	0.9417	0.9050	0.9367	0.9311	0.9386	**0.9534**
		Std	0.0066	0.0099	0.0066	0.0175	0.0102	0.0103	0.0054	**0.0041**
Friedman-Rank			3.29	4.96	3.68	7.72	4.21	5.62	4.60	**1.93**
Final-Rank			2	6	3	8	4	7	5	**1**

Table 14. Ave and Std of all test images for SSIM in Otsu.

Images	TH	Metrics	GWO	IWOA	AGPSO	HSO	DBO	BPBO	GJO	MGJO
Baboon	4	Ave	0.7189	**0.7276**	0.7233	0.6908	0.7208	0.7033	0.7174	0.7251
		Std	0.0123	0.0169	0.0084	0.0512	0.0122	0.0182	0.0239	**0.0028**
	6	Ave	0.8238	0.8166	0.8315	0.7672	0.8158	0.7976	0.8168	**0.8327**
		Std	0.0218	0.0289	0.0152	0.0397	0.0355	0.0271	0.0294	**0.0081**
	8	Ave	0.8710	0.8612	0.8717	0.8175	0.8571	0.8368	0.8676	**0.8856**
		Std	0.0195	0.0246	0.0175	0.0302	0.0311	0.0278	0.0247	**0.0123**
	10	Ave	0.8979	0.8901	0.8970	0.8633	0.9011	0.8753	0.9000	**0.9088**
		Std	0.0168	0.0200	0.0185	0.0339	0.0206	0.0229	0.0218	**0.0105**
Camera	4	Ave	0.6960	0.7248	0.7008	0.6702	0.7026	0.6940	0.6895	**0.7248**
		Std	0.0407	0.0442	**0.0296**	0.0420	0.0387	0.0350	0.0403	0.0344
	6	Ave	0.7821	0.7796	0.7987	0.7427	0.7795	0.7466	0.7825	**0.8013**
		Std	0.0303	0.0351	0.0155	0.0612	0.0372	0.0365	0.0371	**0.0075**
	8	Ave	0.8154	0.8117	0.8296	0.7861	0.8198	0.8015	0.8171	**0.8311**
		Std	0.0349	0.0329	0.0216	0.0585	0.0322	0.0278	0.0292	**0.0106**
	10	Ave	0.8404	0.8355	0.8393	0.8183	0.8461	0.8190	0.8457	**0.8525**
		Std	0.0206	0.0320	0.0236	0.0410	0.0277	0.0239	0.0284	**0.0093**
Face	4	Ave	0.7033	0.7056	0.7064	0.6829	0.7055	**0.7075**	0.7006	0.7066
		Std	0.0104	0.0121	0.0043	0.0266	0.0093	0.0100	0.0176	**0.0019**
	6	Ave	0.7859	0.7804	0.7930	0.7546	0.7882	0.7903	0.7690	**0.7996**
		Std	0.0131	0.0209	0.0083	0.0294	0.0138	0.0133	0.0190	**0.0019**
	8	Ave	0.8396	0.8284	0.8388	0.7935	0.8340	0.8381	0.8220	**0.8537**
		Std	0.0157	0.0122	0.0100	0.0239	0.0113	0.0114	0.0135	**0.0041**
	10	Ave	0.8617	0.8526	0.8730	0.8281	0.8556	0.8661	0.8503	**0.8890**
		Std	0.0128	0.0182	0.0088	0.0207	0.0236	0.0133	0.0156	**0.0047**
Girl	4	Ave	**0.7141**	0.7136	0.7132	0.6763	0.7128	0.7120	0.7110	0.7141
		Std	0.0036	0.0144	0.0110	0.0272	0.0066	0.0099	0.0097	**0.0007**
	6	Ave	0.7530	0.7512	0.7531	0.7280	**0.7652**	0.7473	0.7601	0.7566
		Std	0.0140	0.0211	0.0165	0.0229	0.0196	**0.0138**	0.0206	0.0148
	8	Ave	0.7973	0.7875	0.7925	0.7499	0.8008	0.7804	0.8034	**0.8079**
		Std	0.0152	0.0178	0.0132	0.0236	0.0165	0.0205	0.0196	**0.0106**
	10	Ave	0.8258	0.8135	0.8217	0.7707	0.8265	0.8016	0.8258	**0.8365**
		Std	0.0164	0.0258	0.0171	0.0322	0.0229	0.0134	0.0157	**0.0111**
Hunter	4	Ave	0.6959	0.6976	0.6963	0.6446	0.6961	0.6856	0.6995	**0.7037**
		Std	0.0146	0.0158	0.0125	0.0374	0.0167	0.0181	0.0175	**0.0064**
	6	Ave	0.7722	0.7593	0.7722	0.7133	0.7667	0.7643	0.7706	**0.7771**
		Std	0.0170	0.0172	0.0156	0.0358	0.0198	0.0148	0.0128	**0.0104**
	8	Ave	0.8122	0.8005	0.8111	0.7603	0.8106	0.7947	0.8109	**0.8228**
		Std	0.0109	0.0216	0.0128	0.0283	0.0152	0.0197	0.0162	**0.0091**
	10	Ave	0.8411	0.8304	0.8352	0.7806	0.8373	0.8182	0.8410	**0.8532**
		Std	0.0135	0.0167	0.0195	0.0326	0.0239	0.0176	0.0178	**0.0100**
Lena	4	Ave	0.6743	0.6738	0.6757	0.6575	**0.6759**	0.6728	0.6727	0.6756
		Std	0.0044	0.0046	0.0015	0.0242	0.0030	0.0056	0.0057	**0.0007**
	6	Ave	0.7472	0.7458	0.7468	0.7179	0.7417	0.7366	0.7413	**0.7568**
		Std	0.0179	0.0173	0.0142	0.0206	0.0161	0.0133	0.0239	**0.0055**
	8	Ave	0.7930	0.7857	0.7959	0.7624	0.7837	0.7813	0.7893	**0.8053**
		Std	0.0225	0.0223	0.0198	0.0324	0.0162	0.0189	0.0272	**0.0158**
	10	Ave	0.8207	0.8227	0.8303	0.7902	0.8174	0.8059	0.8151	**0.8364**
		Std	0.0197	0.0221	0.0190	0.0269	0.0239	0.0191	0.0286	**0.0141**
Saturn	4	Ave	0.8302	0.8316	0.8316	0.8236	0.8319	**0.8343**	0.8303	0.8310
		Std	0.0040	0.0076	0.0034	0.0137	0.0049	0.0056	0.0105	**0.0007**
	6	Ave	0.8752	0.8694	0.8783	0.8561	0.8739	0.8748	0.8729	**0.8806**
		Std	0.0060	0.0094	0.0058	0.0162	0.0107	0.0074	0.0076	**0.0023**
	8	Ave	0.9024	0.8981	0.9023	0.8822	0.8983	0.8966	0.8983	**0.9069**
		Std	0.0065	0.0094	0.0054	0.0138	0.0087	0.0090	0.0059	**0.0027**
	10	Ave	0.9213	0.9144	0.9195	0.8978	0.9105	0.9167	0.9119	**0.9267**
		Std	0.0058	0.0057	0.0061	0.0099	0.0092	0.0063	0.0085	**0.0025**
Terrace	4	Ave	0.7184	0.7145	**0.7195**	0.6644	0.7184	0.7101	0.7139	0.7190
		Std	0.0048	0.0144	0.0053	0.0344	0.0070	0.0155	0.0101	**0.0017**
	6	Ave	0.8010	0.7915	0.7954	0.7296	0.8020	0.7825	0.7955	**0.8054**
		Std	0.0120	0.0244	0.0137	0.0339	0.0142	0.0163	0.0189	**0.0053**
	8	Ave	0.8452	0.8329	0.8393	0.7892	0.8383	0.8232	0.8458	**0.8543**
		Std	0.0166	0.0233	0.0177	0.0300	0.0175	0.0166	0.0173	**0.0129**
	10	Ave	0.8735	0.8659	0.8714	0.8160	0.8661	0.8540	0.8773	**0.8894**
		Std	0.0147	0.0210	0.0160	0.0347	0.0210	0.0194	0.0174	**0.0112**
Friedman-Rank			3.58	4.48	4.14	7.55	4.06	5.58	3.97	**2.65**
Final-Rank			2	6	5	8	4	7	3	**1**



Figure 10. Friedman average rank of fitness in Otsu.

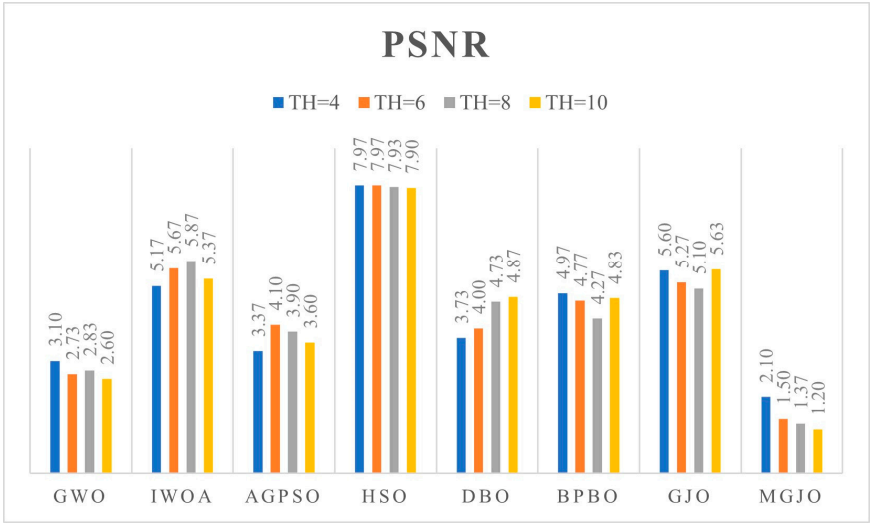


Figure 11. Friedman average rank of PSNR in Otsu.

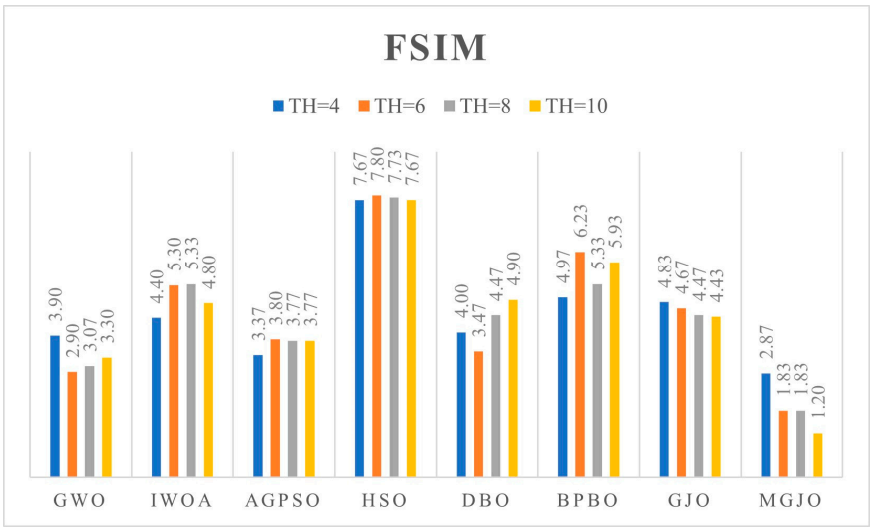


Figure 12. Friedman average rank of FSIM in Otsu.

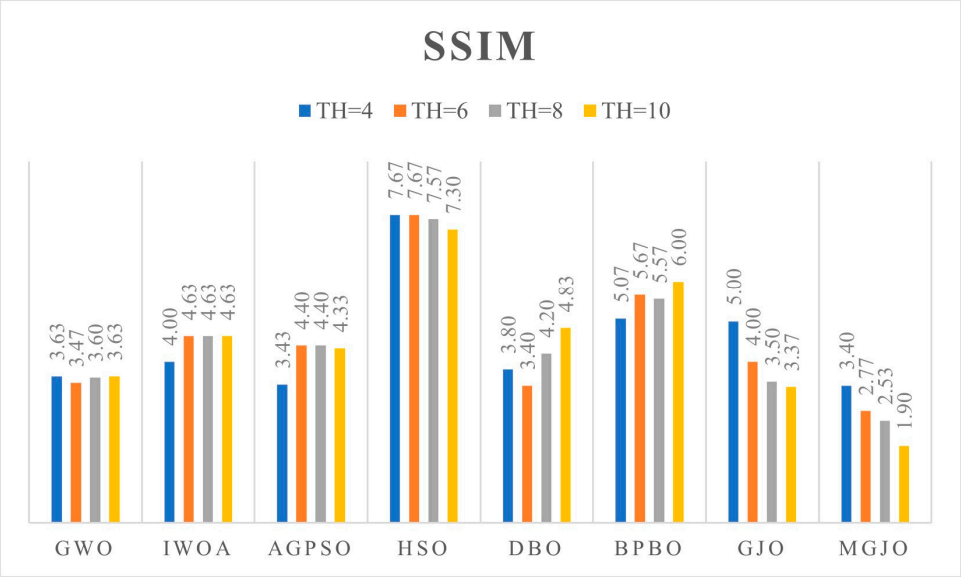


Figure 13. Friedman average rank of SSIM in Otsu.

Figures 10–13, showing the Friedman average rankings, provide a comprehensive validation of MGJO's superiority. In Figure 10 (fitness ranking), MGJO consistently ranks first across all thresholds, far ahead of GJO, which mostly ranks 5–6. In Figure 11 (PSNR ranking), MGJO maintains the top position with an average rank of 1.54, while the next best, GWO, only achieves 2.82. In Figure 12 (FSIM ranking) and Figure 13 (SSIM ranking), MGJO achieves average ranks of 1.93 and 2.65, respectively, consistently outperforming all other algorithms. Notably, as the threshold number increases (TH = 8, 10), MGJO's ranking advantage is not diminished; rather, the synergistic effects of the double-crossover strategy and the global-optimum-based boundary handling mechanism make its superiority even more pronounced under high-threshold conditions. These findings further confirm MGJO's overall excellence in multilevel image segmentation tasks, providing strong support for its application in practical image segmentation scenarios.

5. Summary and Prospect

This study addresses several limitations of the standard Golden Jackal Optimization (GJO) algorithm, including uneven population initialization, insufficient information exchange among individuals, and passive boundary handling, by proposing an improved GJO algorithm (MGJO) and applying it to multilevel image thresholding tasks. At the algorithmic level, MGJO employs a good-point-set population initialization strategy, which enables a wider and more uniform distribution of the initial population within the search space, overcoming the limited exploration range of traditional random initialization. A double-crossover strategy, integrating horizontal and vertical crossovers, not only promotes information sharing among individuals within the population but also achieves fine-grained, dimension-level search, enhancing optimization accuracy in high-dimensional problems. Meanwhile, the global-optimum-based boundary handling mechanism replaces the traditional "resetting-boundary" passive approach, preserving the search direction information of individuals while improving search efficiency near boundary regions.

Comparative experiments on the CEC2017 (dimensions 30 and 100) and CEC2022 (dimensions 10 and 20) test suites against seven mainstream optimization algorithms demonstrate that MGJO excels in optimization accuracy, stability, and exploration–exploitation balance. Statistical analyses using Wilcoxon rank-sum tests and Friedman mean rank tests further confirm the reliability of its performance. In the application of multilevel image

thresholding, with Otsu's method as the objective function, MGJO achieves higher fitness values in 4-, 6-, 8-, and 10-level thresholding tasks on eight benchmark artistic images. The segmented images exhibit superior quality in terms of PSNR, FSIM, and SSIM metrics compared with other algorithms, and MGJO consistently ranks first in Friedman average rankings, fully demonstrating its practical effectiveness in image segmentation tasks.

Despite the outstanding performance of MGJO in numerical optimization and image segmentation, there remains room for further improvement and extension. Future research may proceed in several directions. In terms of algorithmic performance, the current MGJO parameters are mostly fixed or linearly varied; adaptive parameter adjustment mechanisms can be designed using reinforcement learning or chaos theory to adjust parameters in real time according to the characteristics of the optimization problem. MGJO can also be extended to multi-objective optimization to address complex scenarios with multiple conflicting objectives. In terms of application scenarios, the algorithm can be tailored to medical image segmentation by optimizing objective functions for images with uneven grayscale distribution or high noise, supporting clinical diagnosis. Temporal optimization strategies can also be developed to leverage correlations between video frames, enabling efficient video sequence segmentation. Furthermore, MGJO can be integrated with deep learning models to optimize network hyperparameters or initial weights, forming a hybrid framework. Lightweight adaptations and hardware acceleration techniques may be applied for deployment on embedded devices. From a theoretical perspective, establishing a convergence analysis framework would provide rigorous support for algorithmic improvements and further enhance the value of MGJO in engineering optimization problems.

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

Funding: This research received no external funding.

Data Availability Statement: All data in this paper are included in the manuscript.

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

References

1. Zhao, S.; Meng, F.; Cai, L.; Yang, R. Boomerang aerodynamic ellipse optimizer: A human game-inspired optimization technique for numerical optimization and multilevel thresholding image segmentation. *Math. Comput. Simul.* **2025**, *238*, 604–636. [CrossRef]
2. Wang, X.; Snášel, V.; Mirjalili, S.; Pan, J.-S. MAAPO: An innovative membrane algorithm based on artificial protozoa optimizer for multilevel threshold image segmentation. *Artif. Intell. Rev.* **2025**, *58*, 324. [CrossRef]
3. Rao, H.; Jia, H.; Zhang, X.; Abualigah, L. Hybrid Adaptive Crayfish Optimization with Differential Evolution for Color Multi-Threshold Image Segmentation. *Biomimetics* **2025**, *10*, 218. [CrossRef]
4. Huo, Y.; Gang, S.; Guan, C. FCIHMRT: Feature Cross-Layer Interaction Hybrid Method Based on Res2Net and Transformer for Remote Sensing Scene Classification. *Electronics* **2023**, *12*, 4362. [CrossRef]
5. Tamilarasan, A.; Rajamani, D. Towards efficient image segmentation: A fuzzy entropy-based approach using the snake optimizer algorithm. *Results Eng.* **2025**, *26*, 105335. [CrossRef]
6. Ramos-Frutos, J.; Oliva, D.; Miguel-Andrés, I.; Casas-Ordaz, A.; Ramos-Soto, O.; Aranguren, I.; Zapotecas-Martínez, S. Multi-population estimation of distribution algorithm for multilevel thresholding in image segmentation. *Neurocomputing* **2025**, *641*, 130325. [CrossRef]
7. Qiao, Z.; Wu, L.; Heidari, A.A.; Zhao, X.; Chen, H. An enhanced tree-seed algorithm for global optimization and neural architecture search optimization in medical image segmentation. *Biomed. Signal Process. Control* **2025**, *104*, 107457. [CrossRef]
8. Guo, H.; Wang, J.G.; Liu, Y. Multi-threshold image segmentation algorithm based on Aquila optimization. *Vis. Comput.* **2023**, *40*, 2905–2932. [CrossRef]

9. Zheng, J.; Gao, Y.; Zhang, H.; Lei, Y.; Zhang, J. OTSU Multi-Threshold Image Segmentation Based on Improved Particle Swarm Algorithm. *Appl. Sci.* **2022**, *12*, 11514. [CrossRef]
10. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.
11. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
12. Abdel-Basset, M.; Mohamed, R.; Abouhawwash, M. Crested Porcupine Optimizer: A new nature-inspired metaheuristic. *Knowl.-Based Syst.* **2024**, *284*, 111257.
13. Dorigo, M.; Birattari, M.; Stützle, T. Ant Colony Optimization. *Comput. Intell. Mag. IEEE* **2006**, *1*, 28–39.
14. Yang, X.-S.; He, X. Bat algorithm: Literature review and applications. *Int. J. Bio-Inspired Comput.* **2013**, *5*, 141–149. [CrossRef]
15. Wang, R.-B.; Hu, R.-B.; Geng, F.-D.; Xu, L.; Chu, S.-C.; Pan, J.-S.; Meng, Z.-Y.; Mirjalili, S. The Animated Oat Optimization Algorithm: A nature-inspired metaheuristic for engineering optimization and a case study on Wireless Sensor Networks. *Knowl.-Based Syst.* **2025**, *318*, 113589.
16. Hashim, F.A.; Hussien, A.G. Snake Optimizer: A novel meta-heuristic optimization algorithm. *Knowl. Based Syst.* **2022**, *242*, 108320.
17. Xue, J.; Shen, B. A novel swarm intelligence optimization approach: Sparrow search algorithm. *Syst. Sci. Control Eng.* **2020**, *8*, 22–34. [CrossRef]
18. Saremi, S.; Mirjalili, S.; Lewis, A. Grasshopper Optimisation Algorithm: Theory and application. *Adv. Eng. Softw.* **2017**, *105*, 30–47. [CrossRef]
19. Jia, H.; Peng, X.; Lang, C. Remora optimization algorithm. *Expert Syst. Appl.* **2021**, *185*, 115665. [CrossRef]
20. Hayyolalam, V.; Kazem, A.A.P. Black widow optimization algorithm: A novel meta-heuristic approach for solving engineering optimization problems. *Eng. Appl. Artif. Intell.* **2020**, *87*, 103249. [CrossRef]
21. Xue, J.; Shen, B. Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. *J. Supercomput.* **2022**, *79*, 7305–7336. [CrossRef]
22. Mohammadi-Balani, A.; Nayeri, M.D.; Azar, A.; Taghizadeh-Yazdi, M. Golden eagle optimizer: A nature-inspired metaheuristic algorithm. *Comput. Ind. Eng.* **2021**, *152*, 107050. [CrossRef]
23. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191. [CrossRef]
24. Khishe, M.; Mosavi, M.R. Chimp optimization algorithm. *Expert Syst. Appl.* **2020**, *149*, 113338. [CrossRef]
25. Wang, L.; Cao, Q.; Zhang, Z.; Mirjalili, S.; Zhao, W. Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems. *Eng. Appl. Artif. Intell.* **2022**, *114*, 105082. [CrossRef]
26. Aziz, M.A.E.; Ewees, A.A.; Hassanien, A.E. Whale Optimization Algorithm and Moth-Flame Optimization for multilevel thresholding image segmentation. *Expert Syst. Appl.* **2017**, *83*, 242–256. [CrossRef]
27. SMookiah; Parasuraman, K.; Chandar, S.K. Color image segmentation based on improved sine cosine optimization algorithm. *Soft Comput.* **2022**, *26*, 13193–13203. [CrossRef]
28. Aranguren, I.; Valdivia, A.; Morales-Castañeda, B.; Oliva, D.; Elaziz, M.A.; Perez-Cisneros, M. Improving the segmentation of magnetic resonance brain images using the LSHADE optimization algorithm. *Biomed. Signal Process. Control* **2021**, *64*, 102259. [CrossRef]
29. Dhal, K.G.; Das, A.; Ray, S.; Gálvez, J.; Das, S. Nature-Inspired Optimization Algorithms and Their Application in Multi-Thresholding Image Segmentation. *Arch. Comput. Methods Eng.* **2019**, *27*, 855–888. [CrossRef]
30. Jiang, Z.; Zou, F.; Chen, D.; Cao, S.; Liu, H.; Guo, W. An ensemble multi-swarm teaching–learning-based optimization algorithm for function optimization and image segmentation. *Appl. Soft Comput.* **2022**, *130*, 109653. [CrossRef]
31. Elaziz, M.A.; Al-qaness, M.A.A.; Al-Betar, M.A.; Ewees, A.A. Polyp image segmentation based on improved planet optimization algorithm using reptile search algorithm. *Neural Comput. Appl.* **2025**, *37*, 6327–6349. [CrossRef]
32. Premalatha, R.; Dhanalakshmi, P. An innovative segmentation algorithm based on enhanced fuzzy optimization of skin cancer images. *Multimed. Tools Appl.* **2025**, *84*, 40425–40448. [CrossRef]
33. Abdel-Salam, M.; Houssein, E.H.; Emam, M.M.; Samee, N.A.; Gharehchopogh, F.S.; Bacanin, N. EATHOA: Elite-evolved hiking algorithm for global optimization and precise multi-thresholding image segmentation in intracerebral hemorrhage images. *Comput. Biol. Med.* **2025**, *196*, 110835. [CrossRef]
34. Chopra, N.; Ansari, M.M. Golden jackal optimization: A novel nature-inspired optimizer for engineering applications. *Expert Syst. Appl.* **2022**, *198*, 116924. [CrossRef]
35. Yang, W.; Lai, T.; Fang, Y. Multi-Strategy Golden Jackal Optimization for engineering design. *J. Supercomput.* **2025**, *81*, 1–60. [CrossRef]
36. Mohapatra, S.; Mohapatra, P. Fast random opposition-based learning Golden Jackal Optimization algorithm. *Knowl. Based Syst.* **2023**, *275*, 110679. [CrossRef]

37. Hu, G.; Chen, L.; Wei, G. Enhanced golden jackal optimizer-based shape optimization of complex CSGC-Ball surfaces. *Artif. Intell. Rev.* **2023**, *56*, 2407–2475. [CrossRef]
38. Devi, R.M.; Premkumar, M.; Kiruthiga, G.; Sowmya, R. IGJO: An Improved Golden Jackel Optimization Algorithm Using Local Escaping Operator for Feature Selection Problems. *Neural Process. Lett.* **2023**, *55*, 6443–6531. [CrossRef]
39. Bai, J.; Khatir, S.; Abualigah, L.; Wahab, M.A. Ameliorated Golden jackal optimization (AGJO) with enhanced movement and multi-angle position updating strategy for solving engineering problems. *Adv. Eng. Softw.* **2024**, *194*, 103665. [CrossRef]
40. Meng, X.; Tan, L.; Wang, Y. An efficient hybrid differential evolution-golden jackal optimization algorithm for multilevel thresholding image segmentation. *PeerJ Comput. Sci.* **2024**, *10*, e2121. [CrossRef]
41. Mai, X.; Zhong, Y.; Li, L. The Crossover strategy integrated Secretary Bird Optimization Algorithm and its application in engineering design problems. *Electron. Res. Arch.* **2025**, *33*, 471–512. [CrossRef]
42. Fu, Y.; Liu, D.; Fu, S.; Chen, J. Enhanced aquila optimizer based on tent chaotic mapping and new rules. *Sci. Rep.* **2024**, *14*, 3013. [CrossRef] [PubMed]
43. Huang, H.; Wu, R.; Huang, H.; Wei, J.; Han, Z.; Wen, L.; Yuan, Y. Multi-strategy improved artificial rabbit optimization algorithm based on fusion centroid and elite guidance mechanisms. *Comput. Methods Appl. Mech. Eng.* **2024**, *425*, 116915. [CrossRef]
44. Xie, J.; He, J.; Gao, Z.; Wang, S.; Liu, J.; Fan, H.J.H. An enhanced snow ablation optimizer for UAV swarm path planning and engineering design problems. *Heliyon* **2024**, *10*, e37819. [CrossRef]
45. Wu, G.; Mallipeddi, R.; Suganthan, P. Problem Definitions and Evaluation Criteria for the CEC 2017 Competition and Special Session on Constrained Single Objective Real-Parameter Optimization. *Nanyang Technol. Univ. Singap. Tech. Rep.* **2016**, 1–18.
46. Luo, W.; Lin, X.; Li, C.; Yang, S.; Shi, Y. Benchmark functions for CEC 2022 competition on seeking multiple optima in dynamic environments. *arXiv* **2022**, arXiv:2201.00523. [CrossRef]
47. Huang, S.; Liu, D.; Fu, Y.; Chen, J.; He, L.; Yan, J.; Yang, D. Prediction of Self-Care Behaviors in Patients Using High-Density Surface Electromyography Signals and an Improved Whale Optimization Algorithm-Based LSTM Model. *J. Bionic Eng.* **2025**, *22*, 1963–1984. [CrossRef]
48. Mirjalili, S.; Lewis, A.; Sadiq, A.S. Autonomous particles groups for particle swarm optimization. *Arab. J. Sci. Eng.* **2014**, *39*, 4683–4697. [CrossRef]
49. Akbari, E.; Rahimnejad, A.; Gadsden, S.A. Holistic swarm optimization: A novel metaphor-less algorithm guided by whole population information for addressing exploration-exploitation dilemma. *Comput. Methods Appl. Mech. Eng.* **2025**, *445*, 118208. [CrossRef]
50. Ghasemi, M.; Akbari, M.A.; Zare, M.; Mirjalili, S.; Deriche, M.; Abualigah, L. Birds of prey-based optimization (BPBO): A metaheuristic algorithm for optimization. *Evol. Intell.* **2025**, *18*, 88. [CrossRef]
51. Ou, Y.; Qin, F.; Zhou, K.-Q.; Yin, P.-F.; Mo, L.-P.; Zain, A.J.S.M. An improved grey wolf optimizer with multi-strategies coverage in wireless sensor networks. *Symmetry* **2024**, *16*, 286. [CrossRef]
52. Wang, W.-C.; Tian, W.-C.; Xu, D.-M.; Zang, H.-F. Arctic puffin optimization: A bio-inspired metaheuristic algorithm for solving engineering design optimization. *Adv. Eng. Softw.* **2024**, *195*, 103694. [CrossRef]
53. Mohammed, B.O.; Aghdasi, H.S.; Salehpour, P. Dhole optimization algorithm: A new metaheuristic algorithm for solving optimization problems. *Clust. Comput.* **2025**, *28*, 430. [CrossRef]
54. Chai, X.; Wu, Z.; Li, W.; Fan, H.; Sun, X.; Xu, J. Image Segmentation Based on the Optimized K-Means Algorithm with the Improved Hybrid Grey Wolf Optimization: Application in Ore Particle Size Detection. *Sensors* **2025**, *28*, 2785. [CrossRef] [PubMed]
55. Abualigah, L.; Al-Okbi, N.K.; Alomari, S.A.; Almomani, M.H.; Moneam, S.; Yousif, M.A.; Snasel, V.; Saleem, K.; Smerat, A. Optimized image segmentation using an improved reptile search algorithm with Gbest operator for multi-level thresholding. *Sci. Rep.* **2025**, *15*, 12713. [CrossRef]
56. Zhang, Y.; Wang, J.; Zhang, X.; Wang, B. ACPOA: An Adaptive Cooperative Pelican Optimization Algorithm for Global Optimization and Multilevel Thresholding Image Segmentation. *Biomimetics* **2025**, *15*, 596. [CrossRef]
57. Shi, J.; Chen, Y.; Wang, C.; Heidari, A.A.; Liu, L.; Chen, H.; Chen, X.; Sun, L. Multi-threshold image segmentation using new strategies enhanced whale optimization for lupus nephritis pathological images. *Displays* **2024**, *84*, 102799. [CrossRef]
58. Abdel-salam, M.; Houssein, E.H.; Emam, M.M.; Samee, N.A.; Azam, M.T. A novel dynamic Nelder-based Electric Eel Foraging algorithm for global optimization and pathological colorectal cancer image segmentation. *Comput. Biol. Med.* **2025**, *197*, 110982. [CrossRef] [PubMed]

59. Wang, J.; Bei, J.; Song, H.; Zhang, H.; Zhang, P. A whale optimization algorithm with combined mutation and removing similarity for global optimization and multilevel thresholding image segmentation. *Appl. Soft Comput.* **2023**, *137*, 110130. [CrossRef]
60. Jiang, Y.; Yeh, W.-C.; Hao, Z.; Yang, Z. A cooperative honey bee mating algorithm and its application in multi-threshold image segmentation. *Inf. Sci.* **2016**, *369*, 171–183. [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.