biomimetics

MDPI

*Article*

# A Multi-Strategy Enhanced Bionic-Inspired Secretary Bird Optimization Algorithm for Numerical Optimization and Artistic Image Segmentation

**Xuanqi Yuan [1], Jinlu Qin [1] and Xiaohan Zhong [2,3,*]**

[1] School of Packaging Design & Art, Hunan University of Technology, Zhuzhou 412007, China; 14565@hut.edu.cn (X.Y.); 14438@hut.edu.cn (J.Q.)
[2] School of Design, Shunde Polytechnic University, Foshan 528000, China
[3] Department of Design, Chung-Ang University, Seoul 06974, Republic of Korea
* Correspondence: zhannah29@cau.ac.kr

**Abstract**

To address the limitations of the original Secretary Bird Optimization Algorithm (SBOA), such as insufficient population diversity, weak local exploitation ability, and a tendency to become trapped in local optima when solving complex optimization problems, this paper proposes a Multi-Strategy Improved Secretary Bird Optimization Algorithm (MISBOA). First, a chaotic elite initialization strategy is introduced to improve the quality and diversity of the initial population. Second, an adaptive spiral Lévy flight strategy is designed to enhance the balance between global exploration and local exploitation during the iterative process. Third, a dynamic neighborhood-guided mutation strategy is incorporated to maintain population diversity and improve convergence accuracy in the later search stage. To validate the effectiveness of the proposed algorithm, MISBOA is comprehensively evaluated on the IEEE CEC2014, CEC2017, and CEC2020 benchmark suites. Experimental results demonstrate that MISBOA achieves superior convergence speed, optimization accuracy, and robustness compared with several representative metaheuristic algorithms. Furthermore, MISBOA is applied to Otsu-based multilevel threshold image segmentation. The segmentation performance is assessed using PSNR, FSIM, SSIM, and visual quality comparisons. The results indicate that MISBOA can generate more accurate and stable segmentation outcomes, demonstrating its strong potential for solving complex numerical optimization and artistic image segmentation problems.

**Keywords:** metaheuristic algorithm; secretary bird optimization algorithm; multilevel threshold image segmentation; numerical optimization; swarm intelligence algorithms

Check for updates

Academic Editor: Yinggao Yue

Received: 11 May 2026
Revised: 24 May 2026
Accepted: 26 May 2026
Published: 1 June 2026

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

## 1. Introduction

In recent years, driven by the rapid advancements in artificial intelligence, big data, cloud computing, and computer vision technologies, image processing has emerged as one of the most dynamic and practically valuable research directions within the field of information science. Serving as a fundamental basis for image understanding and analysis, image processing techniques are widely applied across a multitude of domains, including medical diagnosis [1], remote sensing [2], intelligent transportation [3], industrial inspection [4], agricultural identification [5], facial recognition [6], and digital media [7]. Particularly against the backdrop of the continuous evolution of deep learning and intelligent perception technologies, there is a growing demand for higher standards regarding image

quality, object recognition accuracy, and automated analysis capabilities [8]; this, in turn, has further propelled the sustained development of image processing-related technologies. Among the myriad tasks within image processing, image segmentation stands out as a core research area in computer vision, primarily because it enables the partitioning of an image into distinct regions characterized by similar features, while simultaneously extracting critical information regarding the boundaries, textures, and structural properties of target objects. High-quality image segmentation results not only enhance the accuracy of subsequent feature extraction and object recognition processes but also significantly boost the overall performance of image analysis systems, thereby attracting increasing attention from researchers.

Image segmentation techniques hold immense significance in practical applications. For instance, in medical image analysis, precise image segmentation enables clinicians to rapidly identify lesion areas, thereby enhancing the efficiency of disease diagnosis [9]. In the realm of remote sensing image processing, image segmentation is utilized for land-cover classification, road extraction, and object monitoring [10]. Within the field of industrial inspection, it facilitates product defect detection and quality assessment [11]. Furthermore, in the domains of artistic image processing and digital media, image segmentation is widely applied in tasks such as style transfer, image restoration, object matting, and the creation of visual effects [12]. Compared to natural images, artistic images typically feature more intricate textural information, richer color variations, and more pronounced characteristics of subjective expression; consequently, their segmentation presents a greater challenge. This is particularly true for artistic images, where distinct regions often lack clear boundaries and frequently contain numerous areas characterized by rich detail, similar color tones, or overlapping textures—conditions that render traditional segmentation methods ill-equipped to yield optimal results. Consequently, a key direction in current research involves exploring how to simultaneously enhance the accuracy and stability of artistic image segmentation while faithfully preserving the image's edge and detailed information.

Currently, researchers have proposed numerous distinct approaches to address the problem of image segmentation, primarily encompassing threshold-based [13], region-based [14], clustering-based [15], edge-detection-based [16], and deep-learning-based methods [17]. Among these, threshold-based segmentation has garnered significant attention due to its simplicity of implementation, computational efficiency, and applicability across a wide variety of image scenarios. In particular, multi-threshold image segmentation methods are capable of effectively extracting information regarding distinct regions within an image, thereby yielding superior segmentation results compared to single-threshold methods [18]. However, as the number of thresholds increases, the image segmentation problem gradually evolves into a complex optimization challenge characterized by high dimensionality, non-linearity, and multi-modality; consequently, traditional exhaustive search methods often suffer from drawbacks such as high computational costs, slow convergence rates, and a susceptibility to becoming trapped in local optima. Therefore, the efficient search for the optimal combination of thresholds has emerged as a critical factor in enhancing image segmentation performance.

To address the aforementioned complex image segmentation problems, researchers have begun exploring the use of optimization algorithms to search for the optimal combination of thresholds, aiming to enhance both the efficiency and accuracy of image segmentation. While traditional optimization methods—such as gradient descent [19], dynamic programming [20], and exhaustive search [21]—can yield satisfactory results when applied to simple, low-dimensional problems, they often suffer from significant drawbacks when confronted with high-dimensional, nonlinear, and multimodal challenges like multi-threshold image segmentation. Specifically, in the context of artistic image

segmentation—characterized by intricate textures and rich color variations—the strong coupling relationships among different thresholds make it exceedingly difficult for traditional methods to achieve satisfactory segmentation results within a finite timeframe.

Against this backdrop, metaheuristic optimization algorithms have gradually emerged as a crucial tool for addressing complex image segmentation problems, owing to their robust global search capabilities, minimal parameter requirements, and reduced reliance on specific problem models. Typically, metaheuristic algorithms simulate natural phenomena—such as biological behaviors, collective cooperation, or evolutionary mechanisms—to iteratively update candidate solutions within the search space, thereby facilitating an efficient search for the optimal solution. Compared to traditional optimization methods, metaheuristic algorithms do not require gradient information and impose fewer constraints regarding the continuity and differentiability of the objective function; consequently, they are particularly well-suited for solving complex, multimodal, and non-differentiable problems.

In recent years, an increasing number of researchers have employed heuristic algorithms to solve image segmentation problems. For instance, to address the problem of multi-threshold image segmentation, Ma et al. proposed an improved multi-threshold image segmentation method based on the Whale Optimization Algorithm [22], utilizing inter-class variance (Otsu's method) as the objective function. Experimental results demonstrate that the proposed RAV-WOA is capable of selecting appropriate thresholds for both grayscale and color images while simultaneously ensuring high efficiency and segmentation quality. To address the image processing task of grayscale multi-level thresholding segmentation, Laith et al. proposed a novel heuristic optimization algorithm that combines the Reptile Search Algorithm with the Salp Swarm Algorithm [23]. This method employs Otsu's variance classification function to determine the optimal thresholds at each level. Experimental results on COVID-19-related images demonstrate that the performance of the proposed algorithm surpasses that of other heuristic optimization algorithms reported in the literature. To address the challenge of processing Alzheimer's disease images—which feature complex textures and rich information content—Yu et al. proposed a novel 3D Rényi entropy model that integrates grayscale intensity [24], nonlocal means, and local entropy information. Simulation results demonstrate the model's outstanding segmentation performance and stability, fully highlighting its immense potential for practical applications in medical image analysis.

The Secretary Bird Optimization Algorithm (SBOA) is a novel population-based metaheuristic algorithm proposed by Fu et al. in 2024, inspired by the survival behaviors of secretary birds in their natural environment [25]. Experimental evaluations—conducted using the CEC-2017 and CEC-2022 benchmark suites, 12 engineering optimization problems, and a 3D path planning task for unmanned aerial vehicles (UAVs)—demonstrated that SBOA is capable of identifying superior solutions at a faster convergence rate, thereby fully showcasing its immense potential for solving real-world optimization problems. Since its inception, the algorithm has garnered widespread attention within the academic community; numerous researchers have subsequently built upon this algorithm, implementing targeted modifications to address their specific optimization challenges and achieve enhanced results. For instance, to address problems related to wireless sensor network deployment and engineering, Meng et al. improved the SBOA by incorporating a differential cooperative search mechanism [26]. Simulation results indicate that, although the improved algorithm exhibits inconsistent performance in high-dimensional multimodal and complex constrained optimization problems, it performs well in the three specific engineering design and wireless sensor deployment problems it was designed to solve. To enhance power generation efficiency, Qin et al. improved the SBOA by incorporating a feedback regulation

mechanism based on incremental PID control [27], a collaborative camouflage strategy, and a cosine-similarity-based update strategy. When applied to the shape optimization problem of Combined Quartic Generalized Spherical Interpolation (CQGBI) curves, this method enables the design of smoother curve shapes—based on parameters derived from the MISBOA approach—thereby boosting power generation efficiency. To address the limitations of existing TDOA-based localization algorithms, Liao et al. proposed an improved Secretary bird Optimization Algorithm for target localization [28]. Simulation results demonstrate that, compared with traditional mathematical algorithms and other meta-heuristic algorithms, this algorithm exhibits superior robustness and localization accuracy across various simulation scenarios.

Although numerous metaheuristic algorithms have been proposed in recent decades, not all algorithms exhibit sufficient adaptability and scalability for complex optimization tasks such as multilevel image segmentation. Compared with many conventional metaphor-based optimizers that rely on relatively simple update mechanisms, the Secretary Bird Optimization Algorithm (SBOA) possesses a distinctive multi-stage search framework inspired by prey searching, exhausting, attacking, and escape behaviors. These mechanisms naturally correspond to different optimization phases, including global exploration, transition search, local exploitation, and diversity maintenance. Such characteristics provide SBOA with stronger behavioral interpretability and greater flexibility for balancing exploration and exploitation.

In addition, SBOA is a recently developed optimization algorithm with considerable research potential. Although it has demonstrated promising performance in several engineering optimization tasks, its search mechanism still suffers from several limitations, including insufficient population diversity, weak local exploitation ability, and premature convergence when solving high-dimensional multimodal problems. These shortcomings become particularly significant in multilevel threshold image segmentation problems, where the search space is highly nonlinear, strongly coupled, and contains numerous local optima.

Therefore, SBOA provides a meaningful and extensible optimization framework for further investigation. Motivated by its unique staged-search mechanism and existing limitations, this study develops an enhanced SBOA to improve population diversity, strengthen exploration–exploitation balance, and enhance convergence robustness for complex numerical optimization and artistic image segmentation tasks.

However, according to the No Free Lunch Theorem, no single optimization algorithm can consistently maintain optimal performance across all optimization problems. In other words, the fact that an algorithm performs exceptionally well on certain problems does not guarantee that it will possess a similar advantage on others. Different optimization problems often feature distinct search space structures, variable dimensions, constraints, and objective function characteristics; consequently, they necessitate the adoption of search mechanisms and update strategies that are specifically tailored to the problem's unique properties. Complex numerical optimization and artistic image segmentation problems, in particular, are frequently characterized by high dimensionality, multi-modality, nonlinearity, and a susceptibility to becoming trapped in local optima; under such conditions, relying solely on the singular search mechanism of a standard algorithm makes it difficult to simultaneously balance global exploration capabilities with local exploitation capabilities. Therefore, building upon the SBOA framework, this paper introduces a suite of enhancement strategies designed to boost population diversity and strengthen the algorithm's ability to escape local optima. By further improving its convergence speed, optimization accuracy, and robustness, these enhancements render the algorithm more effectively suited for addressing image segmentation problems. The specific contributions of this paper are as follows:

1. A chaotic elite initialization strategy is proposed to improve the quality of the initial population. By combining Tent chaotic mapping with elite-guided population reconstruction, the proposed strategy enhances the uniformity and diversity of the population distribution, thereby improving the algorithm's ability to locate promising regions in the early search stage.
2. An adaptive spiral Lévy flight strategy is introduced to strengthen the balance between global exploration and local exploitation. By integrating spiral motion with adaptive Lévy flight, the proposed strategy enables the algorithm to maintain wide-range exploration in the early stage and fine local search in the later stage, thereby improving its ability to escape local optima.
3. A dynamic neighborhood-guided mutation strategy is developed to alleviate the problem of premature convergence. By dynamically adjusting the neighborhood radius and combining neighborhood learning with global-best guidance, the proposed strategy effectively maintains population diversity and enhances convergence accuracy in the later search stage.
4. Extensive numerical optimization experiments are conducted on the IEEE CEC2014, CEC2017, and CEC2020 benchmark suites to evaluate the performance of MISBOA. The experimental results demonstrate that MISBOA achieves superior convergence speed, optimization accuracy, and robustness compared with several state-of-the-art metaheuristic algorithms.
5. MISBOA is successfully applied to Otsu-based multilevel threshold image segmentation. Experimental results on multiple benchmark images, evaluated by PSNR, FSIM, SSIM, and visual quality, verify that the proposed algorithm can generate more accurate and stable segmentation results for complex artistic images.

The remainder of this paper is organized as follows. Section 2 introduces the original Secretary Bird Optimization Algorithm and describes the improvement strategies incorporated into the proposed MISBOA in detail. Section 3 presents a comprehensive experimental evaluation of MISBOA on a series of benchmark functions for numerical optimization, demonstrating its effectiveness and robustness. Section 4 applies the proposed MIGSEA to artistic image segmentation tasks and assesses its performance through Peak Signal-to-Noise Ratio, Structural Similarity Index Measure, Feature Similarity Index, and visual quality comparisons. Finally, Section 5 summarizes the main conclusions of this study and discusses potential directions for future research.

## 2. Multi-Strategy Improved SBOA

The Secretary Bird Optimization Algorithm is a recently proposed swarm intelligence optimization method inspired by the hunting behavior of secretary birds in nature. By simulating different stages of prey searching, chasing, attacking, and escaping, SBOA is able to balance global exploration and local exploitation during the optimization process. Owing to its simple structure, few control parameters, and strong search capability, SBOA has shown promising performance in solving a variety of continuous optimization problems. However, when dealing with complex high-dimensional problems, SBOA may still suffer from insufficient population diversity, weak local exploitation ability, and premature convergence, which can limit its optimization accuracy and robustness.

To address these shortcomings, this section first provides a brief introduction to the original SBOA framework and its mathematical model. Subsequently, three effective improvement strategies are proposed to enhance the exploration capability, exploitation performance, and convergence accuracy of the algorithm. By integrating these strategies into the original SBOA, the proposed MISBOA is expected to achieve better optimization

performance and stronger adaptability when solving numerical optimization and artistic image segmentation problems.

### *2.1. Secretary Bird Optimization Algorithm (SBOA)*

The Secretary Bird Optimization Algorithm is a recently proposed swarm intelligence optimization algorithm inspired by the hunting and escape behaviors of secretary birds in nature. Secretary birds are well known for their ability to hunt snakes in grasslands. During hunting, they first search for prey over a wide range, then continuously circle around the prey to weaken its resistance, and finally launch a rapid and accurate attack. In addition, when facing predators, secretary birds can either camouflage themselves using the surrounding environment or quickly escape by running or flying. Based on these biological behaviors, SBOA divides the optimization process into two stages: hunting strategy and escape strategy, corresponding to the exploration and exploitation phases of the algorithm, respectively.

Assume that the population size is $N$, and the dimensionality of the optimization problem is $dim$. In SBOA, each secretary bird represents a candidate solution, and the population can be represented as:

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,Dim} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,Dim} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N,1} & x_{N,2} & \cdots & x_{N,Dim} \end{bmatrix} \tag{1}$$

where $x_{i,j}$ represents the value of the $j$-th dimension of the $i$-th secretary bird. The position of each individual is randomly initialized within the feasible search space as follows:

$$x_{i,j} = lb_j + r \times \left(ub_j - lb_j\right) \tag{2}$$

where $lb_j$ and $ub_j$ denote the lower and upper bounds of the $j$-th dimension, respectively, and $r \in [0, 1]$ is a random number. After initialization, the fitness values of all individuals are calculated according to the objective function:

$$F = \begin{bmatrix} F(X_1) \\ F(X_2) \\ \vdots \\ F(X_N) \end{bmatrix} \tag{3}$$

where $F(X_i)$ denotes the fitness value of the $i$-th individual. In minimization problems, the individual with the smallest fitness value is considered the current best solution.

The search process of SBOA mainly consists of two phases. The first phase is the hunting stage, which is responsible for enhancing the exploration capability of the algorithm. The second phase is the escape stage, which focuses on improving the local exploitation ability and accelerating convergence. By combining these two stages, SBOA attempts to achieve a balance between global exploration and local exploitation.

#### 2.1.1. Hunting Strategy of Secretary Bird

The hunting behavior of secretary birds is divided into three stages: searching for prey, exhausting prey, and attacking prey. These three stages correspond to different search mechanisms in SBOA.

In the first stage, secretary birds search for prey over a wide area. This stage mainly occurs in the early iterations of the algorithm, where population diversity and global

exploration are important. To simulate this process, SBOA updates the position of each individual according to the difference between two randomly selected individuals:

$$x_{i,j}^{new} = x_{i,j} + \left(x_{r1,j} - x_{r2,j}\right) \times R_1 \tag{4}$$

where $x_{r1,j}$ and $x_{r2,j}$ are two randomly selected candidate solutions, and $R_1$ is a random vector in $[0, 1]$. By using the positional difference between individuals, the algorithm can effectively expand the search range and avoid premature convergence. This mechanism improves the diversity of the population and enhances the global search ability of SBOA.

In the second stage, once the prey is found, secretary birds continuously move around it and provoke it in order to consume its energy. To model this behavior, SBOA combines Brownian motion with the current best solution:

$$x_{i,j}^{new} = x_{best,j} + e^{(t/T)^4} \times (RB - 0.5) \times (x_{best,j} - x_{i,j}) \tag{5}$$

where $x_{best,j}$ is the best position found so far, $RB$ denotes Brownian motion, $t$ is the current iteration number, and $T$ is the maximum number of iterations. Brownian motion introduces random local perturbations, enabling the algorithm to search around the best solution more effectively. At the same time, the exponential term dynamically adjusts the search range during the optimization process. This stage strengthens the exploitation ability of the algorithm near promising regions and improves convergence speed.

In the third stage, when the prey becomes weak, secretary birds launch a rapid and accurate attack. SBOA models this process by introducing Lévy flight:

$$x_{i,j}^{new} = x_{best,j} + \left(1 - \frac{t}{T}\right)^{2\frac{t}{T}} x_{i,j} \times RL \tag{6}$$

where $RL$ denotes the weighted Lévy flight step size. Lévy flight is characterized by frequent short-distance movements and occasional long-distance jumps. Therefore, it allows the algorithm to perform local fine search while still maintaining the ability to jump out of local optima. In addition, the nonlinear weight factor dynamically adjusts the influence of Lévy flight according to the iteration process, helping the algorithm achieve a better balance between exploration and exploitation.

#### 2.1.2. Escape Strategy of Secretary Bird

When secretary birds encounter predators, they typically adopt two different escape behaviors. One is to camouflage themselves in the surrounding environment, while the other is to rapidly escape by running or flying. In SBOA, these two behaviors are modeled as:

$$x_{i,j}^{new} = \begin{cases} x_{best,j} + (2 \times RB - 1) \times \left(1 - \frac{t}{T}\right)^2 \times x_{i,j}, & r_i < 0.5 \\ x_{i,j} + R_2 \times \left(x_{random,j} - K \times x_{i,j}\right), & \text{otherwise} \end{cases} \tag{7}$$

where $R_2$ is a random vector, $x_{random,j}$ is a randomly selected solution, and $K$ is a random integer equal to 1 or 2. In the first case, the individual moves toward the vicinity of the current best solution, which helps the algorithm exploit promising regions. In the second case, the individual updates its position according to a randomly selected solution, which enhances the diversity of the population and helps the algorithm avoid local optima. By alternately applying these two strategies, SBOA is able to maintain a dynamic balance between local search and global exploration, thereby improving its convergence accuracy and robustness. The pseudocode of the SBOA is outlined in Algorithm 1.

**Algorithm 1:** the pseudo-code of the SBOA

1: Initialize the population $X_i$ $(i = 1, 2, \ldots, N)$
2: Evaluate the fitness of all individuals
3: Determine the current best solution $X_{best}$
4: For $t = 1 : T$ do
5: For $i = 1 : N$ do
6 : if $t < 1/3\ T$ then
7 : Update $X_i$ using the prey-searching strategy
8 : else if $1/3\ T \leq t < 2/3\ T$ then
9 : Update $X_i$ using the prey-exhausting strategy
10: else
11 : Update $X_i$ using the prey-attacking strategy
12: end if
13: if $rand < 0.5$ then
14 : Update $X_i$ using the camouflage strategy
15: else
16 : Update $X_i$ using the escape strategy
17: end if
18: Evaluate the new fitness of $X_i$
19: End For
20: Update the current best solution $X_{best}$
21: End For
22: Return $X_{best}$

### *2.2. Proposed MISBOA*

It should be noted that the three proposed enhancement strategies are not directly derived from additional biological behaviors of secretary birds. Instead, they are algorithm-oriented improvement mechanisms designed according to the optimization characteristics and structural limitations of the original SBOA framework. The motivation of these strategies mainly comes from improving population diversity, strengthening the balance between global exploration and local exploitation, and alleviating premature convergence in complex optimization problems. Therefore, the proposed strategies should be regarded as computational enhancement mechanisms integrated into the SBOA framework, rather than newly introduced biological metaphors. Nevertheless, these strategies remain compatible with the staged-search structure of SBOA and further strengthen its optimization capability in different search phases.

The three proposed strategies were not introduced as arbitrary combinations of existing techniques, but were specifically designed to address different structural limitations of the original SBOA. In the original SBOA, the prey-searching stage mainly relies on randomly selected individuals, which may lead to insufficient population diversity and uneven search-space coverage during the early optimization stage. Therefore, the chaotic elite initialization strategy was introduced to improve the uniformity and quality of the initial population distribution.

In addition, although the original SBOA employs Lévy flight during the prey-attacking stage, its search trajectory remains relatively simple and lacks adaptive local refinement capability. To strengthen the transition from global exploration to local exploitation, the adaptive spiral Lévy flight strategy was incorporated to generate more flexible search trajectories and enhance exploitation around promising regions.

Furthermore, the original escape mechanism of SBOA mainly depends on random perturbation and best-solution guidance, which may gradually reduce population diversity

in the later search stage and increase the risk of premature convergence. To alleviate this issue, the dynamic neighborhood-guided mutation strategy was developed to maintain cooperative information exchange among neighboring individuals while gradually enhancing convergence toward elite solutions.

Therefore, the three strategies are functionally complementary and correspond to different optimization stages of SBOA, rather than being simple combinations of generic metaheuristic operators.

#### 2.2.1. Chaotic Elite Initialization Strategy

In the original SBOA, the population is randomly initialized, which may lead to an uneven distribution of individuals and insufficient diversity in the early stage. This problem becomes more severe in multilevel threshold image segmentation because the threshold search space is very large and contains many local optima. To improve the quality of the initial population, a chaotic elite initialization strategy is introduced.

First, the Tent chaotic map is used to generate a uniformly distributed chaotic sequence:

$$z_{k+1} = \begin{cases} \frac{z_k}{\mu}, & z_k < \mu \\ \frac{1-z_k}{1-\mu}, & z_k \geq \mu \end{cases} \tag{8}$$

where $z_k \in [0,1]$ is the chaotic variable, and $\mu$ is the control parameter. In this paper, $\mu = 0.6$, since this value provides better randomness and ergodicity. Then, the chaotic sequence is mapped into the search space:

$$x_{i,j} = lb_j + z_{i,j} \times \left(ub_j - lb_j\right) \tag{9}$$

After initialization, the fitness values of all individuals are calculated, and the top 20% of individuals are selected as elite individuals. The remaining individuals are regenerated around these elite solutions:

$$x_{i,j}^{new} = x_{elite,j} + \lambda \times \left(x_{r1,j} - x_{r2,j}\right) \tag{10}$$

where $x_{elite,j}$ is the position of an elite individual, $x_{r1,j}$ and $x_{r2,j}$ are two randomly selected individuals, and $\lambda$ is the perturbation coefficient. In this study, $\lambda = 0.3$.

Through this strategy, the population distribution becomes more uniform and diverse, which helps the algorithm locate promising regions in the early stage and improves the segmentation quality of complex artistic images. Figure 1 illustrates a schematic diagram of the Chaotic Elite Initialization Strategy.

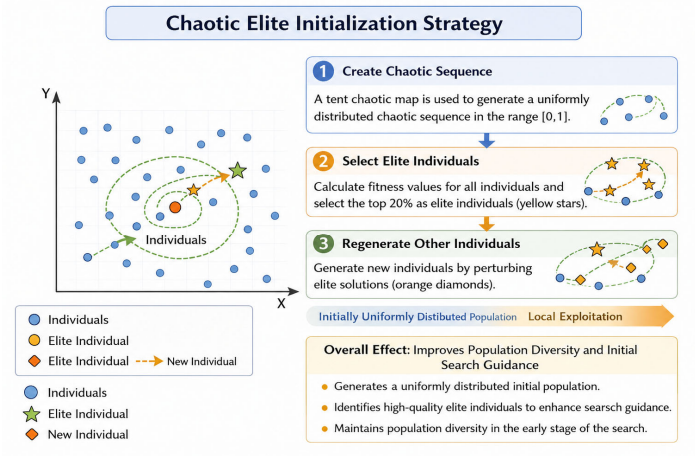



**Figure 1.** Schematic diagram of the Chaotic Elite Initialization Strategy.

#### 2.2.2. Adaptive Spiral Lévy Flight Strategy

In the original SBOA, Lévy flight is only used in the attacking stage, and its search pattern is relatively simple. However, for artistic image segmentation problems, especially multilevel threshold segmentation, the search process often requires both global jumps and fine local adjustment. Therefore, an adaptive spiral Lévy flight strategy is proposed to strengthen the balance between exploration and exploitation.

The position update rule is defined as:

$$x_{i,j}^{new} = x_{best,j} + A \cdot e^{b\theta} \cdot \cos(2\pi\theta) \cdot (x_{best,j} - x_{i,j}) + L \cdot Levy(D) \tag{11}$$

where $A$ is the adaptive weight factor, $b$ is the spiral shape coefficient, $\theta \in [0,1]$ is a random number, and $L$ is the adaptive Lévy flight coefficient. The adaptive weight factor is defined as:

$$A = 2\left(1 - \frac{t}{T}\right) \tag{12}$$

The adaptive Lévy coefficient is defined as:

$$L = 0.5\left(1 - \frac{t}{T}\right) + 0.1 \tag{13}$$

In this paper, the spiral coefficient is set as $b = 1$. In the early stage, larger values of $A$ and $L$ help the algorithm perform wide-range global exploration, while in the later stage, smaller values allow the algorithm to focus on local refinement. Compared with the original Lévy flight mechanism, the proposed strategy can generate more flexible search trajectories and improve the ability of the algorithm to jump out of local optima. Figure 2 illustrates a schematic diagram of the Adaptive Spiral Lévy Flight Strategy.

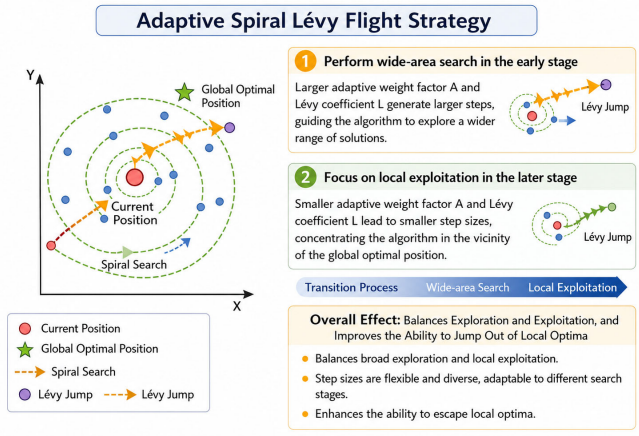



**Figure 2.** Schematic diagram of the Adaptive Spiral Lévy Flight Strategy.

#### 2.2.3. Dynamic Neighborhood-Guided Mutation Strategy

When solving high-dimensional image segmentation problems, the original SBOA may easily lose population diversity in the later stage, causing premature convergence. To address this issue, a dynamic neighborhood-guided mutation strategy is proposed.

First, the neighborhood radius is defined as:

$$R(t) = R_{max}\left(1 - \frac{t}{T}\right) \tag{14}$$

where $R_{max}$ is the maximum neighborhood radius. In this paper, $R_{max} = 0.5 \times (ub - lb)$. For each individual, neighboring individuals within radius $R(t)$ are selected, and the local neighborhood center is calculated as:

$$X_{center} = \frac{1}{M}\sum_{k=1}^{M} X_k \tag{15}$$

where $M$ is the number of neighboring individuals. Then, the mutation strategy is performed as:

$$X_i^{new} = X_i + \alpha(X_{center} - X_i) + \beta(X_{best} - X_i) \tag{16}$$

where $\alpha$ and $\beta$ are two adaptive coefficients:

$$\alpha = 0.5\left(1 - \frac{t}{T}\right), \beta = 0.5\frac{t}{T} \tag{17}$$

At the beginning of the search, a larger $\alpha$ helps individuals learn more from their neighbors and maintain population diversity. In the later stage, a larger $\beta$ makes individuals move closer to the global best solution, thereby improving convergence accuracy.

This strategy allows the algorithm to dynamically adjust the influence of neighboring individuals and elite individuals during the search process, which is especially beneficial for artistic image segmentation tasks with many thresholds and complex texture distributions. Figure 3 illustrates a schematic diagram of the Dynamic Neighborhood-Guided Mutation Strategy. Algorithm 2 shows the pseudocode for MISBOA. Figure 4 shows the execution flowchart of MISBOA.

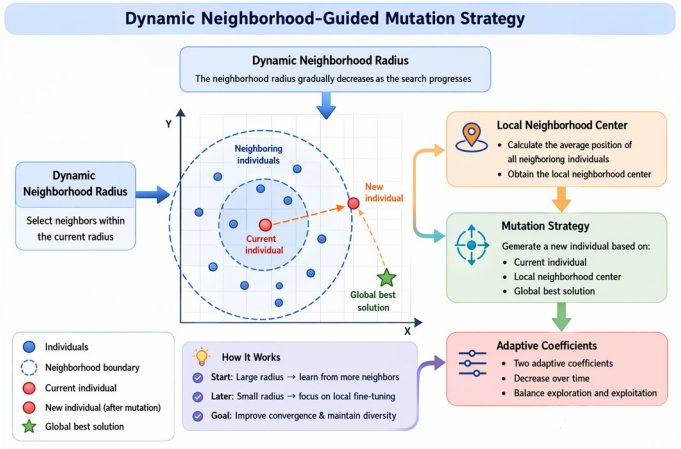



**Figure 3.** Schematic diagram of the Dynamic Neighborhood-Guided Mutation Strategy.

**Algorithm 2:** the pseudo-code of the MISBOA

Initialize Problem Setting ($Dim$, $ub$, $lb$, Pop_size ($N$), Max_Iter ($T$), Current_Iter ($t$))
Initialize the population using Tent chaotic mapping
Evaluate the fitness of all Secretary Birds
Select elite individuals and reconstruct the population
Update Secretary Bird $X_{best}$
For $t = 1 : T$
    For $i = 1 : N$
        Exploration:
        if $t < 1/3\ T$
            Calculate new status of the $i$-th Secretary Bird using prey-searching strategy
            Update the $i$-th Secretary Bird
        else if $1/3\ T < t < 2/3\ T$
            Calculate new status of the $i$-th Secretary Bird using prey-exhausting strategy
            Update the $i$-th Secretary Bird
        else
            Calculate new status of the $i$-th Secretary Bird using adaptive spiral Lévy flight strategy
            Update the $i$-th Secretary Bird
        end if
        Exploitation:
        if $r < 0.5$
            Calculate new status of the $i$-th Secretary Bird using camouflage strategy
        else
            Calculate new status of the $i$-th Secretary Bird using escape strategy
        end if
        Dynamic neighborhood-guided mutation:
        Calculate the dynamic neighborhood radius $R(t)$
        Find neighboring individuals of the $i$-th Secretary Bird
        Calculate the neighborhood center $X_{center}$
        Update the $i$-th Secretary Bird using neighborhood-guided mutation
        Apply boundary control
        Evaluate the fitness of the $i$-th Secretary Bird
    **end for** $i = 1 : N$
    Save best candidate solution so far
**end for** $t = 1 : T$
Output: The best solution obtained by MISBOA for given optimization problem
Return best solution

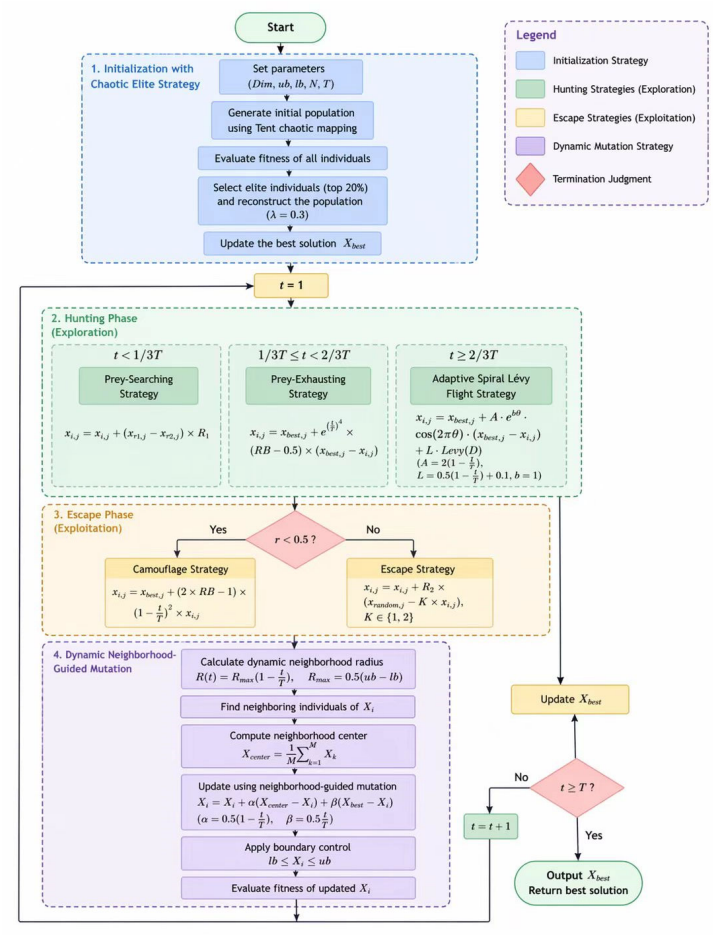



**Figure 4.** MISBOA flowchart.

### *2.3. Time Complexity Analysis of MISBOA*

The computational complexity of an optimization algorithm is an important indicator for evaluating its efficiency and scalability. As per the original SBOA, let $N$ denote the population size, $Dim$ represent the problem dimension, and $T$ be the maximum number of iterations. The time complexity of SBOA mainly consists of population initialization, fitness evaluation, and position updating. Therefore, the overall time complexity of SBOA can be expressed as $O(N \times (T \times Dim + 1))$, which is dominated by the iterative update and fitness calculation processes.

For the proposed MISBOA, three additional strategies are incorporated into the original SBOA framework, namely the chaotic elite initialization strategy, the adaptive spiral Lévy flight strategy, and the dynamic neighborhood-guided mutation strategy. As a result, its computational complexity should be analyzed by considering both the original SBOA operations and the extra cost introduced by these enhancement mechanisms.

First, in the chaotic elite initialization strategy, the Tent chaotic mapping is used to generate the initial population, and then the elite individuals are selected to guide the regeneration of the remaining individuals. The generation of chaotic sequences requires traversing all individuals and dimensions once, resulting in a complexity of $O(N \times Dim)$. The elite selection operation usually requires sorting or ranking the population according to fitness values, whose complexity is $O(NlogN)$. Since initialization is performed only once before the main iteration process, the total additional complexity introduced by this strategy is $O(N \times Dim + NlogN)$.

Second, in the adaptive spiral Lévy flight strategy, the position of each individual is updated using spiral motion and Lévy flight during the iterative process. Since the update

is carried out for each individual in each dimension at every iteration, the additional computational cost of this strategy is $O(T \times N \times Dim)$. It should be noted that Lévy flight itself is essentially a vector generation and scaling operation, so it does not change the order of complexity, but only increases the constant computational overhead.

Third, in the dynamic neighborhood-guided mutation strategy, each individual needs to calculate its neighborhood information and update its position accordingly. If the neighborhood center is estimated by scanning neighboring individuals, the worst-case computational cost for one iteration is $O(N^2 \times Dim)$, because each individual may need to compare itself with all other individuals. Thus, over the whole optimization process, the complexity introduced by this strategy is $O(T \times N^2 \times Dim)$. This term becomes the dominant part of the overall time complexity of MISBOA. Therefore, the total time complexity of MISBOA can be expressed as $O(N \times Dim + N\log N) + O(T \times N \times Dim) + O(T \times N \times N \times Dim)$, which can be further simplified as $O(T \times N^2 \times Dim)$. because the term $O(T \times N^2 \times Dim)$ dominates the remaining lower-order terms.

Compared with the original SBOA, whose complexity is $O(N \times (T \times Dim + 1))$, the proposed MISBOA has a higher computational complexity. The main reason is that the added dynamic neighborhood-guided mutation strategy introduces pairwise neighborhood interactions among individuals, which increases the complexity from approximately linear dependence on $N$ to quadratic dependence on $N$. In other words, SBOA mainly performs individual-level position updates, while MISBOA further considers neighborhood-level cooperative mutation, leading to a greater computational burden.

Nevertheless, although MISBOA requires more computation time than SBOA, this increase is acceptable for two reasons. On the one hand, in practical optimization problems such as multilevel artistic image segmentation, the population size is usually set to a moderate value, so the quadratic term does not become prohibitively expensive. On the other hand, the three proposed strategies significantly enhance population diversity, strengthen the balance between exploration and exploitation, and improve the ability of the algorithm to escape local optima. Therefore, the additional computational cost is exchanged for better optimization accuracy, stronger robustness, and higher segmentation quality, which is worthwhile in complex image segmentation tasks.

## 3. Analysis of Numerical Optimization Experiments

This section presents a comprehensive numerical evaluation of the proposed MISBOA on the IEEE CEC2014 [29], IEEE CEC2017 [30], and CEC2020 benchmark suites [31]. First, a brief introduction to the characteristics of the three benchmark sets is provided. Then, the experimental settings are described in detail to ensure the reproducibility of the results. Subsequently, MISBOA is compared with nine representative optimization algorithms on all benchmark functions.

To guarantee a fair comparison and reduce the influence of randomness, all algorithms were tested under the same conditions, with a population size of 50 and a maximum number of 1000 iterations. Each algorithm was independently executed 30 times, and the experimental performance was assessed in terms of the mean and standard deviation of the obtained results. All experiments in this section were implemented on the MATLAB 2023a platform.

### *3.1. Numerical Optimization Test Set*

The IEEE CEC2014, CEC2017, and CEC2020 test sets, all released by the IEEE Congress on Evolutionary Computation, constitute some of the most widely used standard benchmark sets for validating the performance of heuristic optimization algorithms. These test sets comprise a large collection of optimization functions exhibiting diverse characteristics,

enabling a comprehensive evaluation of an algorithm's search capability, convergence speed, stability, and global optimization ability when solving complex optimization problems.

Specifically, the CEC2014 test suite primarily comprises various problem types, including unimodal functions, multimodal functions, hybrid functions, and composite functions. Unimodal functions are typically employed to assess an algorithm's local exploitation capabilities, whereas multimodal functions serve to evaluate its ability to escape local optima; hybrid and composite functions further heighten problem complexity, thereby enabling the assessment of an algorithm's comprehensive performance within complex search spaces.

In comparison, the CEC2017 test suite features a more complex functional structure, incorporating a greater number of high-dimensional optimization problems characterized by rotation, translation, and non-separability; consequently, it imposes more stringent demands on an algorithm's global exploration capabilities and robustness. This test suite is generally regarded as more challenging than CEC2014, as it is capable of more comprehensively reflecting the performance disparities among different algorithms.

The CEC2020 test suite further increases the proportion of complex hybrid and composite functions, while placing greater emphasis on the capability to solve high-dimensional, complex problems characterized by multiple local optima. Given its more intricate function landscapes and stronger coupling relationships among variables, it is frequently employed to validate the adaptability and generalization capabilities of novel, improved algorithms within complex optimization environments.

Overall, the CEC2014, CEC2017, and CEC2020 test suites encompass a wide range of optimization scenarios—spanning from simple to complex, low-dimensional to high-dimensional, and unimodal to complex combinatorial problems—and are therefore widely utilized for the performance validation and comparative analysis of heuristic algorithms, meta-heuristic algorithms, and swarm intelligence optimization algorithms.

### *3.2. Competitor Algorithms and Parameters Setting*

To validate the performance of MISBOA, numerical optimization experiments are conducted in this subsection to analyze and compare it with nine other novel heuristic algorithms. The selected algorithms include the Bounty Hunter Optimizer (BHO), Newton Downhill Optimizer (NDO), Gekko Japonicus Algorithm (GJA), Stochastic social learning optimization (SSLO), LangEvin Equation (LEE), Snow Ablation Optimizer (SAO), Red-billed Blue Magpie Optimizer (RBMO), Escape Algorithm (ESC), and the standard Secretary bird optimization algorithm (SBOA). In the comparative experiments, the parameter settings of all algorithms were determined according to their original references. For clarity, Table 1 lists the detailed parameter configurations of each algorithm together with the corresponding references for further consultation.

**Table 1.** Parameter Configurations of the Compared Algorithms.

| Algorithms | Parameter Name | Parameter Value | Reference |
|---|---|---|---|
| BHO | nTransfer, rho | 5, 0.25 | [32] |
| NDO | u | 0.5 | [33] |
| GJA | beta_start, beta_end | 1.2, 0.3 | [34] |
| SSLO | CR | 0.9 | [35] |
| LEE | inT, upr | 1, 0.3 | [36] |
| SAO | N1 | 0.5 | [37] |
| RBMO | Epsilon | 0.5 | [38] |
| ESC | eliteSize, beta_base | 5, 1.5 | [39] |
| SBOA | beta | 1.5 | [25] |

### *3.3. Analysis of Numerical Optimization Experimental Results*

#### 3.3.1. Performance Comparison on the IEEE CEC2014 Benchmark Functions

In this section, the numerical optimization performance of MISBOA is evaluated on the 30-dimensional functions of the CEC2014 benchmark suite. MISBOA is compared with nine representative state-of-the-art algorithms, and the corresponding optimization results are reported in Table 2. In the table, "Ave" and "Std" denote the mean and standard deviation of the results obtained from 30 independent runs, respectively. To further illustrate the search behavior of the compared algorithms, the convergence curves of the 10 algorithms are shown in Figure 5. In addition, Figure 6 presents the boxplot results of 30 independent runs for each algorithm, which are used to evaluate the robustness and stability of the algorithms while reducing the influence of random factors.

Figure 5 illustrates the convergence process of each algorithm on some functions of the CEC2014 test set. Overall, MISBOA exhibits a faster descent rate and better final convergence results on most test functions, indicating that it achieves a more effective balance between global search and local exploitation. Especially on functions such as F7, F14, and F20, MISBOA can rapidly reduce the objective function value in the early stages of iteration and maintain a relatively stable convergence trend in subsequent iterations, demonstrating strong search efficiency and stability. On relatively complex problems such as F8, F24, and F27, although some comparative algorithms can gradually approach the optimal region, MISBOA still demonstrates a faster convergence speed and stronger continuous optimization ability, obtaining higher-quality solutions with fewer iterations. In summary, Figure 5 fully illustrates that the proposed improvement strategy can effectively improve the algorithm's optimization efficiency and enhance its ability to escape local optima, thus enabling MISBOA to exhibit superior convergence performance and competitive advantage on the CEC2014 benchmark functions.

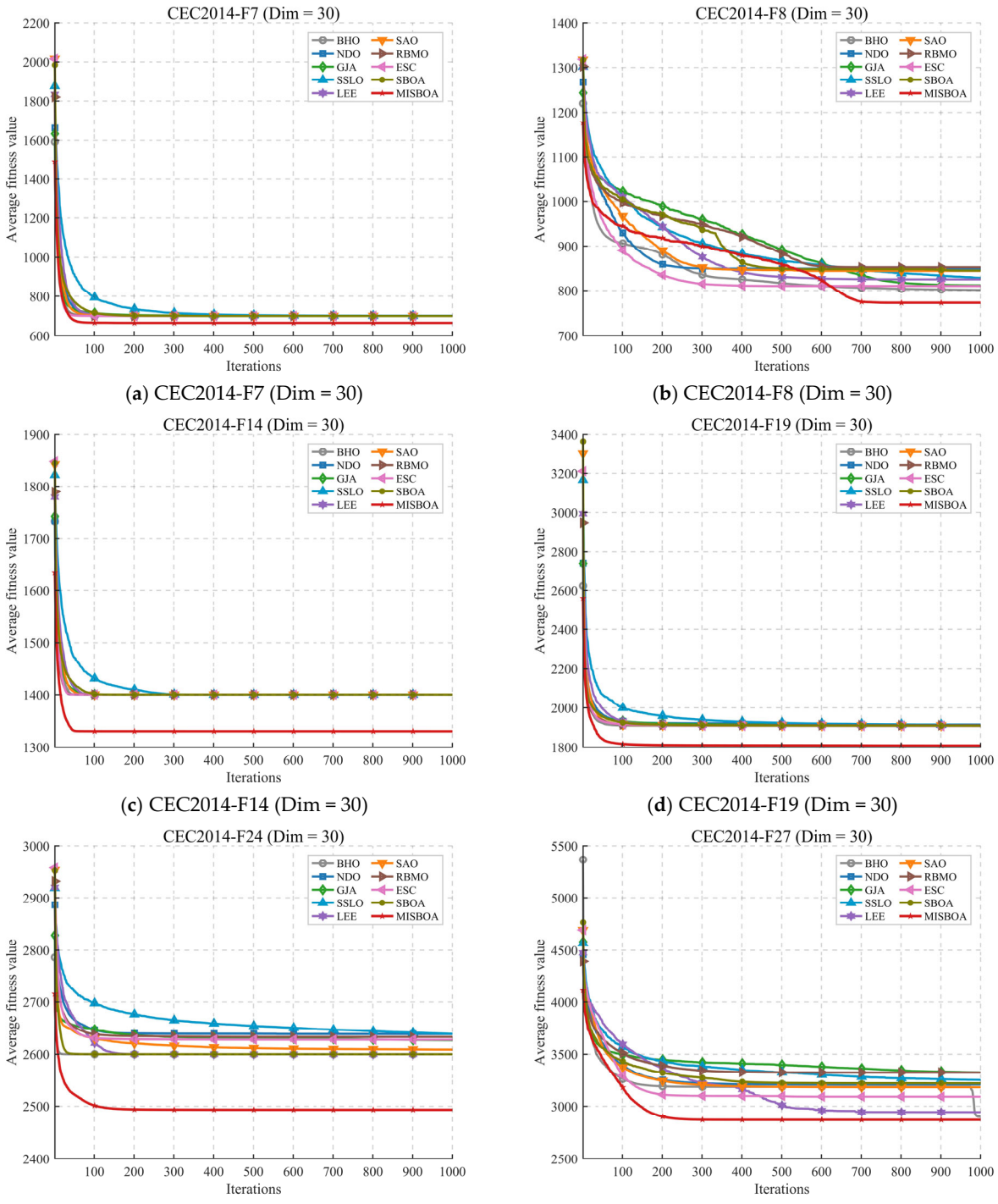
Table 2 presents the mean and standard deviation results of each algorithm on the 30-dimensional CEC2014 benchmark functions. Overall, MISBOA achieves optimal or near-optimal results on most test functions, demonstrating strong optimization accuracy and stability. For unimodal functions such as F1–F4, MISBOA obtains significantly better average fitness values than the compared algorithms, indicating that the proposed improvement strategies effectively enhance local exploitation and convergence accuracy. For multimodal and hybrid functions such as F17, F18, F20, F21, and F22, MISBOA also shows strong global search ability, suggesting that it can better escape local optima and locate high-quality solutions. It is worth noting that on CEC2014-F23, MISBOA reaches the theoretical optimum value of 2500, with a standard deviation close to zero, while none of the other compared algorithms can achieve this result. This indicates that MISBOA not only finds the global optimum on this function but also maintains highly consistent performance across 30 independent runs. In addition, MISBOA exhibits relatively small standard deviations on many functions, further confirming its robustness and stability. Although MISBOA does not obtain the best results on a few functions, its overall performance remains highly competitive. Therefore, the results in Table 2 demonstrate that MISBOA has excellent convergence accuracy, global optimization capability, and robustness on the CEC2014 benchmark suite.

**Table 2.** Performance Comparison of Different Algorithms on the 30-Dimensional CEC2014 Benchmark Functions.

| ID | Items | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CEC2014-F1 | Ave | $2.5066 \times 10^{5}$ | $2.1732 \times 10^{6}$ | $6.0964 \times 10^{6}$ | $4.7766 \times 10^{7}$ | $2.6614 \times 10^{6}$ | $2.0733 \times 10^{6}$ | $1.8911 \times 10^{5}$ | $8.3092 \times 10^{6}$ | $6.0607 \times 10^{6}$ | $2.1767 \times 10^{3}$ |
| | Std | $1.7409 \times 10^{5}$ | $1.2405 \times 10^{6}$ | $2.5346 \times 10^{6}$ | $1.3085 \times 10^{7}$ | $1.1783 \times 10^{6}$ | $1.0938 \times 10^{6}$ | $1.3707 \times 10^{5}$ | $4.6988 \times 10^{6}$ | $4.0035 \times 10^{6}$ | $2.0669 \times 10^{3}$ |
| CEC2014-F2 | Ave | $2.0009 \times 10^{2}$ | $3.2135 \times 10^{3}$ | $1.3679 \times 10^{6}$ | $1.0512 \times 10^{7}$ | $2.0003 \times 10^{2}$ | $1.0908 \times 10^{4}$ | $2.0021 \times 10^{2}$ | $9.2482 \times 10^{3}$ | $8.7464 \times 10^{3}$ | $2.0000 \times 10^{2}$ |
| | Std | $2.2283 \times 10^{-1}$ | $2.5095 \times 10^{3}$ | $2.3735 \times 10^{5}$ | $2.3774 \times 10^{6}$ | $2.5235 \times 10^{-2}$ | $9.9897 \times 10^{3}$ | $2.7067 \times 10^{-1}$ | $6.2854 \times 10^{3}$ | $7.5205 \times 10^{3}$ | $2.4186 \times 10^{-14}$ |
| CEC2014-F3 | Ave | $3.0044 \times 10^{2}$ | $2.9023 \times 10^{3}$ | $4.0530 \times 10^{3}$ | $6.1515 \times 10^{3}$ | $3.0002 \times 10^{2}$ | $8.1133 \times 10^{3}$ | $3.0025 \times 10^{2}$ | $2.6203 \times 10^{3}$ | $3.0248 \times 10^{3}$ | $3.0000 \times 10^{2}$ |
| | Std | $1.3791 \times 10^{00}$ | $1.9766 \times 10^{3}$ | $1.3393 \times 10^{3}$ | $3.0504 \times 10^{3}$ | $6.8199 \times 10^{-3}$ | $3.7347 \times 10^{3}$ | $2.3921 \times 10^{-1}$ | $4.7131 \times 10^{3}$ | $2.1803 \times 10^{3}$ | $4.9510 \times 10^{-14}$ |
| CEC2014-F4 | Ave | $4.3863 \times 10^{2}$ | $4.8282 \times 10^{2}$ | $5.1429 \times 10^{2}$ | $5.5740 \times 10^{2}$ | $4.8438 \times 10^{2}$ | $4.8809 \times 10^{2}$ | $4.7285 \times 10^{2}$ | $5.1470 \times 10^{2}$ | $4.9407 \times 10^{2}$ | $4.0011 \times 10^{2}$ |
| | Std | $3.3035 \times 10^{1}$ | $3.6196 \times 10^{1}$ | $3.1459 \times 10^{1}$ | $2.0410 \times 10^{1}$ | $3.3222 \times 10^{1}$ | $3.6493 \times 10^{1}$ | $2.5577 \times 10^{1}$ | $2.8831 \times 10^{1}$ | $3.0176 \times 10^{1}$ | $2.0105 \times 10^{-1}$ |
| CEC2014-F5 | Ave | $5.2016 \times 10^{2}$ | $5.2057 \times 10^{2}$ | $5.2093 \times 10^{2}$ | $5.2048 \times 10^{2}$ | $5.2054 \times 10^{2}$ | $5.2100 \times 10^{2}$ | $5.2030 \times 10^{2}$ | $5.2097 \times 10^{2}$ | $5.2012 \times 10^{2}$ | $5.2074 \times 10^{2}$ |
| | Std | $2.0395 \times 10^{-1}$ | $1.7020 \times 10^{-1}$ | $5.7840 \times 10^{-2}$ | $5.3882 \times 10^{-2}$ | $2.0339 \times 10^{-1}$ | $6.2369 \times 10^{-2}$ | $1.1585 \times 10^{-1}$ | $5.0138 \times 10^{-2}$ | $6.0937 \times 10^{-2}$ | $9.6401 \times 10^{-2}$ |
| CEC2014-F6 | Ave | $6.0755 \times 10^{2}$ | $6.1234 \times 10^{2}$ | $6.1560 \times 10^{2}$ | $6.2036 \times 10^{2}$ | $6.0837 \times 10^{2}$ | $6.0463 \times 10^{2}$ | $6.0894 \times 10^{2}$ | $6.0249 \times 10^{2}$ | $6.1009 \times 10^{2}$ | $6.0057 \times 10^{2}$ |
| | Std | $4.0067 \times 10^{00}$ | $2.7379 \times 10^{00}$ | $1.9030 \times 10^{00}$ | $1.2525 \times 10^{00}$ | $3.0902 \times 10^{00}$ | $2.1270 \times 10^{00}$ | $3.8291 \times 10^{00}$ | $1.2591 \times 10^{00}$ | $2.6151 \times 10^{00}$ | $7.9730 \times 10^{-1}$ |
| CEC2014-F7 | Ave | $7.0002 \times 10^{2}$ | $7.0002 \times 10^{2}$ | $7.0095 \times 10^{2}$ | $7.0105 \times 10^{2}$ | $7.0000 \times 10^{2}$ | $7.0001 \times 10^{2}$ | $7.0001 \times 10^{2}$ | $7.0000 \times 10^{2}$ | $7.0001 \times 10^{2}$ | $7.0000 \times 10^{2}$ |
| | Std | $1.3126 \times 10^{-2}$ | $4.1455 \times 10^{-2}$ | $3.4725 \times 10^{-2}$ | $1.2865 \times 10^{-2}$ | $4.3456 \times 10^{-3}$ | $1.4565 \times 10^{-2}$ | $8.0940 \times 10^{-3}$ | $1.7810 \times 10^{-3}$ | $1.5925 \times 10^{-2}$ | $4.2222 \times 10^{-14}$ |
| CEC2014-F8 | Ave | $8.0141 \times 10^{2}$ | $8.4968 \times 10^{2}$ | $8.1184 \times 10^{2}$ | $8.2880 \times 10^{2}$ | $8.2534 \times 10^{2}$ | $8.4458 \times 10^{2}$ | $8.5284 \times 10^{2}$ | $8.1038 \times 10^{2}$ | $8.4521 \times 10^{2}$ | $8.1386 \times 10^{2}$ |
| | Std | $1.0506 \times 10^{00}$ | $1.4750 \times 10^{1}$ | $3.0762 \times 10^{00}$ | $3.2199 \times 10^{00}$ | $7.6367 \times 10^{00}$ | $1.0673 \times 10^{1}$ | $1.6732 \times 10^{1}$ | $3.5213 \times 10^{00}$ | $1.4511 \times 10^{1}$ | $4.4337 \times 10^{00}$ |
| CEC2014-F9 | Ave | $9.2952 \times 10^{2}$ | $9.8041 \times 10^{2}$ | $9.6428 \times 10^{2}$ | $1.0032 \times 10^{3}$ | $9.5390 \times 10^{2}$ | $9.5486 \times 10^{2}$ | $9.4735 \times 10^{2}$ | $9.7227 \times 10^{2}$ | $9.6825 \times 10^{2}$ | $9.1844 \times 10^{2}$ |
| | Std | $6.8282 \times 10^{00}$ | $1.8257 \times 10^{1}$ | $1.6642 \times 10^{1}$ | $1.0900 \times 10^{1}$ | $1.4033 \times 10^{1}$ | $1.3836 \times 10^{1}$ | $1.3572 \times 10^{1}$ | $2.2924 \times 10^{1}$ | $1.3417 \times 10^{1}$ | $7.4673 \times 10^{00}$ |
| CEC2014-F10 | Ave | $1.1430 \times 10^{3}$ | $1.6611 \times 10^{3}$ | $1.4575 \times 10^{3}$ | $1.6655 \times 10^{3}$ | $2.8879 \times 10^{3}$ | $2.8874 \times 10^{3}$ | $3.5463 \times 10^{3}$ | $1.2096 \times 10^{3}$ | $1.7793 \times 10^{3}$ | $1.2235 \times 10^{3}$ |
| | Std | $1.2284 \times 10^{2}$ | $3.7858 \times 10^{2}$ | $1.8627 \times 10^{2}$ | $1.5438 \times 10^{2}$ | $4.8829 \times 10^{2}$ | $5.0340 \times 10^{2}$ | $6.1863 \times 10^{2}$ | $1.6878 \times 10^{2}$ | $2.5914 \times 10^{2}$ | $1.4805 \times 10^{2}$ |
| CEC2014-F11 | Ave | $3.5818 \times 10^{3}$ | $4.4586 \times 10^{3}$ | $3.6424 \times 10^{3}$ | $4.7279 \times 10^{3}$ | $4.6031 \times 10^{3}$ | $3.9016 \times 10^{3}$ | $4.5428 \times 10^{3}$ | $6.3632 \times 10^{3}$ | $3.9689 \times 10^{3}$ | $3.3364 \times 10^{3}$ |
| | Std | $4.6536 \times 10^{2}$ | $7.3914 \times 10^{2}$ | $6.7934 \times 10^{2}$ | $2.9476 \times 10^{2}$ | $6.5107 \times 10^{2}$ | $7.3440 \times 10^{2}$ | $4.4563 \times 10^{2}$ | $4.8121 \times 10^{2}$ | $7.0654 \times 10^{2}$ | $6.1142 \times 10^{2}$ |
| CEC2014-F12 | Ave | $1.2004 \times 10^{3}$ | $1.2006 \times 10^{3}$ | $1.2006 \times 10^{3}$ | $1.2007 \times 10^{3}$ | $1.2005 \times 10^{3}$ | $1.2017 \times 10^{3}$ | $1.2008 \times 10^{3}$ | $1.2021 \times 10^{3}$ | $1.2002 \times 10^{3}$ | $1.2005 \times 10^{3}$ |
| | Std | $1.3763 \times 10^{-1}$ | $2.5121 \times 10^{-1}$ | $2.4100 \times 10^{-1}$ | $1.3575 \times 10^{-1}$ | $2.3348 \times 10^{-1}$ | $1.0905 \times 10^{00}$ | $3.1518 \times 10^{-1}$ | $3.7220 \times 10^{-1}$ | $1.1210 \times 10^{-1}$ | $3.7475 \times 10^{-1}$ |
| CEC2014-F13 | Ave | $1.3002 \times 10^{3}$ | $1.3005 \times 10^{3}$ | $1.3004 \times 10^{3}$ | $1.3004 \times 10^{3}$ | $1.3004 \times 10^{3}$ | $1.3005 \times 10^{3}$ | $1.3003 \times 10^{3}$ | $1.3003 \times 10^{3}$ | $1.3004 \times 10^{3}$ | $1.3002 \times 10^{3}$ |
| | Std | $5.8921 \times 10^{-2}$ | $2.7362 \times 10^{-1}$ | $6.0189 \times 10^{-2}$ | $4.3852 \times 10^{-2}$ | $5.9542 \times 10^{-2}$ | $8.7354 \times 10^{-2}$ | $7.5534 \times 10^{-2}$ | $4.4323 \times 10^{-2}$ | $1.0615 \times 10^{-1}$ | $3.5293 \times 10^{-2}$ |
| CEC2014-F14 | Ave | $1.4003 \times 10^{3}$ | $1.4003 \times 10^{3}$ | $1.4003 \times 10^{3}$ | $1.4003 \times 10^{3}$ | $1.4003 \times 10^{3}$ | $1.4004 \times 10^{3}$ | $1.4004 \times 10^{3}$ | $1.4003 \times 10^{3}$ | $1.4003 \times 10^{3}$ | $1.4003 \times 10^{3}$ |
| | Std | $5.7789 \times 10^{-2}$ | $5.5809 \times 10^{-2}$ | $4.7677 \times 10^{-2}$ | $2.7024 \times 10^{-2}$ | $8.2078 \times 10^{-2}$ | $1.5322 \times 10^{-1}$ | $2.2430 \times 10^{-1}$ | $4.5337 \times 10^{-2}$ | $1.0950 \times 10^{-1}$ | $2.7868 \times 10^{-2}$ |
| CEC2014-F15 | Ave | $1.5042 \times 10^{3}$ | $1.5092 \times 10^{3}$ | $1.5184 \times 10^{3}$ | $1.5177 \times 10^{3}$ | $1.5061 \times 10^{3}$ | $1.5159 \times 10^{3}$ | $1.5052 \times 10^{3}$ | $1.5098 \times 10^{3}$ | $1.5071 \times 10^{3}$ | $1.5031 \times 10^{3}$ |
| | Std | $9.7598 \times 10^{-1}$ | $3.4790 \times 10^{00}$ | $3.6426 \times 10^{00}$ | $1.7337 \times 10^{00}$ | $1.7166 \times 10^{00}$ | $3.8003 \times 10^{00}$ | $1.3539 \times 10^{00}$ | $1.1543 \times 10^{00}$ | $2.3906 \times 10^{00}$ | $6.9901 \times 10^{-1}$ |

**Table 2.** *Cont.*

| ID | Items | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CEC2014-F16 | Ave | $1.6104 \times 10^{3}$ | $1.6115 \times 10^{3}$ | $1.6109 \times 10^{3}$ | $1.6115 \times 10^{3}$ | $1.6119 \times 10^{3}$ | $1.6126 \times 10^{3}$ | $1.6114 \times 10^{3}$ | $1.6118 \times 10^{3}$ | $1.6107 \times 10^{3}$ | $1.6099 \times 10^{3}$ |
| | Std | $5.6080 \times 10^{-1}$ | $6.2709 \times 10^{-1}$ | $5.4586 \times 10^{-1}$ | $3.6655 \times 10^{-1}$ | $6.1340 \times 10^{-1}$ | $3.5099 \times 10^{-1}$ | $6.6402 \times 10^{-1}$ | $4.6880 \times 10^{-1}$ | $7.4399 \times 10^{-1}$ | $9.4974 \times 10^{-1}$ |
| CEC2014-F17 | Ave | $5.5852 \times 10^{3}$ | $3.0809 \times 10^{5}$ | $3.6616 \times 10^{5}$ | $2.6380 \times 10^{6}$ | $6.7220 \times 10^{3}$ | $2.4155 \times 10^{5}$ | $2.8542 \times 10^{3}$ | $1.1021 \times 10^{6}$ | $4.1504 \times 10^{5}$ | $2.2941 \times 10^{3}$ |
| | Std | $4.5609 \times 10^{3}$ | $2.0216 \times 10^{5}$ | $2.0292 \times 10^{5}$ | $8.2519 \times 10^{5}$ | $2.8337 \times 10^{3}$ | $2.1127 \times 10^{5}$ | $2.9209 \times 10^{2}$ | $7.8913 \times 10^{5}$ | $2.7741 \times 10^{5}$ | $3.4118 \times 10^{2}$ |
| CEC2014-F18 | Ave | $1.9127 \times 10^{3}$ | $3.9694 \times 10^{3}$ | $2.9382 \times 10^{4}$ | $4.6587 \times 10^{4}$ | $2.2162 \times 10^{3}$ | $4.1689 \times 10^{3}$ | $3.1638 \times 10^{3}$ | $3.3782 \times 10^{3}$ | $4.1572 \times 10^{3}$ | $1.8158 \times 10^{3}$ |
| | Std | $4.5847 \times 10^{1}$ | $2.3759 \times 10^{3}$ | $8.0689 \times 10^{3}$ | $4.4429 \times 10^{4}$ | $8.7391 \times 10^{1}$ | $4.0722 \times 10^{3}$ | $5.0085 \times 10^{3}$ | $1.2416 \times 10^{3}$ | $3.2243 \times 10^{3}$ | $1.8798 \times 10^{1}$ |
| CEC2014-F19 | Ave | $1.9077 \times 10^{3}$ | $1.9145 \times 10^{3}$ | $1.9111 \times 10^{3}$ | $1.9130 \times 10^{3}$ | $1.9093 \times 10^{3}$ | $1.9102 \times 10^{3}$ | $1.9114 \times 10^{3}$ | $1.9075 \times 10^{3}$ | $1.9085 \times 10^{3}$ | $1.9037 \times 10^{3}$ |
| | Std | $1.5887 \times 10^{00}$ | $2.2969 \times 10^{1}$ | $1.5497 \times 10^{00}$ | $1.6027 \times 10^{00}$ | $1.3737 \times 10^{1}$ | $1.0218 \times 10^{1}$ | $1.4669 \times 10^{1}$ | $1.1062 \times 10^{1}$ | $1.0651 \times 10^{1}$ | $6.7297 \times 10^{-1}$ |
| CEC2014-F20 | Ave | $2.2132 \times 10^{3}$ | $1.1512 \times 10^{4}$ | $5.2819 \times 10^{3}$ | $1.0811 \times 10^{4}$ | $2.0948 \times 10^{3}$ | $2.6618 \times 10^{4}$ | $2.0820 \times 10^{3}$ | $6.7316 \times 10^{3}$ | $4.6816 \times 10^{3}$ | $2.0069 \times 10^{3}$ |
| | Std | $1.0351 \times 10^{2}$ | $4.0277 \times 10^{3}$ | $2.5286 \times 10^{3}$ | $3.9214 \times 10^{3}$ | $1.5978 \times 10^{1}$ | $1.1507 \times 10^{4}$ | $2.5385 \times 10^{1}$ | $3.3594 \times 10^{3}$ | $2.7647 \times 10^{3}$ | $2.1031 \times 10^{00}$ |
| CEC2014-F21 | Ave | $2.9180 \times 10^{3}$ | $8.4199 \times 10^{4}$ | $1.4274 \times 10^{5}$ | $4.0361 \times 10^{5}$ | $4.3007 \times 10^{3}$ | $1.5535 \times 10^{5}$ | $2.9610 \times 10^{3}$ | $2.4441 \times 10^{5}$ | $1.1797 \times 10^{5}$ | $2.3591 \times 10^{3}$ |
| | Std | $2.9845 \times 10^{2}$ | $7.0848 \times 10^{4}$ | $1.1743 \times 10^{5}$ | $1.3362 \times 10^{5}$ | $4.0517 \times 10^{2}$ | $9.8093 \times 10^{4}$ | $2.3176 \times 10^{2}$ | $2.1481 \times 10^{5}$ | $1.2468 \times 10^{5}$ | $1.2400 \times 10^{2}$ |
| CEC2014-F22 | Ave | $2.3974 \times 10^{3}$ | $2.6189 \times 10^{3}$ | $2.6263 \times 10^{3}$ | $2.5957 \times 10^{3}$ | $2.4759 \times 10^{3}$ | $2.6471 \times 10^{3}$ | $2.5076 \times 10^{3}$ | $2.4037 \times 10^{3}$ | $2.4288 \times 10^{3}$ | $2.3616 \times 10^{3}$ |
| | Std | $8.9222 \times 10^{1}$ | $1.5423 \times 10^{2}$ | $1.5234 \times 10^{2}$ | $9.8016 \times 10^{1}$ | $1.4043 \times 10^{2}$ | $2.1090 \times 10^{2}$ | $1.0856 \times 10^{2}$ | $1.0767 \times 10^{2}$ | $1.0711 \times 10^{2}$ | $5.2308 \times 10^{1}$ |
| CEC2014-F23 | Ave | $2.5000 \times 10^{3}$ | $2.6152 \times 10^{3}$ | $2.6154 \times 10^{3}$ | $2.6164 \times 10^{3}$ | $2.5000 \times 10^{3}$ | $2.6152 \times 10^{3}$ | $2.6152 \times 10^{3}$ | $2.6152 \times 10^{3}$ | $2.6152 \times 10^{3}$ | $2.6152 \times 10^{3}$ |
| | Std | $0.0000 \times 10^{00}$ | $2.8497 \times 10^{-7}$ | $5.7549 \times 10^{-2}$ | $3.6706 \times 10^{-1}$ | $0.0000 \times 10^{00}$ | $1.1860 \times 10^{-5}$ | $7.4315 \times 10^{-8}$ | $3.1751 \times 10^{-4}$ | $9.0475 \times 10^{-7}$ | $1.4964 \times 10^{-12}$ |
| CEC2014-F24 | Ave | $2.6000 \times 10^{3}$ | $2.6384 \times 10^{3}$ | $2.6264 \times 10^{3}$ | $2.6384 \times 10^{3}$ | $2.6000 \times 10^{3}$ | $2.6086 \times 10^{3}$ | $2.6328 \times 10^{3}$ | $2.6275 \times 10^{3}$ | $2.6000 \times 10^{3}$ | $2.6244 \times 10^{3}$ |
| | Std | $0.0000 \times 10^{00}$ | $9.6732 \times 10^{00}$ | $1.9696 \times 10^{00}$ | $2.2380 \times 10^{00}$ | $0.0000 \times 10^{00}$ | $1.0735 \times 10^{1}$ | $8.6256 \times 10^{00}$ | $4.2284 \times 10^{00}$ | $4.2152 \times 10^{-4}$ | $1.4479 \times 10^{00}$ |
| CEC2014-F25 | Ave | $2.7000 \times 10^{3}$ | $2.7087 \times 10^{3}$ | $2.7097 \times 10^{3}$ | $2.7129 \times 10^{3}$ | $2.7000 \times 10^{3}$ | $2.7033 \times 10^{3}$ | $2.7043 \times 10^{3}$ | $2.7073 \times 10^{3}$ | $2.7015 \times 10^{3}$ | $2.7030 \times 10^{3}$ |
| | Std | $0.0000 \times 10^{00}$ | $2.8358 \times 10^{00}$ | $3.0918 \times 10^{00}$ | $1.9317 \times 10^{00}$ | $0.0000 \times 10^{00}$ | $4.5657 \times 10^{00}$ | $1.8730 \times 10^{00}$ | $1.2664 \times 10^{00}$ | $2.9335 \times 10^{00}$ | $3.0097 \times 10^{-1}$ |
| CEC2014-F26 | Ave | $2.7036 \times 10^{3}$ | $2.7042 \times 10^{3}$ | $2.7635 \times 10^{3}$ | $2.7004 \times 10^{3}$ | $2.7003 \times 10^{3}$ | $2.7237 \times 10^{3}$ | $2.7225 \times 10^{3}$ | $2.7069 \times 10^{3}$ | $2.7303 \times 10^{3}$ | $2.7002 \times 10^{3}$ |
| | Std | $1.8209 \times 10^{1}$ | $2.0533 \times 10^{1}$ | $4.8873 \times 10^{1}$ | $5.1958 \times 10^{-2}$ | $6.5127 \times 10^{-2}$ | $4.2809 \times 10^{1}$ | $6.7846 \times 10^{1}$ | $2.5378 \times 10^{1}$ | $4.6428 \times 10^{1}$ | $3.2830 \times 10^{-2}$ |
| CEC2014-F27 | Ave | $2.9000 \times 10^{3}$ | $3.2097 \times 10^{3}$ | $3.3272 \times 10^{3}$ | $3.2543 \times 10^{3}$ | $2.9429 \times 10^{3}$ | $3.1839 \times 10^{3}$ | $3.3264 \times 10^{3}$ | $3.0923 \times 10^{3}$ | $3.2220 \times 10^{3}$ | $3.0209 \times 10^{3}$ |
| | Std | $0.0000 \times 10^{00}$ | $1.2366 \times 10^{2}$ | $1.9481 \times 10^{2}$ | $1.0923 \times 10^{2}$ | $8.8021 \times 10^{1}$ | $8.9489 \times 10^{1}$ | $1.4673 \times 10^{2}$ | $4.6584 \times 10^{1}$ | $1.2048 \times 10^{2}$ | $3.8252 \times 10^{1}$ |
| CEC2014-F28 | Ave | $3.2592 \times 10^{3}$ | $3.8593 \times 10^{3}$ | $3.9855 \times 10^{3}$ | $3.7712 \times 10^{3}$ | $3.0000 \times 10^{3}$ | $3.8219 \times 10^{3}$ | $3.9091 \times 10^{3}$ | $3.6541 \times 10^{3}$ | $3.6844 \times 10^{3}$ | $3.6785 \times 10^{3}$ |
| | Std | $2.3529 \times 10^{2}$ | $2.4361 \times 10^{2}$ | $4.3070 \times 10^{2}$ | $2.7873 \times 10^{1}$ | $0.0000 \times 10^{00}$ | $2.3672 \times 10^{2}$ | $2.0851 \times 10^{2}$ | $4.4931 \times 10^{1}$ | $6.2813 \times 10^{1}$ | $2.3489 \times 10^{1}$ |
| CEC2014-F29 | Ave | $6.8506 \times 10^{4}$ | $4.2437 \times 10^{3}$ | $2.9974 \times 10^{5}$ | $1.1505 \times 10^{4}$ | $6.8265 \times 10^{5}$ | $5.9885 \times 10^{5}$ | $4.7270 \times 10^{6}$ | $4.5430 \times 10^{3}$ | $5.1207 \times 10^{3}$ | $3.4499 \times 10^{3}$ |
| | Std | $2.3017 \times 10^{5}$ | $3.3950 \times 10^{2}$ | $1.5902 \times 10^{6}$ | $2.1707 \times 10^{4}$ | $2.5939 \times 10^{6}$ | $2.2620 \times 10^{6}$ | $6.1732 \times 10^{6}$ | $4.2237 \times 10^{2}$ | $1.5422 \times 10^{3}$ | $2.2320 \times 10^{2}$ |
| CEC2014-F30 | Ave | $5.0853 \times 10^{3}$ | $5.5940 \times 10^{3}$ | $6.2655 \times 10^{3}$ | $1.3548 \times 10^{4}$ | $6.1903 \times 10^{3}$ | $6.0456 \times 10^{3}$ | $6.9538 \times 10^{3}$ | $6.8012 \times 10^{3}$ | $6.0340 \times 10^{3}$ | $3.6211 \times 10^{3}$ |
| | Std | $4.2035 \times 10^{3}$ | $9.2568 \times 10^{2}$ | $8.2302 \times 10^{2}$ | $2.7440 \times 10^{3}$ | $1.1597 \times 10^{3}$ | $1.0430 \times 10^{3}$ | $2.5866 \times 10^{3}$ | $2.6277 \times 10^{3}$ | $9.8982 \times 10^{2}$ | $2.0235 \times 10^{2}$ |




(**a**) CEC2014-F7 (Dim = 30)

(**b**) CEC2014-F8 (Dim = 30)

(**c**) CEC2014-F14 (Dim = 30)

(**d**) CEC2014-F19 (Dim = 30)

(**e**) CEC2014-F24 (Dim = 30)

(**f**) CEC2014-F27 (Dim = 30)

**Figure 5.** Convergence Speed Comparison of Different Algorithms on the CEC2014 Test Suite.

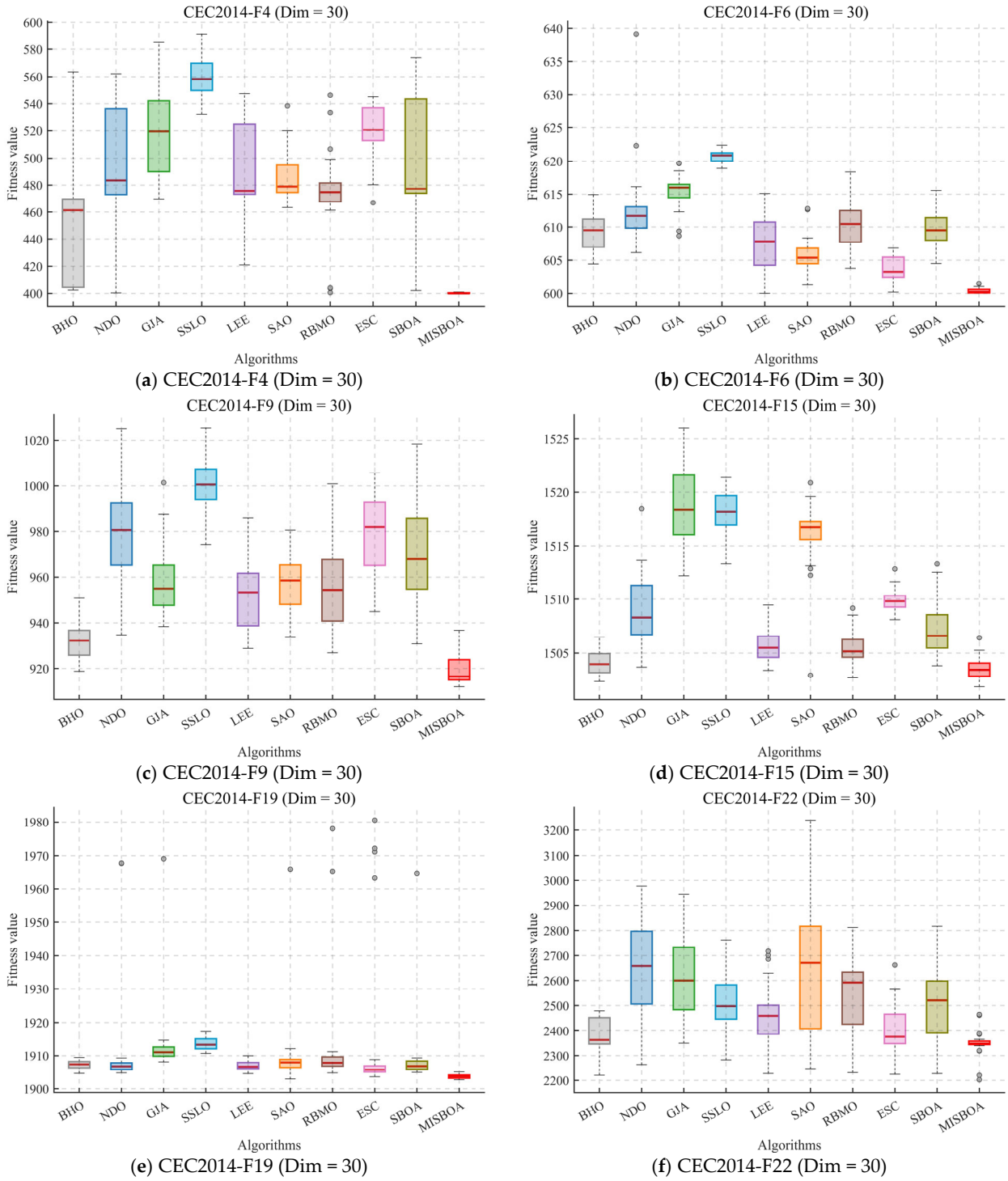



**Figure 6.** Boxplot-Based Performance Comparison of Different Algorithms on the CEC2014 Test Suite.

Figure 6 shows the box plot results of each algorithm on some functions of the CEC2014 test set, reflecting the distribution, stability, and resistance to random disturbances of different algorithms in 30 independent runs. Overall, MISBOA generally has a smaller box width, a better median position, and a shorter upper and lower bound, indicating that it maintains high consistency and stability across multiple independent runs. Particularly on functions F7, F14, F20, F24, and F30, MISBOA's box plot distribution is more concentrated with fewer outliers, indicating that it not only obtains better solutions but also exhibits less fluctuation and strong robustness. In contrast, some of the compared algorithms have wider boxes and more outliers, indicating that these algorithms are susceptible to random

initialization and local optima, leading to unstable results. Overall, Figure 6 further verifies the stability and reliability of MISBOA on the CEC2014 test set, demonstrating that the proposed improvement strategy can effectively improve the algorithm's robustness and reduce the impact of random factors on the optimization results.

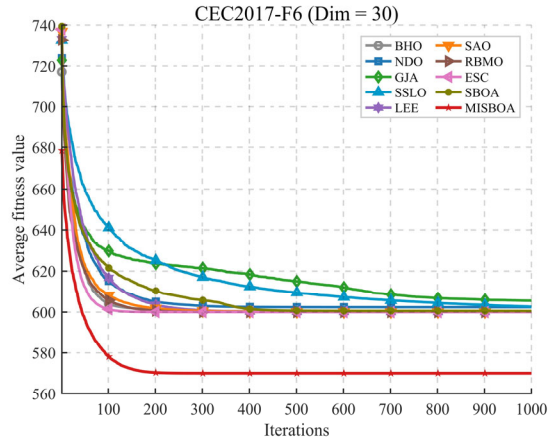
#### 3.3.2. Performance Comparison on the IEEE CEC2017 Benchmark Functions

In this section, we evaluate the numerical optimization performance of MISBOA using the 30-dimensional and 100-dimensional functions from the CEC2017 benchmark suite. Specifically, MISBOA is compared against nine representative state-of-the-art algorithms, and the corresponding optimization results are presented in Tables 3 and 4. In these tables, "Ave" and "Std" denote the mean and standard deviation, respectively, of the results obtained from 30 independent runs. To further elucidate the search behaviors of the compared algorithms, Figure 7 displays the convergence curves for these 10 algorithms. Furthermore, Figure 8 presents box plots of the results for each algorithm across the 30 independent runs; these results serve to assess the robustness and stability of the algorithms while simultaneously helping to mitigate the influence of stochastic factors.

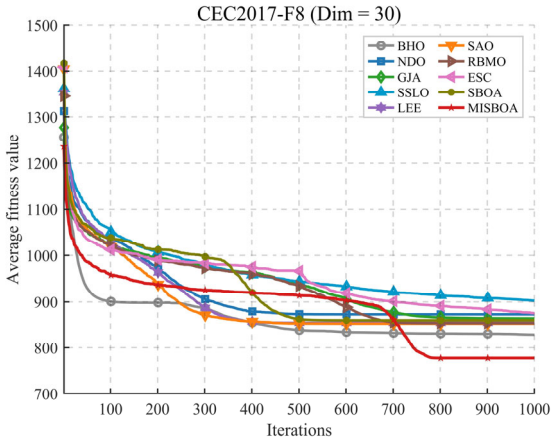
Figure 7 shows the convergence curves of each algorithm on some functions of the CEC2017 test set. Overall, MISBOA exhibits faster convergence speed and better final convergence accuracy on most test functions, indicating that it can more effectively balance global search and local exploitation. In the early stages of iteration, MISBOA often rapidly reduces the objective function value and quickly enters the advantageous search region; in later iterations, its convergence curve continues to decline, unlike some of the comparative algorithms which plateau prematurely, demonstrating its strong ability to escape local optima and its continuous optimization capability. Especially on complex mixed and combined functions, MISBOA's curve is usually below all other algorithms, and it can reach a better solution with fewer iterations, showing its higher search efficiency in handling high-dimensional, nonlinear, and multi-modal optimization problems. In contrast, while some of the comparative algorithms show a faster decline trend in the early stages, they are prone to getting trapped in local optima later, resulting in lower final convergence accuracy. Overall, Figure 7 fully illustrates that the chaotic elite initialization, adaptive spiral Lévy flight, and dynamic neighborhood-guided mutation strategies introduced by MISBOA can effectively enhance the algorithm's exploration and development capabilities, thereby achieving better convergence performance on the CEC2017 test set.

Tables 3 and 4 present the mean and standard deviation results of each algorithm on the 30-dimensional and 100-dimensional CEC2017 benchmark functions, respectively. Overall, MISBOA achieved optimal or near-optimal performance in both dimensional settings, demonstrating strong global search capability, local exploitation capability, and good dimensional adaptability. In the 30-dimensional test, MISBOA exhibited lower mean fitness values and smaller standard deviations on most unimodal, multimodal, mixed, and combined functions, indicating that it not only obtains higher-quality solutions but also achieves more stable results. As the problem dimension increased to 100 dimensions, the performance of most compared algorithms showed a significant decline, with increased mean values and greater volatility, indicating that they are more prone to getting trapped in local optima or premature convergence in high-dimensional complex problems. However, MISBOA still maintained superior solution results and a smaller standard deviation, continuing to outperform SBOA and other compared algorithms on most test functions, demonstrating strong robustness and scalability. This demonstrates that the proposed chaotic elite initialization strategy improves population diversity, the adaptive spiral Lévy flight strategy enhances global search capabilities, and the dynamic neighborhood-guided mutation strategy further improves the algorithm's local exploitation capabilities in high-

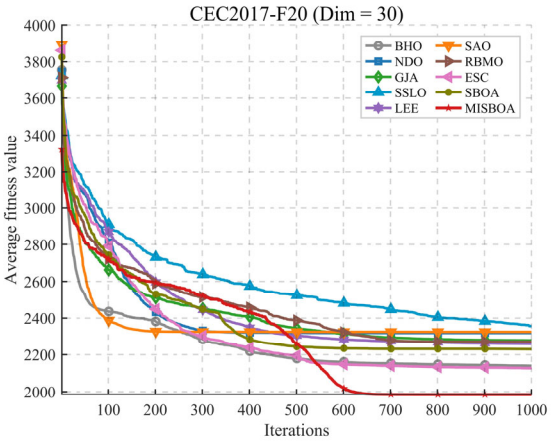
dimensional complex search spaces. Therefore, MISBOA's superior performance on the 30-dimensional and 100-dimensional CEC2017 benchmark functions fully validates the effectiveness and generalization ability of the proposed improved strategy.



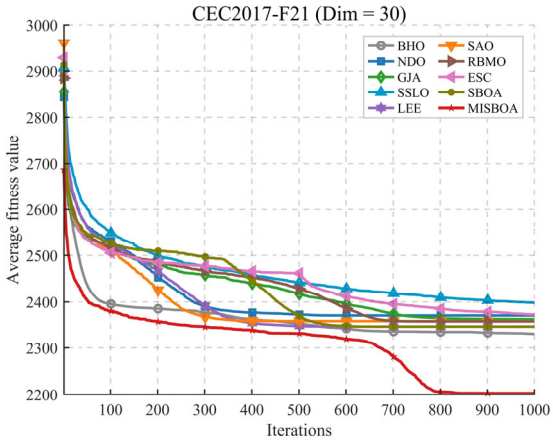

(**a**) CEC2017-F6 (Dim = 30)



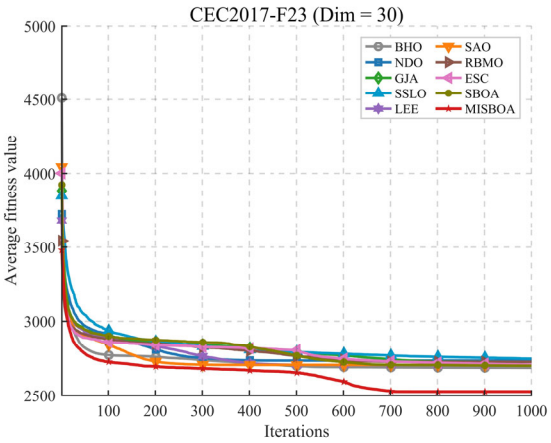

(**b**) CEC2017-F8 (Dim = 30)



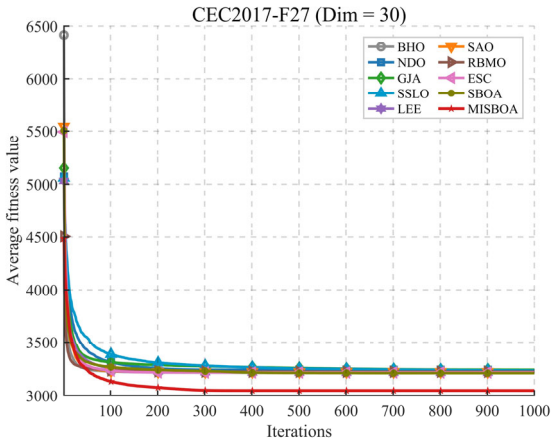

(**c**) CEC2017-F20 (Dim = 30)




(**d**) CEC2017-F21 (Dim = 30)

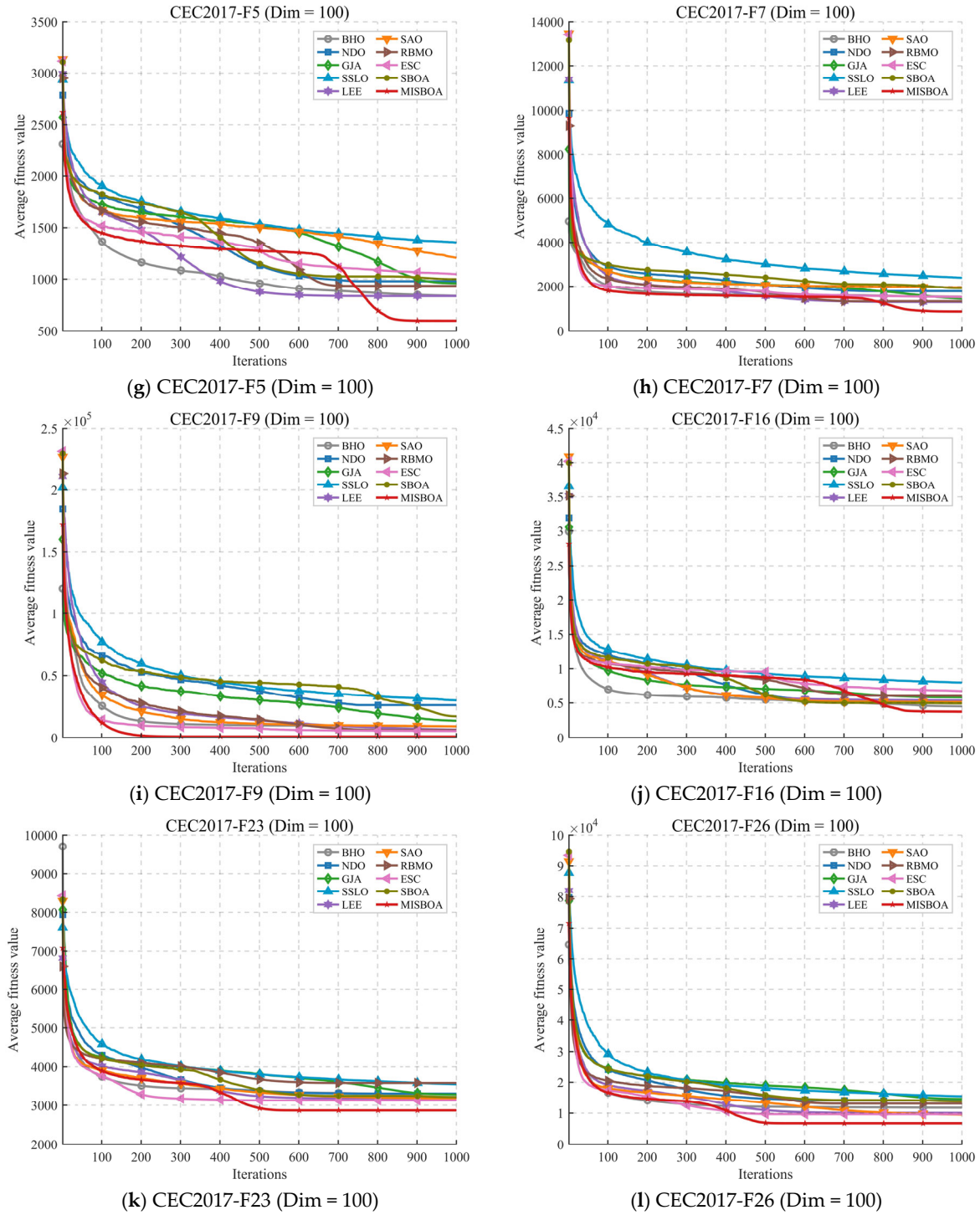



(**e**) CEC2017-F23 (Dim = 30)




(**f**) CEC2017-F27 (Dim = 30)

**Figure 7.** *Cont.*




(**g**) CEC2017-F5 (Dim = 100)

(**h**) CEC2017-F7 (Dim = 100)

(**i**) CEC2017-F9 (Dim = 100)

(**j**) CEC2017-F16 (Dim = 100)

(**k**) CEC2017-F23 (Dim = 100)

(**l**) CEC2017-F26 (Dim = 100)

**Figure 7.** Convergence Speed Comparison of Different Algorithms on the CEC2017 Test Suite.

**Table 3.** Performance Comparison of Different Algorithms on the 30-Dimensional CEC2017 Benchmark Functions.

| ID | Items | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CEC2017-F1 | Ave | $1.0740 \times 10^{2}$ | $3.6289 \times 10^{3}$ | $1.2483 \times 10^{6}$ | $1.2195 \times 10^{7}$ | $1.6391 \times 10^{2}$ | $4.2386 \times 10^{3}$ | $4.0946 \times 10^{2}$ | $2.8119 \times 10^{3}$ | $5.2727 \times 10^{3}$ | $1.0000 \times 10^{2}$ |
| | Std | $1.4039 \times 10^{1}$ | $4.1975 \times 10^{3}$ | $2.0568 \times 10^{5}$ | $4.0018 \times 10^{6}$ | $6.8159 \times 10^{1}$ | $5.8184 \times 10^{3}$ | $3.4828 \times 10^{2}$ | $3.6231 \times 10^{3}$ | $5.3558 \times 10^{3}$ | $2.6782 \times 10^{-14}$ |
| CEC2017-F3 | Ave | $3.0046 \times 10^{2}$ | $5.7796 \times 10^{4}$ | $3.1905 \times 10^{3}$ | $1.0716 \times 10^{5}$ | $1.0232 \times 10^{3}$ | $7.1659 \times 10^{4}$ | $3.0078 \times 10^{2}$ | $4.4081 \times 10^{4}$ | $5.9832 \times 10^{3}$ | $3.0000 \times 10^{2}$ |
| | Std | $4.3867 \times 10^{-1}$ | $1.2400 \times 10^{4}$ | $1.3429 \times 10^{3}$ | $1.4848 \times 10^{4}$ | $4.8830 \times 10^{2}$ | $2.1870 \times 10^{4}$ | $8.5749 \times 10^{-1}$ | $1.3502 \times 10^{4}$ | $3.1803 \times 10^{3}$ | $3.2600 \times 10^{-9}$ |
| CEC2017-F4 | Ave | $4.6916 \times 10^{2}$ | $4.7957 \times 10^{2}$ | $5.0089 \times 10^{2}$ | $5.3286 \times 10^{2}$ | $4.9795 \times 10^{2}$ | $4.9503 \times 10^{2}$ | $4.7856 \times 10^{2}$ | $5.0186 \times 10^{2}$ | $4.9559 \times 10^{2}$ | $4.2585 \times 10^{2}$ |
| | Std | $2.5160 \times 10^{1}$ | $3.4673 \times 10^{1}$ | $2.3959 \times 10^{1}$ | $1.2040 \times 10^{1}$ | $1.4348 \times 10^{1}$ | $1.4288 \times 10^{1}$ | $2.4333 \times 10^{1}$ | $2.3444 \times 10^{1}$ | $2.3162 \times 10^{1}$ | $2.8065 \times 10^{1}$ |
| CEC2017-F5 | Ave | $5.2875 \times 10^{2}$ | $5.8186 \times 10^{2}$ | $5.6398 \times 10^{2}$ | $6.0096 \times 10^{2}$ | $5.4852 \times 10^{2}$ | $5.5250 \times 10^{2}$ | $5.4794 \times 10^{2}$ | $5.7739 \times 10^{2}$ | $5.6455 \times 10^{2}$ | $5.2189 \times 10^{2}$ |
| | Std | $6.5946 \times 10^{00}$ | $4.6020 \times 10^{1}$ | $1.9621 \times 10^{1}$ | $1.0160 \times 10^{1}$ | $1.3792 \times 10^{1}$ | $1.2937 \times 10^{1}$ | $1.0442 \times 10^{1}$ | $1.9085 \times 10^{1}$ | $1.5616 \times 10^{1}$ | $6.3789 \times 10^{00}$ |
| CEC2017-F6 | Ave | $6.0028 \times 10^{2}$ | $6.0241 \times 10^{2}$ | $6.0563 \times 10^{2}$ | $6.0266 \times 10^{2}$ | $6.0000 \times 10^{2}$ | $6.0004 \times 10^{2}$ | $6.0004 \times 10^{2}$ | $6.0000 \times 10^{2}$ | $6.0055 \times 10^{2}$ | $6.0000 \times 10^{2}$ |
| | Std | $2.3263 \times 10^{-1}$ | $2.9305 \times 10^{00}$ | $1.9400 \times 10^{00}$ | $3.3107 \times 10^{-1}$ | $2.0635 \times 10^{-4}$ | $6.1913 \times 10^{-2}$ | $5.3407 \times 10^{-2}$ | $4.2105 \times 10^{-3}$ | $6.1343 \times 10^{-1}$ | $1.3378 \times 10^{-6}$ |
| CEC2017-F7 | Ave | $7.6558 \times 10^{2}$ | $8.3293 \times 10^{2}$ | $8.0812 \times 10^{2}$ | $8.9383 \times 10^{2}$ | $7.8168 \times 10^{2}$ | $8.2386 \times 10^{2}$ | $7.7783 \times 10^{2}$ | $8.2475 \times 10^{2}$ | $8.0460 \times 10^{2}$ | $7.4829 \times 10^{2}$ |
| | Std | $9.4642 \times 10^{00}$ | $2.8546 \times 10^{1}$ | $1.5932 \times 10^{1}$ | $1.2263 \times 10^{1}$ | $1.4760 \times 10^{1}$ | $6.3526 \times 10^{1}$ | $1.3283 \times 10^{1}$ | $1.5747 \times 10^{1}$ | $2.8026 \times 10^{1}$ | $6.1372 \times 10^{00}$ |
| CEC2017-F8 | Ave | $8.2869 \times 10^{2}$ | $8.7289 \times 10^{2}$ | $8.6388 \times 10^{2}$ | $9.0264 \times 10^{2}$ | $8.5423 \times 10^{2}$ | $8.5255 \times 10^{2}$ | $8.5372 \times 10^{2}$ | $8.7526 \times 10^{2}$ | $8.5984 \times 10^{2}$ | $8.1834 \times 10^{2}$ |
| | Std | $7.4096 \times 10^{00}$ | $1.9211 \times 10^{1}$ | $1.5371 \times 10^{1}$ | $1.0846 \times 10^{1}$ | $1.4316 \times 10^{1}$ | $1.4265 \times 10^{1}$ | $1.7585 \times 10^{1}$ | $2.1218 \times 10^{1}$ | $1.7338 \times 10^{1}$ | $4.3853 \times 10^{00}$ |
| CEC2017-F9 | Ave | $9.2509 \times 10^{2}$ | $1.2225 \times 10^{3}$ | $9.7483 \times 10^{2}$ | $1.6693 \times 10^{3}$ | $9.0152 \times 10^{2}$ | $9.0546 \times 10^{2}$ | $9.0890 \times 10^{2}$ | $9.0084 \times 10^{2}$ | $9.5482 \times 10^{2}$ | $9.0003 \times 10^{2}$ |
| | Std | $2.6963 \times 10^{1}$ | $3.3651 \times 10^{2}$ | $4.1124 \times 10^{1}$ | $1.7538 \times 10^{2}$ | $3.5595 \times 10^{00}$ | $1.2454 \times 10^{1}$ | $1.4554 \times 10^{1}$ | $1.5142 \times 10^{00}$ | $8.4825 \times 10^{1}$ | $8.6397 \times 10^{-2}$ |
| CEC2017-F10 | Ave | $3.8951 \times 10^{3}$ | $4.8517 \times 10^{3}$ | $4.0983 \times 10^{3}$ | $4.8336 \times 10^{3}$ | $4.4588 \times 10^{3}$ | $3.8680 \times 10^{3}$ | $4.4890 \times 10^{3}$ | $6.4250 \times 10^{3}$ | $4.2045 \times 10^{3}$ | $3.2613 \times 10^{3}$ |
| | Std | $4.8122 \times 10^{2}$ | $6.9530 \times 10^{2}$ | $6.7577 \times 10^{2}$ | $3.8861 \times 10^{2}$ | $6.2092 \times 10^{2}$ | $5.0078 \times 10^{2}$ | $6.0145 \times 10^{2}$ | $4.8994 \times 10^{2}$ | $5.8371 \times 10^{2}$ | $5.3609 \times 10^{2}$ |
| CEC2017-F11 | Ave | $1.1998 \times 10^{3}$ | $1.1783 \times 10^{3}$ | $1.2064 \times 10^{3}$ | $1.6102 \times 10^{3}$ | $1.1671 \times 10^{3}$ | $1.1665 \times 10^{3}$ | $1.1596 \times 10^{3}$ | $1.1741 \times 10^{3}$ | $1.1764 \times 10^{3}$ | $1.1211 \times 10^{3}$ |
| | Std | $3.2665 \times 10^{1}$ | $3.3516 \times 10^{1}$ | $3.4591 \times 10^{1}$ | $1.6262 \times 10^{2}$ | $3.0603 \times 10^{1}$ | $4.6103 \times 10^{1}$ | $3.0849 \times 10^{1}$ | $2.4623 \times 10^{1}$ | $3.3387 \times 10^{1}$ | $2.1412 \times 10^{1}$ |
| CEC2017-F12 | Ave | $2.7334 \times 10^{4}$ | $1.0588 \times 10^{5}$ | $2.3382 \times 10^{6}$ | $5.2432 \times 10^{6}$ | $1.7946 \times 10^{5}$ | $3.2335 \times 10^{5}$ | $1.8770 \times 10^{4}$ | $7.1805 \times 10^{5}$ | $3.4547 \times 10^{5}$ | $2.7938 \times 10^{3}$ |
| | Std | $1.8891 \times 10^{4}$ | $8.7463 \times 10^{4}$ | $1.2377 \times 10^{6}$ | $2.2234 \times 10^{6}$ | $2.6087 \times 10^{5}$ | $2.7853 \times 10^{5}$ | $1.4759 \times 10^{4}$ | $6.2879 \times 10^{5}$ | $3.1649 \times 10^{5}$ | $7.8023 \times 10^{2}$ |
| CEC2017-F13 | Ave | $3.5923 \times 10^{3}$ | $1.2230 \times 10^{4}$ | $1.0536 \times 10^{5}$ | $1.9841 \times 10^{5}$ | $7.0239 \times 10^{3}$ | $1.8695 \times 10^{4}$ | $4.5262 \times 10^{3}$ | $1.1995 \times 10^{4}$ | $2.0372 \times 10^{4}$ | $1.3489 \times 10^{3}$ |
| | Std | $1.2176 \times 10^{3}$ | $1.0232 \times 10^{4}$ | $3.0763 \times 10^{4}$ | $1.8661 \times 10^{5}$ | $5.9008 \times 10^{3}$ | $1.8662 \times 10^{4}$ | $1.0995 \times 10^{4}$ | $9.4711 \times 10^{3}$ | $2.0948 \times 10^{4}$ | $3.6688 \times 10^{1}$ |
| CEC2017-F14 | Ave | $1.5448 \times 10^{3}$ | $1.6923 \times 10^{4}$ | $1.4493 \times 10^{4}$ | $9.8324 \times 10^{4}$ | $1.5059 \times 10^{3}$ | $2.4081 \times 10^{4}$ | $1.4594 \times 10^{3}$ | $4.9109 \times 10^{4}$ | $1.8495 \times 10^{4}$ | $1.4282 \times 10^{3}$ |
| | Std | $6.1477 \times 10^{1}$ | $1.2532 \times 10^{4}$ | $1.2729 \times 10^{4}$ | $6.9423 \times 10^{4}$ | $1.0893 \times 10^{1}$ | $1.9232 \times 10^{4}$ | $1.3063 \times 10^{1}$ | $5.9674 \times 10^{4}$ | $1.7469 \times 10^{4}$ | $3.6274 \times 10^{00}$ |
| CEC2017-F15 | Ave | $1.7486 \times 10^{3}$ | $8.7618 \times 10^{3}$ | $3.3355 \times 10^{4}$ | $4.0356 \times 10^{4}$ | $1.7647 \times 10^{3}$ | $5.5854 \times 10^{3}$ | $1.6751 \times 10^{3}$ | $1.0353 \times 10^{4}$ | $1.1047 \times 10^{4}$ | $1.5143 \times 10^{3}$ |
| | Std | $1.1110 \times 10^{2}$ | $7.6780 \times 10^{3}$ | $1.4281 \times 10^{4}$ | $2.9036 \times 10^{4}$ | $6.0349 \times 10^{1}$ | $7.5008 \times 10^{3}$ | $6.4461 \times 10^{1}$ | $8.5785 \times 10^{3}$ | $8.3287 \times 10^{3}$ | $6.7724 \times 10^{00}$ |
| CEC2017-F16 | Ave | $1.9866 \times 10^{3}$ | $2.4280 \times 10^{3}$ | $2.4762 \times 10^{3}$ | $2.5923 \times 10^{3}$ | $2.2445 \times 10^{3}$ | $2.3497 \times 10^{3}$ | $2.2201 \times 10^{3}$ | $2.0926 \times 10^{3}$ | $2.2408 \times 10^{3}$ | $2.0377 \times 10^{3}$ |
| | Std | $2.0850 \times 10^{2}$ | $3.0132 \times 10^{2}$ | $2.2409 \times 10^{2}$ | $1.3612 \times 10^{2}$ | $2.6898 \times 10^{2}$ | $2.6690 \times 10^{2}$ | $1.8825 \times 10^{2}$ | $2.1612 \times 10^{2}$ | $2.2161 \times 10^{2}$ | $2.3614 \times 10^{2}$ |
| CEC2017-F17 | Ave | $1.8056 \times 10^{3}$ | $2.0274 \times 10^{3}$ | $1.9327 \times 10^{3}$ | $1.9934 \times 10^{3}$ | $1.9042 \times 10^{3}$ | $2.0287 \times 10^{3}$ | $1.9497 \times 10^{3}$ | $1.8140 \times 10^{3}$ | $1.8556 \times 10^{3}$ | $1.7439 \times 10^{3}$ |
| | Std | $4.6432 \times 10^{1}$ | $1.7959 \times 10^{2}$ | $1.0746 \times 10^{2}$ | $8.1197 \times 10^{1}$ | $1.1721 \times 10^{2}$ | $2.0915 \times 10^{2}$ | $9.7818 \times 10^{1}$ | $8.1015 \times 10^{1}$ | $8.2337 \times 10^{1}$ | $3.8824 \times 10^{1}$ |
| CEC2017-F18 | Ave | $3.5039 \times 10^{3}$ | $2.4241 \times 10^{5}$ | $2.3512 \times 10^{5}$ | $6.8135 \times 10^{5}$ | $4.9840 \times 10^{3}$ | $3.8670 \times 10^{5}$ | $1.9144 \times 10^{3}$ | $5.5822 \times 10^{5}$ | $2.6816 \times 10^{5}$ | $1.8752 \times 10^{3}$ |
| | Std | $1.6423 \times 10^{3}$ | $2.3965 \times 10^{5}$ | $2.3267 \times 10^{5}$ | $3.3020 \times 10^{5}$ | $3.2704 \times 10^{3}$ | $4.3688 \times 10^{5}$ | $3.3725 \times 10^{1}$ | $6.4101 \times 10^{5}$ | $1.4387 \times 10^{5}$ | $4.4420 \times 10^{1}$ |
| CEC2017-F19 | Ave | $2.0563 \times 10^{3}$ | $1.0845 \times 10^{4}$ | $4.5153 \times 10^{4}$ | $5.1731 \times 10^{4}$ | $1.9893 \times 10^{3}$ | $6.4958 \times 10^{3}$ | $1.9441 \times 10^{3}$ | $1.2806 \times 10^{4}$ | $1.3130 \times 10^{4}$ | $1.9129 \times 10^{3}$ |
| | Std | $6.4774 \times 10^{1}$ | $1.2607 \times 10^{4}$ | $2.7267 \times 10^{4}$ | $3.2759 \times 10^{4}$ | $2.8480 \times 10^{1}$ | $3.7784 \times 10^{3}$ | $1.7652 \times 10^{1}$ | $1.3053 \times 10^{4}$ | $1.5142 \times 10^{4}$ | $4.4667 \times 10^{00}$ |
| CEC2017-F20 | Ave | $2.1398 \times 10^{3}$ | $2.3182 \times 10^{3}$ | $2.2774 \times 10^{3}$ | $2.3562 \times 10^{3}$ | $2.2589 \times 10^{3}$ | $2.3239 \times 10^{3}$ | $2.2698 \times 10^{3}$ | $2.1269 \times 10^{3}$ | $2.2297 \times 10^{3}$ | $2.0891 \times 10^{3}$ |
| | Std | $6.1937 \times 10^{1}$ | $1.4433 \times 10^{2}$ | $1.1556 \times 10^{2}$ | $9.3653 \times 10^{1}$ | $1.0974 \times 10^{2}$ | $1.7134 \times 10^{2}$ | $1.2085 \times 10^{2}$ | $6.6802 \times 10^{1}$ | $1.0293 \times 10^{2}$ | $6.1402 \times 10^{1}$ |
| CEC2017-F21 | Ave | $2.3304 \times 10^{3}$ | $2.3706 \times 10^{3}$ | $2.3624 \times 10^{3}$ | $2.3982 \times 10^{3}$ | $2.3464 \times 10^{3}$ | $2.3581 \times 10^{3}$ | $2.3575 \times 10^{3}$ | $2.3727 \times 10^{3}$ | $2.3461 \times 10^{3}$ | $2.3188 \times 10^{3}$ |
| | Std | $6.7739 \times 10^{00}$ | $2.1658 \times 10^{1}$ | $1.5823 \times 10^{1}$ | $1.2376 \times 10^{1}$ | $1.3204 \times 10^{1}$ | $1.6117 \times 10^{1}$ | $1.7772 \times 10^{1}$ | $1.9821 \times 10^{1}$ | $1.1288 \times 10^{1}$ | $6.8398 \times 10^{00}$ |
| CEC2017-F22 | Ave | $2.3003 \times 10^{3}$ | $2.4268 \times 10^{3}$ | $2.3120 \times 10^{3}$ | $2.3240 \times 10^{3}$ | $3.4112 \times 10^{3}$ | $2.8764 \times 10^{3}$ | $4.3287 \times 10^{3}$ | $3.3803 \times 10^{3}$ | $2.3008 \times 10^{3}$ | $2.3000 \times 10^{3}$ |
| | Std | $1.0373 \times 10^{00}$ | $5.3421 \times 10^{2}$ | $7.9133 \times 10^{-1}$ | $5.6797 \times 10^{00}$ | $1.7464 \times 10^{3}$ | $1.1937 \times 10^{3}$ | $1.9617 \times 10^{3}$ | $2.2038 \times 10^{3}$ | $1.2847 \times 10^{00}$ | $1.6889 \times 10^{-13}$ |

**Table 3.** *Cont.*

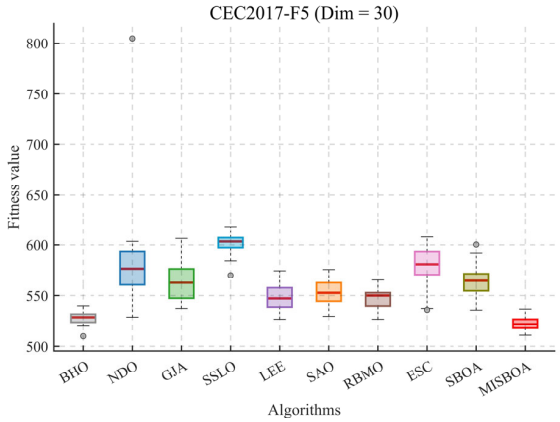
| ID | Items | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CEC2017-F23 | Ave | $2.6854 \times 10^{3}$ | $2.7337 \times 10^{3}$ | $2.7251 \times 10^{3}$ | $2.7465 \times 10^{3}$ | $2.7041 \times 10^{3}$ | $2.7049 \times 10^{3}$ | $2.7253 \times 10^{3}$ | $2.7068 \times 10^{3}$ | $2.6996 \times 10^{3}$ | $2.6600 \times 10^{3}$ |
| | Std | $8.5479 \times 10^{00}$ | $2.3965 \times 10^{1}$ | $2.1614 \times 10^{1}$ | $1.1534 \times 10^{1}$ | $1.3833 \times 10^{1}$ | $1.3782 \times 10^{1}$ | $2.6433 \times 10^{1}$ | $2.2609 \times 10^{1}$ | $1.6947 \times 10^{1}$ | $1.0466 \times 10^{1}$ |
| CEC2017-F24 | Ave | $2.8560 \times 10^{3}$ | $2.9096 \times 10^{3}$ | $2.8888 \times 10^{3}$ | $2.9149 \times 10^{3}$ | $2.8783 \times 10^{3}$ | $2.8798 \times 10^{3}$ | $2.8995 \times 10^{3}$ | $2.9132 \times 10^{3}$ | $2.8692 \times 10^{3}$ | $2.8320 \times 10^{3}$ |
| | Std | $1.0336 \times 10^{1}$ | $2.6077 \times 10^{1}$ | $1.8817 \times 10^{1}$ | $1.4235 \times 10^{1}$ | $1.8662 \times 10^{1}$ | $1.6584 \times 10^{1}$ | $3.2291 \times 10^{1}$ | $1.4565 \times 10^{1}$ | $1.5848 \times 10^{1}$ | $5.8100 \times 10^{00}$ |
| CEC2017-F25 | Ave | $2.8893 \times 10^{3}$ | $2.8908 \times 10^{3}$ | $2.9135 \times 10^{3}$ | $2.9207 \times 10^{3}$ | $2.8873 \times 10^{3}$ | $2.8867 \times 10^{3}$ | $2.8892 \times 10^{3}$ | $2.8889 \times 10^{3}$ | $2.8933 \times 10^{3}$ | $2.8869 \times 10^{3}$ |
| | Std | $6.6324 \times 10^{00}$ | $9.7651 \times 10^{00}$ | $1.8745 \times 10^{1}$ | $8.8034 \times 10^{00}$ | $8.2619 \times 10^{-1}$ | $1.8729 \times 10^{00}$ | $6.6018 \times 10^{00}$ | $4.3333 \times 10^{00}$ | $1.2431 \times 10^{1}$ | $1.0303 \times 10^{-1}$ |
| CEC2017-F26 | Ave | $3.4460 \times 10^{3}$ | $4.1009 \times 10^{3}$ | $3.6682 \times 10^{3}$ | $4.6996 \times 10^{3}$ | $4.2180 \times 10^{3}$ | $4.1323 \times 10^{3}$ | $4.3594 \times 10^{3}$ | $3.9437 \times 10^{3}$ | $3.7868 \times 10^{3}$ | $3.6386 \times 10^{3}$ |
| | Std | $5.8271 \times 10^{2}$ | $8.9344 \times 10^{2}$ | $9.8808 \times 10^{2}$ | $1.5191 \times 10^{2}$ | $1.3092 \times 10^{2}$ | $3.7319 \times 10^{2}$ | $4.9309 \times 10^{2}$ | $1.5738 \times 10^{2}$ | $6.3072 \times 10^{2}$ | $1.0213 \times 10^{2}$ |
| CEC2017-F27 | Ave | $3.2135 \times 10^{3}$ | $3.2407 \times 10^{3}$ | $3.2421 \times 10^{3}$ | $3.2345 \times 10^{3}$ | $3.2141 \times 10^{3}$ | $3.2227 \times 10^{3}$ | $3.2239 \times 10^{3}$ | $3.2173 \times 10^{3}$ | $3.2108 \times 10^{3}$ | $3.2090 \times 10^{3}$ |
| | Std | $1.0289 \times 10^{1}$ | $1.9179 \times 10^{1}$ | $1.8671 \times 10^{1}$ | $7.1712 \times 10^{00}$ | $1.1141 \times 10^{1}$ | $1.3915 \times 10^{1}$ | $2.2498 \times 10^{1}$ | $8.2055 \times 10^{00}$ | $1.1409 \times 10^{1}$ | $1.1028 \times 10^{1}$ |
| CEC2017-F28 | Ave | $3.1712 \times 10^{3}$ | $3.2288 \times 10^{3}$ | $3.2447 \times 10^{3}$ | $3.2882 \times 10^{3}$ | $3.2284 \times 10^{3}$ | $3.2159 \times 10^{3}$ | $3.3235 \times 10^{3}$ | $3.2243 \times 10^{3}$ | $3.2196 \times 10^{3}$ | $3.1335 \times 10^{3}$ |
| | Std | $5.7453 \times 10^{1}$ | $1.0445 \times 10^{2}$ | $2.0971 \times 10^{1}$ | $1.1244 \times 10^{1}$ | $2.6018 \times 10^{1}$ | $1.8035 \times 10^{1}$ | $6.0940 \times 10^{2}$ | $1.7275 \times 10^{1}$ | $2.3788 \times 10^{1}$ | $5.2481 \times 10^{1}$ |
| CEC2017-F29 | Ave | $3.4945 \times 10^{3}$ | $3.6068 \times 10^{3}$ | $3.6311 \times 10^{3}$ | $3.6685 \times 10^{3}$ | $3.6090 \times 10^{3}$ | $3.6583 \times 10^{3}$ | $3.6456 \times 10^{3}$ | $3.4485 \times 10^{3}$ | $3.5081 \times 10^{3}$ | $3.3630 \times 10^{3}$ |
| | Std | $9.7275 \times 10^{1}$ | $1.5391 \times 10^{2}$ | $1.8566 \times 10^{2}$ | $6.5626 \times 10^{1}$ | $1.3069 \times 10^{2}$ | $1.3517 \times 10^{2}$ | $1.3415 \times 10^{2}$ | $7.4082 \times 10^{1}$ | $1.1446 \times 10^{2}$ | $1.9823 \times 10^{1}$ |
| CEC2017-F30 | Ave | $7.3532 \times 10^{3}$ | $9.3334 \times 10^{3}$ | $1.6175 \times 10^{5}$ | $9.5922 \times 10^{4}$ | $1.7642 \times 10^{4}$ | $8.5175 \times 10^{3}$ | $7.0105 \times 10^{3}$ | $1.0911 \times 10^{4}$ | $2.0223 \times 10^{4}$ | $5.3068 \times 10^{3}$ |
| | Std | $2.2403 \times 10^{3}$ | $3.0397 \times 10^{3}$ | $9.9323 \times 10^{4}$ | $3.5134 \times 10^{4}$ | $5.8532 \times 10^{3}$ | $3.6042 \times 10^{3}$ | $1.6945 \times 10^{3}$ | $4.4873 \times 10^{3}$ | $2.4201 \times 10^{4}$ | $1.3538 \times 10^{2}$ |

**Table 4.** Performance Comparison of Different Algorithms on the 100-Dimensional CEC2017 Benchmark Functions.

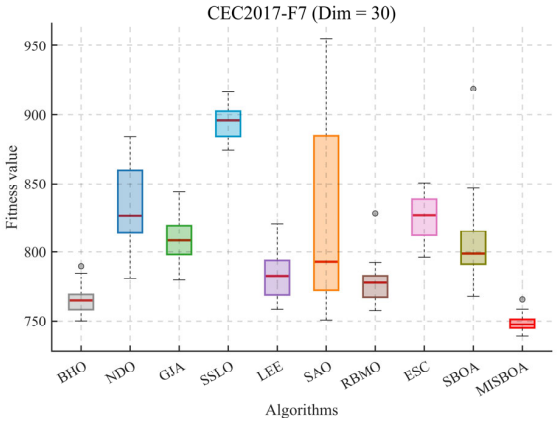
| ID | Items | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CEC2017-F1 | Ave | $2.3656 \times 10^{5}$ | $2.1626 \times 10^{8}$ | $2.8232 \times 10^{7}$ | $1.3296 \times 10^{10}$ | $4.6316 \times 10^{4}$ | $1.4426 \times 10^{8}$ | $3.9710 \times 10^{7}$ | $8.6900 \times 10^{9}$ | $2.3778 \times 10^{8}$ | $6.7162 \times 10^{3}$ |
| | Std | $1.9031 \times 10^{5}$ | $4.8936 \times 10^{8}$ | $3.2417 \times 10^{6}$ | $1.0328 \times 10^{9}$ | $9.7020 \times 10^{4}$ | $6.7772 \times 10^{7}$ | $2.6587 \times 10^{7}$ | $3.4946 \times 10^{9}$ | $5.0375 \times 10^{8}$ | $5.6948 \times 10^{3}$ |
| CEC2017-F3 | Ave | $1.1238 \times 10^{5}$ | $4.9515 \times 10^{5}$ | $2.9307 \times 10^{5}$ | $5.6949 \times 10^{5}$ | $2.3626 \times 10^{5}$ | $7.9730 \times 10^{5}$ | $7.9903 \times 10^{4}$ | $4.8855 \times 10^{5}$ | $2.4441 \times 10^{5}$ | $3.1722 \times 10^{5}$ |
| | Std | $1.8219 \times 10^{4}$ | $5.6885 \times 10^{4}$ | $4.2376 \times 10^{4}$ | $4.9903 \times 10^{4}$ | $2.7139 \times 10^{4}$ | $1.7806 \times 10^{5}$ | $1.5551 \times 10^{4}$ | $5.0067 \times 10^{4}$ | $2.3827 \times 10^{4}$ | $2.2250 \times 10^{4}$ |
| CEC2017-F4 | Ave | $7.1858 \times 10^{2}$ | $9.2884 \times 10^{2}$ | $9.2122 \times 10^{2}$ | $2.5730 \times 10^{3}$ | $7.4919 \times 10^{2}$ | $7.6995 \times 10^{2}$ | $8.5117 \times 10^{2}$ | $1.4783 \times 10^{3}$ | $9.5445 \times 10^{2}$ | $6.4263 \times 10^{2}$ |
| | Std | $5.6997 \times 10^{1}$ | $7.6884 \times 10^{1}$ | $6.5320 \times 10^{1}$ | $1.2746 \times 10^{2}$ | $6.0462 \times 10^{1}$ | $5.1533 \times 10^{1}$ | $5.8474 \times 10^{1}$ | $2.7996 \times 10^{2}$ | $8.7135 \times 10^{1}$ | $5.0854 \times 10^{1}$ |
| CEC2017-F5 | Ave | $8.3803 \times 10^{2}$ | $9.7909 \times 10^{2}$ | $9.5883 \times 10^{2}$ | $1.3573 \times 10^{3}$ | $8.3476 \times 10^{2}$ | $1.2069 \times 10^{3}$ | $9.3365 \times 10^{2}$ | $1.0478 \times 10^{3}$ | $9.9938 \times 10^{2}$ | $6.2995 \times 10^{2}$ |
| | Std | $5.4671 \times 10^{1}$ | $7.0918 \times 10^{1}$ | $8.2643 \times 10^{1}$ | $3.7609 \times 10^{1}$ | $5.9760 \times 10^{1}$ | $2.3507 \times 10^{2}$ | $7.0651 \times 10^{1}$ | $1.2999 \times 10^{2}$ | $7.5345 \times 10^{1}$ | $2.1719 \times 10^{1}$ |
| CEC2017-F6 | Ave | $6.1784 \times 10^{2}$ | $6.2240 \times 10^{2}$ | $6.2534 \times 10^{2}$ | $6.2600 \times 10^{2}$ | $6.0807 \times 10^{2}$ | $6.1170 \times 10^{2}$ | $6.1111 \times 10^{2}$ | $6.0584 \times 10^{2}$ | $6.2608 \times 10^{2}$ | $6.0115 \times 10^{2}$ |
| | Std | $3.8183 \times 10^{00}$ | $5.4665 \times 10^{00}$ | $3.6385 \times 10^{00}$ | $1.1616 \times 10^{00}$ | $3.8598 \times 10^{00}$ | $3.0288 \times 10^{00}$ | $3.4458 \times 10^{00}$ | $1.3344 \times 10^{00}$ | $5.5798 \times 10^{00}$ | $4.4594 \times 10^{-1}$ |
| CEC2017-F7 | Ave | $1.5294 \times 10^{3}$ | $1.7855 \times 10^{3}$ | $1.4494 \times 10^{3}$ | $2.3961 \times 10^{3}$ | $1.2976 \times 10^{3}$ | $1.9557 \times 10^{3}$ | $1.3352 \times 10^{3}$ | $1.5178 \times 10^{3}$ | $1.9246 \times 10^{3}$ | $9.1940 \times 10^{2}$ |
| | Std | $1.2869 \times 10^{2}$ | $1.8724 \times 10^{2}$ | $6.5704 \times 10^{1}$ | $4.2081 \times 10^{1}$ | $7.2251 \times 10^{1}$ | $5.0828 \times 10^{1}$ | $8.3361 \times 10^{1}$ | $1.1265 \times 10^{2}$ | $2.0701 \times 10^{2}$ | $2.3951 \times 10^{1}$ |
| CEC2017-F8 | Ave | $1.1862 \times 10^{3}$ | $1.3151 \times 10^{3}$ | $1.2322 \times 10^{3}$ | $1.6595 \times 10^{3}$ | $1.1113 \times 10^{3}$ | $1.4270 \times 10^{3}$ | $1.2227 \times 10^{3}$ | $1.3644 \times 10^{3}$ | $1.2882 \times 10^{3}$ | $9.3150 \times 10^{2}$ |
| | Std | $6.1575 \times 10^{1}$ | $8.5719 \times 10^{1}$ | $6.1181 \times 10^{1}$ | $3.2115 \times 10^{1}$ | $5.1768 \times 10^{1}$ | $2.1662 \times 10^{2}$ | $1.0006 \times 10^{2}$ | $1.1717 \times 10^{2}$ | $6.3832 \times 10^{1}$ | $2.1689 \times 10^{1}$ |
| CEC2017-F9 | Ave | $9.1382 \times 10^{3}$ | $2.6044 \times 10^{4}$ | $1.3531 \times 10^{4}$ | $2.9839 \times 10^{4}$ | $6.5359 \times 10^{3}$ | $9.0870 \times 10^{3}$ | $6.1915 \times 10^{3}$ | $5.3717 \times 10^{3}$ | $1.7130 \times 10^{4}$ | $9.4629 \times 10^{2}$ |
| | Std | $2.3868 \times 10^{3}$ | $9.0571 \times 10^{3}$ | $5.6651 \times 10^{3}$ | $4.1849 \times 10^{3}$ | $1.6765 \times 10^{3}$ | $2.8502 \times 10^{3}$ | $2.3205 \times 10^{3}$ | $1.6787 \times 10^{3}$ | $3.2980 \times 10^{3}$ | $3.4376 \times 10^{1}$ |

**Table 4.** *Cont.*

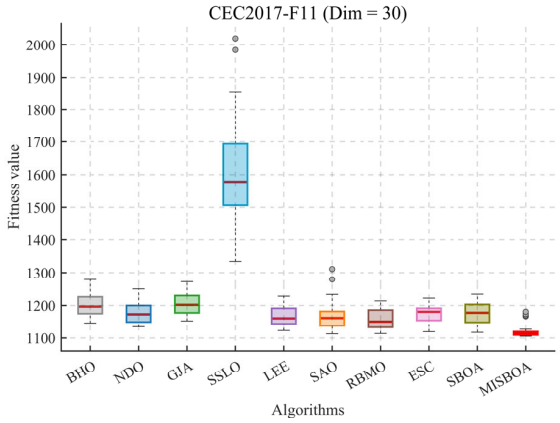
| ID | Items | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CEC2017-F10 | Ave | $1.5224 \times 10^{4}$ | $2.3979 \times 10^{4}$ | $1.4171 \times 10^{4}$ | $2.1661 \times 10^{4}$ | $1.5521 \times 10^{4}$ | $2.0339 \times 10^{4}$ | $1.9326 \times 10^{4}$ | $2.8178 \times 10^{4}$ | $1.4382 \times 10^{4}$ | $2.9217 \times 10^{4}$ |
| | Std | $1.1596 \times 10^{3}$ | $4.0393 \times 10^{3}$ | $1.1554 \times 10^{3}$ | $5.2214 \times 10^{2}$ | $1.2388 \times 10^{3}$ | $6.9978 \times 10^{3}$ | $1.7473 \times 10^{3}$ | $6.6470 \times 10^{2}$ | $1.8747 \times 10^{3}$ | $8.9907 \times 10^{2}$ |
| CEC2017-F11 | Ave | $2.6851 \times 10^{3}$ | $5.7817 \times 10^{4}$ | $3.7572 \times 10^{3}$ | $1.1568 \times 10^{5}$ | $1.5462 \times 10^{4}$ | $1.2635 \times 10^{5}$ | $3.4177 \times 10^{3}$ | $4.2050 \times 10^{4}$ | $1.2155 \times 10^{4}$ | $2.6260 \times 10^{3}$ |
| | Std | $1.7968 \times 10^{2}$ | $1.9018 \times 10^{4}$ | $7.5299 \times 10^{2}$ | $1.5916 \times 10^{4}$ | $4.2986 \times 10^{3}$ | $3.7462 \times 10^{4}$ | $3.4655 \times 10^{2}$ | $9.7885 \times 10^{3}$ | $4.7945 \times 10^{3}$ | $1.8456 \times 10^{2}$ |
| CEC2017-F12 | Ave | $7.4148 \times 10^{6}$ | $4.1636 \times 10^{7}$ | $1.7013 \times 10^{8}$ | $3.3919 \times 10^{9}$ | $2.6075 \times 10^{7}$ | $4.4456 \times 10^{7}$ | $3.2439 \times 10^{7}$ | $3.6758 \times 10^{8}$ | $5.4397 \times 10^{7}$ | $3.6697 \times 10^{6}$ |
| | Std | $2.8801 \times 10^{6}$ | $2.5223 \times 10^{7}$ | $5.2121 \times 10^{7}$ | $3.6296 \times 10^{8}$ | $1.0571 \times 10^{7}$ | $2.6693 \times 10^{7}$ | $2.7889 \times 10^{7}$ | $2.5097 \times 10^{8}$ | $2.6916 \times 10^{7}$ | $1.5510 \times 10^{6}$ |
| CEC2017-F13 | Ave | $1.3917 \times 10^{4}$ | $8.9109 \times 10^{3}$ | $8.3288 \times 10^{5}$ | $3.2118 \times 10^{7}$ | $6.3794 \times 10^{3}$ | $8.0810 \times 10^{3}$ | $1.5315 \times 10^{4}$ | $3.4792 \times 10^{4}$ | $1.0632 \times 10^{4}$ | $4.0617 \times 10^{3}$ |
| | Std | $6.0828 \times 10^{3}$ | $7.1554 \times 10^{3}$ | $1.0378 \times 10^{5}$ | $1.0888 \times 10^{7}$ | $4.8554 \times 10^{3}$ | $5.4849 \times 10^{3}$ | $2.3616 \times 10^{4}$ | $1.1822 \times 10^{4}$ | $9.4458 \times 10^{3}$ | $2.2374 \times 10^{3}$ |
| CEC2017-F14 | Ave | $5.6774 \times 10^{4}$ | $1.0487 \times 10^{6}$ | $1.5424 \times 10^{6}$ | $1.6347 \times 10^{7}$ | $4.0947 \times 10^{5}$ | $8.3518 \times 10^{5}$ | $2.3522 \times 10^{3}$ | $4.7472 \times 10^{6}$ | $1.7399 \times 10^{6}$ | $2.5560 \times 10^{4}$ |
| | Std | $3.5301 \times 10^{4}$ | $5.6758 \times 10^{5}$ | $4.2439 \times 10^{5}$ | $4.5129 \times 10^{6}$ | $2.2547 \times 10^{5}$ | $4.1761 \times 10^{5}$ | $1.9075 \times 10^{2}$ | $3.1375 \times 10^{6}$ | $9.3930 \times 10^{5}$ | $1.7086 \times 10^{4}$ |
| CEC2017-F15 | Ave | $6.8646 \times 10^{3}$ | $5.2421 \times 10^{3}$ | $3.2655 \times 10^{5}$ | $2.1906 \times 10^{6}$ | $6.0308 \times 10^{3}$ | $5.0891 \times 10^{3}$ | $6.7004 \times 10^{3}$ | $1.6607 \times 10^{4}$ | $6.3186 \times 10^{3}$ | $2.5555 \times 10^{3}$ |
| | Std | $6.9871 \times 10^{3}$ | $3.5998 \times 10^{3}$ | $5.1891 \times 10^{4}$ | $1.2305 \times 10^{6}$ | $5.9219 \times 10^{3}$ | $4.1559 \times 10^{3}$ | $7.5220 \times 10^{3}$ | $5.4254 \times 10^{3}$ | $5.0903 \times 10^{3}$ | $1.0446 \times 10^{3}$ |
| CEC2017-F16 | Ave | $4.5108 \times 10^{3}$ | $5.2095 \times 10^{3}$ | $5.8084 \times 10^{3}$ | $7.9535 \times 10^{3}$ | $5.3750 \times 10^{3}$ | $5.1321 \times 10^{3}$ | $6.0382 \times 10^{3}$ | $6.7069 \times 10^{3}$ | $4.9341 \times 10^{3}$ | $3.9377 \times 10^{3}$ |
| | Std | $6.7589 \times 10^{2}$ | $7.7525 \times 10^{2}$ | $6.2921 \times 10^{2}$ | $2.7254 \times 10^{2}$ | $7.2387 \times 10^{2}$ | $6.2585 \times 10^{2}$ | $7.5143 \times 10^{2}$ | $7.5548 \times 10^{2}$ | $7.1780 \times 10^{2}$ | $6.2776 \times 10^{2}$ |
| CEC2017-F17 | Ave | $4.2193 \times 10^{3}$ | $4.6872 \times 10^{3}$ | $5.3200 \times 10^{3}$ | $6.3280 \times 10^{3}$ | $4.8059 \times 10^{3}$ | $5.1044 \times 10^{3}$ | $5.1280 \times 10^{3}$ | $5.0645 \times 10^{3}$ | $4.5074 \times 10^{3}$ | $3.7065 \times 10^{3}$ |
| | Std | $5.2659 \times 10^{2}$ | $5.4104 \times 10^{2}$ | $5.8614 \times 10^{2}$ | $2.5693 \times 10^{2}$ | $5.0609 \times 10^{2}$ | $1.0927 \times 10^{3}$ | $5.5306 \times 10^{2}$ | $4.5268 \times 10^{2}$ | $4.6538 \times 10^{2}$ | $3.8564 \times 10^{2}$ |
| CEC2017-F18 | Ave | $1.7430 \times 10^{5}$ | $5.0972 \times 10^{6}$ | $1.7135 \times 10^{6}$ | $1.8128 \times 10^{7}$ | $7.9193 \times 10^{5}$ | $3.4358 \times 10^{6}$ | $8.7935 \times 10^{4}$ | $8.3247 \times 10^{6}$ | $3.4329 \times 10^{6}$ | $7.8461 \times 10^{4}$ |
| | Std | $7.4403 \times 10^{4}$ | $3.0340 \times 10^{6}$ | $5.1663 \times 10^{5}$ | $4.8052 \times 10^{6}$ | $4.1902 \times 10^{5}$ | $2.1150 \times 10^{6}$ | $3.9863 \times 10^{4}$ | $3.4027 \times 10^{6}$ | $1.7623 \times 10^{6}$ | $1.9852 \times 10^{4}$ |
| CEC2017-F19 | Ave | $6.4859 \times 10^{3}$ | $5.7996 \times 10^{3}$ | $5.9772 \times 10^{5}$ | $4.6921 \times 10^{6}$ | $6.0335 \times 10^{3}$ | $4.9609 \times 10^{3}$ | $6.9109 \times 10^{3}$ | $2.5038 \times 10^{4}$ | $8.6746 \times 10^{3}$ | $3.3521 \times 10^{3}$ |
| | Std | $6.2478 \times 10^{3}$ | $4.7900 \times 10^{3}$ | $1.6030 \times 10^{5}$ | $2.7171 \times 10^{6}$ | $6.1895 \times 10^{3}$ | $3.0320 \times 10^{3}$ | $5.4175 \times 10^{3}$ | $4.6720 \times 10^{4}$ | $9.0864 \times 10^{3}$ | $1.3850 \times 10^{3}$ |
| CEC2017-F20 | Ave | $4.5133 \times 10^{3}$ | $5.6774 \times 10^{3}$ | $4.6167 \times 10^{3}$ | $6.0873 \times 10^{3}$ | $5.0227 \times 10^{3}$ | $5.4992 \times 10^{3}$ | $4.9890 \times 10^{3}$ | $5.7320 \times 10^{3}$ | $4.3832 \times 10^{3}$ | $4.3067 \times 10^{3}$ |
| | Std | $3.9588 \times 10^{2}$ | $6.5777 \times 10^{2}$ | $4.8729 \times 10^{2}$ | $2.7658 \times 10^{2}$ | $4.7156 \times 10^{2}$ | $1.4038 \times 10^{3}$ | $5.0777 \times 10^{2}$ | $3.3659 \times 10^{2}$ | $5.1315 \times 10^{2}$ | $6.5664 \times 10^{2}$ |
| CEC2017-F21 | Ave | $2.7132 \times 10^{3}$ | $2.8101 \times 10^{3}$ | $2.7456 \times 10^{3}$ | $3.1871 \times 10^{3}$ | $2.6387 \times 10^{3}$ | $2.9009 \times 10^{3}$ | $2.8686 \times 10^{3}$ | $2.7763 \times 10^{3}$ | $2.7793 \times 10^{3}$ | $2.4481 \times 10^{3}$ |
| | Std | $6.8494 \times 10^{1}$ | $6.4857 \times 10^{1}$ | $5.9494 \times 10^{1}$ | $3.3741 \times 10^{1}$ | $5.0480 \times 10^{1}$ | $2.1268 \times 10^{2}$ | $1.0823 \times 10^{2}$ | $1.4014 \times 10^{2}$ | $5.8507 \times 10^{1}$ | $2.0727 \times 10^{1}$ |
| CEC2017-F22 | Ave | $1.9147 \times 10^{4}$ | $2.3964 \times 10^{4}$ | $1.4600 \times 10^{4}$ | $2.3618 \times 10^{4}$ | $1.8503 \times 10^{4}$ | $2.2047 \times 10^{4}$ | $2.2662 \times 10^{4}$ | $2.9752 \times 10^{4}$ | $1.8465 \times 10^{4}$ | $3.0181 \times 10^{4}$ |
| | Std | $3.4351 \times 10^{3}$ | $3.8542 \times 10^{3}$ | $5.7353 \times 10^{3}$ | $2.3026 \times 10^{3}$ | $1.2258 \times 10^{3}$ | $6.9765 \times 10^{3}$ | $1.7331 \times 10^{3}$ | $9.3367 \times 10^{2}$ | $1.8855 \times 10^{3}$ | $4.6507 \times 10^{3}$ |
| CEC2017-F23 | Ave | $3.2901 \times 10^{3}$ | $3.3057 \times 10^{3}$ | $3.2787 \times 10^{3}$ | $3.5467 \times 10^{3}$ | $3.1765 \times 10^{3}$ | $3.1796 \times 10^{3}$ | $3.5807 \times 10^{3}$ | $3.1279 \times 10^{3}$ | $3.2147 \times 10^{3}$ | $3.0198 \times 10^{3}$ |
| | Std | $6.1841 \times 10^{1}$ | $8.6901 \times 10^{1}$ | $6.0468 \times 10^{1}$ | $2.3372 \times 10^{1}$ | $4.8020 \times 10^{1}$ | $5.6195 \times 10^{1}$ | $1.6950 \times 10^{2}$ | $5.1835 \times 10^{1}$ | $5.0010 \times 10^{1}$ | $3.2517 \times 10^{1}$ |
| CEC2017-F24 | Ave | $3.8708 \times 10^{3}$ | $3.9951 \times 10^{3}$ | $3.6959 \times 10^{3}$ | $4.1662 \times 10^{3}$ | $3.6581 \times 10^{3}$ | $3.6391 \times 10^{3}$ | $4.2663 \times 10^{3}$ | $3.6577 \times 10^{3}$ | $3.7900 \times 10^{3}$ | $3.4339 \times 10^{3}$ |
| | Std | $1.1900 \times 10^{2}$ | $1.1000 \times 10^{2}$ | $7.3386 \times 10^{1}$ | $3.2190 \times 10^{1}$ | $6.7183 \times 10^{1}$ | $6.4127 \times 10^{1}$ | $2.0348 \times 10^{2}$ | $8.3090 \times 10^{1}$ | $9.1430 \times 10^{1}$ | $3.3121 \times 10^{1}$ |
| CEC2017-F25 | Ave | $3.3821 \times 10^{3}$ | $3.6049 \times 10^{3}$ | $3.6035 \times 10^{3}$ | $5.6725 \times 10^{3}$ | $3.4252 \times 10^{3}$ | $3.4859 \times 10^{3}$ | $3.5348 \times 10^{3}$ | $4.3815 \times 10^{3}$ | $3.6110 \times 10^{3}$ | $3.2703 \times 10^{3}$ |
| | Std | $5.1848 \times 10^{1}$ | $9.8536 \times 10^{1}$ | $6.9321 \times 10^{1}$ | $1.6037 \times 10^{2}$ | $6.6191 \times 10^{1}$ | $5.0934 \times 10^{1}$ | $7.9807 \times 10^{1}$ | $3.0052 \times 10^{2}$ | $1.2027 \times 10^{2}$ | $5.0850 \times 10^{1}$ |
| CEC2017-F26 | Ave | $1.1720 \times 10^{4}$ | $1.4015 \times 10^{4}$ | $1.4567 \times 10^{4}$ | $1.5412 \times 10^{4}$ | $1.0052 \times 10^{4}$ | $9.4798 \times 10^{3}$ | $1.3063 \times 10^{4}$ | $9.6390 \times 10^{3}$ | $1.3782 \times 10^{4}$ | $7.0934 \times 10^{3}$ |
| | Std | $1.2317 \times 10^{3}$ | $2.0469 \times 10^{3}$ | $5.8420 \times 10^{3}$ | $3.6692 \times 10^{2}$ | $7.5648 \times 10^{2}$ | $5.1247 \times 10^{2}$ | $1.7015 \times 10^{3}$ | $6.5037 \times 10^{2}$ | $3.2203 \times 10^{3}$ | $2.3801 \times 10^{2}$ |
| CEC2017-F27 | Ave | $3.6557 \times 10^{3}$ | $3.7558 \times 10^{3}$ | $3.7740 \times 10^{3}$ | $3.8946 \times 10^{3}$ | $3.5658 \times 10^{3}$ | $3.4287 \times 10^{3}$ | $3.5226 \times 10^{3}$ | $3.7109 \times 10^{3}$ | $3.5892 \times 10^{3}$ | $3.5068 \times 10^{3}$ |
| | Std | $9.4630 \times 10^{1}$ | $1.0374 \times 10^{2}$ | $1.2417 \times 10^{2}$ | $6.0549 \times 10^{1}$ | $9.9748 \times 10^{1}$ | $4.0292 \times 10^{1}$ | $6.2107 \times 10^{1}$ | $7.9100 \times 10^{1}$ | $7.5451 \times 10^{1}$ | $3.6778 \times 10^{1}$ |
| CEC2017-F28 | Ave | $3.4883 \times 10^{3}$ | $3.8545 \times 10^{3}$ | $3.7399 \times 10^{3}$ | $6.9463 \times 10^{3}$ | $3.5473 \times 10^{3}$ | $3.5151 \times 10^{3}$ | $3.7818 \times 10^{3}$ | $6.2668 \times 10^{3}$ | $3.7350 \times 10^{3}$ | $3.4021 \times 10^{3}$ |
| | Std | $3.9728 \times 10^{1}$ | $1.7091 \times 10^{2}$ | $5.7352 \times 10^{1}$ | $4.3816 \times 10^{2}$ | $3.0065 \times 10^{1}$ | $3.4859 \times 10^{1}$ | $1.4084 \times 10^{2}$ | $7.9983 \times 10^{2}$ | $5.8356 \times 10^{1}$ | $3.7373 \times 10^{1}$ |
| CEC2017-F29 | Ave | $6.2833 \times 10^{3}$ | $6.4310 \times 10^{3}$ | $7.2054 \times 10^{3}$ | $8.2914 \times 10^{3}$ | $6.8524 \times 10^{3}$ | $6.0556 \times 10^{3}$ | $7.2724 \times 10^{3}$ | $6.1302 \times 10^{3}$ | $6.3823 \times 10^{3}$ | $5.1616 \times 10^{3}$ |
| | Std | $5.1510 \times 10^{2}$ | $4.4050 \times 10^{2}$ | $4.8165 \times 10^{2}$ | $3.1281 \times 10^{2}$ | $5.2648 \times 10^{2}$ | $5.5642 \times 10^{2}$ | $6.2413 \times 10^{2}$ | $7.1563 \times 10^{2}$ | $5.1835 \times 10^{2}$ | $3.6989 \times 10^{2}$ |
| CEC2017-F30 | Ave | $5.1238 \times 10^{4}$ | $1.8996 \times 10^{5}$ | $4.0863 \times 10^{6}$ | $2.1047 \times 10^{7}$ | $9.0194 \times 10^{4}$ | $2.7021 \times 10^{4}$ | $7.0792 \times 10^{4}$ | $6.2361 \times 10^{6}$ | $5.7704 \times 10^{4}$ | $1.1205 \times 10^{4}$ |
| | Std | $3.7408 \times 10^{4}$ | $1.7521 \times 10^{5}$ | $1.1122 \times 10^{6}$ | $4.3527 \times 10^{6}$ | $4.1427 \times 10^{4}$ | $2.1326 \times 10^{4}$ | $1.2400 \times 10^{5}$ | $1.1453 \times 10^{7}$ | $4.9220 \times 10^{4}$ | $2.8422 \times 10^{3}$ |



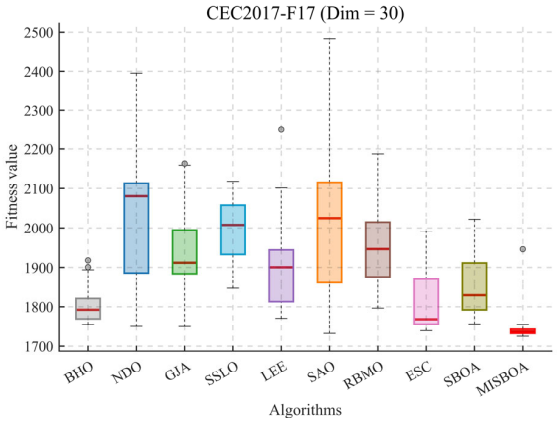

(**a**) CEC2017-F5 (Dim = 30)



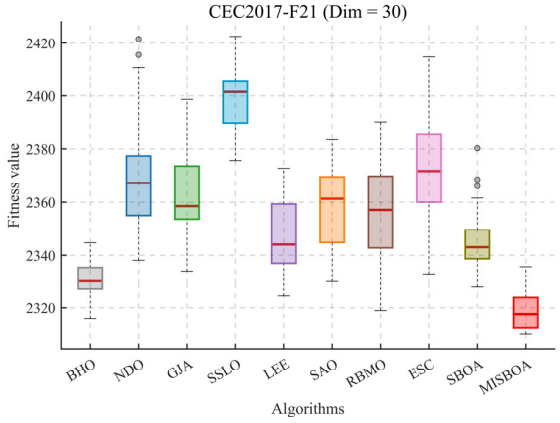

(**b**) CEC2017-F7 (Dim = 30)



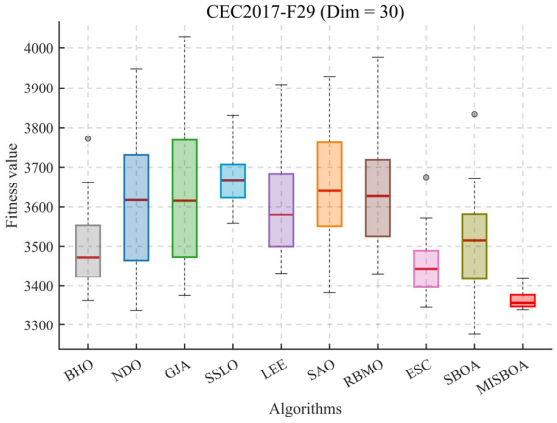

(**c**) CEC2017-F11 (Dim = 30)



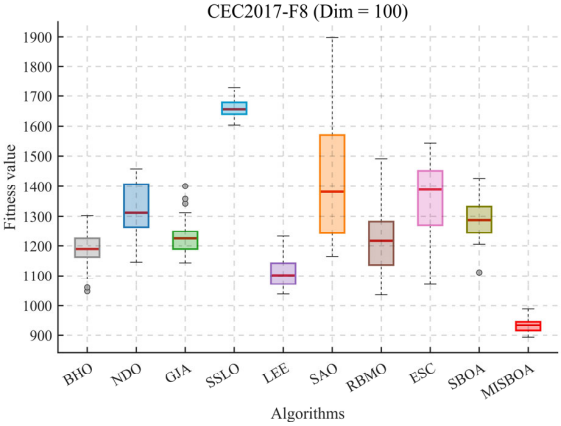

(**d**) CEC2017-F17 (Dim = 30)



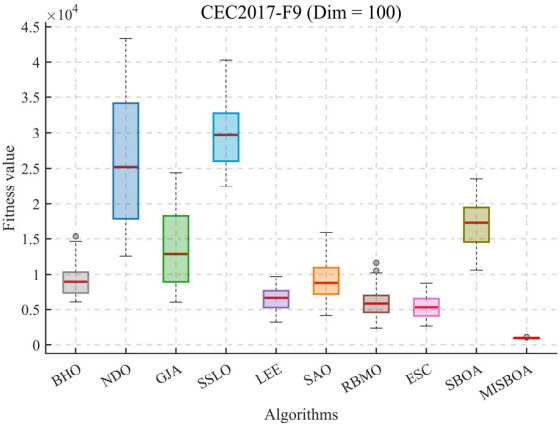

(**e**) CEC2017-F21 (Dim = 30)




(**f**) CEC2017-F29 (Dim = 30)

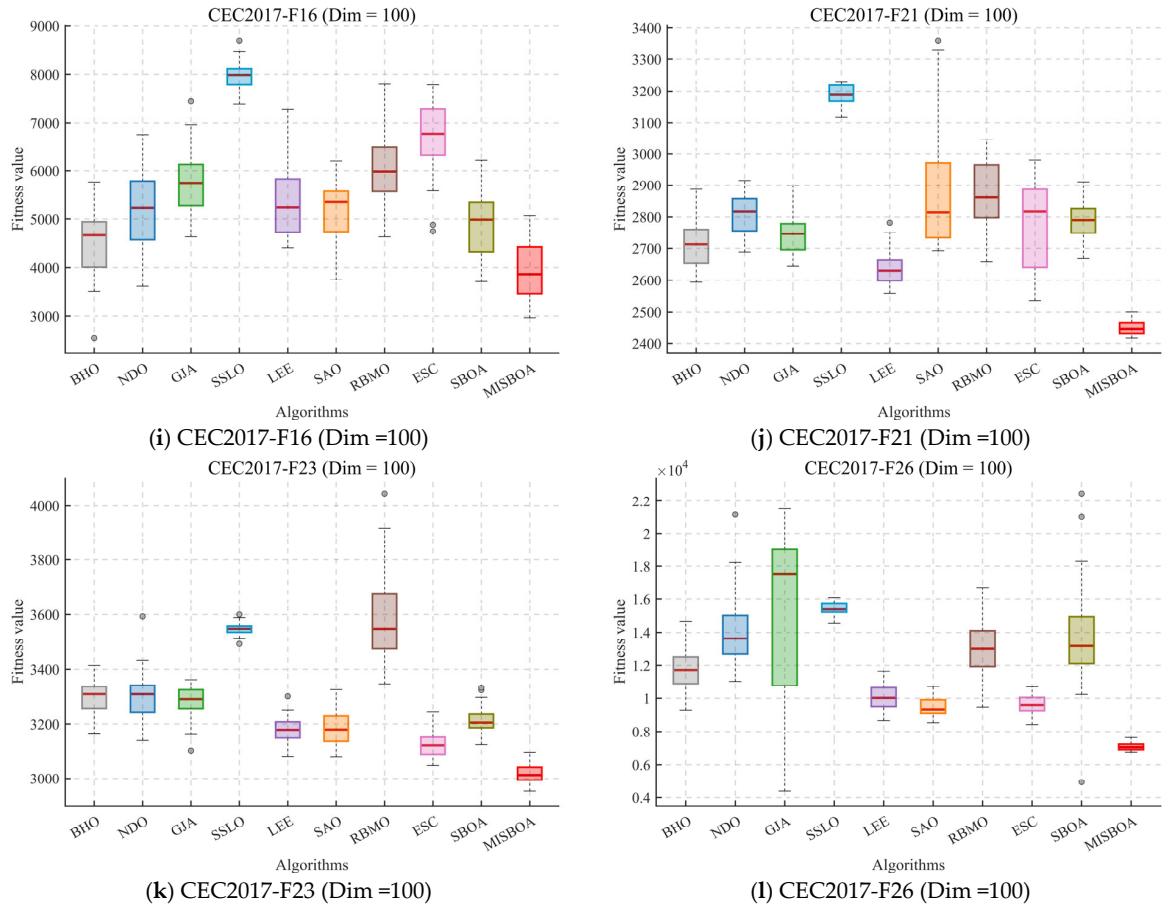



(**g**) CEC2017-F8 (Dim =100)




(**h**) CEC2017-F9 (Dim =100)

**Figure 8.** *Cont.*




(**i**) CEC2017-F16 (Dim =100) (**j**) CEC2017-F21 (Dim =100)

(**k**) CEC2017-F23 (Dim =100) (**l**) CEC2017-F26 (Dim =100)

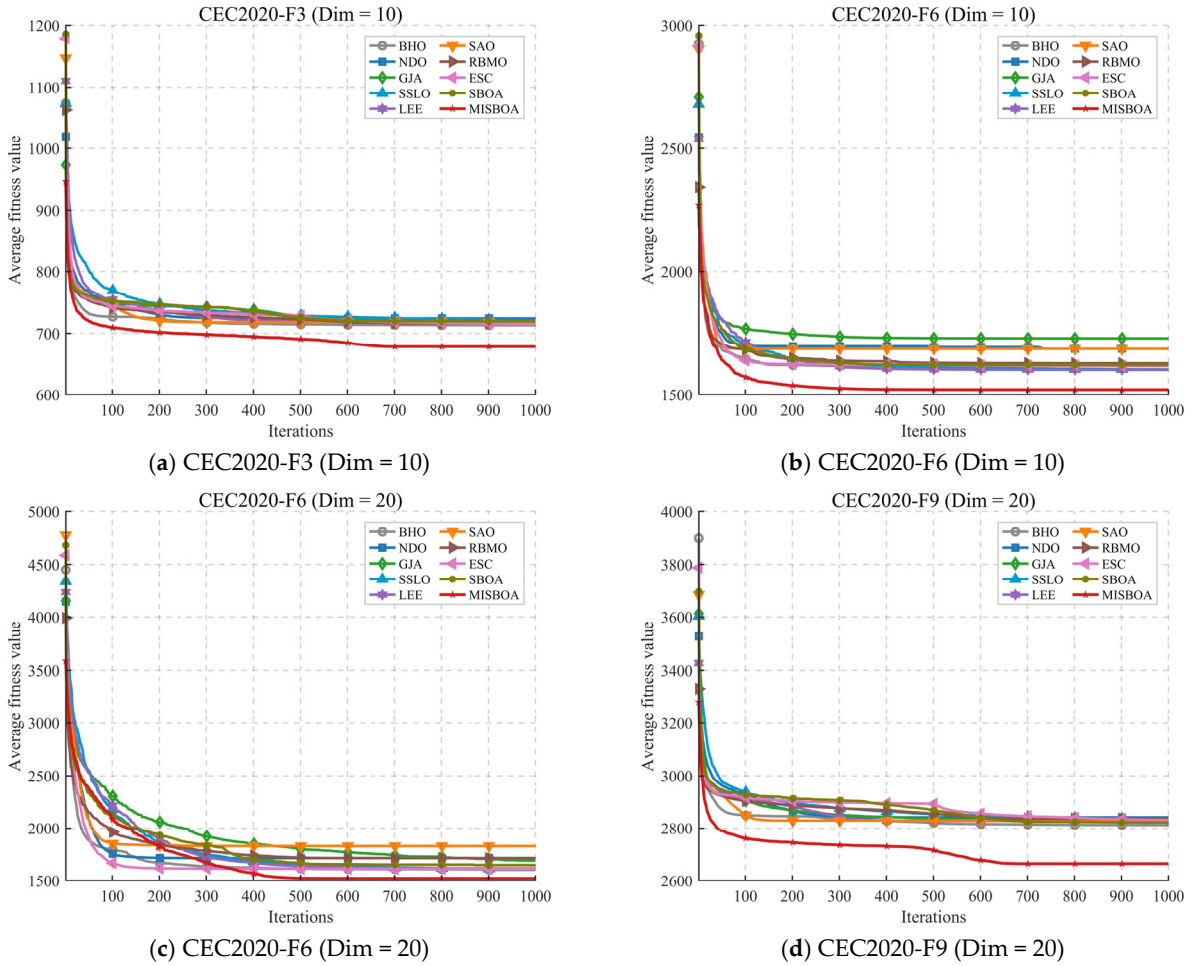
**Figure 8.** Boxplot-Based Performance Comparison of Different Algorithms on the CEC2017 Test Suite.

Figure 8 shows the box plot results of each algorithm on some functions of the CEC2017 test set, used to further evaluate the stability and robustness of different algorithms in 30 independent runs. As can be seen from the figure, MISBOA has a narrower overall box, a better median position, and a shorter upper and lower whisker range, indicating that it has smaller result fluctuations and higher consistency in multiple independent runs. Especially on some complex mixed and combined functions, the box plot distribution of MISBOA is more concentrated, with fewer outliers, indicating that it can maintain good stability in complex search environments and effectively reduce the impact of random factors on the optimization results. In contrast, although some of the compared algorithms can obtain good optimal values on a few functions, their boxes are wider, their upper and lower whiskers are longer, and they are accompanied by more outliers, indicating that the results of these algorithms fluctuate more and are easily affected by the initial population and local optima, resulting in relatively poor stability. Overall, Figure 8 further validates the robustness and reliability of MISBOA on the CEC2017 test set, demonstrating that the proposed improvement strategy not only enhances the algorithm's optimization ability but also strengthens its stable performance in complex high-dimensional optimization problems.

#### 3.3.3. Performance Comparison on the IEEE CEC2020 Benchmark Functions

In this section, we evaluate the numerical optimization performance of MISBOA based on the 10-dimensional and 20-dimensional functions of the CEC2020 benchmark test suite. Specifically, MISBOA is compared against nine representative state-of-the-art algorithms, and the corresponding optimization results are presented in Tables 5 and 6. In these tables, "Ave" and "Std" denote the mean and standard deviation, respectively, of the results obtained from 30 independent runs. To further elucidate the search behaviors of the

compared algorithms, Figure 9 displays the convergence curves for these 10 algorithms. Furthermore, Figure 10 presents the box plots of the results for each algorithm across the 30 independent runs; these results serve to assess the robustness and stability of the algorithms while helping to mitigate the influence of stochastic factors.




(**a**) CEC2020-F3 (Dim = 10) (**b**) CEC2020-F6 (Dim = 10)

(**c**) CEC2020-F6 (Dim = 20) (**d**) CEC2020-F9 (Dim = 20)

**Figure 9.** Convergence Speed Comparison of Different Algorithms on the CEC2020 Test Suite.

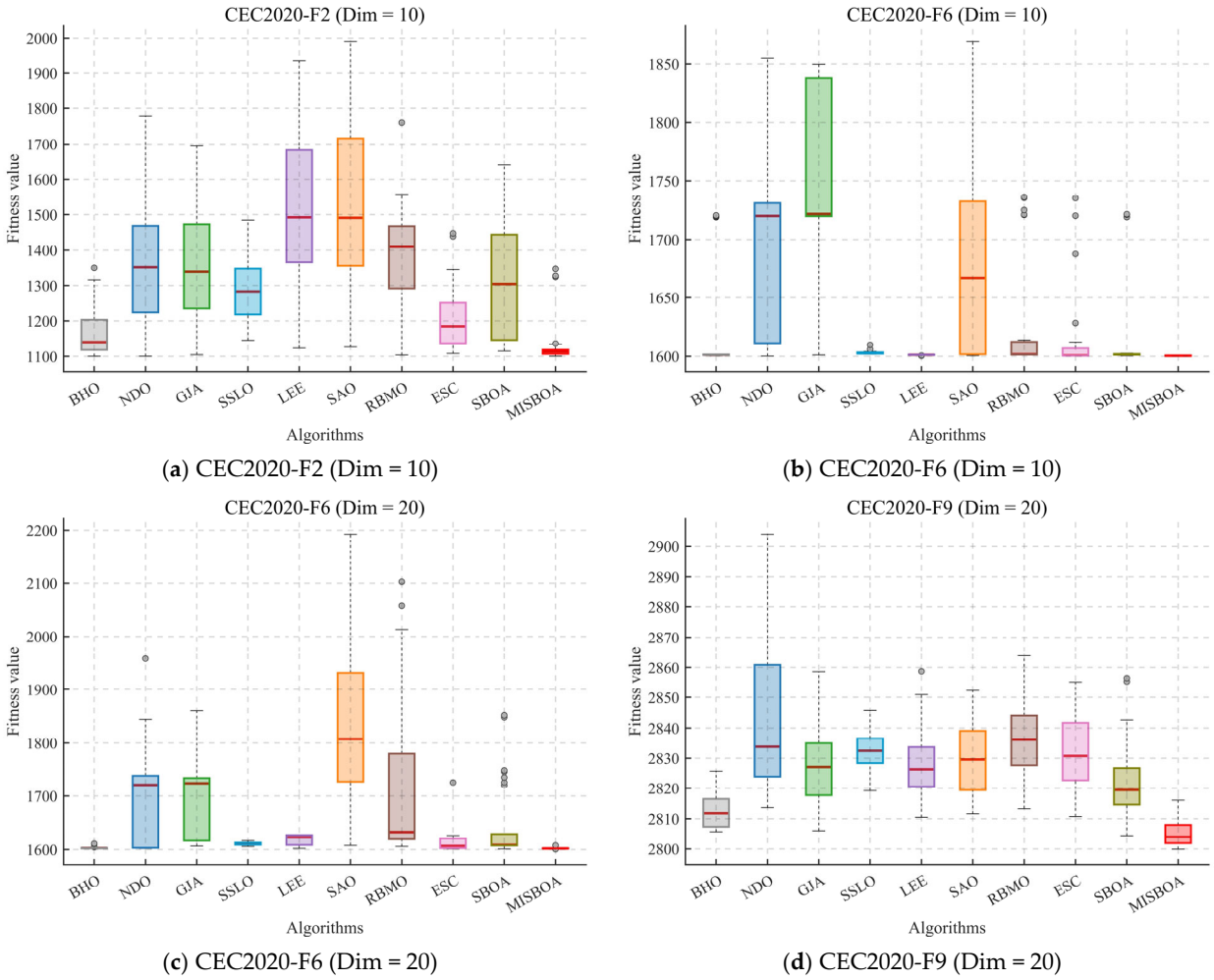
Figure 9 shows the convergence curves of various algorithms on some functions of the CEC2020 test set. Overall, MISBOA exhibits faster convergence speed and better final optimization results on most functions. Especially on complex mixed and combined functions, MISBOA can rapidly reduce the objective function value in the early stages of iteration and maintain a downward trend in subsequent iterations, indicating its strong global search and local optimization capabilities. In contrast, some of the compared algorithms, while converging quickly in the early stages, tend to stagnate in the middle and later stages, making it difficult to further approach the global optimum; others show a slower rate of decrease throughout the optimization process, indicating a strong local optimum trap in complex high-dimensional search spaces. MISBOA's convergence curve is generally lower than other algorithms and can obtain better solutions with fewer iterations, demonstrating its higher search efficiency and stronger continuous optimization capability. In summary, Figure 9 illustrates that the proposed chaotic elite initialization, adaptive spiral Lévy flight, and dynamic neighborhood-guided mutation strategies can effectively enhance the algorithm's exploration and development capabilities on complex test functions of CEC2020, thereby enabling MISBOA to achieve better convergence performance and higher optimization accuracy.

**Table 5.** Performance Comparison of Different Algorithms on the 10-Dimensional CEC2020 Benchmark Functions.

| ID | Items | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CEC2020-F1 | Ave | $1.0000 \times 10^{2}$ | $2.0320 \times 10^{3}$ | $5.1276 \times 10^{4}$ | $1.7130 \times 10^{3}$ | $1.0000 \times 10^{2}$ | $1.4940 \times 10^{3}$ | $1.0000 \times 10^{2}$ | $1.4128 \times 10^{3}$ | $3.4987 \times 10^{3}$ | $1.0000 \times 10^{2}$ |
| | Std | $0.0000 \times 10^{00}$ | $2.2764 \times 10^{3}$ | $1.5061 \times 10^{4}$ | $1.4145 \times 10^{3}$ | $6.1731 \times 10^{-6}$ | $1.7886 \times 10^{3}$ | $1.0998 \times 10^{-9}$ | $1.8063 \times 10^{3}$ | $3.8100 \times 10^{3}$ | $0.0000 \times 10^{00}$ |
| CEC2020-F2 | Ave | $1.1673 \times 10^{3}$ | $1.3691 \times 10^{3}$ | $1.3519 \times 10^{3}$ | $1.2821 \times 10^{3}$ | $1.5156 \times 10^{3}$ | $1.5442 \times 10^{3}$ | $1.3842 \times 10^{3}$ | $1.2153 \times 10^{3}$ | $1.3187 \times 10^{3}$ | $1.1353 \times 10^{3}$ |
| | Std | $6.6420 \times 10^{1}$ | $1.9253 \times 10^{2}$ | $1.6758 \times 10^{2}$ | $8.7577 \times 10^{1}$ | $1.9987 \times 10^{2}$ | $2.4909 \times 10^{2}$ | $1.4950 \times 10^{2}$ | $9.5647 \times 10^{1}$ | $1.5524 \times 10^{2}$ | $6.7669 \times 10^{1}$ |
| CEC2020- F3 | Ave | $7.1302 \times 10^{2}$ | $7.2386 \times 10^{2}$ | $7.1878 \times 10^{2}$ | $7.2011 \times 10^{2}$ | $7.1867 \times 10^{2}$ | $7.1790 \times 10^{2}$ | $7.1610 \times 10^{2}$ | $7.1495 \times 10^{2}$ | $7.1926 \times 10^{2}$ | $7.1432 \times 10^{2}$ |
| | Std | $1.1134 \times 10^{00}$ | $6.2820 \times 10^{00}$ | $3.5085 \times 10^{00}$ | $2.6383 \times 10^{00}$ | $4.3009 \times 10^{00}$ | $3.3847 \times 10^{00}$ | $4.0017 \times 10^{00}$ | $1.8063 \times 10^{00}$ | $5.2250 \times 10^{00}$ | $1.9631 \times 10^{00}$ |
| CEC2020- F4 | Ave | $1.9005 \times 10^{3}$ | $1.9011 \times 10^{3}$ | $1.9009 \times 10^{3}$ | $1.9011 \times 10^{3}$ | $1.9008 \times 10^{3}$ | $1.9009 \times 10^{3}$ | $1.9007 \times 10^{3}$ | $1.9011 \times 10^{3}$ | $1.9009 \times 10^{3}$ | $1.9006 \times 10^{3}$ |
| | Std | $2.1196 \times 10^{-1}$ | $5.2379 \times 10^{-1}$ | $3.8904 \times 10^{-1}$ | $2.2809 \times 10^{-1}$ | $2.2350 \times 10^{-1}$ | $3.4985 \times 10^{-1}$ | $1.8610 \times 10^{-1}$ | $2.0396 \times 10^{-1}$ | $3.1218 \times 10^{-1}$ | $2.0031 \times 10^{-1}$ |
| CEC2020- F5 | Ave | $1.8181 \times 10^{3}$ | $5.4815 \times 10^{3}$ | $3.1354 \times 10^{3}$ | $1.2650 \times 10^{4}$ | $1.7566 \times 10^{3}$ | $3.3108 \times 10^{3}$ | $1.7488 \times 10^{3}$ | $5.1592 \times 10^{3}$ | $2.9188 \times 10^{3}$ | $1.7079 \times 10^{3}$ |
| | Std | $9.0924 \times 10^{1}$ | $2.9586 \times 10^{3}$ | $1.2181 \times 10^{3}$ | $6.7002 \times 10^{3}$ | $4.0437 \times 10^{1}$ | $1.1072 \times 10^{3}$ | $3.3008 \times 10^{1}$ | $3.0830 \times 10^{3}$ | $1.1159 \times 10^{3}$ | $9.0020 \times 10^{00}$ |
| CEC2020- F6 | Ave | $1.6170 \times 10^{3}$ | $1.6874 \times 10^{3}$ | $1.7296 \times 10^{3}$ | $1.6031 \times 10^{3}$ | $1.6012 \times 10^{3}$ | $1.6875 \times 10^{3}$ | $1.6282 \times 10^{3}$ | $1.6179 \times 10^{3}$ | $1.6214 \times 10^{3}$ | $1.6005 \times 10^{3}$ |
| | Std | $4.0928 \times 10^{1}$ | $7.9769 \times 10^{1}$ | $8.2648 \times 10^{1}$ | $1.6116 \times 10^{00}$ | $3.3895 \times 10^{-1}$ | $9.4651 \times 10^{1}$ | $5.1459 \times 10^{1}$ | $3.9963 \times 10^{1}$ | $4.5224 \times 10^{1}$ | $2.3636 \times 10^{-1}$ |
| CEC2020- F7 | Ave | $2.1107 \times 10^{3}$ | $2.4271 \times 10^{3}$ | $3.5007 \times 10^{3}$ | $2.8563 \times 10^{3}$ | $2.1179 \times 10^{3}$ | $8.4079 \times 10^{3}$ | $2.1129 \times 10^{3}$ | $2.6306 \times 10^{3}$ | $2.2323 \times 10^{3}$ | $2.1003 \times 10^{3}$ |
| | Std | $2.5176 \times 10^{1}$ | $1.2338 \times 10^{3}$ | $1.2997 \times 10^{3}$ | $6.1264 \times 10^{2}$ | $8.9514 \times 10^{00}$ | $4.9000 \times 10^{3}$ | $1.5142 \times 10^{1}$ | $8.6007 \times 10^{2}$ | $1.1022 \times 10^{2}$ | $2.8797 \times 10^{-1}$ |
| CEC2020- F8 | Ave | $2.2941 \times 10^{3}$ | $2.2992 \times 10^{3}$ | $2.3042 \times 10^{3}$ | $2.2974 \times 10^{3}$ | $2.2922 \times 10^{3}$ | $2.2982 \times 10^{3}$ | $2.2978 \times 10^{3}$ | $2.3003 \times 10^{3}$ | $2.2941 \times 10^{3}$ | $2.2933 \times 10^{3}$ |
| | Std | $2.3273 \times 10^{1}$ | $1.4046 \times 10^{1}$ | $9.9570 \times 10^{-1}$ | $1.0513 \times 10^{1}$ | $2.6286 \times 10^{1}$ | $1.3058 \times 10^{1}$ | $1.6289 \times 10^{1}$ | $2.6509 \times 10^{-1}$ | $2.4049 \times 10^{1}$ | $2.5371 \times 10^{1}$ |
| CEC2020- F9 | Ave | $2.6007 \times 10^{3}$ | $2.6307 \times 10^{3}$ | $2.7046 \times 10^{3}$ | $2.6949 \times 10^{3}$ | $2.6995 \times 10^{3}$ | $2.7444 \times 10^{3}$ | $2.7141 \times 10^{3}$ | $2.7366 \times 10^{3}$ | $2.7063 \times 10^{3}$ | $2.6076 \times 10^{3}$ |
| | Std | $1.1717 \times 10^{2}$ | $1.1994 \times 10^{2}$ | $8.1504 \times 10^{1}$ | $6.3528 \times 10^{1}$ | $9.0782 \times 10^{1}$ | $6.1567 \times 10^{00}$ | $7.2693 \times 10^{1}$ | $2.6107 \times 10^{00}$ | $8.2351 \times 10^{1}$ | $1.1705 \times 10^{2}$ |
| CEC2020- F10 | Ave | $2.9178 \times 10^{3}$ | $2.9268 \times 10^{3}$ | $2.9231 \times 10^{3}$ | $2.8942 \times 10^{3}$ | $2.9091 \times 10^{3}$ | $2.9366 \times 10^{3}$ | $2.9140 \times 10^{3}$ | $2.9351 \times 10^{3}$ | $2.9122 \times 10^{3}$ | $2.9133 \times 10^{3}$ |
| | Std | $2.2831 \times 10^{1}$ | $2.3330 \times 10^{1}$ | $2.2924 \times 10^{1}$ | $3.3719 \times 10^{1}$ | $2.0275 \times 10^{1}$ | $1.9358 \times 10^{1}$ | $2.2738 \times 10^{1}$ | $2.0629 \times 10^{1}$ | $2.1337 \times 10^{1}$ | $2.2138 \times 10^{1}$ |

**Table 6.** Performance Comparison of Different Algorithms on the 20-Dimensional CEC2020 Benchmark Functions.

| ID | Items | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CEC2020-F1 | Ave | $1.0000 \times 10^{2}$ | $2.6474 \times 10^{3}$ | $3.6400 \times 10^{5}$ | $9.6632 \times 10^{4}$ | $1.0003 \times 10^{2}$ | $1.8602 \times 10^{3}$ | $1.0006 \times 10^{2}$ | $1.4788 \times 10^{3}$ | $3.8620 \times 10^{3}$ | $1.0000 \times 10^{2}$ |
| | Std | $5.2596 \times 10^{-7}$ | $2.6801 \times 10^{3}$ | $6.0721 \times 10^{4}$ | $4.9678 \times 10^{4}$ | $3.3184 \times 10^{-2}$ | $2.5999 \times 10^{3}$ | $9.5109 \times 10^{-2}$ | $1.6985 \times 10^{3}$ | $3.8546 \times 10^{3}$ | $2.6389 \times 10^{-15}$ |
| CEC2020-F2 | Ave | $1.3133 \times 10^{3}$ | $1.6728 \times 10^{3}$ | $1.7697 \times 10^{3}$ | $1.7812 \times 10^{3}$ | $2.3493 \times 10^{3}$ | $2.1489 \times 10^{3}$ | $2.4181 \times 10^{3}$ | $1.3455 \times 10^{3}$ | $1.8534 \times 10^{3}$ | $1.2239 \times 10^{3}$ |
| | Std | $1.2688 \times 10^{2}$ | $2.7333 \times 10^{2}$ | $3.1742 \times 10^{2}$ | $1.3158 \times 10^{2}$ | $5.4236 \times 10^{2}$ | $4.3716 \times 10^{2}$ | $3.6818 \times 10^{2}$ | $2.1398 \times 10^{2}$ | $3.2082 \times 10^{2}$ | $1.1880 \times 10^{2}$ |
| CEC2020- F3 | Ave | $7.2467 \times 10^{2}$ | $7.6072 \times 10^{2}$ | $7.5152 \times 10^{2}$ | $7.5446 \times 10^{2}$ | $7.4120 \times 10^{2}$ | $7.4191 \times 10^{2}$ | $7.4805 \times 10^{2}$ | $7.3167 \times 10^{2}$ | $7.4779 \times 10^{2}$ | $7.2780 \times 10^{2}$ |
| | Std | $1.9301 \times 10^{00}$ | $2.9888 \times 10^{1}$ | $7.1785 \times 10^{00}$ | $5.3007 \times 10^{00}$ | $7.8036 \times 10^{00}$ | $7.4569 \times 10^{00}$ | $8.3167 \times 10^{00}$ | $4.7777 \times 10^{00}$ | $1.1321 \times 10^{1}$ | $3.5604 \times 10^{00}$ |
| CEC2020- F4 | Ave | $1.9012 \times 10^{3}$ | $1.9034 \times 10^{3}$ | $1.9060 \times 10^{3}$ | $1.9038 \times 10^{3}$ | $1.9028 \times 10^{3}$ | $1.9025 \times 10^{3}$ | $1.9025 \times 10^{3}$ | $1.9033 \times 10^{3}$ | $1.9027 \times 10^{3}$ | $1.9015 \times 10^{3}$ |
| | Std | $2.8079 \times 10^{-1}$ | $1.8456 \times 10^{00}$ | $1.6494 \times 10^{00}$ | $4.6540 \times 10^{-1}$ | $1.1288 \times 10^{00}$ | $1.8444 \times 10^{00}$ | $9.1032 \times 10^{-1}$ | $6.5264 \times 10^{-1}$ | $9.5541 \times 10^{-1}$ | $3.4225 \times 10^{-1}$ |
| CEC2020- F5 | Ave | $2.5462 \times 10^{3}$ | $8.7376 \times 10^{4}$ | $9.7698 \times 10^{4}$ | $4.3548 \times 10^{5}$ | $2.5373 \times 10^{3}$ | $1.0054 \times 10^{5}$ | $2.1211 \times 10^{3}$ | $2.2152 \times 10^{5}$ | $1.1113 \times 10^{5}$ | $1.8584 \times 10^{3}$ |
| | Std | $3.1032 \times 10^{2}$ | $5.9723 \times 10^{4}$ | $3.4071 \times 10^{4}$ | $2.4844 \times 10^{5}$ | $2.5393 \times 10^{2}$ | $5.0959 \times 10^{4}$ | $1.2771 \times 10^{2}$ | $1.7251 \times 10^{5}$ | $9.7544 \times 10^{4}$ | $7.7977 \times 10^{1}$ |
| CEC2020- F6 | Ave | $1.6032 \times 10^{3}$ | $1.7148 \times 10^{3}$ | $1.6912 \times 10^{3}$ | $1.6114 \times 10^{3}$ | $1.6182 \times 10^{3}$ | $1.8301 \times 10^{3}$ | $1.7152 \times 10^{3}$ | $1.6133 \times 10^{3}$ | $1.6486 \times 10^{3}$ | $1.6023 \times 10^{3}$ |
| | Std | $2.2062 \times 10^{00}$ | $9.8260 \times 10^{1}$ | $7.8896 \times 10^{1}$ | $3.0660 \times 10^{00}$ | $1.0292 \times 10^{1}$ | $1.6252 \times 10^{2}$ | $1.4309 \times 10^{2}$ | $2.2822 \times 10^{1}$ | $7.2413 \times 10^{1}$ | $1.5090 \times 10^{00}$ |
| CEC2020- F7 | Ave | $2.3531 \times 10^{3}$ | $8.9650 \times 10^{4}$ | $4.8649 \times 10^{4}$ | $1.0608 \times 10^{5}$ | $2.7291 \times 10^{3}$ | $9.2191 \times 10^{4}$ | $2.4673 \times 10^{3}$ | $8.4762 \times 10^{4}$ | $3.0750 \times 10^{4}$ | $2.1406 \times 10^{3}$ |
| | Std | $1.3704 \times 10^{2}$ | $8.6398 \times 10^{4}$ | $3.4140 \times 10^{4}$ | $6.3115 \times 10^{4}$ | $1.8521 \times 10^{2}$ | $8.6143 \times 10^{4}$ | $1.1016 \times 10^{2}$ | $8.5509 \times 10^{4}$ | $3.2334 \times 10^{4}$ | $5.1018 \times 10^{1}$ |
| CEC2020- F8 | Ave | $2.3004 \times 10^{3}$ | $2.3008 \times 10^{3}$ | $2.3081 \times 10^{3}$ | $2.3019 \times 10^{3}$ | $2.5158 \times 10^{3}$ | $2.3537 \times 10^{3}$ | $2.4607 \times 10^{3}$ | $2.4741 \times 10^{3}$ | $2.3003 \times 10^{3}$ | $2.3000 \times 10^{3}$ |
| | Std | $5.5270 \times 10^{-1}$ | $7.0819 \times 10^{-1}$ | $8.8076 \times 10^{-1}$ | $5.3323 \times 10^{-1}$ | $6.6153 \times 10^{2}$ | $2.9269 \times 10^{2}$ | $6.1192 \times 10^{2}$ | $6.6256 \times 10^{2}$ | $6.1521 \times 10^{-1}$ | $1.1942 \times 10^{-13}$ |
| CEC2020- F9 | Ave | $2.8122 \times 10^{3}$ | $2.8420 \times 10^{3}$ | $2.8279 \times 10^{3}$ | $2.8326 \times 10^{3}$ | $2.8284 \times 10^{3}$ | $2.8295 \times 10^{3}$ | $2.8368 \times 10^{3}$ | $2.8315 \times 10^{3}$ | $2.8224 \times 10^{3}$ | $2.8050 \times 10^{3}$ |
| | Std | $5.2927 \times 10^{00}$ | $2.3999 \times 10^{1}$ | $1.2236 \times 10^{1}$ | $6.0194 \times 10^{00}$ | $1.1552 \times 10^{1}$ | $1.1981 \times 10^{1}$ | $1.2564 \times 10^{1}$ | $1.1963 \times 10^{1}$ | $1.2817 \times 10^{1}$ | $4.0310 \times 10^{00}$ |
| CEC2020- F10 | Ave | $2.9146 \times 10^{3}$ | $2.9511 \times 10^{3}$ | $2.9701 \times 10^{3}$ | $2.9176 \times 10^{3}$ | $2.9219 \times 10^{3}$ | $2.9259 \times 10^{3}$ | $2.9246 \times 10^{3}$ | $2.9211 \times 10^{3}$ | $2.9342 \times 10^{3}$ | $2.9136 \times 10^{3}$ |
| | Std | $6.8313 \times 10^{00}$ | $3.3864 \times 10^{1}$ | $2.6325 \times 10^{1}$ | $4.1139 \times 10^{00}$ | $2.2081 \times 10^{1}$ | $2.3885 \times 10^{1}$ | $2.3990 \times 10^{1}$ | $1.7529 \times 10^{1}$ | $2.8985 \times 10^{1}$ | $6.6357 \times 10^{-1}$ |




(**a**) CEC2020-F2 (Dim = 10)

(**b**) CEC2020-F6 (Dim = 10)

(**c**) CEC2020-F6 (Dim = 20)

(**d**) CEC2020-F9 (Dim = 20)

**Figure 10.** Boxplot-Based Performance Comparison of Different Algorithms on the CEC2020 Test Suite.

Figure 10 shows the box plot results of each algorithm on some functions of the CEC2020 test set, used to further evaluate the stability and robustness of different algorithms in multiple independent runs. As can be seen from the figure, MISBOA has narrower box sizes, lower medians, and shorter upper and lower bounds on most test functions, indicating that it maintains high consistency and small result fluctuations in 30 independent runs. Especially on complex mixed and combined functions, MISBOA has fewer outliers and a more concentrated box distribution, indicating that it can effectively reduce the impact of random initialization and complex search spaces on the optimization results, thus achieving more stable performance. In contrast, while some of the compared algorithms achieve better results on a few functions, their box sizes are wider and have more outliers, indicating that these algorithms are easily affected by random factors and have relatively poor stability. Overall, Figure 10 further verifies the robustness and reliability of MISBOA on the CEC2020 test set, demonstrating that the proposed improvement strategies can not only improve the convergence accuracy of the algorithms but also enhance their stability in complex high-dimensional optimization problems.

Tables 5 and 6 present the mean and standard deviation results of each algorithm on the 10-dimensional and 20-dimensional CEC2020 benchmark functions, respectively. Overall, MISBOA achieved optimal or near-optimal performance in both dimensional settings, demonstrating strong search capabilities and good dimensionality adaptability. In the 10-dimensional test, MISBOA exhibited lower mean fitness values and smaller

standard deviations on most functions, indicating that it not only obtains better solutions but also possesses high stability and robustness. As the problem dimension increased to 20 dimensions, the performance of most compared algorithms decreased to varying degrees, manifested as increased mean values, wider fluctuation ranges, and significantly increased standard deviations. This suggests that these algorithms are more prone to getting trapped in local optima in more complex search spaces. However, MISBOA still maintained superior solution accuracy and a small standard deviation on most test functions, demonstrating strong global search capabilities and stability in high-dimensional complex problems. Compared to the standard SBOA, MISBOA demonstrates significant improvements on most functions, further proving that the chaotic elite initialization strategy, the adaptive spiral Lévy flight strategy, and the dynamic neighborhood-guided mutation strategy play a positive role in enhancing population diversity, balancing exploration and exploitation, and improving local search accuracy. Therefore, the experimental results of MISBOA on the 10-dimensional and 20-dimensional CEC2020 benchmark functions fully validate its excellent optimization performance and good generalization ability.

### *3.4. Statistical Analysis of Algorithm Performance*

Statistical analysis plays a crucial role in evaluating the performance of heuristic and meta-heuristic algorithms. Given that such algorithms typically involve random initialization and stochastic search mechanisms, the results obtained from a single run may not accurately reflect their true optimization capabilities. Consequently, relying solely on the best or average results is often insufficient to provide a fair and reliable comparison. Statistical analysis effectively mitigates the influence of randomness and reveals whether the observed performance differences between algorithms are statistically significant and robust. Furthermore, statistical methods offer deeper insights into an algorithm's robustness, consistency, and overall superiority across various benchmark problems. Therefore, incorporating statistical analysis is indispensable for conducting a comprehensive and compelling evaluation of heuristic algorithms. In this subsection, we subjected the proposed MISBOA to the Wilcoxon signed-rank test and the Friedman Mean Rank Test; the specific details are presented below.

#### 3.4.1. Wilcoxon Signed-Rank Test

In the performance evaluation of heuristic algorithms, relying solely on mean values, standard deviations, or convergence curves often proves insufficient to comprehensively reflect the true differences between algorithms; this is because heuristic algorithms exhibit significant stochasticity, leading to potentially substantial fluctuations across different independent runs [40]. Consequently, it is essential to incorporate statistical methods to further validate experimental results. As a non-parametric statistical test, the Wilcoxon signed-rank test does not require data to follow a normal distribution, making it particularly well-suited for comparing the performance of heuristic algorithms across multiple independent runs. By conducting a significance analysis on the performance differences between two algorithms across various test functions, one can effectively determine whether any observed performance improvement is statistically significant rather than merely attributable to random factors. Furthermore, the Wilcoxon signed-rank test provides an intuitive visualization of the extent of the improved algorithm's superiority over its counterparts through W/T/L (Win/Tie/Loss) results; as such, this test has emerged as a crucial tool for assessing the reliability and superiority of heuristic optimization algorithms. In this section, we performed the Wilcoxon signed-rank test on the MISBOA at a significance level ($\alpha = 0.05$); the experimental results are shown in Table 7.

**Table 7.** Experimental Results of the Wilcoxon Signed-Rank Test. (W/T/L).

| MISBOA.VS | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA |
|---|---|---|---|---|---|---|---|---|---|
| CEC2014 (dim = 30) | 23/7/0 | 27/2/1 | 28/2/0 | 28/1/1 | 22/2/6 | 27/2/1 | 28/1/1 | 25/4/1 | 23/3/4 |
| CEC2017 (dim = 30) | 27/2/1 | 30/0/0 | 29/1/0 | 30/0/0 | 29/1/0 | 29/1/0 | 29/1/0 | 28/2/0 | 28/2/0 |
| CEC2017 (dim = 100) | 24/3/3 | 28/0/2 | 26/1/3 | 28/0/2 | 25/2/3 | 26/1/3 | 26/2/2 | 28/1/1 | 26/1/3 |
| CEC2020 (dim = 10) | 5/3/2 | 9/1/0 | 9/1/0 | 8/2/0 | 8/2/0 | 10/0/0 | 8/2/0 | 10/0/0 | 9/1/0 |
| CEC2020 (dim = 20) | 7/1/2 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 9/1/0 | 10/0/0 | 9/1/0 | 10/0/0 |

Table 7 presents the W/T/L results—based on the Wilcoxon signed-rank test—comparing MISBOA against other benchmark algorithms. As indicated by the results, MISBOA demonstrates a distinct statistical advantage across various test sets and under different dimensionality conditions. On the CEC2014 (dim = 30) test set, MISBOA achieved a higher number of wins compared to the majority of the algorithms; for instance, its W/T/L scores against GJA and RBMO were 28/2/0 and 28/1/1, respectively. On the CEC2017 (dim = 30) test set, MISBOA's superiority was even more pronounced, achieving a score of 30/0/0 against both NDO and SSLO, thereby indicating that it significantly outperformed these benchmark algorithms across all test functions. Even on the higher-dimensional CEC2017 (dim = 100) test set, MISBOA maintained a high winning ratio, achieving W/T/L scores of 28/0/2 against both NDO and SSLO, and 28/1/1 against ESC. Regarding the CEC2020 test set, MISBOA similarly exhibited significant advantages under both dim = 10 and dim = 20 conditions, with multiple comparative results reaching 10/0/0. Overall, MISBOA consistently yielded results significantly superior to those of the benchmark algorithms across the majority of test functions, demonstrating that the proposed multi-strategy improvement mechanism effectively enhances the algorithm's global search capability, local exploitation capability, and overall stability.

#### 3.4.2. Friedman Mean Rank Test

Since the performance of heuristic algorithms often varies significantly across different test functions, relying solely on the optimal values, mean values, or standard deviations obtained from a single function makes it difficult to comprehensively evaluate an algorithm's overall performance [41]. Consequently, in scenarios involving multiple algorithms and multiple test problems, it is necessary to employ more comprehensive statistical analysis methods to facilitate a holistic comparison. The Friedman Mean Rank Test is a non-parametric statistical method well-suited for comparing multiple algorithms across multiple test functions; it evaluates the relative merits of algorithms from a global perspective by ranking their performance on each test function and subsequently calculating their average ranks. Compared to a mere comparison of numerical results, the Friedman Mean Rank Test effectively mitigates the impact of random fluctuations and isolated outliers, thereby providing a more objective reflection of an algorithm's stability, robustness, and comprehensive optimization capabilities across diverse problems. As a result, this method has become a widely adopted and essential statistical tool in the performance analysis of heuristic algorithms. In this subsection, we subjected MISBOA to the Friedman Mean Rank Test; the experimental results are presented in the table below.

Table 8 presents the results of the Friedman Mean Rank Test for each algorithm across various test sets. As the results indicate, MISBOA achieved the lowest mean rank (M.R.) and the highest final rank (T.R.) across all test sets and under varying dimensional conditions, demonstrating its superior overall performance. In the CEC2014 (dim = 30) test set, MISBOA attained a mean rank of 2.00 and a final rank of 1st—significantly outperforming the other comparative algorithms—with BHO securing second place with a mean rank of 2.30. In the CEC2017 (dim = 30) and CEC2017 (dim = 100) test sets, MISBOA achieved mean ranks

of 1.03 and 1.87, respectively, ranking 1st in both instances; this demonstrates that the algorithm maintains strong competitiveness in complex optimization problems of both medium and high dimensions. Notably, in the CEC2017 (dim = 30) test set, MISBOA's mean rank approached the theoretical optimum of 1, indicating that it achieved the best results across the vast majority of the test functions. For the CEC2020 test set, MISBOA's superiority was equally evident. Under the conditions of dim = 10 and dim = 20, its mean ranks were 1.30 and 1.20, respectively—firmly securing the first-place position in both cases—while the mean ranks of the other algorithms generally exceeded 2, with some even surpassing 7. This suggests that MISBOA not only possesses robust global search capabilities but also maintains stable optimization performance across test functions of varying dimensions and types. Overall, MISBOA achieved the best comprehensive ranking across all test sets, demonstrating that the proposed multi-strategy improvement mechanism effectively enhances the algorithm's convergence performance, search precision, and stability, thereby enabling it to exhibit superior overall competitiveness across a wide range of complex optimization problems.

**Table 8.** Friedman mean rank test result.

| **Suites** | **CEC2014** | | **CEC2017** | | | | **CEC2020** | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Dimensions** | **30** | | **30** | | **100** | | **10** | | **20** | |
| **Algorithms** | ***M.R*** | ***T.R*** | ***M.R*** | ***T.R*** | ***M.R*** | ***T.R*** | ***M.R*** | ***T.R*** | ***M.R*** | ***T.R*** |
| BHO | 2.30 | 2 | 3.10 | 2 | 3.90 | 3 | 2.20 | 2 | 2.10 | 2 |
| NDO | 6.43 | 7 | 6.67 | 8 | 6.73 | 8 | 7.50 | 8 | 7.00 | 8 |
| GJA | 7.47 | 9 | 7.60 | 9 | 6.17 | 7 | 7.90 | 10 | 7.80 | 9 |
| SSLO | 8.40 | 10 | 9.77 | 10 | 9.70 | 10 | 7.10 | 7 | 8.40 | 10 |
| LEE | 4.37 | 3 | 4.53 | 3 | 3.67 | 2 | 5.30 | 4 | 5.00 | 3 |
| SAO | 7.00 | 8 | 5.50 | 5 | 5.27 | 5 | 7.70 | 9 | 6.20 | 7 |
| RBMO | 5.33 | 4 | 4.53 | 3 | 5.20 | 4 | 4.80 | 3 | 5.70 | 4 |
| ESC | 6.17 | 6 | 6.57 | 7 | 7.00 | 9 | 5.60 | 5 | 5.90 | 6 |
| SBOA | 5.53 | 5 | 5.70 | 6 | 5.50 | 6 | 5.60 | 5 | 5.70 | 4 |
| MISBOA | 2.00 | 1 | 1.03 | 1 | 1.87 | 1 | 1.30 | 1 | 1.20 | 1 |

### *3.5. Comparison with CEC Competition Winners*

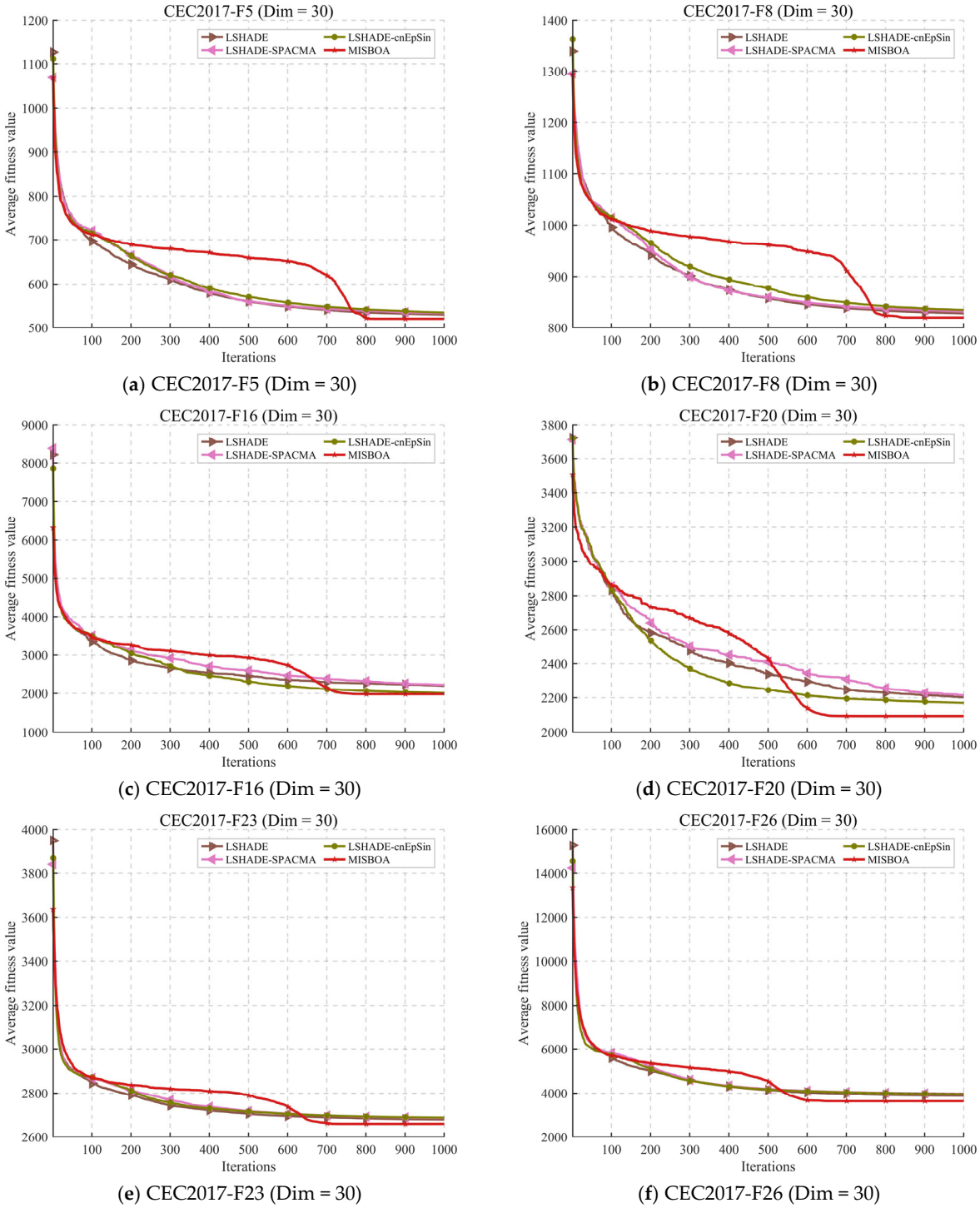
Comparison with CEC competition winners is of great significance in the evaluation of metaheuristic optimization algorithms because the winning algorithms in CEC competitions usually represent the state-of-the-art optimization performance under standardized benchmark environments. These algorithms are typically developed through extensive research and rigorous experimental validation, exhibiting strong global search capability, convergence accuracy, and robustness on complex optimization problems. Therefore, comparing the proposed algorithm with CEC competition winners can provide a more convincing assessment of its optimization effectiveness and competitiveness. Moreover, such comparisons help demonstrate whether the proposed algorithm can maintain superior performance when facing highly competitive and challenging optimization scenarios. Motivated by this, this subsection conducts experimental comparisons between MISBOA and several representative CEC competition-winning algorithms to further verify the effectiveness, robustness, and competitiveness of the proposed method. The experimental results are shown below.

Figure 11 presents the convergence curve comparisons between MISBOA and several representative CEC competition-winning algorithms, including LSHADE, LSHADE-cnEpSin, and LSHADE-SPACMA, on selected CEC2017 benchmark functions with 30 dimensions. Overall, MISBOA demonstrates highly competitive convergence behavior and

achieves superior optimization performance on most test functions. Specifically, on functions such as F5, F8, and F16, although MISBOA exhibits a relatively slower convergence rate during the early and middle stages of the iteration process, it maintains stronger continuous optimization capability in the later stages and ultimately obtains lower fitness values than the compared winner algorithms. This phenomenon indicates that the proposed strategies effectively enhance the algorithm's ability to escape local optima and maintain population diversity during the later search stage. On more complex functions such as F20, F23, and F26, MISBOA not only converges steadily throughout the optimization process but also achieves the best final convergence accuracy among all compared algorithms. In particular, the convergence curves of MISBOA exhibit more stable descending trends and lower final fitness values, demonstrating its strong balance between global exploration and local exploitation. Overall, the experimental results in Figure 11 verify that MISBOA possesses strong competitiveness even when compared with highly advanced CEC competition-winning algorithms, thereby further confirming the effectiveness, robustness, and optimization capability of the proposed enhancement strategies.

Figure 12 presents the boxplot comparisons between MISBOA and several representative CEC competition-winning algorithms on selected CEC2017 benchmark functions with 30 dimensions. The boxplots are used to evaluate the distribution characteristics, robustness, and stability of the algorithms over 30 independent runs. Overall, MISBOA exhibits significantly superior distribution characteristics compared with the competing algorithms, demonstrating stronger robustness and more stable optimization performance. Specifically, on functions such as F11, F15, F21, and F29, MISBOA achieves lower median fitness values and narrower box ranges, indicating that the proposed algorithm can consistently obtain high-quality solutions with relatively small fluctuations across multiple independent runs. In addition, MISBOA generally produces fewer outliers and shorter upper and lower whiskers, which further demonstrates its strong resistance to random disturbances and its ability to maintain stable convergence behavior. Although on certain functions such as F4 the distribution range of MISBOA is relatively wider during some runs, its median fitness value still remains significantly better than those of the compared algorithms, indicating superior overall optimization capability. Particularly on F6, MISBOA shows an extremely concentrated distribution with almost negligible variance, demonstrating excellent convergence stability and reliability. Overall, the experimental results shown in Figure 12 further confirm that MISBOA not only achieves competitive convergence accuracy when compared with advanced CEC competition-winning algorithms, but also maintains strong robustness, stability, and consistency in repeated optimization experiments.

Table 9 presents the comparative experimental results between MISBOA and several representative CEC competition-winning algorithms on the 30-dimensional CEC2017 benchmark functions. The compared algorithms include LSHADE, LSHADE-SPACMA, and LSHADE-cnEpSin, which are widely recognized as highly competitive optimization algorithms in CEC competitions. Overall, MISBOA demonstrates highly competitive optimization performance and achieves superior or near-optimal results on most benchmark functions. Specifically, MISBOA obtains the best average fitness values on a large number of functions, including F1, F3, F4, F5, F6, F8, F9, F10, F11, F15, F16, F17, F20, F21, F22, F23, F24, F27, F28, F29, and F30, indicating its strong global optimization capability and convergence accuracy. Particularly on complex multimodal and hybrid functions such as F15, F20, F21, and F29, MISBOA exhibits significantly lower fitness values than the compared winner algorithms, demonstrating that the proposed strategies can effectively enhance exploration capability and improve the ability to escape local optima.




(**a**) CEC2017-F5 (Dim = 30)

(**b**) CEC2017-F8 (Dim = 30)

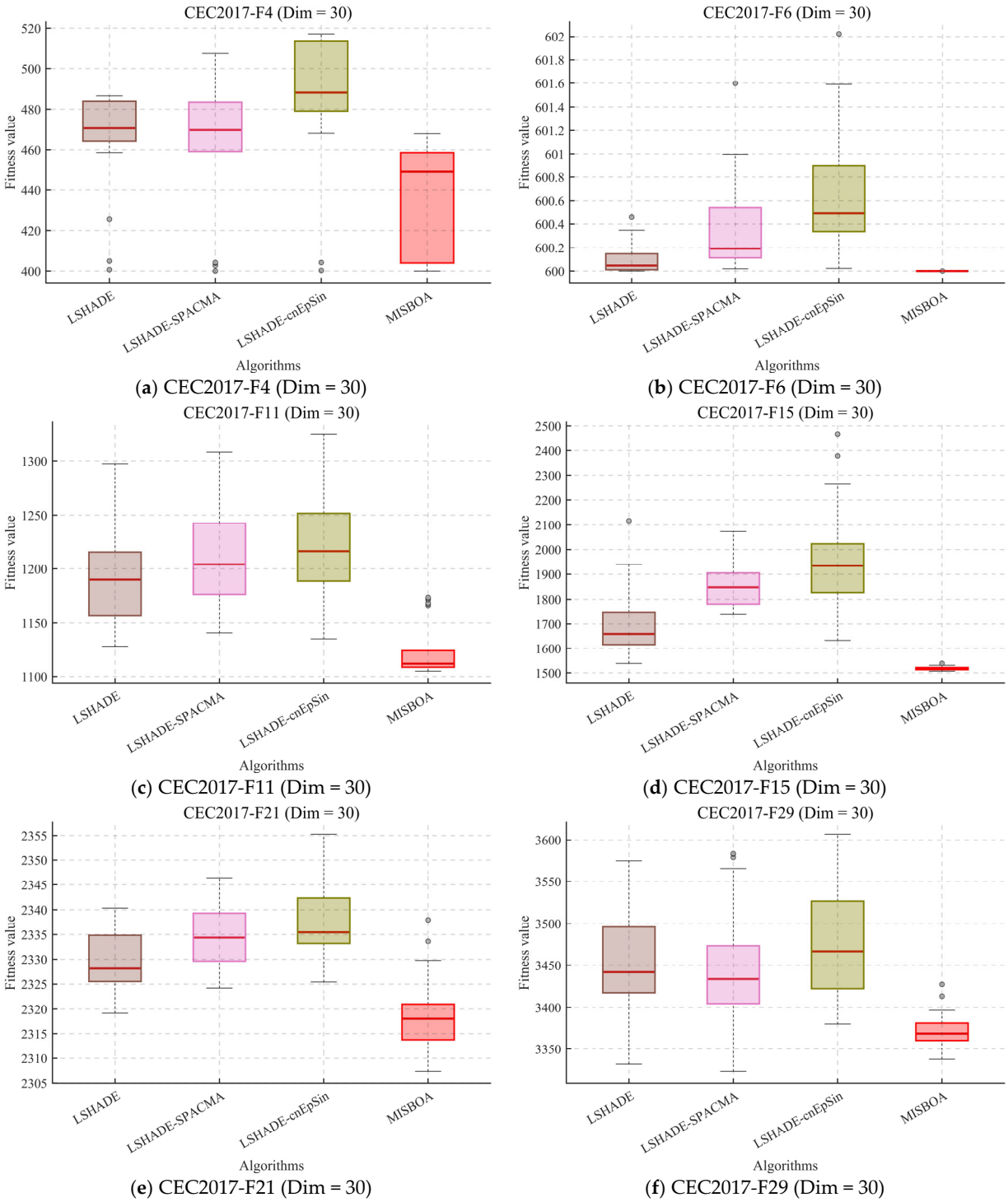
(**c**) CEC2017-F16 (Dim = 30)

(**d**) CEC2017-F20 (Dim = 30)

(**e**) CEC2017-F23 (Dim = 30)

(**f**) CEC2017-F26 (Dim = 30)

**Figure 11.** Convergence curves compared to the CEC winner algorithm.




(**a**) CEC2017-F4 (Dim = 30)

(**b**) CEC2017-F6 (Dim = 30)

(**c**) CEC2017-F11 (Dim = 30)

(**d**) CEC2017-F15 (Dim = 30)

(**e**) CEC2017-F21 (Dim = 30)

(**f**) CEC2017-F29 (Dim = 30)

**Figure 12.** Box plots comparing the algorithm with the CEC winner.

**Table 9.** Experimental results compared with CEC winner algorithm.

| Algorithm | Items | LSHADE | LSHADE-SPACMA | LSHADE-cnEpSin | MISBOA |
|---|---|---|---|---|---|
| CEC2017-F1 | Ave | $1.0708 \times 10^{2}$ | $1.0001 \times 10^{2}$ | $1.7712 \times 10^{3}$ | $1.0000 \times 10^{2}$ |
| | Std | $1.7515 \times 10^{1}$ | $4.8689 \times 10^{-2}$ | $2.4661 \times 10^{3}$ | $2.0270 \times 10^{-14}$ |
| CEC2017-F3 | Ave | $4.3045 \times 10^{3}$ | $3.5032 \times 10^{4}$ | $3.9070 \times 10^{3}$ | $3.0000 \times 10^{2}$ |
| | Std | $2.0679 \times 10^{4}$ | $5.4744 \times 10^{4}$ | $6.6174 \times 10^{3}$ | $1.8533 \times 10^{-10}$ |
| CEC2017-F4 | Ave | $4.6797 \times 10^{2}$ | $4.5903 \times 10^{2}$ | $4.8859 \times 10^{2}$ | $4.3582 \times 10^{2}$ |
| | Std | $2.1733 \times 10^{1}$ | $3.3214 \times 10^{1}$ | $2.8736 \times 10^{1}$ | $2.6922 \times 10^{1}$ |
| CEC2017-F5 | Ave | $5.2977 \times 10^{2}$ | $5.3351 \times 10^{2}$ | $5.3480 \times 10^{2}$ | $5.2040 \times 10^{2}$ |
| | Std | $4.8004 \times 10^{00}$ | $5.2945 \times 10^{00}$ | $6.2344 \times 10^{00}$ | $5.3405 \times 10^{00}$ |
| CEC2017-F6 | Ave | $6.0009 \times 10^{2}$ | $6.0035 \times 10^{2}$ | $6.0067 \times 10^{2}$ | $6.0000 \times 10^{2}$ |
| | Std | $1.1217 \times 10^{-1}$ | $3.5811 \times 10^{-1}$ | $4.8942 \times 10^{-1}$ | $1.9196 \times 10^{-6}$ |
| CEC2017-F7 | Ave | $7.6497 \times 10^{2}$ | $7.6628 \times 10^{2}$ | $7.6962 \times 10^{2}$ | $7.4610 \times 10^{2}$ |
| | Std | $7.7942 \times 10^{00}$ | $8.9243 \times 10^{00}$ | $9.1815 \times 10^{00}$ | $4.7083 \times 10^{00}$ |
| CEC2017-F8 | Ave | $8.2823 \times 10^{2}$ | $8.3177 \times 10^{2}$ | $8.3493 \times 10^{2}$ | $8.1987 \times 10^{2}$ |
| | Std | $6.3110 \times 10^{00}$ | $8.2330 \times 10^{00}$ | $5.2959 \times 10^{00}$ | $5.8919 \times 10^{00}$ |
| CEC2017-F9 | Ave | $9.2084 \times 10^{2}$ | $9.2396 \times 10^{2}$ | $9.2642 \times 10^{2}$ | $9.0001 \times 10^{2}$ |
| | Std | $1.5364 \times 10^{1}$ | $2.9162 \times 10^{1}$ | $2.3478 \times 10^{1}$ | $2.7318 \times 10^{-2}$ |
| CEC2017-F10 | Ave | $3.5280 \times 10^{3}$ | $3.7472 \times 10^{3}$ | $3.7694 \times 10^{3}$ | $3.3798 \times 10^{3}$ |
| | Std | $3.0783 \times 10^{2}$ | $4.4807 \times 10^{2}$ | $3.3263 \times 10^{2}$ | $5.7834 \times 10^{2}$ |
| CEC2017-F11 | Ave | $1.1932 \times 10^{3}$ | $1.2112 \times 10^{3}$ | $1.2193 \times 10^{3}$ | $1.1251 \times 10^{3}$ |
| | Std | $4.5062 \times 10^{1}$ | $4.4834 \times 10^{1}$ | $4.3971 \times 10^{1}$ | $2.5571 \times 10^{1}$ |
| CEC2017-F12 | Ave | $2.6469 \times 10^{4}$ | $2.9180 \times 10^{4}$ | $2.7267 \times 10^{4}$ | $2.5468 \times 10^{3}$ |
| | Std | $2.0869 \times 10^{4}$ | $2.0623 \times 10^{4}$ | $1.5669 \times 10^{4}$ | $5.6942 \times 10^{2}$ |
| CEC2017-F13 | Ave | $1.6453 \times 10^{3}$ | $4.5828 \times 10^{3}$ | $4.1883 \times 10^{3}$ | $1.3452 \times 10^{3}$ |
| | Std | $2.0936 \times 10^{2}$ | $1.6759 \times 10^{3}$ | $1.1327 \times 10^{3}$ | $3.2223 \times 10^{1}$ |
| CEC2017-F14 | Ave | $1.4891 \times 10^{3}$ | $1.5474 \times 10^{3}$ | $1.5493 \times 10^{3}$ | $1.4281 \times 10^{3}$ |
| | Std | $2.8302 \times 10^{1}$ | $5.1296 \times 10^{1}$ | $4.5612 \times 10^{1}$ | $3.6879 \times 10^{00}$ |
| CEC2017-F15 | Ave | $1.6953 \times 10^{3}$ | $1.8583 \times 10^{3}$ | $1.9588 \times 10^{3}$ | $1.5176 \times 10^{3}$ |
| | Std | $1.2336 \times 10^{2}$ | $9.7534 \times 10^{1}$ | $1.9385 \times 10^{2}$ | $6.9654 \times 10^{00}$ |
| CEC2017-F16 | Ave | $2.1973 \times 10^{3}$ | $2.2197 \times 10^{3}$ | $2.0098 \times 10^{3}$ | $1.9806 \times 10^{3}$ |
| | Std | $2.2161 \times 10^{2}$ | $2.0916 \times 10^{2}$ | $1.9007 \times 10^{2}$ | $2.3374 \times 10^{2}$ |
| CEC2017-F17 | Ave | $1.8120 \times 10^{3}$ | $1.8293 \times 10^{3}$ | $1.8281 \times 10^{3}$ | $1.7369 \times 10^{3}$ |
| | Std | $4.4827 \times 10^{1}$ | $5.4628 \times 10^{1}$ | $6.3643 \times 10^{1}$ | $7.6043 \times 10^{00}$ |
| CEC2017-F18 | Ave | $2.0552 \times 10^{3}$ | $7.9281 \times 10^{3}$ | $6.0787 \times 10^{3}$ | $1.8657 \times 10^{3}$ |
| | Std | $2.1935 \times 10^{2}$ | $7.1912 \times 10^{3}$ | $4.3390 \times 10^{3}$ | $3.4847 \times 10^{1}$ |
| CEC2017-F19 | Ave | $1.9953 \times 10^{3}$ | $2.0495 \times 10^{3}$ | $2.0631 \times 10^{3}$ | $1.9130 \times 10^{3}$ |
| | Std | $5.4975 \times 10^{1}$ | $8.1484 \times 10^{1}$ | $6.6187 \times 10^{1}$ | $3.8464 \times 10^{00}$ |
| CEC2017-F20 | Ave | $2.2023 \times 10^{3}$ | $2.2135 \times 10^{3}$ | $2.1694 \times 10^{3}$ | $2.0936 \times 10^{3}$ |
| | Std | $8.0431 \times 10^{1}$ | $1.0684 \times 10^{2}$ | $6.7029 \times 10^{1}$ | $6.0960 \times 10^{1}$ |
| CEC2017-F21 | Ave | $2.3295 \times 10^{3}$ | $2.3346 \times 10^{3}$ | $2.3381 \times 10^{3}$ | $2.3185 \times 10^{3}$ |
| | Std | $5.6685 \times 10^{00}$ | $6.5256 \times 10^{00}$ | $8.2807 \times 10^{00}$ | $6.7052 \times 10^{00}$ |
| CEC2017-F22 | Ave | $2.4894 \times 10^{3}$ | $2.3012 \times 10^{3}$ | $2.3798 \times 10^{3}$ | $2.3000 \times 10^{3}$ |
| | Std | $7.0965 \times 10^{2}$ | $1.7589 \times 10^{00}$ | $4.3251 \times 10^{2}$ | $1.4626 \times 10^{-13}$ |
| CEC2017-F23 | Ave | $2.6797 \times 10^{3}$ | $2.6870 \times 10^{3}$ | $2.6885 \times 10^{3}$ | $2.6598 \times 10^{3}$ |
| | Std | $6.1832 \times 10^{00}$ | $8.7358 \times 10^{00}$ | $9.3977 \times 10^{00}$ | $9.6610 \times 10^{00}$ |
| CEC2017-F24 | Ave | $2.8487 \times 10^{3}$ | $2.8611 \times 10^{3}$ | $2.8574 \times 10^{3}$ | $2.8339 \times 10^{3}$ |
| | Std | $5.3234 \times 10^{00}$ | $1.0471 \times 10^{1}$ | $1.1075 \times 10^{1}$ | $5.6674 \times 10^{00}$ |
| CEC2017-F25 | Ave | $2.8886 \times 10^{3}$ | $2.8914 \times 10^{3}$ | $2.8893 \times 10^{3}$ | $2.8869 \times 10^{3}$ |
| | Std | $6.8082 \times 10^{00}$ | $9.1486 \times 10^{00}$ | $3.8007 \times 10^{00}$ | $8.2095 \times 10^{-2}$ |
| CEC2017-F26 | Ave | $3.8914 \times 10^{3}$ | $3.9787 \times 10^{3}$ | $3.9573 \times 10^{3}$ | $3.6409 \times 10^{3}$ |
| | Std | $1.2628 \times 10^{2}$ | $2.4512 \times 10^{2}$ | $2.5347 \times 10^{2}$ | $9.3325 \times 10^{1}$ |

**Table 9.** *Cont.*

| Algorithm | Items | LSHADE | LSHADE-SPACMA | LSHADE-cnEpSin | MISBOA |
|---|---|---|---|---|---|
| CEC2017-F27 | Ave | $3.2163 \times 10^{3}$ | $3.2222 \times 10^{3}$ | $3.2174 \times 10^{3}$ | $3.2107 \times 10^{3}$ |
| | Std | $9.9239 \times 10^{00}$ | $9.9340 \times 10^{00}$ | $1.0903 \times 10^{1}$ | $7.2401 \times 10^{00}$ |
| CEC2017-F28 | Ave | $3.1916 \times 10^{3}$ | $3.1553 \times 10^{3}$ | $3.2064 \times 10^{3}$ | $3.1290 \times 10^{3}$ |
| | Std | $5.4058 \times 10^{1}$ | $5.9772 \times 10^{1}$ | $3.6538 \times 10^{1}$ | $4.8940 \times 10^{1}$ |
| CEC2017-F29 | Ave | $3.4567 \times 10^{3}$ | $3.4442 \times 10^{3}$ | $3.4773 \times 10^{3}$ | $3.3715 \times 10^{3}$ |
| | Std | $6.0365 \times 10^{1}$ | $6.8641 \times 10^{1}$ | $6.8330 \times 10^{1}$ | $1.9535 \times 10^{1}$ |
| CEC2017-F30 | Ave | $5.8124 \times 10^{3}$ | $8.5509 \times 10^{3}$ | $7.9381 \times 10^{3}$ | $5.2900 \times 10^{3}$ |
| | Std | $6.6711 \times 10^{2}$ | $3.2951 \times 10^{3}$ | $2.6561 \times 10^{3}$ | $1.6663 \times 10^{2}$ |

In addition, the standard deviation results further verify the robustness and stability of MISBOA. On many benchmark functions, MISBOA achieves relatively smaller standard deviation values, indicating that the proposed algorithm maintains stable optimization performance and strong resistance to random disturbances over 30 independent runs. Especially on functions such as F1, F6, F10, F15, and F29, MISBOA not only achieves the best average performance but also exhibits highly concentrated result distributions, demonstrating excellent convergence reliability and consistency. Although MISBOA does not achieve the absolute best performance on a few functions, its results remain highly competitive and generally close to those of the compared algorithms. Overall, the experimental results in Table 9 demonstrate that MISBOA possesses strong competitiveness even when compared with advanced CEC competition-winning algorithms, thereby further validating the effectiveness of the proposed enhancement strategies in improving convergence accuracy, robustness, and optimization capability on complex optimization problems.

### *3.6. Strategy Effectiveness Analysis*

Strategy effectiveness analysis plays a crucial role in the evaluation of metaheuristic optimization algorithms because it can directly reveal the contribution of each improvement strategy to the overall optimization performance. Since many improved metaheuristic algorithms are constructed by integrating multiple enhancement mechanisms, merely reporting the final optimization results is often insufficient to fully demonstrate the effectiveness and rationality of the proposed strategies. Through strategy effectiveness analysis, it is possible to investigate whether each strategy can effectively improve specific optimization behaviors, such as population diversity maintenance, exploration capability, exploitation performance, convergence speed, and the ability to escape local optima. In addition, such analysis helps verify the complementarity and cooperative interaction among different strategies, thereby providing more convincing evidence for the effectiveness and necessity of the proposed improvements. Therefore, to comprehensively evaluate the contribution of each proposed strategy in MISBOA, this subsection conducts a detailed strategy effectiveness analysis experiment. Among them, SBOA1 indicates that the Chaotic Elite Initialization Strategy algorithm has been added to the SBOA, SBOA2 indicates that the Adaptive Spiral Lévy Flight Strategy algorithm has been added to the SBOA, and SBOA3 indicates that the Dynamic Neighborhood-Guided Mutation Strategy algorithm has been added to the SBOA.

Table 10 presents the strategy effectiveness analysis results of MISBOA on the CEC2014, CEC2017, and CEC2020 benchmark suites under different dimensional settings. The experimental results are evaluated using Mean Rank (M.R) and Total Rank (T.R), where lower values indicate better overall optimization performance. Overall, as more improvement strategies are incorporated into the original SBOA framework, the ranking performance of the algorithm consistently improves across all benchmark suites, demonstrating the effectiveness and complementarity of the proposed strategies.

**Table 10.** Strategy effectiveness analysis experimental results.

| **Suites** | **CEC2014** | | **CEC2017** | | | | **CEC2020** | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Dimensions** | **30** | | **30** | | **100** | | **10** | | **20** | |
| **Algorithms** | ***M.R*** | ***T.R*** | ***M.R*** | ***T.R*** | ***M.R*** | ***T.R*** | ***M.R*** | ***T.R*** | ***M.R*** | ***T.R*** |
| SBOA | 4.40 | 5 | 4.97 | 5 | 4.83 | 5 | 4.20 | 5 | 4.40 | 5 |
| SBOA1 | 3.43 | 4 | 3.50 | 4 | 3.40 | 4 | 3.80 | 4 | 3.90 | 4 |
| SBOA2 | 3.10 | 3 | 2.83 | 3 | 3.13 | 3 | 3.00 | 2 | 3.10 | 3 |
| SBOA3 | 2.57 | 2 | 2.67 | 2 | 2.30 | 2 | 3.00 | 2 | 2.60 | 2 |
| MISBOA | 1.50 | 1 | 1.03 | 1 | 1.33 | 1 | 1.00 | 1 | 1.00 | 1 |

Specifically, the original SBOA generally exhibits the worst ranking performance on most benchmark sets, indicating that the standard SBOA still suffers from limitations such as insufficient population diversity and weak balance between exploration and exploitation when solving complex optimization problems. After introducing the first improvement strategy, SBOA1 shows noticeable performance improvement on all test suites, suggesting that the proposed initialization enhancement mechanism can effectively improve the early-stage search capability and population quality. Subsequently, SBOA2 further improves the mean rank values, indicating that the adaptive spiral Lévy flight strategy effectively strengthens the transition between global exploration and local exploitation. Moreover, SBOA3 achieves additional ranking improvements compared with SBOA2, demonstrating that the dynamic neighborhood-guided mutation strategy can effectively alleviate premature convergence and enhance population diversity maintenance in the later search stage.

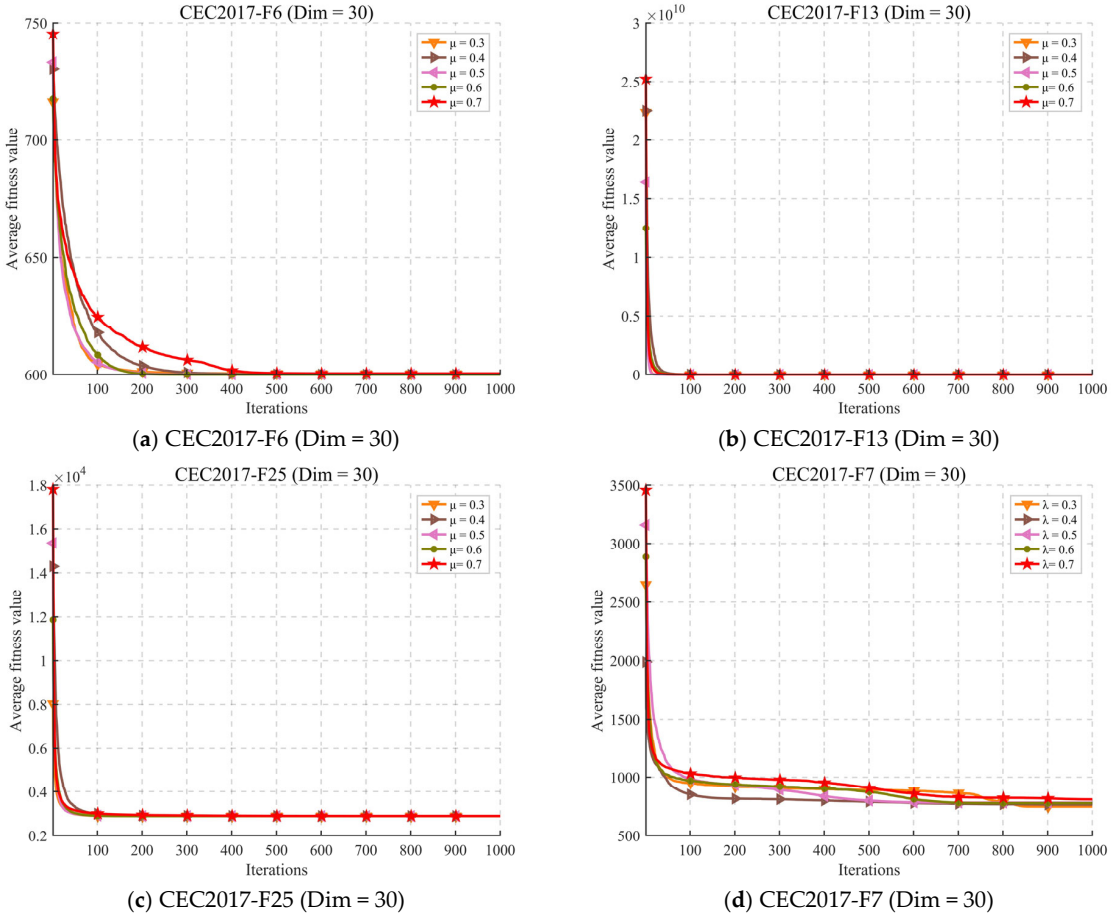
Finally, MISBOA consistently achieves the best Mean Rank and Total Rank values on all benchmark suites and dimensional settings, including CEC2014, CEC2017 (30D and 100D), and CEC2020 (10D and 20D). In particular, MISBOA obtains rank values close to 1 on almost all test sets, indicating that the complete integration of the three proposed strategies provides the strongest optimization capability and the most stable performance among all compared variants. Overall, the results in Table 10 clearly verify that each proposed strategy contributes positively to the optimization performance of MISBOA, while the cooperative interaction among the three strategies further enhances the convergence accuracy, robustness, and global optimization capability of the algorithm.

### *3.7. Parameter Sensitivity Analysis*

Parameter sensitivity analysis plays a crucial role in the performance evaluation of metaheuristic optimization algorithms, as the optimization behavior of these algorithms is often heavily influenced by the choice of control parameters. Different parameter settings significantly affect the balance between global exploration and local exploitation, convergence speed, population diversity, and solution accuracy. Therefore, conducting parameter sensitivity analysis is essential for assessing the stability, robustness, and adaptability of the proposed algorithm under different parameter configurations. Furthermore, parameter sensitivity analysis helps reveal the impact of key parameters on the algorithm's search dynamics and provides guidance for selecting appropriate parameter values in practical

optimization problems. Moreover, it can further verify whether the proposed algorithm can maintain stable optimization performance under different parameter conditions, thereby enhancing the reliability and persuasiveness of the experimental results. Therefore, to comprehensively evaluate the robustness and parameter adaptability of MISBOA, this section will conduct detailed parameter sensitivity analysis experiments on several representative hyperparameters.

Figure 13 shows the parameter sensitivity analysis results of the MISBOA for several representative hyperparameters. Overall, MISBOA maintains relatively stable convergence behavior under different parameter configurations, indicating strong robustness and parameter adaptability. Specifically, different values of the control parameters have little impact on the convergence speed and final optimization accuracy of some benchmark functions; however, the overall convergence trend remains highly consistent. Among the tested parameter settings, moderate parameter values generally achieve better convergence performance and stability, reflecting a more balanced trade-off between global exploration and local exploitation. Furthermore, MISBOA maintains stable optimization capabilities under different parameter combinations, further validating the reliability and robustness of the proposed improvement strategy.




(**a**) CEC2017-F6 (Dim = 30)

(**b**) CEC2017-F13 (Dim = 30)

(**c**) CEC2017-F25 (Dim = 30)

(**d**) CEC2017-F7 (Dim = 30)

**Figure 13.** *Cont.*

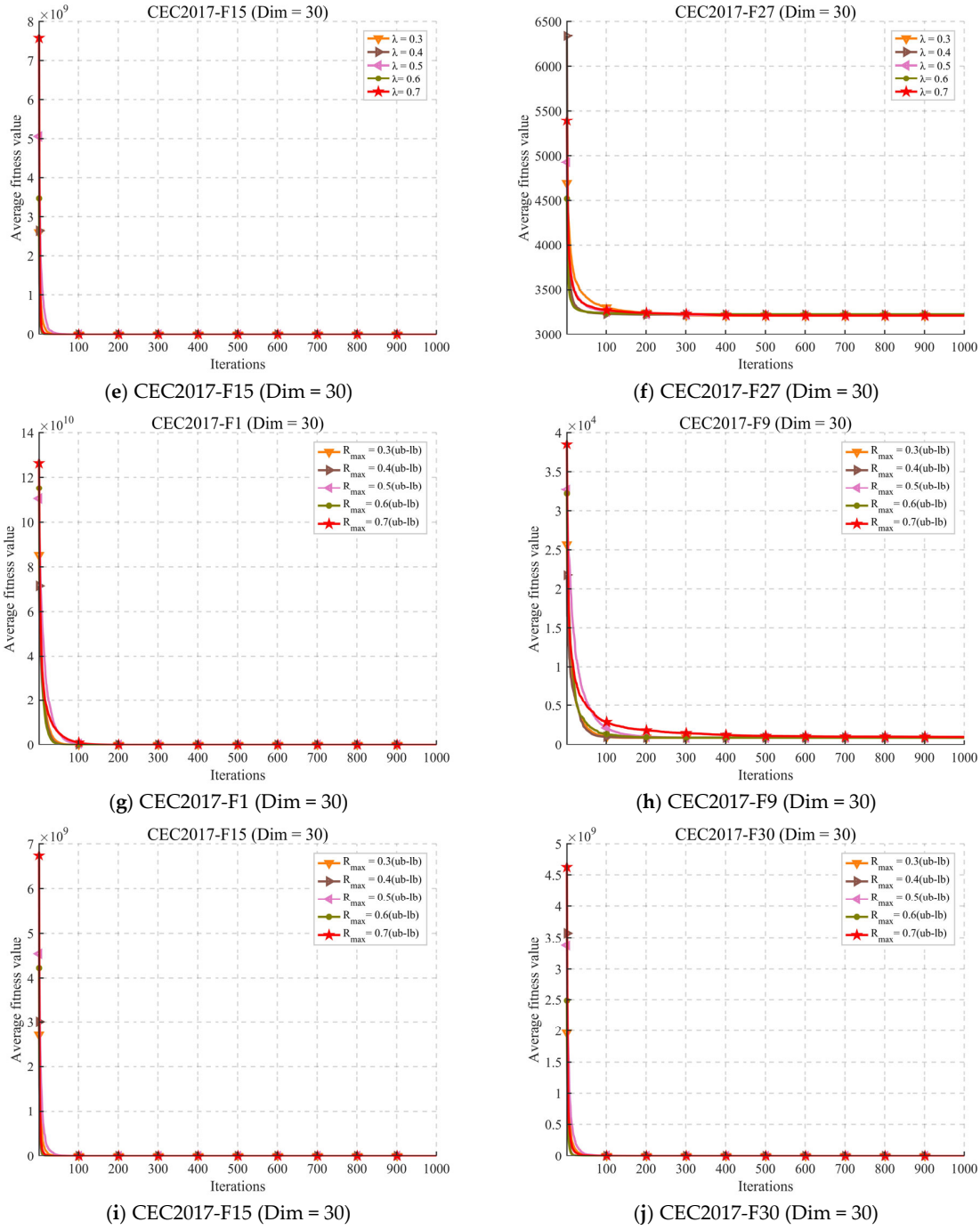



**Figure 13.** Parameter sensitivity analysis experimental results.

## 4. Multilevel Thresholding-Based Image Segmentation Model

In the context of multi-threshold image segmentation, the selection of thresholds directly impacts the effectiveness of distinguishing target regions from background regions. Common threshold selection methods primarily include Otsu's method and Kapur's entropy method. Specifically, Otsu's method identifies the optimal threshold by maximizing the inter-class variance, whereas Kapur's entropy method achieves image segmentation by maximizing the information entropy of each region. Given its advantages—including computational simplicity, robust stability, and sensitivity to grayscale distribution—this paper adopts Otsu's method as the objective function for multi-threshold image segmentation [22,23].

Assuming that grayscale image $I$ comprises a total of $L$ grayscale levels, and the number of pixels corresponding to the $i$-th grayscale level is $n_i$, then the total number of pixels in the image can be expressed as:

$$N = \sum_{i=0}^{L-1} n_i, \tag{18}$$

Accordingly, the probability of occurrence of the $i$-th grayscale level is:

$$P_i = \frac{n_i}{N}, i = 0, 1, \ldots, L-1 \tag{19}$$

where $P_i \geq 0$, and satisfying $\sum_{i=0}^{L-1} P_i = 1$.

For single-threshold segmentation, when the gray value $t$ is selected as the threshold, the image is divided into two parts: the target region and the background region. The gray value range $[0,\ t]$ corresponds to the target region, and the gray value range $[t+1,\ L-1]$ corresponds to the background region. Let the probabilities of the two regions be $\omega_0$ and $\omega_1$, and their average gray values be $\mu_0$ and $\mu_1$, respectively. The average gray value of the entire image is $\mu$. Then:

$$\omega_0 = \sum_{i=0}^{t} P_i \tag{20}$$

$$\mu_0 = \frac{\sum_{i=0}^{t} iP_i}{\omega_0} \tag{21}$$

$$\omega_1 = \sum_{i=t+1}^{L-1} P_i \tag{22}$$

$$\mu_1 = \frac{\sum_{i=t+1}^{L-1} iP_i}{\omega_1} \tag{23}$$

$$\mu = \sum_{i=0}^{L-1} iP_i \tag{24}$$

Based on this, the inter-class variance of single-threshold segmentation can be written as:

$$\nu(t) = \omega_0(\mu_0 - \mu)^2 + \omega_1(\mu_1 - \mu)^2 \tag{25}$$

For multi-threshold segmentation, if $k$ thresholds $t_1, t_2, \ldots, t_k$ are set, the image will be divided into $k+1$ gray-level regions. Correspondingly, the inter-class variance in the multi-threshold case can be further extended as follows:

$$\nu(t_1, t_2, \ldots, t_k) = \sum_{j=0}^{k} \omega_j(\mu_j - \mu)^2 \tag{26}$$

Wherein, the probability and average gray level of the $j$-th region are defined as follows:

$$\omega_j = \sum_{i=t_j+1}^{t_{j+1}} P_i \tag{27}$$

$$\mu_j = \frac{\sum_{i=t_j+1}^{t_{j+1}} iP_i}{\omega_j} \tag{28}$$

Therefore, the multi-threshold image segmentation problem can be transformed into an optimization problem, namely, finding a set of optimal thresholds that maximizes the inter-class variance:

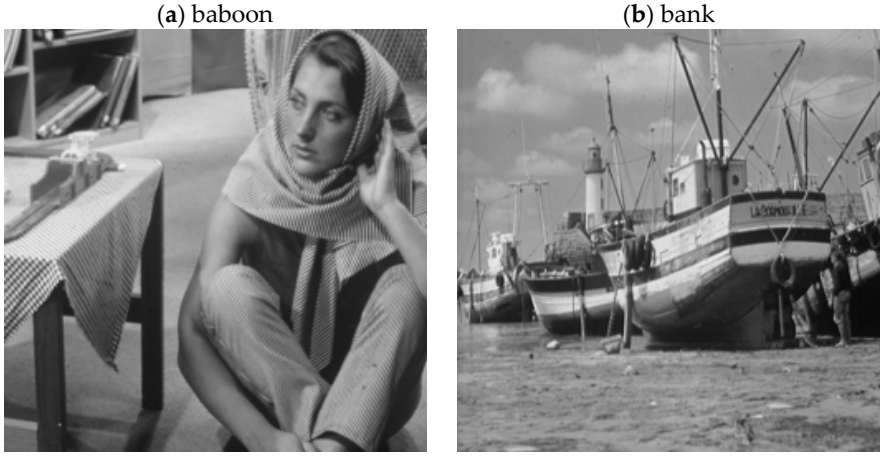
$$T_{best} = \text{argmax}\nu(t_1, t_2, \ldots, t_k) \tag{29}$$

where $T_{best}$ represents the optimal combination of thresholds.

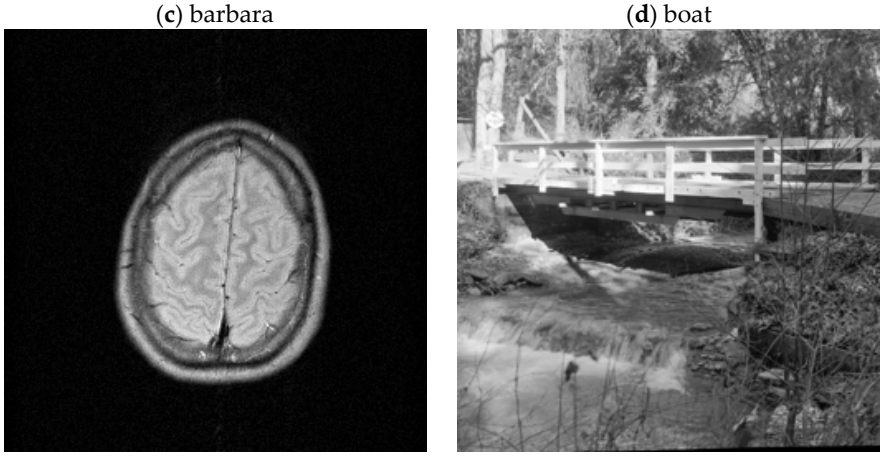
To verify the effectiveness of the proposed algorithm in multi-threshold image segmentation, this paper selects 10 typical test images for experimental analysis, as shown in Figure 14, and conducts comparative tests using different threshold numbers. All algorithms are run independently 30 times under the same parameter conditions to ensure the fairness and reliability of the experimental results. Finally, the segmentation accuracy and stability of the algorithm are comprehensively evaluated using the mean and standard deviation of the objective function values.



(**a**) baboon (**b**) bank

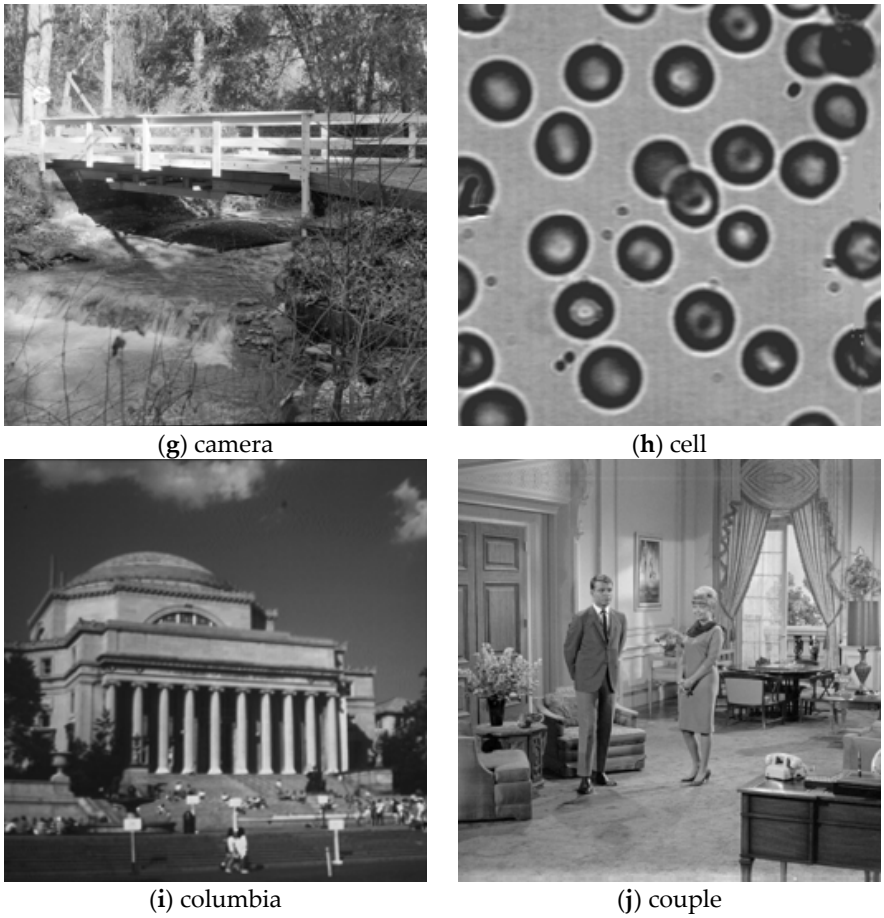


(**c**) barbara (**d**) boat



(**e**) brain (**f**) bridge

**Figure 14.** *Cont.*



(**g**) camera (**h**) cell

(**i**) columbia (**j**) couple

**Figure 14.** Benchmark Test Images Used for Segmentation Evaluation.

### *4.1. Evaluation Metrics*

To comprehensively evaluate the quality of image segmentation results, this paper uses Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and Feature Similarity Index (FSIM) as evaluation metrics.

PSNR measures the degree of distortion between the original image and the segmented image; a higher value indicates higher image fidelity. The definition of PSNR is as follows:

$$PSNR = 10log_{10}(\frac{255^2}{MSE}), \tag{30}$$

The mean squared error (MSE) can be expressed as:

$$MSE = \frac{1}{MN}\sum_{j=1}^{M}\sum_{k=1}^{N}\left[I(j,k) - I'(j,k)\right]^2, \tag{31}$$

where $M \times N$ represents the image size, and $I$ and $I'$ represent the original image and the segmented image, respectively.

SSIM measures the similarity between two images in terms of brightness, contrast, and structural information. A value closer to 1 indicates a closer resemblance between the segmented image and the original image. The formula for calculating SSIM is:

$$SSIM(I, I') = \frac{(2\mu_I\mu_{I'} + C_1)(2\sigma_{II'} + C_2)}{(\mu_I^2 + \mu_{I'}^2 + C_1)(\sigma_I^2 + \sigma_{I'}^2 + C_2)}, \tag{32}$$

where $\mu_I$ and $\mu_{I'}$ represent the average gray values of the two images, respectively, $\sigma_I^2$ and $\sigma_{I'}^2$ represent variance, $\sigma_{II'}$ represents the covariance.

Unlike PSNR and SSIM, FSIM focuses more on edge, texture, and local feature information in an image, thus more effectively reflecting the preservation of image details. The expression for FSIM is:

$$FSIM(I, I') = \frac{\sum_{x \in \Omega} S_L(x) \cdot PC_m(x)}{\sum_{x \in \Omega} PC_m(x)}, \tag{33}$$

where $\Omega$ represents the entire image region, $S_L(x)$ represents the local feature similarity at pixel $x$, and $PC_m(x)$ represents the phase consistency feature.

Overall, higher PSNR, SSIM, and FSIM values mean that the segmented image can better preserve the grayscale information, structural features, and details of the original image. Therefore, these metrics can comprehensively evaluate the performance of multi-threshold image segmentation algorithms.

### *4.2. MISBOA Analysis for Multilevel Threshold Segmentation Based on Otsu Criterion*

This study applies the MISBOA to multilevel image thresholding and adopts the Otsu criterion as the objective function to search for the optimal threshold values for ten benchmark images. The segmentation performance of the proposed method is assessed by maximizing the Otsu objective function and further evaluated using three widely used image quality indicators, namely Peak Signal-to-Noise Ratio, Feature Similarity Index, and Structural Similarity Index. Larger values of these metrics indicate that the segmented image preserves more structural information, texture details, and visual quality from the original image. The experimental results are shown below.

Figure 15 shows the experimental results of MISBOA in the Otsu-based multi-threshold image segmentation task. As can be seen from the figure, with the increase in the number of thresholds, the target region, edge contours, and texture details in the image are gradually separated more clearly, indicating that MISBOA can effectively search for better threshold combinations, thereby improving segmentation quality. At lower threshold numbers, the segmentation results can already distinguish the foreground and background well and retain the main structural information of the image; as the number of thresholds further increases, the grayscale levels, local textures, and edge transitions in the image become richer and more natural, making the overall segmentation results more readable and visually continuous. For some test images with complex textures, uneven grayscale distribution, or blurred boundaries, MISBOA can still maintain the integrity of the target region well, reduce detail loss and over segmentation, demonstrating strong global search and local optimization capabilities. Overall, Figure 15 illustrates that MISBOA can obtain clear, stable, and visually high-quality segmentation results in the multi-threshold image segmentation task, verifying its effectiveness and practical value in complex image processing problems.

Table 11 presents the mean fitness (Ave) and standard deviation (Std) of different algorithms in the multi-threshold image segmentation task based on the Otsu objective function. Overall, the fitness values of each algorithm show an increasing trend with the increase in the number of thresholds, indicating that more thresholds can more fully exploit the gray-level distribution information in the image, thus achieving better segmentation results. However, compared with other comparative algorithms, MISBOA achieved a higher mean fitness value and a smaller standard deviation in most test images and under different threshold settings, indicating that it not only searches for better threshold combinations but also has better stability and consistency in multiple independent runs. Especially on images with complex textures, uneven gray-level distribution, and blurred edges, MISBOA's advantage is more pronounced, with its Ave value significantly better than other algorithms, while its Std value remains at a low level. This shows that the proposed improvement strategy can effectively enhance population diversity, improve global search ability, and reduce the probability of the algorithm getting trapped in local

optima. Overall, the experimental results in Table 11 fully demonstrate that MISBOA has higher optimization accuracy, stronger robustness, and better generalization ability in multi-threshold image segmentation tasks based on the Otsu objective function.

Table 12 presents the mean and standard deviation of PSNR for different algorithms on various test images in the Otsu multi-threshold segmentation task. Overall, the PSNR values of all algorithms generally improve with the increase in the number of thresholds, indicating that more thresholds can more fully preserve the grayscale information and detailed features in the original image, thereby reducing segmentation distortion. Compared to other comparison algorithms, MISBOA achieved higher PSNR values on most test images and under different threshold settings, with a smaller corresponding standard deviation, indicating that it can consistently obtain higher-quality segmentation results in multiple independent runs. Especially on images with complex textures, rich grayscale variations, and blurred edges, MISBOA's advantages are more pronounced. Its higher PSNR value indicates that the segmented image better preserves the information of the original image, with lower pixel errors and less distortion. Meanwhile, the smaller Std value further demonstrates that MISBOA has good stability and robustness, and is not easily affected by random initialization and local optima. Overall, the results in Table 12 show that MISBOA achieves higher image reconstruction quality and better visual fidelity in Otsu-based multi-threshold image segmentation tasks.

Table 13 presents the mean and standard deviation of the FSIM scores for different algorithms on various test images in the Otsu multi-threshold image segmentation task. Overall, the FSIM scores of all algorithms generally improve with increasing threshold number, indicating that more thresholds better preserve edge, texture, and structural features in the image. Compared to other comparison algorithms, MISBOA achieved higher FSIM scores on most test images and under different threshold settings, with a smaller corresponding standard deviation. This suggests that it not only more effectively preserves important image features but also exhibits better stability and consistency in multiple independent runs. MISBOA's advantage is particularly evident in images with rich textures, complex boundaries, and numerous details. Its higher FSIM score indicates that the segmentation results better preserve the edge contours, local textures, and visually salient features of the target region, making the segmented image more consistent with human visual perception. Meanwhile, the smaller standard deviation (Std) score further demonstrates that MISBOA performs more stably on complex images and is less susceptible to random factors. Overall, the experimental results in Table 13 show that MISBOA has stronger feature preservation capabilities and higher visual quality in Otsu-based multi-threshold image segmentation tasks.

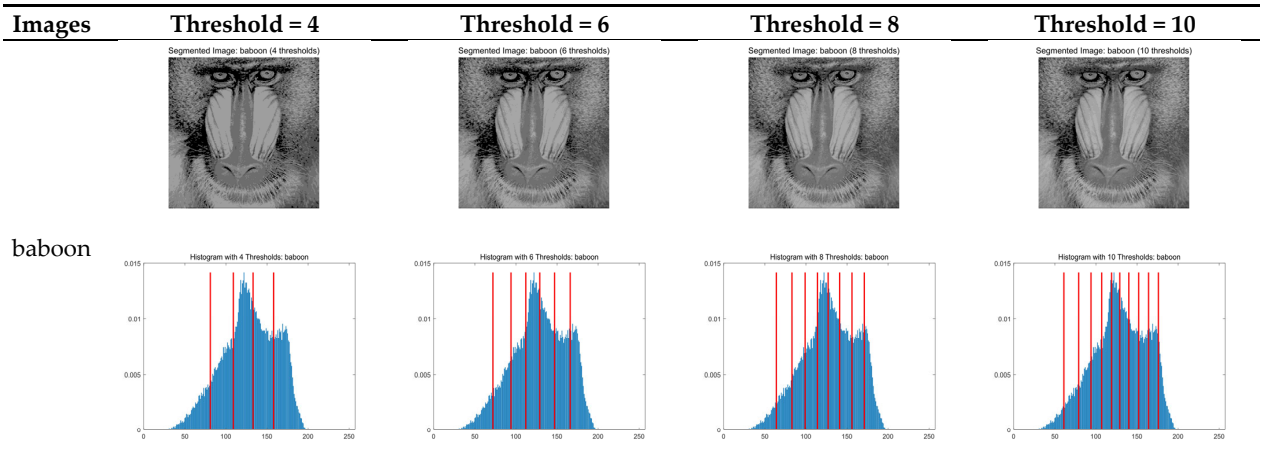



**Figure 15.** *Cont.*

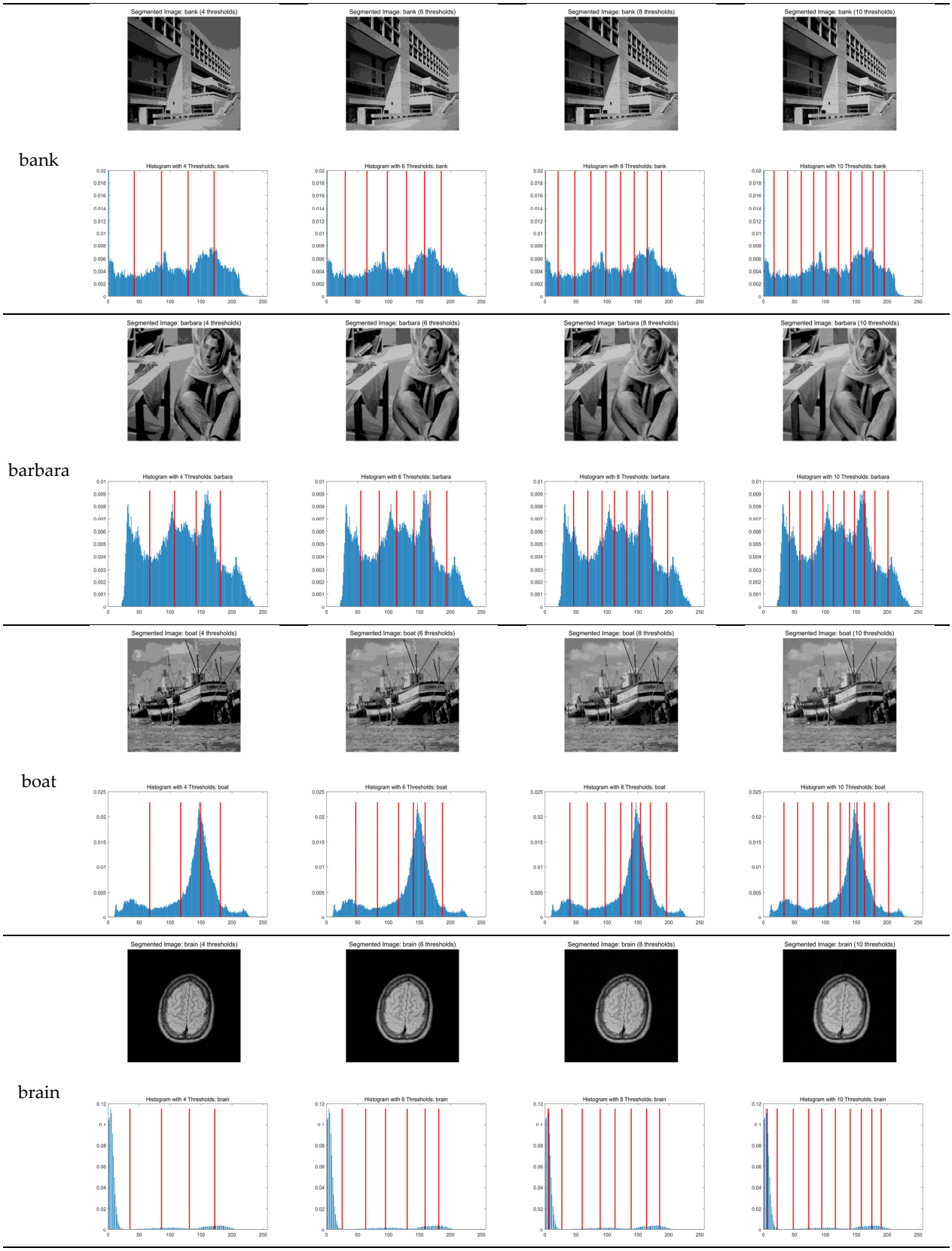



**Figure 15.** *Cont*.

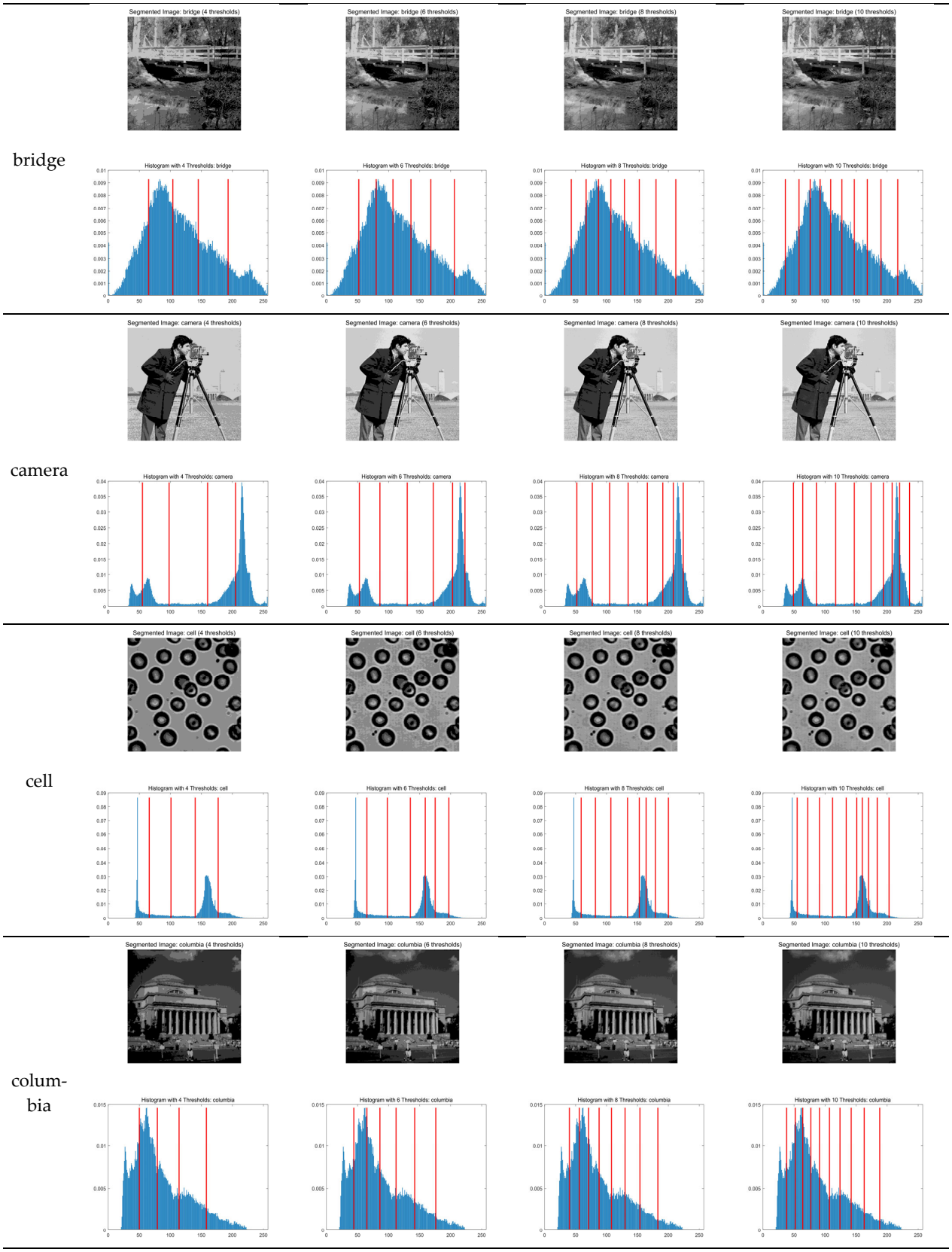



**Figure 15.** *Cont.*

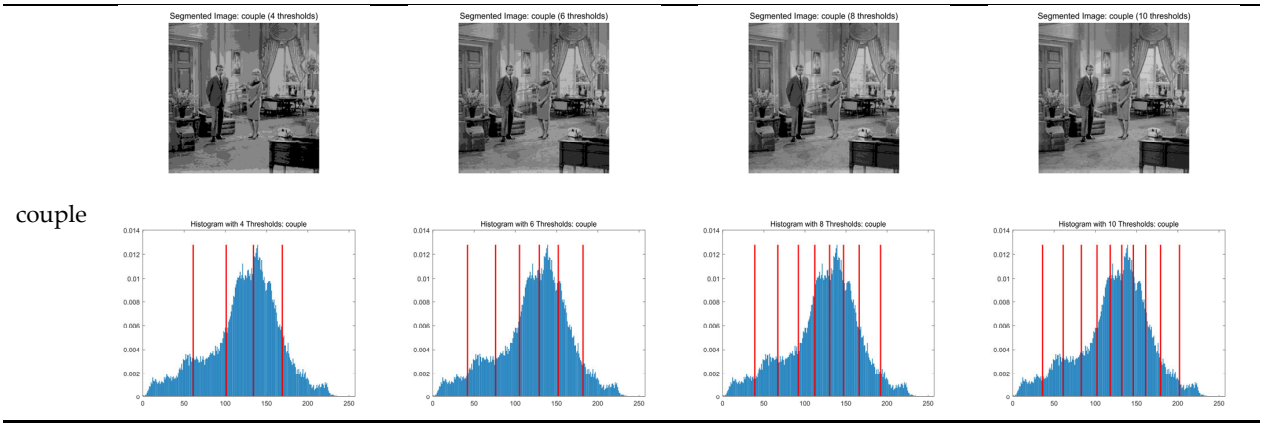



**Figure 15.** Performance of MISBOA in Otsu-Based Multilevel Threshold Segmentation.

Table 14 presents the mean and standard deviation of SSIM values for different algorithms on various test images in the Otsu multi-threshold image segmentation task. Overall, the SSIM values of all algorithms show an upward trend with increasing threshold number, indicating that a higher threshold number helps to more accurately preserve structural information and regional hierarchy in the image. Compared to other comparison algorithms, MISBOA achieved higher SSIM values on most test images and under different threshold settings, with a smaller standard deviation. This indicates that it not only better preserves the brightness, contrast, and structural features of the original image but also exhibits high stability and consistency in multiple independent runs. Especially in images with complex textures, blurred edges, and uneven grayscale distributions, MISBOA still maintains a high SSIM value, demonstrating that its segmentation results can more completely preserve the structural information of the target region, making the segmented image visually more natural and continuous. Meanwhile, the lower standard deviation further illustrates that MISBOA has good robustness in complex multi-threshold search spaces and is not easily affected by random initialization and local optima. Overall, the experimental results in Table 14 show that MISBOA has stronger structure preservation ability and better visual consistency in Otsu-based multi-threshold image segmentation tasks.

Table 15 presents the runtime comparison results of different algorithms on multiple benchmark images for multilevel threshold image segmentation tasks. Overall, MISBOA requires slightly more computational time than the original SBOA on most test images due to the introduction of multiple enhancement strategies, particularly the adaptive spiral Lévy flight and dynamic neighborhood-guided mutation mechanisms, which increase the computational complexity of the optimization process. However, the increase in runtime remains relatively limited and acceptable. For example, on most benchmark images, the runtime difference between MISBOA and SBOA is only a few seconds, while MISBOA achieves significantly better segmentation quality and optimization performance in terms of objective fitness values and image quality metrics. In addition, compared with several other algorithms such as ESC, GJA, and SSLO, MISBOA still maintains relatively competitive computational efficiency. Particularly, ESC exhibits substantially higher runtime on almost all benchmark images, whereas MISBOA achieves a better balance between segmentation accuracy and computational cost. Therefore, although MISBOA introduces a moderate increase in runtime compared with SBOA, this additional computational overhead is acceptable for multilevel threshold image segmentation problems considering the significant improvements in convergence accuracy, robustness, and segmentation quality achieved by the proposed algorithm.

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

| Image | Threshold | Metric | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| baboon | 4 | Ave | $1.008 \times 10^{3}$ | $1.008 \times 10^{3}$ | $1.008 \times 10^{3}$ | $1.008 \times 10^{3}$ | $1.008 \times 10^{3}$ | $1.008 \times 10^{3}$ | $1.008 \times 10^{3}$ | $1.008 \times 10^{3}$ | $1.008 \times 10^{3}$ | $1.008 \times 10^{3}$ |
| | | Std | $7.357 \times 10^{-3}$ | $9.781 \times 10^{-2}$ | $2.313 \times 10^{-13}$ | $8.290 \times 10^{-2}$ | $2.313 \times 10^{-13}$ | $2.313 \times 10^{-13}$ | $2.313 \times 10^{-13}$ | $1.873 \times 10^{-2}$ | $4.823 \times 10^{-3}$ | $4.823 \times 10^{-3}$ |
| | 6 | Ave | $1.036 \times 10^{3}$ | $1.002 \times 10^{3}$ | $1.036 \times 10^{3}$ | $1.036 \times 10^{3}$ | $1.036 \times 10^{3}$ | $1.036 \times 10^{3}$ | $1.036 \times 10^{3}$ | $1.036 \times 10^{3}$ | $1.036 \times 10^{3}$ | $1.036 \times 10^{3}$ |
| | | Std | $3.299 \times 10^{-2}$ | $1.892 \times 10^{2}$ | $1.600 \times 10^{-2}$ | $2.586 \times 10^{-1}$ | $4.625 \times 10^{-13}$ | $2.194 \times 10^{-2}$ | $1.836 \times 10^{-2}$ | $2.540 \times 10^{-1}$ | $3.795 \times 10^{-3}$ | $5.532 \times 10^{-4}$ |
| | 8 | Ave | $1.049 \times 10^{3}$ | $7.690 \times 10^{2}$ | $1.049 \times 10^{3}$ | $8.380 \times 10^{2}$ | $1.049 \times 10^{3}$ | $1.049 \times 10^{3}$ | $1.014 \times 10^{3}$ | $1.048 \times 10^{3}$ | $1.049 \times 10^{3}$ | $1.049 \times 10^{3}$ |
| | | Std | $1.739 \times 10^{-1}$ | $4.717 \times 10^{2}$ | $5.449 \times 10^{-2}$ | $4.262 \times 10^{2}$ | $3.416 \times 10^{-3}$ | $5.035 \times 10^{-2}$ | $1.915 \times 10^{2}$ | $4.951 \times 10^{-1}$ | $4.070 \times 10^{-2}$ | $8.525 \times 10^{-2}$ |
| | 10 | Ave | $1.055 \times 10^{3}$ | $4.221 \times 10^{2}$ | $1.055 \times 10^{3}$ | $5.965 \times 10^{2}$ | $1.056 \times 10^{3}$ | $1.020 \times 10^{3}$ | $8.444 \times 10^{2}$ | $1.054 \times 10^{3}$ | $9.851 \times 10^{2}$ | $1.056 \times 10^{3}$ |
| | | Std | $2.146 \times 10^{-1}$ | $5.258 \times 10^{2}$ | $1.358 \times 10^{-1}$ | $5.305 \times 10^{2}$ | $2.799 \times 10^{-2}$ | $1.927 \times 10^{2}$ | $4.294 \times 10^{2}$ | $4.437 \times 10^{-1}$ | $2.678 \times 10^{2}$ | $6.553 \times 10^{-2}$ |
| bank | 4 | Ave | $3.602 \times 10^{3}$ | $3.602 \times 10^{3}$ | $3.602 \times 10^{3}$ | $3.602 \times 10^{3}$ | $3.602 \times 10^{3}$ | $3.602 \times 10^{3}$ | $3.602 \times 10^{3}$ | $3.602 \times 10^{3}$ | $3.602 \times 10^{3}$ | $3.602 \times 10^{3}$ |
| | | Std | $2.776 \times 10^{-2}$ | $3.824 \times 10^{-2}$ | $2.957 \times 10^{-3}$ | $1.088 \times 10^{-1}$ | $1.850 \times 10^{-12}$ | $2.183 \times 10^{-1}$ | $3.139 \times 10^{-3}$ | $2.454 \times 10^{-1}$ | $5.688 \times 10^{-3}$ | $2.957 \times 10^{-3}$ |
| | 6 | Ave | $3.681 \times 10^{3}$ | $3.681 \times 10^{3}$ | $3.681 \times 10^{3}$ | $3.680 \times 10^{3}$ | $3.681 \times 10^{3}$ | $3.681 \times 10^{3}$ | $3.681 \times 10^{3}$ | $3.680 \times 10^{3}$ | $3.681 \times 10^{3}$ | $3.681 \times 10^{3}$ |
| | | Std | $6.435 \times 10^{-2}$ | $7.107 \times 10^{-1}$ | $2.099 \times 10^{-2}$ | $8.351 \times 10^{-1}$ | $1.388 \times 10^{-12}$ | $1.548 \times 10^{-2}$ | $3.033 \times 10^{-2}$ | $9.351 \times 10^{-1}$ | $2.164 \times 10^{-2}$ | $1.388 \times 10^{-12}$ |
| | 8 | Ave | $3.711 \times 10^{3}$ | $3.711 \times 10^{3}$ | $3.712 \times 10^{3}$ | $3.710 \times 10^{3}$ | $3.712 \times 10^{3}$ | $3.711 \times 10^{3}$ | $3.711 \times 10^{3}$ | $3.709 \times 10^{3}$ | $3.711 \times 10^{3}$ | $3.712 \times 10^{3}$ |
| | | Std | $2.104 \times 10^{-1}$ | $1.322 \times 10^{00}$ | $5.751 \times 10^{-2}$ | $8.201 \times 10^{-1}$ | $8.268 \times 10^{-3}$ | $1.745 \times 10^{00}$ | $3.917 \times 10^{-1}$ | $1.204 \times 10^{00}$ | $4.103 \times 10^{-1}$ | $2.077 \times 10^{-1}$ |
| | 10 | Ave | $3.727 \times 10^{3}$ | $3.727 \times 10^{3}$ | $3.728 \times 10^{3}$ | $3.725 \times 10^{3}$ | $3.728 \times 10^{3}$ | $3.727 \times 10^{3}$ | $3.727 \times 10^{3}$ | $3.724 \times 10^{3}$ | $3.727 \times 10^{3}$ | $3.728 \times 10^{3}$ |
| | | Std | $5.755 \times 10^{-1}$ | $1.304 \times 10^{00}$ | $1.962 \times 10^{-1}$ | $1.129 \times 10^{00}$ | $3.469 \times 10^{-3}$ | $1.279 \times 10^{00}$ | $8.652 \times 10^{-1}$ | $1.088 \times 10^{00}$ | $9.422 \times 10^{-1}$ | $3.982 \times 10^{-1}$ |
| barbara | 4 | Ave | $2.645 \times 10^{3}$ | $2.645 \times 10^{3}$ | $2.645 \times 10^{3}$ | $2.645 \times 10^{3}$ | $2.645 \times 10^{3}$ | $2.645 \times 10^{3}$ | $2.645 \times 10^{3}$ | $2.645 \times 10^{3}$ | $2.645 \times 10^{3}$ | $2.645 \times 10^{3}$ |
| | | Std | $1.850 \times 10^{-12}$ | $1.464 \times 10^{-2}$ | $1.850 \times 10^{-12}$ | $7.256 \times 10^{-2}$ | $1.850 \times 10^{-12}$ | $1.850 \times 10^{-12}$ | $1.850 \times 10^{-12}$ | $4.736 \times 10^{-2}$ | $1.850 \times 10^{-12}$ | $1.850 \times 10^{-12}$ |
| | 6 | Ave | $2.701 \times 10^{3}$ | $2.701 \times 10^{3}$ | $2.702 \times 10^{3}$ | $2.701 \times 10^{3}$ | $2.702 \times 10^{3}$ | $2.702 \times 10^{3}$ | $2.702 \times 10^{3}$ | $2.701 \times 10^{3}$ | $2.702 \times 10^{3}$ | $2.702 \times 10^{3}$ |
| | | Std | $4.806 \times 10^{-2}$ | $4.179 \times 10^{-1}$ | $1.692 \times 10^{-2}$ | $5.258 \times 10^{-1}$ | $9.250 \times 10^{-13}$ | $1.167 \times 10^{-2}$ | $4.875 \times 10^{-2}$ | $6.425 \times 10^{-1}$ | $1.017 \times 10^{-2}$ | $9.250 \times 10^{-13}$ |
| | 8 | Ave | $2.727 \times 10^{3}$ | $2.727 \times 10^{3}$ | $2.727 \times 10^{3}$ | $2.726 \times 10^{3}$ | $2.727 \times 10^{3}$ | $2.727 \times 10^{3}$ | $2.727 \times 10^{3}$ | $2.725 \times 10^{3}$ | $2.727 \times 10^{3}$ | $2.727 \times 10^{3}$ |
| | | Std | $9.586 \times 10^{-2}$ | $3.919 \times 10^{-1}$ | $3.799 \times 10^{-2}$ | $5.743 \times 10^{-1}$ | $4.625 \times 10^{-13}$ | $2.125 \times 10^{-2}$ | $3.254 \times 10^{-1}$ | $7.797 \times 10^{-1}$ | $5.484 \times 10^{-2}$ | $2.079 \times 10^{-2}$ |
| | 10 | Ave | $2.739 \times 10^{3}$ | $2.739 \times 10^{3}$ | $2.740 \times 10^{3}$ | $2.738 \times 10^{3}$ | $2.740 \times 10^{3}$ | $2.740 \times 10^{3}$ | $2.739 \times 10^{3}$ | $2.737 \times 10^{3}$ | $2.740 \times 10^{3}$ | $2.740 \times 10^{3}$ |
| | | Std | $3.057 \times 10^{-1}$ | $1.140 \times 10^{00}$ | $2.730 \times 10^{-1}$ | $6.164 \times 10^{-1}$ | $1.327 \times 10^{-2}$ | $2.295 \times 10^{-1}$ | $4.912 \times 10^{-1}$ | $8.514 \times 10^{-1}$ | $4.416 \times 10^{-1}$ | $9.999 \times 10^{-2}$ |
| boat | 4 | Ave | $1.970 \times 10^{3}$ | $1.970 \times 10^{3}$ | $1.970 \times 10^{3}$ | $1.970 \times 10^{3}$ | $1.970 \times 10^{3}$ | $1.970 \times 10^{3}$ | $1.970 \times 10^{3}$ | $1.970 \times 10^{3}$ | $1.970 \times 10^{3}$ | $1.970 \times 10^{3}$ |
| | | Std | $2.376 \times 10^{-3}$ | $2.694 \times 10^{-2}$ | $0.000 \times 10^{00}$ | $1.342 \times 10^{-1}$ | $0.000 \times 10^{00}$ | $0.000 \times 10^{00}$ | $0.000 \times 10^{00}$ | $4.724 \times 10^{-2}$ | $0.000 \times 10^{00}$ | $0.000 \times 10^{00}$ |
| | 6 | Ave | $2.025 \times 10^{3}$ | $2.025 \times 10^{3}$ | $2.025 \times 10^{3}$ | $2.024 \times 10^{3}$ | $2.025 \times 10^{3}$ | $2.025 \times 10^{3}$ | $2.025 \times 10^{3}$ | $2.025 \times 10^{3}$ | $2.025 \times 10^{3}$ | $2.025 \times 10^{3}$ |
| | | Std | $8.649 \times 10^{-2}$ | $9.811 \times 10^{-2}$ | $2.050 \times 10^{-2}$ | $5.452 \times 10^{-1}$ | $1.156 \times 10^{-12}$ | $1.837 \times 10^{-2}$ | $2.599 \times 10^{-2}$ | $3.903 \times 10^{-1}$ | $2.405 \times 10^{-2}$ | $7.595 \times 10^{-3}$ |
| | 8 | Ave | $2.049 \times 10^{3}$ | $2.048 \times 10^{3}$ | $2.049 \times 10^{3}$ | $2.047 \times 10^{3}$ | $2.049 \times 10^{3}$ | $2.049 \times 10^{3}$ | $2.049 \times 10^{3}$ | $2.047 \times 10^{3}$ | $2.049 \times 10^{3}$ | $2.049 \times 10^{3}$ |
| | | Std | $1.587 \times 10^{-1}$ | $7.285 \times 10^{-1}$ | $4.102 \times 10^{-2}$ | $6.925 \times 10^{-1}$ | $1.307 \times 10^{-3}$ | $2.958 \times 10^{-2}$ | $2.787 \times 10^{-1}$ | $9.528 \times 10^{-1}$ | $2.484 \times 10^{-1}$ | $5.789 \times 10^{-2}$ |
| | 10 | Ave | $2.060 \times 10^{3}$ | $2.060 \times 10^{3}$ | $2.061 \times 10^{3}$ | $2.059 \times 10^{3}$ | $2.061 \times 10^{3}$ | $2.061 \times 10^{3}$ | $2.061 \times 10^{3}$ | $2.059 \times 10^{3}$ | $2.061 \times 10^{3}$ | $2.061 \times 10^{3}$ |
| | | Std | $3.238 \times 10^{-1}$ | $6.520 \times 10^{-1}$ | $1.551 \times 10^{-1}$ | $5.862 \times 10^{-1}$ | $1.344 \times 10^{-2}$ | $1.674 \times 10^{-1}$ | $2.361 \times 10^{-1}$ | $7.034 \times 10^{-1}$ | $2.476 \times 10^{-1}$ | $1.490 \times 10^{-1}$ |
| brain | 4 | Ave | $3.731 \times 10^{3}$ | $3.731 \times 10^{3}$ | $3.731 \times 10^{3}$ | $3.731 \times 10^{3}$ | $3.731 \times 10^{3}$ | $3.731 \times 10^{3}$ | $3.731 \times 10^{3}$ | $3.731 \times 10^{3}$ | $3.731 \times 10^{3}$ | $3.731 \times 10^{3}$ |
| | | Std | $2.313 \times 10^{-12}$ | $5.278 \times 10^{-2}$ | $2.313 \times 10^{-12}$ | $3.779 \times 10^{-2}$ | $2.313 \times 10^{-12}$ | $2.313 \times 10^{-12}$ | $2.313 \times 10^{-12}$ | $4.472 \times 10^{-2}$ | $2.313 \times 10^{-12}$ | $2.313 \times 10^{-12}$ |
| | 6 | Ave | $3.752 \times 10^{3}$ | $3.751 \times 10^{3}$ | $3.751 \times 10^{3}$ | $3.751 \times 10^{3}$ | $3.751 \times 10^{3}$ | $3.751 \times 10^{3}$ | $3.751 \times 10^{3}$ | $3.751 \times 10^{3}$ | $3.751 \times 10^{3}$ | $3.751 \times 10^{3}$ |
| | | Std | $8.481 \times 10^{-2}$ | $6.276 \times 10^{-1}$ | $4.957 \times 10^{-1}$ | $2.890 \times 10^{-1}$ | $4.166 \times 10^{-1}$ | $5.986 \times 10^{-1}$ | $3.631 \times 10^{-1}$ | $5.265 \times 10^{-1}$ | $6.086 \times 10^{-1}$ | $5.403 \times 10^{-1}$ |
| | 8 | Ave | $3.764 \times 10^{3}$ | $3.763 \times 10^{3}$ | $3.764 \times 10^{3}$ | $3.763 \times 10^{3}$ | $3.764 \times 10^{3}$ | $3.763 \times 10^{3}$ | $3.764 \times 10^{3}$ | $3.763 \times 10^{3}$ | $3.764 \times 10^{3}$ | $3.764 \times 10^{3}$ |
| | | Std | $1.417 \times 10^{-1}$ | $8.931 \times 10^{-1}$ | $2.822 \times 10^{-2}$ | $2.554 \times 10^{-1}$ | $5.019 \times 10^{-3}$ | $1.255 \times 10^{00}$ | $9.029 \times 10^{-2}$ | $5.291 \times 10^{-1}$ | $7.783 \times 10^{-2}$ | $6.148 \times 10^{-2}$ |
| | 10 | Ave | $3.769 \times 10^{3}$ | $3.769 \times 10^{3}$ | $3.769 \times 10^{3}$ | $3.769 \times 10^{3}$ | $3.769 \times 10^{3}$ | $3.769 \times 10^{3}$ | $3.769 \times 10^{3}$ | $3.768 \times 10^{3}$ | $3.769 \times 10^{3}$ | $3.769 \times 10^{3}$ |
| | | Std | $1.224 \times 10^{-1}$ | $3.473 \times 10^{-1}$ | $9.266 \times 10^{-2}$ | $2.411 \times 10^{-1}$ | $2.170 \times 10^{-2}$ | $8.466 \times 10^{-2}$ | $1.374 \times 10^{-1}$ | $3.721 \times 10^{-1}$ | $1.247 \times 10^{-1}$ | $2.270 \times 10^{-1}$ |

**Table 11.** *Cont.*

| Image | Threshold | Metric | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| bridge | 4 | Ave | $2.640 \times 10^{3}$ | $2.640 \times 10^{3}$ | $2.640 \times 10^{3}$ | $2.640 \times 10^{3}$ | $2.640 \times 10^{3}$ | $2.640 \times 10^{3}$ | $2.640 \times 10^{3}$ | $2.640 \times 10^{3}$ | $2.640 \times 10^{3}$ | $2.640 \times 10^{3}$ |
| | | Std | $6.308 \times 10^{-3}$ | $3.850 \times 10^{-2}$ | $1.388 \times 10^{-12}$ | $7.806 \times 10^{-2}$ | $1.388 \times 10^{-12}$ | $1.388 \times 10^{-12}$ | $1.388 \times 10^{-12}$ | $1.691 \times 10^{-1}$ | $1.388 \times 10^{-12}$ | $1.388 \times 10^{-12}$ |
| | 6 | Ave | $2.719 \times 10^{3}$ | $2.718 \times 10^{3}$ | $2.719 \times 10^{3}$ | $2.718 \times 10^{3}$ | $2.719 \times 10^{3}$ | $2.719 \times 10^{3}$ | $2.719 \times 10^{3}$ | $2.718 \times 10^{3}$ | $2.719 \times 10^{3}$ | $2.719 \times 10^{3}$ |
| | | Std | $7.795 \times 10^{-2}$ | $5.065 \times 10^{-1}$ | $1.242 \times 10^{-2}$ | $7.046 \times 10^{-1}$ | $2.313 \times 10^{-12}$ | $2.204 \times 10^{-2}$ | $7.592 \times 10^{-2}$ | $8.565 \times 10^{-1}$ | $1.690 \times 10^{-2}$ | $1.074 \times 10^{-2}$ |
| | 8 | Ave | $2.752 \times 10^{3}$ | $2.751 \times 10^{3}$ | $2.752 \times 10^{3}$ | $2.750 \times 10^{3}$ | $2.752 \times 10^{3}$ | $2.752 \times 10^{3}$ | $2.752 \times 10^{3}$ | $2.750 \times 10^{3}$ | $2.752 \times 10^{3}$ | $2.752 \times 10^{3}$ |
| | | Std | $2.085 \times 10^{-1}$ | $9.252 \times 10^{-1}$ | $4.128 \times 10^{-2}$ | $1.026 \times 10^{00}$ | $5.943 \times 10^{-3}$ | $2.048 \times 10^{-2}$ | $3.281 \times 10^{-1}$ | $1.165 \times 10^{00}$ | $1.005 \times 10^{-1}$ | $2.886 \times 10^{-2}$ |
| | 10 | Ave | $2.768 \times 10^{3}$ | $2.767 \times 10^{3}$ | $2.768 \times 10^{3}$ | $2.766 \times 10^{3}$ | $2.768 \times 10^{3}$ | $2.768 \times 10^{3}$ | $2.768 \times 10^{3}$ | $2.766 \times 10^{3}$ | $2.768 \times 10^{3}$ | $2.768 \times 10^{3}$ |
| | | Std | $4.287 \times 10^{-1}$ | $1.534 \times 10^{00}$ | $1.480 \times 10^{-1}$ | $7.036 \times 10^{-1}$ | $2.744 \times 10^{-2}$ | $6.493 \times 10^{-1}$ | $3.813 \times 10^{-1}$ | $8.860 \times 10^{-1}$ | $2.553 \times 10^{-1}$ | $6.314 \times 10^{-1}$ |
| camera | 4 | Ave | $4.601 \times 10^{3}$ | $4.600 \times 10^{3}$ | $4.600 \times 10^{3}$ | $4.601 \times 10^{3}$ | $4.600 \times 10^{3}$ | $4.599 \times 10^{3}$ | $4.601 \times 10^{3}$ | $4.601 \times 10^{3}$ | $4.601 \times 10^{3}$ | $4.601 \times 10^{3}$ |
| | | Std | $6.121 \times 10^{-1}$ | $1.125 \times 10^{00}$ | $1.018 \times 10^{00}$ | $3.118 \times 10^{-1}$ | $9.838 \times 10^{-1}$ | $1.267 \times 10^{00}$ | $6.774 \times 10^{-3}$ | $5.419 \times 10^{-1}$ | $8.166 \times 10^{-1}$ | $1.116 \times 10^{-2}$ |
| | 6 | Ave | $4.652 \times 10^{3}$ | $4.652 \times 10^{3}$ | $4.652 \times 10^{3}$ | $4.651 \times 10^{3}$ | $4.652 \times 10^{3}$ | $4.652 \times 10^{3}$ | $4.652 \times 10^{3}$ | $4.651 \times 10^{3}$ | $4.652 \times 10^{3}$ | $4.652 \times 10^{3}$ |
| | | Std | $6.146 \times 10^{-2}$ | $3.386 \times 10^{-1}$ | $8.109 \times 10^{-3}$ | $4.799 \times 10^{-1}$ | $9.250 \times 10^{-13}$ | $3.344 \times 10^{-3}$ | $6.793 \times 10^{-2}$ | $3.450 \times 10^{-1}$ | $3.532 \times 10^{-3}$ | $7.707 \times 10^{-3}$ |
| | 8 | Ave | $4.670 \times 10^{3}$ | $4.670 \times 10^{3}$ | $4.669 \times 10^{3}$ | $4.669 \times 10^{3}$ | $4.669 \times 10^{3}$ | $4.670 \times 10^{3}$ | $4.670 \times 10^{3}$ | $4.669 \times 10^{3}$ | $4.670 \times 10^{3}$ | $4.670 \times 10^{3}$ |
| | | Std | $5.763 \times 10^{-1}$ | $9.557 \times 10^{-1}$ | $6.473 \times 10^{-1}$ | $7.773 \times 10^{-1}$ | $6.870 \times 10^{-1}$ | $7.990 \times 10^{-1}$ | $8.342 \times 10^{-1}$ | $6.971 \times 10^{-1}$ | $8.696 \times 10^{-1}$ | $8.593 \times 10^{-1}$ |
| | 10 | Ave | $4.681 \times 10^{3}$ | $4.680 \times 10^{3}$ | $4.681 \times 10^{3}$ | $4.679 \times 10^{3}$ | $4.681 \times 10^{3}$ | $4.681 \times 10^{3}$ | $4.681 \times 10^{3}$ | $4.679 \times 10^{3}$ | $4.681 \times 10^{3}$ | $4.681 \times 10^{3}$ |
| | | Std | $1.987 \times 10^{-1}$ | $1.246 \times 10^{00}$ | $1.085 \times 10^{-1}$ | $6.064 \times 10^{-1}$ | $4.589 \times 10^{-2}$ | $4.651 \times 10^{-2}$ | $4.228 \times 10^{-1}$ | $7.403 \times 10^{-1}$ | $1.043 \times 10^{-1}$ | $3.906 \times 10^{-1}$ |
| cell | 4 | Ave | $2.567 \times 10^{3}$ | $2.567 \times 10^{3}$ | $2.567 \times 10^{3}$ | $2.567 \times 10^{3}$ | $2.567 \times 10^{3}$ | $2.567 \times 10^{3}$ | $2.567 \times 10^{3}$ | $2.567 \times 10^{3}$ | $2.567 \times 10^{3}$ | $2.567 \times 10^{3}$ |
| | | Std | $5.157 \times 10^{-3}$ | $9.250 \times 10^{-13}$ | $9.250 \times 10^{-13}$ | $4.078 \times 10^{-2}$ | $9.250 \times 10^{-13}$ | $9.250 \times 10^{-13}$ | $9.250 \times 10^{-13}$ | $3.711 \times 10^{-3}$ | $9.250 \times 10^{-13}$ | $9.250 \times 10^{-13}$ |
| | 6 | Ave | $2.596 \times 10^{3}$ | $2.596 \times 10^{3}$ | $2.596 \times 10^{3}$ | $2.596 \times 10^{3}$ | $2.596 \times 10^{3}$ | $2.596 \times 10^{3}$ | $2.596 \times 10^{3}$ | $2.596 \times 10^{3}$ | $2.596 \times 10^{3}$ | $2.596 \times 10^{3}$ |
| | | Std | $9.540 \times 10^{-2}$ | $2.254 \times 10^{-1}$ | $1.417 \times 10^{-1}$ | $2.079 \times 10^{-1}$ | $1.338 \times 10^{-1}$ | $1.443 \times 10^{-1}$ | $1.484 \times 10^{-1}$ | $1.685 \times 10^{-1}$ | $1.166 \times 10^{-1}$ | $8.480 \times 10^{-2}$ |
| | 8 | Ave | $2.612 \times 10^{3}$ | $2.612 \times 10^{3}$ | $2.612 \times 10^{3}$ | $2.611 \times 10^{3}$ | $2.612 \times 10^{3}$ | $2.612 \times 10^{3}$ | $2.612 \times 10^{3}$ | $2.611 \times 10^{3}$ | $2.612 \times 10^{3}$ | $2.612 \times 10^{3}$ |
| | | Std | $1.108 \times 10^{-1}$ | $4.209 \times 10^{-1}$ | $1.781 \times 10^{-1}$ | $3.816 \times 10^{-1}$ | $1.511 \times 10^{-3}$ | $5.411 \times 10^{-2}$ | $9.064 \times 10^{-2}$ | $3.157 \times 10^{-1}$ | $1.087 \times 10^{-1}$ | $4.626 \times 10^{-2}$ |
| | 10 | Ave | $2.619 \times 10^{3}$ | $2.619 \times 10^{3}$ | $2.619 \times 10^{3}$ | $2.618 \times 10^{3}$ | $2.620 \times 10^{3}$ | $2.619 \times 10^{3}$ | $2.619 \times 10^{3}$ | $2.618 \times 10^{3}$ | $2.620 \times 10^{3}$ | $2.620 \times 10^{3}$ |
| | | Std | $2.674 \times 10^{-1}$ | $5.644 \times 10^{-1}$ | $3.128 \times 10^{-1}$ | $5.584 \times 10^{-1}$ | $4.021 \times 10^{-3}$ | $6.670 \times 10^{-1}$ | $4.164 \times 10^{-1}$ | $3.553 \times 10^{-1}$ | $3.119 \times 10^{-1}$ | $3.201 \times 10^{-1}$ |
| columbia | 4 | Ave | $1.827 \times 10^{3}$ | $1.827 \times 10^{3}$ | $1.827 \times 10^{3}$ | $1.827 \times 10^{3}$ | $1.827 \times 10^{3}$ | $1.827 \times 10^{3}$ | $1.827 \times 10^{3}$ | $1.827 \times 10^{3}$ | $1.827 \times 10^{3}$ | $1.827 \times 10^{3}$ |
| | | Std | $9.250 \times 10^{-13}$ | $2.239 \times 10^{-2}$ | $9.250 \times 10^{-13}$ | $7.610 \times 10^{-2}$ | $9.250 \times 10^{-13}$ | $1.075 \times 10^{-3}$ | $7.197 \times 10^{-4}$ | $3.142 \times 10^{-2}$ | $9.250 \times 10^{-13}$ | $9.250 \times 10^{-13}$ |
| | 6 | Ave | $1.878 \times 10^{3}$ | $1.878 \times 10^{3}$ | $1.878 \times 10^{3}$ | $1.877 \times 10^{3}$ | $1.878 \times 10^{3}$ | $1.878 \times 10^{3}$ | $1.878 \times 10^{3}$ | $1.878 \times 10^{3}$ | $1.878 \times 10^{3}$ | $1.878 \times 10^{3}$ |
| | | Std | $2.831 \times 10^{-2}$ | $3.070 \times 10^{-1}$ | $1.985 \times 10^{-2}$ | $2.857 \times 10^{-1}$ | $6.938 \times 10^{-13}$ | $2.776 \times 10^{-2}$ | $4.494 \times 10^{-2}$ | $5.555 \times 10^{-1}$ | $3.277 \times 10^{-2}$ | $2.655 \times 10^{-3}$ |
| | 8 | Ave | $1.899 \times 10^{3}$ | $1.899 \times 10^{3}$ | $1.899 \times 10^{3}$ | $1.898 \times 10^{3}$ | $1.899 \times 10^{3}$ | $1.899 \times 10^{3}$ | $1.899 \times 10^{3}$ | $1.898 \times 10^{3}$ | $1.899 \times 10^{3}$ | $1.899 \times 10^{3}$ |
| | | Std | $9.978 \times 10^{-2}$ | $8.472 \times 10^{-1}$ | $6.044 \times 10^{-2}$ | $5.039 \times 10^{-1}$ | $9.072 \times 10^{-3}$ | $3.866 \times 10^{-2}$ | $1.108 \times 10^{-1}$ | $6.579 \times 10^{-1}$ | $4.299 \times 10^{-2}$ | $2.129 \times 10^{-2}$ |
| | 10 | Ave | $1.910 \times 10^{3}$ | $1.910 \times 10^{3}$ | $1.910 \times 10^{3}$ | $1.908 \times 10^{3}$ | $1.910 \times 10^{3}$ | $1.910 \times 10^{3}$ | $1.910 \times 10^{3}$ | $1.908 \times 10^{3}$ | $1.910 \times 10^{3}$ | $1.910 \times 10^{3}$ |
| | | Std | $2.226 \times 10^{-1}$ | $6.125 \times 10^{-1}$ | $1.688 \times 10^{-1}$ | $6.336 \times 10^{-1}$ | $4.588 \times 10^{-3}$ | $1.607 \times 10^{-1}$ | $2.833 \times 10^{-1}$ | $6.890 \times 10^{-1}$ | $1.852 \times 10^{-1}$ | $4.951 \times 10^{-2}$ |
| couple | 4 | Ave | $1.733 \times 10^{3}$ | $1.733 \times 10^{3}$ | $1.733 \times 10^{3}$ | $1.733 \times 10^{3}$ | $1.733 \times 10^{3}$ | $1.733 \times 10^{3}$ | $1.733 \times 10^{3}$ | $1.733 \times 10^{3}$ | $1.733 \times 10^{3}$ | $1.733 \times 10^{3}$ |
| | | Std | $2.928 \times 10^{-2}$ | $1.957 \times 10^{-1}$ | $1.156 \times 10^{-12}$ | $9.090 \times 10^{-2}$ | $1.156 \times 10^{-12}$ | $1.588 \times 10^{-2}$ | $9.365 \times 10^{-3}$ | $1.592 \times 10^{-1}$ | $1.588 \times 10^{-2}$ | $1.156 \times 10^{-12}$ |
| | 6 | Ave | $1.799 \times 10^{3}$ | $1.799 \times 10^{3}$ | $1.799 \times 10^{3}$ | $1.798 \times 10^{3}$ | $1.799 \times 10^{3}$ | $1.799 \times 10^{3}$ | $1.799 \times 10^{3}$ | $1.798 \times 10^{3}$ | $1.799 \times 10^{3}$ | $1.799 \times 10^{3}$ |
| | | Std | $8.390 \times 10^{-2}$ | $9.501 \times 10^{-2}$ | $2.431 \times 10^{-2}$ | $6.145 \times 10^{-1}$ | $2.367 \times 10^{-3}$ | $1.792 \times 10^{-2}$ | $3.931 \times 10^{-2}$ | $7.056 \times 10^{-1}$ | $1.401 \times 10^{-2}$ | $2.367 \times 10^{-3}$ |
| | 8 | Ave | $1.826 \times 10^{3}$ | $1.826 \times 10^{3}$ | $1.826 \times 10^{3}$ | $1.825 \times 10^{3}$ | $1.826 \times 10^{3}$ | $1.826 \times 10^{3}$ | $1.826 \times 10^{3}$ | $1.825 \times 10^{3}$ | $1.826 \times 10^{3}$ | $1.826 \times 10^{3}$ |
| | | Std | $1.840 \times 10^{-1}$ | $7.990 \times 10^{-1}$ | $5.759 \times 10^{-2}$ | $8.329 \times 10^{-1}$ | $0.000 \times 10^{00}$ | $1.800 \times 10^{-2}$ | $1.184 \times 10^{-1}$ | $9.247 \times 10^{-1}$ | $1.309 \times 10^{-1}$ | $2.180 \times 10^{-2}$ |
| | 10 | Ave | $1.839 \times 10^{3}$ | $1.839 \times 10^{3}$ | $1.840 \times 10^{3}$ | $1.838 \times 10^{3}$ | $1.840 \times 10^{3}$ | $1.840 \times 10^{3}$ | $1.839 \times 10^{3}$ | $1.837 \times 10^{3}$ | $1.839 \times 10^{3}$ | $1.840 \times 10^{3}$ |
| | | Std | $4.315 \times 10^{-1}$ | $1.314 \times 10^{00}$ | $1.323 \times 10^{-1}$ | $6.939 \times 10^{-1}$ | $7.430 \times 10^{-3}$ | $2.443 \times 10^{-1}$ | $3.022 \times 10^{-1}$ | $9.746 \times 10^{-1}$ | $5.120 \times 10^{-1}$ | $1.463 \times 10^{-1}$ |

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

| Image | Threshold | Metric | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| baboon | 4 | Ave | $2.0043 \times 10^{1}$ | $2.0013 \times 10^{1}$ | $2.0043 \times 10^{1}$ | $1.9984 \times 10^{1}$ | $2.0043 \times 10^{1}$ | $2.0043 \times 10^{1}$ | $2.0043 \times 10^{1}$ | $2.0027 \times 10^{1}$ | $2.0038 \times 10^{1}$ | $2.0038 \times 10^{1}$ |
| | | Std | $1.8220 \times 10^{-3}$ | $9.3751 \times 10^{-2}$ | $7.2269 \times 10^{-15}$ | $1.5140 \times 10^{-1}$ | $7.2269 \times 10^{-15}$ | $7.2269 \times 10^{-15}$ | $7.2269 \times 10^{-15}$ | $4.9761 \times 10^{-2}$ | $2.8753 \times 10^{-2}$ | $2.8753 \times 10^{-2}$ |
| | 6 | Ave | $2.2783 \times 10^{1}$ | $2.2199 \times 10^{1}$ | $2.2788 \times 10^{1}$ | $2.2706 \times 10^{1}$ | $2.2732 \times 10^{1}$ | $2.2729 \times 10^{1}$ | $2.2733 \times 10^{1}$ | $2.2634 \times 10^{1}$ | $2.2772 \times 10^{1}$ | $2.2747 \times 10^{1}$ |
| | | Std | $1.6022 \times 10^{-1}$ | $3.1269 \times 10^{00}$ | $1.3727 \times 10^{-1}$ | $4.5581 \times 10^{-1}$ | $7.2269 \times 10^{-15}$ | $7.9669 \times 10^{-2}$ | $5.2945 \times 10^{-2}$ | $4.4727 \times 10^{-1}$ | $1.1033 \times 10^{-1}$ | $5.6263 \times 10^{-2}$ |
| | 8 | Ave | $2.5170 \times 10^{1}$ | $1.9878 \times 10^{1}$ | $2.5180 \times 10^{1}$ | $2.1069 \times 10^{1}$ | $2.5346 \times 10^{1}$ | $2.5187 \times 10^{1}$ | $2.4900 \times 10^{1}$ | $2.4725 \times 10^{1}$ | $2.5248 \times 10^{1}$ | $2.5180 \times 10^{1}$ |
| | | Std | $4.9283 \times 10^{-1}$ | $8.6792 \times 10^{00}$ | $3.0325 \times 10^{-1}$ | $7.7697 \times 10^{00}$ | $4.0809 \times 10^{-2}$ | $3.5620 \times 10^{-1}$ | $1.0685 \times 10^{00}$ | $7.3364 \times 10^{-1}$ | $3.7311 \times 10^{-1}$ | $3.9277 \times 10^{-1}$ |
| | 10 | Ave | $2.6875 \times 10^{1}$ | $1.4206 \times 10^{1}$ | $2.6782 \times 10^{1}$ | $1.7943 \times 10^{1}$ | $2.6825 \times 10^{1}$ | $2.6506 \times 10^{1}$ | $2.5146 \times 10^{1}$ | $2.6034 \times 10^{1}$ | $2.6305 \times 10^{1}$ | $2.6810 \times 10^{1}$ |
| | | Std | $5.0870 \times 10^{-1}$ | $1.0532 \times 10^{1}$ | $3.5161 \times 10^{-1}$ | $1.0714 \times 10^{1}$ | $1.0775 \times 10^{-1}$ | $1.8776 \times 10^{00}$ | $3.8589 \times 10^{00}$ | $8.3462 \times 10^{-1}$ | $3.4924 \times 10^{00}$ | $3.0576 \times 10^{-1}$ |
| bank | 4 | Ave | $2.0250 \times 10^{1}$ | $2.0247 \times 10^{1}$ | $2.0253 \times 10^{1}$ | $2.0234 \times 10^{1}$ | $2.0254 \times 10^{1}$ | $2.0245 \times 10^{1}$ | $2.0253 \times 10^{1}$ | $2.0236 \times 10^{1}$ | $2.0252 \times 10^{1}$ | $2.0253 \times 10^{1}$ |
| | | Std | $1.2877 \times 10^{-2}$ | $1.3920 \times 10^{-2}$ | $2.3359 \times 10^{-3}$ | $1.9698 \times 10^{-2}$ | $1.4454 \times 10^{-14}$ | $2.7991 \times 10^{-2}$ | $3.4256 \times 10^{-3}$ | $2.9514 \times 10^{-2}$ | $5.0318 \times 10^{-3}$ | $2.3359 \times 10^{-3}$ |
| | 6 | Ave | $2.3098 \times 10^{1}$ | $2.3087 \times 10^{1}$ | $2.3105 \times 10^{1}$ | $2.3003 \times 10^{1}$ | $2.3097 \times 10^{1}$ | $2.3101 \times 10^{1}$ | $2.3100 \times 10^{1}$ | $2.3082 \times 10^{1}$ | $2.3101 \times 10^{1}$ | $2.3097 \times 10^{1}$ |
| | | Std | $1.2198 \times 10^{-2}$ | $4.7784 \times 10^{-2}$ | $6.9615 \times 10^{-3}$ | $7.7247 \times 10^{-2}$ | $3.6134 \times 10^{-15}$ | $5.6647 \times 10^{-3}$ | $4.7340 \times 10^{-3}$ | $6.0110 \times 10^{-2}$ | $1.1827 \times 10^{-2}$ | $3.6134 \times 10^{-15}$ |
| | 8 | Ave | $2.5276 \times 10^{1}$ | $2.5212 \times 10^{1}$ | $2.5284 \times 10^{1}$ | $2.5094 \times 10^{1}$ | $2.5293 \times 10^{1}$ | $2.5258 \times 10^{1}$ | $2.5274 \times 10^{1}$ | $2.5107 \times 10^{1}$ | $2.5282 \times 10^{1}$ | $2.5296 \times 10^{1}$ |
| | | Std | $4.6213 \times 10^{-2}$ | $1.3222 \times 10^{-1}$ | $1.7980 \times 10^{-2}$ | $1.0749 \times 10^{-1}$ | $1.3611 \times 10^{-2}$ | $1.6128 \times 10^{-1}$ | $3.6399 \times 10^{-2}$ | $1.3672 \times 10^{-1}$ | $4.1472 \times 10^{-2}$ | $1.8264 \times 10^{-2}$ |
| | 10 | Ave | $2.6852 \times 10^{1}$ | $2.6815 \times 10^{1}$ | $2.6918 \times 10^{1}$ | $2.6607 \times 10^{1}$ | $2.6931 \times 10^{1}$ | $2.6884 \times 10^{1}$ | $2.6873 \times 10^{1}$ | $2.6588 \times 10^{1}$ | $2.6890 \times 10^{1}$ | $2.6909 \times 10^{1}$ |
| | | Std | $8.3915 \times 10^{-2}$ | $1.5534 \times 10^{-1}$ | $3.8050 \times 10^{-2}$ | $1.3514 \times 10^{-1}$ | $8.2387 \times 10^{-3}$ | $1.5275 \times 10^{-1}$ | $9.3005 \times 10^{-2}$ | $1.4243 \times 10^{-1}$ | $1.0596 \times 10^{-1}$ | $7.9087 \times 10^{-2}$ |
| barbara | 4 | Ave | $1.8769 \times 10^{1}$ | $1.8769 \times 10^{1}$ | $1.8769 \times 10^{1}$ | $1.8772 \times 10^{1}$ | $1.8769 \times 10^{1}$ | $1.8769 \times 10^{1}$ | $1.8769 \times 10^{1}$ | $1.8769 \times 10^{1}$ | $1.8769 \times 10^{1}$ | $1.8769 \times 10^{1}$ |
| | | Std | $7.2269 \times 10^{-15}$ | $2.3013 \times 10^{-4}$ | $7.2269 \times 10^{-15}$ | $1.9208 \times 10^{-2}$ | $7.2269 \times 10^{-15}$ | $7.2269 \times 10^{-15}$ | $7.2269 \times 10^{-15}$ | $1.9589 \times 10^{-2}$ | $7.2269 \times 10^{-15}$ | $7.2269 \times 10^{-15}$ |
| | 6 | Ave | $2.1076 \times 10^{1}$ | $2.1117 \times 10^{1}$ | $2.1067 \times 10^{1}$ | $2.1064 \times 10^{1}$ | $2.1054 \times 10^{1}$ | $2.1056 \times 10^{1}$ | $2.1072 \times 10^{1}$ | $2.1061 \times 10^{1}$ | $2.1063 \times 10^{1}$ | $2.1054 \times 10^{1}$ |
| | | Std | $3.7126 \times 10^{-2}$ | $9.0396 \times 10^{-2}$ | $2.7941 \times 10^{-2}$ | $1.6756 \times 10^{-1}$ | $0.0000 \times 10^{00}$ | $1.1628 \times 10^{-2}$ | $3.3870 \times 10^{-2}$ | $1.3449 \times 10^{-1}$ | $2.2699 \times 10^{-2}$ | $0.0000 \times 10^{00}$ |
| | 8 | Ave | $2.3107 \times 10^{1}$ | $2.3161 \times 10^{1}$ | $2.3113 \times 10^{1}$ | $2.2993 \times 10^{1}$ | $2.3183 \times 10^{1}$ | $2.3149 \times 10^{1}$ | $2.3175 \times 10^{1}$ | $2.2827 \times 10^{1}$ | $2.3111 \times 10^{1}$ | $2.3163 \times 10^{1}$ |
| | | Std | $1.5207 \times 10^{-1}$ | $2.5714 \times 10^{-1}$ | $1.0024 \times 10^{-1}$ | $3.5611 \times 10^{-1}$ | $1.0840 \times 10^{-14}$ | $6.1531 \times 10^{-2}$ | $1.2344 \times 10^{-1}$ | $3.1101 \times 10^{-1}$ | $1.1358 \times 10^{-1}$ | $5.8799 \times 10^{-2}$ |
| | 10 | Ave | $2.4595 \times 10^{1}$ | $2.4585 \times 10^{1}$ | $2.4634 \times 10^{1}$ | $2.4525 \times 10^{1}$ | $2.4636 \times 10^{1}$ | $2.4631 \times 10^{1}$ | $2.4600 \times 10^{1}$ | $2.4207 \times 10^{1}$ | $2.4550 \times 10^{1}$ | $2.4641 \times 10^{1}$ |
| | | Std | $2.0911 \times 10^{-1}$ | $2.3445 \times 10^{-1}$ | $1.7159 \times 10^{-1}$ | $3.6423 \times 10^{-1}$ | $5.2702 \times 10^{-2}$ | $1.1395 \times 10^{-1}$ | $2.0340 \times 10^{-1}$ | $5.6873 \times 10^{-1}$ | $2.8118 \times 10^{-1}$ | $8.2002 \times 10^{-2}$ |
| boat | 4 | Ave | $1.9897 \times 10^{1}$ | $1.9901 \times 10^{1}$ | $1.9898 \times 10^{1}$ | $1.9892 \times 10^{1}$ | $1.9898 \times 10^{1}$ | $1.9898 \times 10^{1}$ | $1.9898 \times 10^{1}$ | $1.9898 \times 10^{1}$ | $1.9898 \times 10^{1}$ | $1.9898 \times 10^{1}$ |
| | | Std | $1.2417 \times 10^{-3}$ | $1.2035 \times 10^{-2}$ | $1.0840 \times 10^{-14}$ | $4.7619 \times 10^{-2}$ | $1.0840 \times 10^{-14}$ | $1.0840 \times 10^{-14}$ | $1.0840 \times 10^{-14}$ | $1.9761 \times 10^{-2}$ | $1.0840 \times 10^{-14}$ | $1.0840 \times 10^{-14}$ |
| | 6 | Ave | $2.3032 \times 10^{1}$ | $2.3036 \times 10^{1}$ | $2.3045 \times 10^{1}$ | $2.2965 \times 10^{1}$ | $2.3062 \times 10^{1}$ | $2.3039 \times 10^{1}$ | $2.3042 \times 10^{1}$ | $2.2971 \times 10^{1}$ | $2.3032 \times 10^{1}$ | $2.3058 \times 10^{1}$ |
| | | Std | $5.2951 \times 10^{-2}$ | $5.1484 \times 10^{-2}$ | $2.5596 \times 10^{-2}$ | $1.4360 \times 10^{-1}$ | $1.0840 \times 10^{-14}$ | $3.5706 \times 10^{-2}$ | $4.0395 \times 10^{-2}$ | $8.7513 \times 10^{-2}$ | $3.2469 \times 10^{-2}$ | $1.0742 \times 10^{-2}$ |
| | 8 | Ave | $2.5054 \times 10^{1}$ | $2.5088 \times 10^{1}$ | $2.5086 \times 10^{1}$ | $2.5125 \times 10^{1}$ | $2.5128 \times 10^{1}$ | $2.5115 \times 10^{1}$ | $2.5137 \times 10^{1}$ | $2.4897 \times 10^{1}$ | $2.5107 \times 10^{1}$ | $2.5093 \times 10^{1}$ |
| | | Std | $1.2334 \times 10^{-1}$ | $1.7846 \times 10^{-1}$ | $7.1810 \times 10^{-2}$ | $2.9238 \times 10^{-1}$ | $5.0066 \times 10^{-2}$ | $5.9528 \times 10^{-2}$ | $9.6351 \times 10^{-2}$ | $3.1784 \times 10^{-1}$ | $1.2537 \times 10^{-1}$ | $6.6630 \times 10^{-2}$ |
| | 10 | Ave | $2.6904 \times 10^{1}$ | $2.6848 \times 10^{1}$ | $2.6872 \times 10^{1}$ | $2.6734 \times 10^{1}$ | $2.6992 \times 10^{1}$ | $2.7013 \times 10^{1}$ | $2.6895 \times 10^{1}$ | $2.6471 \times 10^{1}$ | $2.6783 \times 10^{1}$ | $2.6883 \times 10^{1}$ |
| | | Std | $2.7577 \times 10^{-1}$ | $3.3114 \times 10^{-1}$ | $2.1533 \times 10^{-1}$ | $3.5248 \times 10^{-1}$ | $2.8711 \times 10^{-2}$ | $1.5284 \times 10^{-1}$ | $2.1437 \times 10^{-1}$ | $3.7559 \times 10^{-1}$ | $2.9500 \times 10^{-1}$ | $1.9512 \times 10^{-1}$ |
| brain | 4 | Ave | $2.5074 \times 10^{1}$ | $2.5074 \times 10^{1}$ | $2.5080 \times 10^{1}$ | $2.5067 \times 10^{1}$ | $2.5066 \times 10^{1}$ | $2.5072 \times 10^{1}$ | $2.5056 \times 10^{1}$ | $2.5079 \times 10^{1}$ | $2.5060 \times 10^{1}$ | $2.5084 \times 10^{1}$ |
| | | Std | $5.2921 \times 10^{-2}$ | $5.4889 \times 10^{-2}$ | $5.5675 \times 10^{-2}$ | $7.4146 \times 10^{-2}$ | $4.8494 \times 10^{-2}$ | $5.8925 \times 10^{-2}$ | $4.7507 \times 10^{-2}$ | $4.8104 \times 10^{-2}$ | $5.2566 \times 10^{-2}$ | $4.7593 \times 10^{-2}$ |
| | 6 | Ave | $2.7315 \times 10^{1}$ | $2.7510 \times 10^{1}$ | $2.7572 \times 10^{1}$ | $2.7195 \times 10^{1}$ | $2.7556 \times 10^{1}$ | $2.7474 \times 10^{1}$ | $2.7324 \times 10^{1}$ | $2.7329 \times 10^{1}$ | $2.7489 \times 10^{1}$ | $2.7410 \times 10^{1}$ |
| | | Std | $9.9469 \times 10^{-2}$ | $2.2300 \times 10^{-1}$ | $1.2711 \times 10^{-1}$ | $1.4883 \times 10^{-1}$ | $9.7655 \times 10^{-2}$ | $1.9204 \times 10^{-1}$ | $1.1499 \times 10^{-1}$ | $2.4033 \times 10^{-1}$ | $1.6065 \times 10^{-1}$ | $1.3366 \times 10^{-1}$ |
| | 8 | Ave | $2.9518 \times 10^{1}$ | $2.9438 \times 10^{1}$ | $2.9532 \times 10^{1}$ | $2.9290 \times 10^{1}$ | $2.9554 \times 10^{1}$ | $2.9494 \times 10^{1}$ | $2.9509 \times 10^{1}$ | $2.9309 \times 10^{1}$ | $2.9534 \times 10^{1}$ | $2.9555 \times 10^{1}$ |
| | | Std | $1.0328 \times 10^{-1}$ | $1.7318 \times 10^{-1}$ | $7.7984 \times 10^{-2}$ | $1.8247 \times 10^{-1}$ | $5.8882 \times 10^{-2}$ | $2.0732 \times 10^{-1}$ | $9.2775 \times 10^{-2}$ | $1.8462 \times 10^{-1}$ | $6.3379 \times 10^{-2}$ | $6.2975 \times 10^{-2}$ |
| | 10 | Ave | $3.0948 \times 10^{1}$ | $3.0910 \times 10^{1}$ | $3.1031 \times 10^{1}$ | $3.0692 \times 10^{1}$ | $3.1096 \times 10^{1}$ | $3.0978 \times 10^{1}$ | $3.0940 \times 10^{1}$ | $3.0669 \times 10^{1}$ | $3.0980 \times 10^{1}$ | $3.0980 \times 10^{1}$ |
| | | Std | $1.3441 \times 10^{-1}$ | $1.8819 \times 10^{-1}$ | $1.3002 \times 10^{-1}$ | $2.2361 \times 10^{-1}$ | $8.2212 \times 10^{-2}$ | $1.6043 \times 10^{-1}$ | $1.7196 \times 10^{-1}$ | $2.2679 \times 10^{-1}$ | $1.9969 \times 10^{-1}$ | $2.0238 \times 10^{-1}$ |

**Table 12.** *Cont.*

| Image | Threshold | Metric | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| bridge | 4 | Ave | $1.8547 \times 10^{1}$ | $1.8563 \times 10^{1}$ | $1.8546 \times 10^{1}$ | $1.8562 \times 10^{1}$ | $1.8546 \times 10^{1}$ | $1.8546 \times 10^{1}$ | $1.8546 \times 10^{1}$ | $1.8551 \times 10^{1}$ | $1.8546 \times 10^{1}$ | $1.8546 \times 10^{1}$ |
| | | Std | $1.4255 \times 10^{-3}$ | $4.4020 \times 10^{-2}$ | $3.6134 \times 10^{-15}$ | $6.0258 \times 10^{-2}$ | $3.6134 \times 10^{-15}$ | $3.6134 \times 10^{-15}$ | $3.6134 \times 10^{-15}$ | $6.4280 \times 10^{-2}$ | $3.6134 \times 10^{-15}$ | $3.6134 \times 10^{-15}$ |
| | 6 | Ave | $2.1658 \times 10^{1}$ | $2.1672 \times 10^{1}$ | $2.1668 \times 10^{1}$ | $2.1654 \times 10^{1}$ | $2.1659 \times 10^{1}$ | $2.1648 \times 10^{1}$ | $2.1650 \times 10^{1}$ | $2.1584 \times 10^{1}$ | $2.1663 \times 10^{1}$ | $2.1657 \times 10^{1}$ |
| | | Std | $7.2423 \times 10^{-2}$ | $1.2255 \times 10^{-1}$ | $2.8597 \times 10^{-2}$ | $1.9936 \times 10^{-1}$ | $0.0000 \times 10^{00}$ | $2.2355 \times 10^{-2}$ | $4.6133 \times 10^{-2}$ | $2.3117 \times 10^{-1}$ | $2.6904 \times 10^{-2}$ | $2.0880 \times 10^{-2}$ |
| | 8 | Ave | $2.3981 \times 10^{1}$ | $2.3866 \times 10^{1}$ | $2.4011 \times 10^{1}$ | $2.3821 \times 10^{1}$ | $2.4015 \times 10^{1}$ | $2.4009 \times 10^{1}$ | $2.3965 \times 10^{1}$ | $2.3788 \times 10^{1}$ | $2.3989 \times 10^{1}$ | $2.4014 \times 10^{1}$ |
| | | Std | $1.3956 \times 10^{-1}$ | $2.1980 \times 10^{-1}$ | $6.8789 \times 10^{-2}$ | $3.5727 \times 10^{-1}$ | $3.2111 \times 10^{-2}$ | $5.6899 \times 10^{-2}$ | $1.4357 \times 10^{-1}$ | $3.3445 \times 10^{-1}$ | $1.2304 \times 10^{-1}$ | $6.5003 \times 10^{-2}$ |
| | 10 | Ave | $2.5767 \times 10^{1}$ | $2.5634 \times 10^{1}$ | $2.5807 \times 10^{1}$ | $2.5643 \times 10^{1}$ | $2.5909 \times 10^{1}$ | $2.5869 \times 10^{1}$ | $2.5805 \times 10^{1}$ | $2.5462 \times 10^{1}$ | $2.5799 \times 10^{1}$ | $2.5890 \times 10^{1}$ |
| | | Std | $2.1725 \times 10^{-1}$ | $3.4655 \times 10^{-1}$ | $1.5148 \times 10^{-1}$ | $2.3667 \times 10^{-1}$ | $8.8194 \times 10^{-2}$ | $3.1488 \times 10^{-1}$ | $2.2674 \times 10^{-1}$ | $3.3442 \times 10^{-1}$ | $2.4669 \times 10^{-1}$ | $2.2270 \times 10^{-1}$ |
| camera | 4 | Ave | $1.9670 \times 10^{1}$ | $1.9059 \times 10^{1}$ | $1.8903 \times 10^{1}$ | $1.9831 \times 10^{1}$ | $1.9205 \times 10^{1}$ | $1.8351 \times 10^{1}$ | $1.9871 \times 10^{1}$ | $1.9704 \times 10^{1}$ | $1.9508 \times 10^{1}$ | $1.9863 \times 10^{1}$ |
| | | Std | $5.4887 \times 10^{-1}$ | $9.4311 \times 10^{-1}$ | $9.2168 \times 10^{-1}$ | $1.0457 \times 10^{-1}$ | $8.9029 \times 10^{-1}$ | $8.8888 \times 10^{-1}$ | $1.7329 \times 10^{-3}$ | $3.4369 \times 10^{-1}$ | $7.3899 \times 10^{-1}$ | $3.4447 \times 10^{-2}$ |
| | 6 | Ave | $2.1879 \times 10^{1}$ | $2.1887 \times 10^{1}$ | $2.1895 \times 10^{1}$ | $2.1801 \times 10^{1}$ | $2.1935 \times 10^{1}$ | $2.1922 \times 10^{1}$ | $2.1913 \times 10^{1}$ | $2.1823 \times 10^{1}$ | $2.1905 \times 10^{1}$ | $2.1914 \times 10^{1}$ |
| | | Std | $6.4135 \times 10^{-2}$ | $1.0619 \times 10^{-1}$ | $5.1106 \times 10^{-2}$ | $2.5455 \times 10^{-1}$ | $3.6134 \times 10^{-15}$ | $2.7301 \times 10^{-2}$ | $3.4446 \times 10^{-2}$ | $1.4054 \times 10^{-1}$ | $4.2733 \times 10^{-2}$ | $4.1597 \times 10^{-2}$ |
| | 8 | Ave | $2.3022 \times 10^{1}$ | $2.3144 \times 10^{1}$ | $2.3276 \times 10^{1}$ | $2.2985 \times 10^{1}$ | $2.3295 \times 10^{1}$ | $2.3229 \times 10^{1}$ | $2.3182 \times 10^{1}$ | $2.2970 \times 10^{1}$ | $2.3227 \times 10^{1}$ | $2.3196 \times 10^{1}$ |
| | | Std | $1.7916 \times 10^{-1}$ | $1.6877 \times 10^{-1}$ | $1.6904 \times 10^{-1}$ | $3.0685 \times 10^{-1}$ | $1.5584 \times 10^{-1}$ | $1.7319 \times 10^{-1}$ | $2.1028 \times 10^{-1}$ | $3.5030 \times 10^{-1}$ | $2.3951 \times 10^{-1}$ | $2.0685 \times 10^{-1}$ |
| | 10 | Ave | $2.4374 \times 10^{1}$ | $2.4273 \times 10^{1}$ | $2.4301 \times 10^{1}$ | $2.4349 \times 10^{1}$ | $2.4395 \times 10^{1}$ | $2.4296 \times 10^{1}$ | $2.4285 \times 10^{1}$ | $2.3961 \times 10^{1}$ | $2.4330 \times 10^{1}$ | $2.4300 \times 10^{1}$ |
| | | Std | $2.8828 \times 10^{-1}$ | $3.9483 \times 10^{-1}$ | $2.8288 \times 10^{-1}$ | $4.4959 \times 10^{-1}$ | $2.1134 \times 10^{-1}$ | $2.2902 \times 10^{-1}$ | $2.8784 \times 10^{-1}$ | $4.4186 \times 10^{-1}$ | $3.2961 \times 10^{-1}$ | $2.9287 \times 10^{-1}$ |
| cell | 4 | Ave | $1.8639 \times 10^{1}$ | $1.8639 \times 10^{1}$ | $1.8639 \times 10^{1}$ | $1.8654 \times 10^{1}$ | $1.8639 \times 10^{1}$ | $1.8639 \times 10^{1}$ | $1.8639 \times 10^{1}$ | $1.8639 \times 10^{1}$ | $1.8639 \times 10^{1}$ | $1.8639 \times 10^{1}$ |
| | | Std | $2.1159 \times 10^{-4}$ | $1.0840 \times 10^{-14}$ | $1.0840 \times 10^{-14}$ | $4.4564 \times 10^{-2}$ | $1.0840 \times 10^{-14}$ | $1.0840 \times 10^{-14}$ | $1.0840 \times 10^{-14}$ | $1.5226 \times 10^{-4}$ | $1.0840 \times 10^{-14}$ | $1.0840 \times 10^{-14}$ |
| | 6 | Ave | $1.9899 \times 10^{1}$ | $1.9898 \times 10^{1}$ | $1.9869 \times 10^{1}$ | $1.9956 \times 10^{1}$ | $1.9940 \times 10^{1}$ | $1.9834 \times 10^{1}$ | $1.9827 \times 10^{1}$ | $1.9806 \times 10^{1}$ | $1.9838 \times 10^{1}$ | $1.9879 \times 10^{1}$ |
| | | Std | $2.0047 \times 10^{-1}$ | $2.2446 \times 10^{-1}$ | $2.1960 \times 10^{-1}$ | $2.0985 \times 10^{-1}$ | $1.9591 \times 10^{-1}$ | $2.3455 \times 10^{-1}$ | $2.4327 \times 10^{-1}$ | $2.2655 \times 10^{-1}$ | $2.5270 \times 10^{-1}$ | $2.4851 \times 10^{-1}$ |
| | 8 | Ave | $2.0723 \times 10^{1}$ | $2.0730 \times 10^{1}$ | $2.0757 \times 10^{1}$ | $2.0716 \times 10^{1}$ | $2.0716 \times 10^{1}$ | $2.0713 \times 10^{1}$ | $2.0727 \times 10^{1}$ | $2.0656 \times 10^{1}$ | $2.0749 \times 10^{1}$ | $2.0755 \times 10^{1}$ |
| | | Std | $7.8055 \times 10^{-2}$ | $1.1628 \times 10^{-1}$ | $6.2506 \times 10^{-2}$ | $2.3728 \times 10^{-1}$ | $2.7727 \times 10^{-2}$ | $5.9719 \times 10^{-2}$ | $7.0103 \times 10^{-2}$ | $1.5750 \times 10^{-1}$ | $5.7536 \times 10^{-2}$ | $3.6102 \times 10^{-2}$ |
| | 10 | Ave | $2.1243 \times 10^{1}$ | $2.1380 \times 10^{1}$ | $2.1320 \times 10^{1}$ | $2.1322 \times 10^{1}$ | $2.1377 \times 10^{1}$ | $2.1304 \times 10^{1}$ | $2.1265 \times 10^{1}$ | $2.1153 \times 10^{1}$ | $2.1311 \times 10^{1}$ | $2.1303 \times 10^{1}$ |
| | | Std | $1.2296 \times 10^{-1}$ | $2.1800 \times 10^{-1}$ | $1.1314 \times 10^{-1}$ | $3.1032 \times 10^{-1}$ | $1.2424 \times 10^{-3}$ | $1.2084 \times 10^{-1}$ | $1.2562 \times 10^{-1}$ | $2.2105 \times 10^{-1}$ | $1.2946 \times 10^{-1}$ | $1.0266 \times 10^{-1}$ |
| columbia | 4 | Ave | $1.9999 \times 10^{1}$ | $1.9999 \times 10^{1}$ | $1.9999 \times 10^{1}$ | $1.9993 \times 10^{1}$ | $1.9999 \times 10^{1}$ | $1.9999 \times 10^{1}$ | $1.9999 \times 10^{1}$ | $1.9988 \times 10^{1}$ | $1.9999 \times 10^{1}$ | $1.9999 \times 10^{1}$ |
| | | Std | $1.4454 \times 10^{-14}$ | $4.2826 \times 10^{-2}$ | $1.4454 \times 10^{-14}$ | $6.5606 \times 10^{-2}$ | $1.4454 \times 10^{-14}$ | $2.6687 \times 10^{-4}$ | $1.7862 \times 10^{-4}$ | $4.2424 \times 10^{-2}$ | $1.4454 \times 10^{-14}$ | $1.4454 \times 10^{-14}$ |
| | 6 | Ave | $2.2301 \times 10^{1}$ | $2.2310 \times 10^{1}$ | $2.2314 \times 10^{1}$ | $2.2384 \times 10^{1}$ | $2.2320 \times 10^{1}$ | $2.2345 \times 10^{1}$ | $2.2323 \times 10^{1}$ | $2.2198 \times 10^{1}$ | $2.2321 \times 10^{1}$ | $2.2321 \times 10^{1}$ |
| | | Std | $5.8477 \times 10^{-2}$ | $1.5826 \times 10^{-1}$ | $4.3031 \times 10^{-2}$ | $2.8971 \times 10^{-1}$ | $7.2269 \times 10^{-15}$ | $5.9940 \times 10^{-2}$ | $7.2414 \times 10^{-2}$ | $1.7924 \times 10^{-1}$ | $8.1013 \times 10^{-2}$ | $2.6730 \times 10^{-3}$ |
| | 8 | Ave | $2.4073 \times 10^{1}$ | $2.3964 \times 10^{1}$ | $2.4067 \times 10^{1}$ | $2.3952 \times 10^{1}$ | $2.4074 \times 10^{1}$ | $2.4036 \times 10^{1}$ | $2.4036 \times 10^{1}$ | $2.3935 \times 10^{1}$ | $2.4044 \times 10^{1}$ | $2.4080 \times 10^{1}$ |
| | | Std | $1.2835 \times 10^{-1}$ | $2.9780 \times 10^{-1}$ | $7.7280 \times 10^{-2}$ | $3.7273 \times 10^{-1}$ | $3.6052 \times 10^{-2}$ | $9.6682 \times 10^{-2}$ | $1.2093 \times 10^{-1}$ | $2.8412 \times 10^{-1}$ | $1.1075 \times 10^{-1}$ | $3.5891 \times 10^{-2}$ |
| | 10 | Ave | $2.5385 \times 10^{1}$ | $2.5300 \times 10^{1}$ | $2.5337 \times 10^{1}$ | $2.5521 \times 10^{1}$ | $2.5261 \times 10^{1}$ | $2.5365 \times 10^{1}$ | $2.5266 \times 10^{1}$ | $2.4936 \times 10^{1}$ | $2.5310 \times 10^{1}$ | $2.5371 \times 10^{1}$ |
| | | Std | $2.7417 \times 10^{-1}$ | $3.2611 \times 10^{-1}$ | $2.4305 \times 10^{-1}$ | $6.2033 \times 10^{-1}$ | $7.7994 \times 10^{-2}$ | $2.6089 \times 10^{-1}$ | $2.4412 \times 10^{-1}$ | $3.6354 \times 10^{-1}$ | $2.9047 \times 10^{-1}$ | $1.8206 \times 10^{-1}$ |
| couple | 4 | Ave | $2.0264 \times 10^{1}$ | $2.0246 \times 10^{1}$ | $2.0269 \times 10^{1}$ | $2.0247 \times 10^{1}$ | $2.0269 \times 10^{1}$ | $2.0255 \times 10^{1}$ | $2.0265 \times 10^{1}$ | $2.0242 \times 10^{1}$ | $2.0255 \times 10^{1}$ | $2.0269 \times 10^{1}$ |
| | | Std | $2.1817 \times 10^{-2}$ | $4.7972 \times 10^{-2}$ | $7.2269 \times 10^{-15}$ | $5.5179 \times 10^{-2}$ | $7.2269 \times 10^{-15}$ | $2.6489 \times 10^{-2}$ | $1.5622 \times 10^{-2}$ | $7.1147 \times 10^{-2}$ | $2.6489 \times 10^{-2}$ | $7.2269 \times 10^{-15}$ |
| | 6 | Ave | $2.3446 \times 10^{1}$ | $2.3454 \times 10^{1}$ | $2.3460 \times 10^{1}$ | $2.3379 \times 10^{1}$ | $2.3460 \times 10^{1}$ | $2.3465 \times 10^{1}$ | $2.3462 \times 10^{1}$ | $2.3391 \times 10^{1}$ | $2.3462 \times 10^{1}$ | $2.3460 \times 10^{1}$ |
| | | Std | $2.5052 \times 10^{-2}$ | $2.1507 \times 10^{-2}$ | $1.2529 \times 10^{-2}$ | $8.0557 \times 10^{-2}$ | $3.2024 \times 10^{-3}$ | $1.0898 \times 10^{-2}$ | $1.3520 \times 10^{-2}$ | $8.0754 \times 10^{-2}$ | $8.4720 \times 10^{-3}$ | $3.2024 \times 10^{-3}$ |
| | 8 | Ave | $2.5361 \times 10^{1}$ | $2.5366 \times 10^{1}$ | $2.5391 \times 10^{1}$ | $2.5195 \times 10^{1}$ | $2.5426 \times 10^{1}$ | $2.5421 \times 10^{1}$ | $2.5406 \times 10^{1}$ | $2.5271 \times 10^{1}$ | $2.5412 \times 10^{1}$ | $2.5420 \times 10^{1}$ |
| | | Std | $4.5081 \times 10^{-2}$ | $1.2493 \times 10^{-1}$ | $2.1050 \times 10^{-2}$ | $1.5715 \times 10^{-1}$ | $3.6134 \times 10^{-15}$ | $1.2129 \times 10^{-2}$ | $2.6196 \times 10^{-2}$ | $1.1442 \times 10^{-1}$ | $1.6682 \times 10^{-2}$ | $9.8911 \times 10^{-3}$ |
| | 10 | Ave | $2.6788 \times 10^{1}$ | $2.6788 \times 10^{1}$ | $2.6831 \times 10^{1}$ | $2.6652 \times 10^{1}$ | $2.6831 \times 10^{1}$ | $2.6864 \times 10^{1}$ | $2.6883 \times 10^{1}$ | $2.6609 \times 10^{1}$ | $2.6881 \times 10^{1}$ | $2.6843 \times 10^{1}$ |
| | | Std | $7.5709 \times 10^{-2}$ | $1.8703 \times 10^{-1}$ | $4.7342 \times 10^{-2}$ | $1.4799 \times 10^{-1}$ | $9.5275 \times 10^{-3}$ | $5.6289 \times 10^{-2}$ | $8.6158 \times 10^{-2}$ | $2.0722 \times 10^{-1}$ | $6.9185 \times 10^{-2}$ | $3.6745 \times 10^{-2}$ |

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

| Image | Threshold | Metric | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| baboon | 4 | Ave | $8.3368 \times 10^{-1}$ | $8.3309 \times 10^{-1}$ | $8.3367 \times 10^{-1}$ | $8.3259 \times 10^{-1}$ | $8.3367 \times 10^{-1}$ | $8.3367 \times 10^{-1}$ | $8.3367 \times 10^{-1}$ | $8.3340 \times 10^{-1}$ | $8.3356 \times 10^{-1}$ | $8.3356 \times 10^{-1}$ |
| | | Std | $9.9178 \times 10^{-5}$ | $1.6870 \times 10^{-3}$ | $2.2584 \times 10^{-16}$ | $3.0753 \times 10^{-3}$ | $2.2584 \times 10^{-16}$ | $2.2584 \times 10^{-16}$ | $2.2584 \times 10^{-16}$ | $7.9317 \times 10^{-4}$ | $6.0223 \times 10^{-4}$ | $6.0223 \times 10^{-4}$ |
| | 6 | Ave | $8.8549 \times 10^{-1}$ | inf | $8.8557 \times 10^{-1}$ | $8.8393 \times 10^{-1}$ | $8.8462 \times 10^{-1}$ | $8.8465 \times 10^{-1}$ | $8.8474 \times 10^{-1}$ | $8.8281 \times 10^{-1}$ | $8.8538 \times 10^{-1}$ | $8.8489 \times 10^{-1}$ |
| | | Std | $2.7147 \times 10^{-3}$ | inf | $2.3959 \times 10^{-3}$ | $6.3538 \times 10^{-3}$ | $3.3876 \times 10^{-16}$ | $1.2321 \times 10^{-3}$ | $8.3541 \times 10^{-4}$ | $6.3012 \times 10^{-3}$ | $1.8339 \times 10^{-3}$ | $9.6359 \times 10^{-4}$ |
| | 8 | Ave | $9.2119 \times 10^{-1}$ | inf | $9.2117 \times 10^{-1}$ | inf | $9.2286 \times 10^{-1}$ | $9.2133 \times 10^{-1}$ | $9.1655 \times 10^{-1}$ | $9.1388 \times 10^{-1}$ | $9.2214 \times 10^{-1}$ | $9.2129 \times 10^{-1}$ |
| | | Std | $6.4167 \times 10^{-3}$ | inf | $3.6074 \times 10^{-3}$ | inf | $6.2423 \times 10^{-4}$ | $3.7032 \times 10^{-3}$ | $1.9484 \times 10^{-2}$ | $9.8909 \times 10^{-3}$ | $4.5064 \times 10^{-3}$ | $4.8059 \times 10^{-3}$ |
| | 10 | Ave | $9.4242 \times 10^{-1}$ | inf | $9.4213 \times 10^{-1}$ | inf | $9.4324 \times 10^{-1}$ | $9.3501 \times 10^{-1}$ | $8.9844 \times 10^{-1}$ | $9.3048 \times 10^{-1}$ | $9.2100 \times 10^{-1}$ | $9.4292 \times 10^{-1}$ |
| | | Std | $6.3689 \times 10^{-3}$ | inf | $3.4091 \times 10^{-3}$ | inf | $5.5858 \times 10^{-4}$ | $4.1254 \times 10^{-2}$ | $1.0871 \times 10^{-1}$ | $9.2048 \times 10^{-3}$ | $1.0052 \times 10^{-1}$ | $3.5963 \times 10^{-3}$ |
| bank | 4 | Ave | $8.3499 \times 10^{-1}$ | $8.3504 \times 10^{-1}$ | $8.3493 \times 10^{-1}$ | $8.3505 \times 10^{-1}$ | $8.3490 \times 10^{-1}$ | $8.3487 \times 10^{-1}$ | $8.3491 \times 10^{-1}$ | $8.3499 \times 10^{-1}$ | $8.3498 \times 10^{-1}$ | $8.3493 \times 10^{-1}$ |
| | | Std | $2.2030 \times 10^{-4}$ | $2.6605 \times 10^{-4}$ | $1.2955 \times 10^{-4}$ | $6.2546 \times 10^{-4}$ | $1.1292 \times 10^{-16}$ | $3.0615 \times 10^{-4}$ | $2.0398 \times 10^{-5}$ | $5.9176 \times 10^{-4}$ | $2.1620 \times 10^{-4}$ | $1.2955 \times 10^{-4}$ |
| | 6 | Ave | $8.8128 \times 10^{-1}$ | $8.8145 \times 10^{-1}$ | $8.8145 \times 10^{-1}$ | $8.8091 \times 10^{-1}$ | $8.8128 \times 10^{-1}$ | $8.8143 \times 10^{-1}$ | $8.8139 \times 10^{-1}$ | $8.8106 \times 10^{-1}$ | $8.8119 \times 10^{-1}$ | $8.8128 \times 10^{-1}$ |
| | | Std | $4.8838 \times 10^{-4}$ | $7.1891 \times 10^{-4}$ | $3.1737 \times 10^{-4}$ | $1.5432 \times 10^{-3}$ | $5.6460 \times 10^{-16}$ | $2.7124 \times 10^{-4}$ | $2.3995 \times 10^{-4}$ | $1.0533 \times 10^{-3}$ | $2.2669 \times 10^{-4}$ | $5.6460 \times 10^{-16}$ |
| | 8 | Ave | $9.0952 \times 10^{-1}$ | $9.0904 \times 10^{-1}$ | $9.0990 \times 10^{-1}$ | $9.0831 \times 10^{-1}$ | $9.0982 \times 10^{-1}$ | $9.0964 \times 10^{-1}$ | $9.0976 \times 10^{-1}$ | $9.0835 \times 10^{-1}$ | $9.0975 \times 10^{-1}$ | $9.0970 \times 10^{-1}$ |
| | | Std | $1.5825 \times 10^{-3}$ | $2.7873 \times 10^{-3}$ | $8.3354 \times 10^{-4}$ | $3.7422 \times 10^{-3}$ | $1.2788 \times 10^{-4}$ | $2.3413 \times 10^{-3}$ | $1.3162 \times 10^{-3}$ | $2.8426 \times 10^{-3}$ | $1.2912 \times 10^{-3}$ | $4.6018 \times 10^{-4}$ |
| | 10 | Ave | $9.3119 \times 10^{-1}$ | $9.3044 \times 10^{-1}$ | $9.3116 \times 10^{-1}$ | $9.2921 \times 10^{-1}$ | $9.3127 \times 10^{-1}$ | $9.3104 \times 10^{-1}$ | $9.3129 \times 10^{-1}$ | $9.2686 \times 10^{-1}$ | $9.3114 \times 10^{-1}$ | $9.3148 \times 10^{-1}$ |
| | | Std | $1.7641 \times 10^{-3}$ | $2.6827 \times 10^{-3}$ | $9.8791 \times 10^{-4}$ | $3.4240 \times 10^{-3}$ | $2.0227 \times 10^{-4}$ | $2.0609 \times 10^{-3}$ | $1.3813 \times 10^{-3}$ | $2.6745 \times 10^{-3}$ | $2.0855 \times 10^{-3}$ | $1.0583 \times 10^{-3}$ |
| barbara | 4 | Ave | $8.1307 \times 10^{-1}$ | $8.1305 \times 10^{-1}$ | $8.1307 \times 10^{-1}$ | $8.1309 \times 10^{-1}$ | $8.1307 \times 10^{-1}$ | $8.1307 \times 10^{-1}$ | $8.1307 \times 10^{-1}$ | $8.1305 \times 10^{-1}$ | $8.1307 \times 10^{-1}$ | $8.1307 \times 10^{-1}$ |
| | | Std | $5.6460 \times 10^{-16}$ | $1.2093 \times 10^{-4}$ | $5.6460 \times 10^{-16}$ | $4.6029 \times 10^{-4}$ | $5.6460 \times 10^{-16}$ | $5.6460 \times 10^{-16}$ | $5.6460 \times 10^{-16}$ | $2.5892 \times 10^{-4}$ | $5.6460 \times 10^{-16}$ | $5.6460 \times 10^{-16}$ |
| | 6 | Ave | $8.6385 \times 10^{-1}$ | $8.6414 \times 10^{-1}$ | $8.6380 \times 10^{-1}$ | $8.6318 \times 10^{-1}$ | $8.6351 \times 10^{-1}$ | $8.6360 \times 10^{-1}$ | $8.6376 \times 10^{-1}$ | $8.6354 \times 10^{-1}$ | $8.6359 \times 10^{-1}$ | $8.6351 \times 10^{-1}$ |
| | | Std | $3.9678 \times 10^{-4}$ | $6.5980 \times 10^{-4}$ | $2.6551 \times 10^{-4}$ | $1.3910 \times 10^{-3}$ | $5.6460 \times 10^{-16}$ | $1.1160 \times 10^{-4}$ | $3.9417 \times 10^{-4}$ | $1.2120 \times 10^{-3}$ | $1.5899 \times 10^{-4}$ | $5.6460 \times 10^{-16}$ |
| | 8 | Ave | $8.9112 \times 10^{-1}$ | $8.9162 \times 10^{-1}$ | $8.9132 \times 10^{-1}$ | $8.9010 \times 10^{-1}$ | $8.9151 \times 10^{-1}$ | $8.9144 \times 10^{-1}$ | $8.9153 \times 10^{-1}$ | $8.8963 \times 10^{-1}$ | $8.9132 \times 10^{-1}$ | $8.9154 \times 10^{-1}$ |
| | | Std | $9.7223 \times 10^{-4}$ | $1.0178 \times 10^{-3}$ | $5.6094 \times 10^{-4}$ | $1.5840 \times 10^{-3}$ | $0.0000 \times 10^{00}$ | $1.9333 \times 10^{-4}$ | $4.9461 \times 10^{-4}$ | $1.3220 \times 10^{-3}$ | $3.9638 \times 10^{-4}$ | $1.5889 \times 10^{-4}$ |
| | 10 | Ave | $9.1120 \times 10^{-1}$ | $9.1073 \times 10^{-1}$ | $9.1185 \times 10^{-1}$ | $9.0847 \times 10^{-1}$ | $9.1231 \times 10^{-1}$ | $9.1223 \times 10^{-1}$ | $9.1152 \times 10^{-1}$ | $9.0668 \times 10^{-1}$ | $9.1186 \times 10^{-1}$ | $9.1226 \times 10^{-1}$ |
| | | Std | $1.4046 \times 10^{-3}$ | $1.8660 \times 10^{-3}$ | $1.0481 \times 10^{-3}$ | $2.1187 \times 10^{-3}$ | $3.6625 \times 10^{-4}$ | $5.5281 \times 10^{-4}$ | $1.0763 \times 10^{-3}$ | $2.6066 \times 10^{-3}$ | $1.2123 \times 10^{-3}$ | $6.4469 \times 10^{-4}$ |
| boat | 4 | Ave | $8.1304 \times 10^{-1}$ | $8.1305 \times 10^{-1}$ | $8.1304 \times 10^{-1}$ | $8.1282 \times 10^{-1}$ | $8.1304 \times 10^{-1}$ | $8.1304 \times 10^{-1}$ | $8.1304 \times 10^{-1}$ | $8.1297 \times 10^{-1}$ | $8.1304 \times 10^{-1}$ | $8.1304 \times 10^{-1}$ |
| | | Std | $3.7162 \times 10^{-6}$ | $7.8733 \times 10^{-5}$ | $3.3876 \times 10^{-16}$ | $7.4146 \times 10^{-4}$ | $3.3876 \times 10^{-16}$ | $3.3876 \times 10^{-16}$ | $3.3876 \times 10^{-16}$ | $2.0443 \times 10^{-4}$ | $3.3876 \times 10^{-16}$ | $3.3876 \times 10^{-16}$ |
| | 6 | Ave | $8.7865 \times 10^{-1}$ | $8.7880 \times 10^{-1}$ | $8.7874 \times 10^{-1}$ | $8.7765 \times 10^{-1}$ | $8.7886 \times 10^{-1}$ | $8.7872 \times 10^{-1}$ | $8.7877 \times 10^{-1}$ | $8.7821 \times 10^{-1}$ | $8.7881 \times 10^{-1}$ | $8.7885 \times 10^{-1}$ |
| | | Std | $4.2906 \times 10^{-4}$ | $3.5662 \times 10^{-4}$ | $4.3243 \times 10^{-4}$ | $1.4671 \times 10^{-3}$ | $1.1292 \times 10^{-16}$ | $2.1917 \times 10^{-4}$ | $3.1326 \times 10^{-4}$ | $9.3243 \times 10^{-4}$ | $3.3064 \times 10^{-4}$ | $2.0089 \times 10^{-4}$ |
| | 8 | Ave | $9.1173 \times 10^{-1}$ | $9.1137 \times 10^{-1}$ | $9.1181 \times 10^{-1}$ | $9.0959 \times 10^{-1}$ | $9.1192 \times 10^{-1}$ | $9.1199 \times 10^{-1}$ | $9.1183 \times 10^{-1}$ | $9.0859 \times 10^{-1}$ | $9.1211 \times 10^{-1}$ | $9.1180 \times 10^{-1}$ |
| | | Std | $1.0177 \times 10^{-3}$ | $1.5367 \times 10^{-3}$ | $6.1039 \times 10^{-4}$ | $1.9458 \times 10^{-3}$ | $8.6306 \times 10^{-6}$ | $1.5156 \times 10^{-4}$ | $9.0352 \times 10^{-4}$ | $2.1600 \times 10^{-3}$ | $7.7144 \times 10^{-4}$ | $3.7235 \times 10^{-4}$ |
| | 10 | Ave | $9.3275 \times 10^{-1}$ | $9.3212 \times 10^{-1}$ | $9.3230 \times 10^{-1}$ | $9.2889 \times 10^{-1}$ | $9.3348 \times 10^{-1}$ | $9.3283 \times 10^{-1}$ | $9.3257 \times 10^{-1}$ | $9.2701 \times 10^{-1}$ | $9.3286 \times 10^{-1}$ | $9.3314 \times 10^{-1}$ |
| | | Std | $1.0256 \times 10^{-3}$ | $1.7207 \times 10^{-3}$ | $9.5327 \times 10^{-4}$ | $2.1764 \times 10^{-3}$ | $1.6442 \times 10^{-4}$ | $8.8873 \times 10^{-4}$ | $1.1292 \times 10^{-3}$ | $3.2697 \times 10^{-3}$ | $9.5286 \times 10^{-4}$ | $6.5670 \times 10^{-4}$ |
| brain | 4 | Ave | $6.7513 \times 10^{-1}$ | $6.7505 \times 10^{-1}$ | $6.7511 \times 10^{-1}$ | $6.7510 \times 10^{-1}$ | $6.7510 \times 10^{-1}$ | $6.7511 \times 10^{-1}$ | $6.7508 \times 10^{-1}$ | $6.7514 \times 10^{-1}$ | $6.7511 \times 10^{-1}$ | $6.7512 \times 10^{-1}$ |
| | | Std | $8.7986 \times 10^{-5}$ | $2.4925 \times 10^{-4}$ | $8.0070 \times 10^{-5}$ | $1.6006 \times 10^{-4}$ | $8.4112 \times 10^{-5}$ | $8.4774 \times 10^{-5}$ | $8.5347 \times 10^{-5}$ | $1.7712 \times 10^{-4}$ | $8.0714 \times 10^{-5}$ | $7.5690 \times 10^{-5}$ |
| | 6 | Ave | $9.2039 \times 10^{-1}$ | $7.6846 \times 10^{-1}$ | $7.3908 \times 10^{-1}$ | $9.1923 \times 10^{-1}$ | $7.2396 \times 10^{-1}$ | $7.9130 \times 10^{-1}$ | $8.9728 \times 10^{-1}$ | $8.6572 \times 10^{-1}$ | $8.1456 \times 10^{-1}$ | $8.6060 \times 10^{-1}$ |
| | | Std | $2.7556 \times 10^{-3}$ | $1.0840 \times 10^{-1}$ | $9.3017 \times 10^{-2}$ | $3.0315 \times 10^{-3}$ | $7.9135 \times 10^{-2}$ | $1.1401 \times 10^{-1}$ | $6.9108 \times 10^{-2}$ | $9.7400 \times 10^{-2}$ | $1.1568 \times 10^{-1}$ | $1.0263 \times 10^{-1}$ |
| | 8 | Ave | $9.4994 \times 10^{-1}$ | $9.4187 \times 10^{-1}$ | $9.5034 \times 10^{-1}$ | $9.4804 \times 10^{-1}$ | $9.5030 \times 10^{-1}$ | $9.4202 \times 10^{-1}$ | $9.5016 \times 10^{-1}$ | $9.4651 \times 10^{-1}$ | $9.5091 \times 10^{-1}$ | $9.5043 \times 10^{-1}$ |
| | | Std | $1.6432 \times 10^{-3}$ | $4.1456 \times 10^{-2}$ | $1.5399 \times 10^{-3}$ | $1.9501 \times 10^{-3}$ | $1.2931 \times 10^{-3}$ | $4.4793 \times 10^{-2}$ | $1.6697 \times 10^{-3}$ | $3.5200 \times 10^{-3}$ | $1.4171 \times 10^{-3}$ | $1.5326 \times 10^{-3}$ |
| | 10 | Ave | $9.6396 \times 10^{-1}$ | $9.6207 \times 10^{-1}$ | $9.6353 \times 10^{-1}$ | $9.6145 \times 10^{-1}$ | $9.6367 \times 10^{-1}$ | $9.6490 \times 10^{-1}$ | $9.6397 \times 10^{-1}$ | $9.6043 \times 10^{-1}$ | $9.6430 \times 10^{-1}$ | $9.6463 \times 10^{-1}$ |
| | | Std | $1.6824 \times 10^{-3}$ | $3.0844 \times 10^{-3}$ | $1.8047 \times 10^{-3}$ | $2.2796 \times 10^{-3}$ | $1.3262 \times 10^{-3}$ | $1.4194 \times 10^{-3}$ | $2.2229 \times 10^{-3}$ | $2.8039 \times 10^{-3}$ | $1.8369 \times 10^{-3}$ | $1.9837 \times 10^{-3}$ |

**Table 13.** *Cont.*

| Image | Threshold | Metric | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| bridge | 4 | Ave | $7.9571 \times 10^{-1}$ | $7.9607 \times 10^{-1}$ | $7.9569 \times 10^{-1}$ | $7.9619 \times 10^{-1}$ | $7.9569 \times 10^{-1}$ | $7.9569 \times 10^{-1}$ | $7.9569 \times 10^{-1}$ | $7.9568 \times 10^{-1}$ | $7.9569 \times 10^{-1}$ | $7.9569 \times 10^{-1}$ |
| | | Std | $7.7060 \times 10^{-5}$ | $1.0378 \times 10^{-3}$ | $5.6460 \times 10^{-16}$ | $1.4516 \times 10^{-3}$ | $5.6460 \times 10^{-16}$ | $5.6460 \times 10^{-16}$ | $5.6460 \times 10^{-16}$ | $1.6023 \times 10^{-3}$ | $5.6460 \times 10^{-16}$ | $5.6460 \times 10^{-16}$ |
| | 6 | Ave | $8.7209 \times 10^{-1}$ | $8.7230 \times 10^{-1}$ | $8.7265 \times 10^{-1}$ | $8.7283 \times 10^{-1}$ | $8.7244 \times 10^{-1}$ | $8.7236 \times 10^{-1}$ | $8.7207 \times 10^{-1}$ | $8.7016 \times 10^{-1}$ | $8.7254 \times 10^{-1}$ | $8.7242 \times 10^{-1}$ |
| | | Std | $1.3789 \times 10^{-3}$ | $1.7030 \times 10^{-3}$ | $7.8498 \times 10^{-4}$ | $3.6708 \times 10^{-3}$ | $4.5168 \times 10^{-16}$ | $4.0512 \times 10^{-4}$ | $6.5360 \times 10^{-4}$ | $4.0374 \times 10^{-3}$ | $5.9794 \times 10^{-4}$ | $3.2904 \times 10^{-4}$ |
| | 8 | Ave | $9.1221 \times 10^{-1}$ | $9.1053 \times 10^{-1}$ | $9.1258 \times 10^{-1}$ | $9.0932 \times 10^{-1}$ | $9.1258 \times 10^{-1}$ | $9.1273 \times 10^{-1}$ | $9.1221 \times 10^{-1}$ | $9.0820 \times 10^{-1}$ | $9.1260 \times 10^{-1}$ | $9.1274 \times 10^{-1}$ |
| | | Std | $2.6828 \times 10^{-3}$ | $3.2676 \times 10^{-3}$ | $1.3492 \times 10^{-3}$ | $7.0761 \times 10^{-3}$ | $5.3245 \times 10^{-4}$ | $7.8558 \times 10^{-4}$ | $1.8129 \times 10^{-3}$ | $6.6615 \times 10^{-3}$ | $1.8963 \times 10^{-3}$ | $1.1146 \times 10^{-3}$ |
| | 10 | Ave | $9.3534 \times 10^{-1}$ | $9.3413 \times 10^{-1}$ | $9.3640 \times 10^{-1}$ | $9.3453 \times 10^{-1}$ | $9.3755 \times 10^{-1}$ | $9.3666 \times 10^{-1}$ | $9.3594 \times 10^{-1}$ | $9.3006 \times 10^{-1}$ | $9.3631 \times 10^{-1}$ | $9.3722 \times 10^{-1}$ |
| | | Std | $2.7665 \times 10^{-3}$ | $4.9611 \times 10^{-3}$ | $1.3143 \times 10^{-3}$ | $3.5396 \times 10^{-3}$ | $1.0765 \times 10^{-3}$ | $3.5689 \times 10^{-3}$ | $1.9903 \times 10^{-3}$ | $4.7943 \times 10^{-3}$ | $2.7980 \times 10^{-3}$ | $3.1010 \times 10^{-3}$ |
| camera | 4 | Ave | $8.3308 \times 10^{-1}$ | $8.3513 \times 10^{-1}$ | $8.3661 \times 10^{-1}$ | $8.3243 \times 10^{-1}$ | $8.3541 \times 10^{-1}$ | $8.3414 \times 10^{-1}$ | $8.3279 \times 10^{-1}$ | $8.3110 \times 10^{-1}$ | $8.3421 \times 10^{-1}$ | $8.3260 \times 10^{-1}$ |
| | | Std | $2.4931 \times 10^{-3}$ | $4.6444 \times 10^{-3}$ | $3.6630 \times 10^{-3}$ | $2.2114 \times 10^{-3}$ | $3.5382 \times 10^{-3}$ | $8.1730 \times 10^{-3}$ | $1.1035 \times 10^{-4}$ | $3.2811 \times 10^{-3}$ | $2.9369 \times 10^{-3}$ | $7.6966 \times 10^{-4}$ |
| | 6 | Ave | $8.7790 \times 10^{-1}$ | $8.7797 \times 10^{-1}$ | $8.7831 \times 10^{-1}$ | $8.7547 \times 10^{-1}$ | $8.7905 \times 10^{-1}$ | $8.7879 \times 10^{-1}$ | $8.7859 \times 10^{-1}$ | $8.7712 \times 10^{-1}$ | $8.7850 \times 10^{-1}$ | $8.7868 \times 10^{-1}$ |
| | | Std | $1.0028 \times 10^{-3}$ | $2.0275 \times 10^{-3}$ | $8.4153 \times 10^{-4}$ | $4.2740 \times 10^{-3}$ | $3.3876 \times 10^{-16}$ | $5.2412 \times 10^{-4}$ | $6.2342 \times 10^{-4}$ | $2.2453 \times 10^{-3}$ | $7.5318 \times 10^{-4}$ | $7.0857 \times 10^{-4}$ |
| | 8 | Ave | $9.0161 \times 10^{-1}$ | $9.0215 \times 10^{-1}$ | $9.0248 \times 10^{-1}$ | $9.0016 \times 10^{-1}$ | $9.0351 \times 10^{-1}$ | $9.0317 \times 10^{-1}$ | $9.0252 \times 10^{-1}$ | $8.9909 \times 10^{-1}$ | $9.0306 \times 10^{-1}$ | $9.0289 \times 10^{-1}$ |
| | | Std | $1.8965 \times 10^{-3}$ | $1.0651 \times 10^{-3}$ | $1.1945 \times 10^{-3}$ | $4.7787 \times 10^{-3}$ | $8.3352 \times 10^{-4}$ | $9.5864 \times 10^{-4}$ | $9.7287 \times 10^{-4}$ | $4.1231 \times 10^{-3}$ | $1.2348 \times 10^{-3}$ | $7.8222 \times 10^{-4}$ |
| | 10 | Ave | $9.1897 \times 10^{-1}$ | $9.1751 \times 10^{-1}$ | $9.1918 \times 10^{-1}$ | $9.1737 \times 10^{-1}$ | $9.1960 \times 10^{-1}$ | $9.1943 \times 10^{-1}$ | $9.1880 \times 10^{-1}$ | $9.1442 \times 10^{-1}$ | $9.1934 \times 10^{-1}$ | $9.1907 \times 10^{-1}$ |
| | | Std | $1.3728 \times 10^{-3}$ | $2.9766 \times 10^{-3}$ | $1.1734 \times 10^{-3}$ | $3.2815 \times 10^{-3}$ | $6.9378 \times 10^{-4}$ | $7.7839 \times 10^{-4}$ | $1.3788 \times 10^{-3}$ | $3.6976 \times 10^{-3}$ | $8.6426 \times 10^{-4}$ | $1.3506 \times 10^{-3}$ |
| cell | 4 | Ave | $8.7881 \times 10^{-1}$ | $8.7881 \times 10^{-1}$ | $8.7881 \times 10^{-1}$ | $8.7921 \times 10^{-1}$ | $8.7881 \times 10^{-1}$ | $8.7881 \times 10^{-1}$ | $8.7881 \times 10^{-1}$ | $8.7881 \times 10^{-1}$ | $8.7881 \times 10^{-1}$ | $8.7881 \times 10^{-1}$ |
| | | Std | $2.1679 \times 10^{-5}$ | $0.0000 \times 10^{00}$ | $0.0000 \times 10^{00}$ | $8.9181 \times 10^{-4}$ | $0.0000 \times 10^{00}$ | $0.0000 \times 10^{00}$ | $0.0000 \times 10^{00}$ | $1.5601 \times 10^{-5}$ | $0.0000 \times 10^{00}$ | $0.0000 \times 10^{00}$ |
| | 6 | Ave | $9.0001 \times 10^{-1}$ | $9.0101 \times 10^{-1}$ | $9.0162 \times 10^{-1}$ | $9.0071 \times 10^{-1}$ | $8.9979 \times 10^{-1}$ | $9.0213 \times 10^{-1}$ | $9.0137 \times 10^{-1}$ | $9.0088 \times 10^{-1}$ | $9.0020 \times 10^{-1}$ | $8.9901 \times 10^{-1}$ |
| | | Std | $2.1633 \times 10^{-3}$ | $3.2026 \times 10^{-3}$ | $3.0904 \times 10^{-3}$ | $3.6562 \times 10^{-3}$ | $3.1560 \times 10^{-3}$ | $2.7138 \times 10^{-3}$ | $2.9883 \times 10^{-3}$ | $2.5322 \times 10^{-3}$ | $2.8643 \times 10^{-3}$ | $2.0424 \times 10^{-3}$ |
| | 8 | Ave | $9.1747 \times 10^{-1}$ | $9.1613 \times 10^{-1}$ | $9.1709 \times 10^{-1}$ | $9.1369 \times 10^{-1}$ | $9.1808 \times 10^{-1}$ | $9.1790 \times 10^{-1}$ | $9.1758 \times 10^{-1}$ | $9.1568 \times 10^{-1}$ | $9.1763 \times 10^{-1}$ | $9.1776 \times 10^{-1}$ |
| | | Std | $1.2966 \times 10^{-3}$ | $3.2993 \times 10^{-3}$ | $1.7528 \times 10^{-3}$ | $4.3344 \times 10^{-3}$ | $1.7585 \times 10^{-4}$ | $4.4744 \times 10^{-4}$ | $8.4041 \times 10^{-4}$ | $3.0147 \times 10^{-3}$ | $1.0134 \times 10^{-3}$ | $5.2775 \times 10^{-4}$ |
| | 10 | Ave | $9.1943 \times 10^{-1}$ | $9.1436 \times 10^{-1}$ | $9.1785 \times 10^{-1}$ | $9.1521 \times 10^{-1}$ | $9.1806 \times 10^{-1}$ | $9.1901 \times 10^{-1}$ | $9.1959 \times 10^{-1}$ | $9.1759 \times 10^{-1}$ | $9.1853 \times 10^{-1}$ | $9.1947 \times 10^{-1}$ |
| | | Std | $2.8304 \times 10^{-3}$ | $8.9320 \times 10^{-3}$ | $4.1635 \times 10^{-3}$ | $7.8579 \times 10^{-3}$ | $1.4524 \times 10^{-4}$ | $2.2851 \times 10^{-3}$ | $2.7134 \times 10^{-3}$ | $4.4777 \times 10^{-3}$ | $4.1297 \times 10^{-3}$ | $3.2286 \times 10^{-3}$ |
| columbia | 4 | Ave | $8.1554 \times 10^{-1}$ | $8.1541 \times 10^{-1}$ | $8.1554 \times 10^{-1}$ | $8.1542 \times 10^{-1}$ | $8.1554 \times 10^{-1}$ | $8.1544 \times 10^{-1}$ | $8.1550 \times 10^{-1}$ | $8.1531 \times 10^{-1}$ | $8.1554 \times 10^{-1}$ | $8.1554 \times 10^{-1}$ |
| | | Std | $2.2584 \times 10^{-16}$ | $6.5508 \times 10^{-4}$ | $2.2584 \times 10^{-16}$ | $6.8100 \times 10^{-4}$ | $2.2584 \times 10^{-16}$ | $2.3172 \times 10^{-4}$ | $1.5510 \times 10^{-4}$ | $4.9364 \times 10^{-4}$ | $2.2584 \times 10^{-16}$ | $2.2584 \times 10^{-16}$ |
| | 6 | Ave | $8.6893 \times 10^{-1}$ | $8.6875 \times 10^{-1}$ | $8.6894 \times 10^{-1}$ | $8.6893 \times 10^{-1}$ | $8.6962 \times 10^{-1}$ | $8.6883 \times 10^{-1}$ | $8.6888 \times 10^{-1}$ | $8.6732 \times 10^{-1}$ | $8.6907 \times 10^{-1}$ | $8.6956 \times 10^{-1}$ |
| | | Std | $8.0032 \times 10^{-4}$ | $7.4952 \times 10^{-4}$ | $6.9489 \times 10^{-4}$ | $2.6061 \times 10^{-3}$ | $4.5168 \times 10^{-16}$ | $7.6014 \times 10^{-4}$ | $7.7918 \times 10^{-4}$ | $1.7120 \times 10^{-3}$ | $6.5433 \times 10^{-4}$ | $2.9061 \times 10^{-4}$ |
| | 8 | Ave | $8.9769 \times 10^{-1}$ | $8.9718 \times 10^{-1}$ | $8.9763 \times 10^{-1}$ | $8.9551 \times 10^{-1}$ | $8.9759 \times 10^{-1}$ | $8.9746 \times 10^{-1}$ | $8.9764 \times 10^{-1}$ | $8.9582 \times 10^{-1}$ | $8.9758 \times 10^{-1}$ | $8.9741 \times 10^{-1}$ |
| | | Std | $8.5901 \times 10^{-4}$ | $2.8874 \times 10^{-3}$ | $7.3566 \times 10^{-4}$ | $2.2873 \times 10^{-3}$ | $5.5705 \times 10^{-4}$ | $6.2579 \times 10^{-4}$ | $6.8143 \times 10^{-4}$ | $2.3523 \times 10^{-3}$ | $5.5947 \times 10^{-4}$ | $5.5263 \times 10^{-4}$ |
| | 10 | Ave | $9.1606 \times 10^{-1}$ | $9.1617 \times 10^{-1}$ | $9.1661 \times 10^{-1}$ | $9.1494 \times 10^{-1}$ | $9.1599 \times 10^{-1}$ | $9.1625 \times 10^{-1}$ | $9.1693 \times 10^{-1}$ | $9.1112 \times 10^{-1}$ | $9.1667 \times 10^{-1}$ | $9.1597 \times 10^{-1}$ |
| | | Std | $1.5457 \times 10^{-3}$ | $2.1005 \times 10^{-3}$ | $1.4163 \times 10^{-3}$ | $3.5849 \times 10^{-3}$ | $4.5407 \times 10^{-4}$ | $9.6993 \times 10^{-4}$ | $9.8669 \times 10^{-4}$ | $2.8319 \times 10^{-3}$ | $9.5957 \times 10^{-4}$ | $5.5185 \times 10^{-4}$ |
| couple | 4 | Ave | $8.0086 \times 10^{-1}$ | $8.0061 \times 10^{-1}$ | $8.0091 \times 10^{-1}$ | $8.0072 \times 10^{-1}$ | $8.0091 \times 10^{-1}$ | $8.0072 \times 10^{-1}$ | $8.0085 \times 10^{-1}$ | $8.0041 \times 10^{-1}$ | $8.0072 \times 10^{-1}$ | $8.0091 \times 10^{-1}$ |
| | | Std | $3.5224 \times 10^{-4}$ | $7.8442 \times 10^{-4}$ | $0.0000 \times 10^{00}$ | $1.1877 \times 10^{-3}$ | $0.0000 \times 10^{00}$ | $3.4992 \times 10^{-4}$ | $2.0637 \times 10^{-4}$ | $1.5658 \times 10^{-3}$ | $3.4992 \times 10^{-4}$ | $0.0000 \times 10^{00}$ |
| | 6 | Ave | $8.7275 \times 10^{-1}$ | $8.7306 \times 10^{-1}$ | $8.7320 \times 10^{-1}$ | $8.7232 \times 10^{-1}$ | $8.7307 \times 10^{-1}$ | $8.7332 \times 10^{-1}$ | $8.7319 \times 10^{-1}$ | $8.7148 \times 10^{-1}$ | $8.7310 \times 10^{-1}$ | $8.7307 \times 10^{-1}$ |
| | | Std | $9.7153 \times 10^{-4}$ | $5.9879 \times 10^{-4}$ | $5.1333 \times 10^{-4}$ | $1.8619 \times 10^{-3}$ | $1.4990 \times 10^{-4}$ | $4.3455 \times 10^{-4}$ | $5.5403 \times 10^{-4}$ | $2.2623 \times 10^{-3}$ | $4.7986 \times 10^{-4}$ | $1.4990 \times 10^{-4}$ |
| | 8 | Ave | $9.1112 \times 10^{-1}$ | $9.1130 \times 10^{-1}$ | $9.1157 \times 10^{-1}$ | $9.0910 \times 10^{-1}$ | $9.1166 \times 10^{-1}$ | $9.1164 \times 10^{-1}$ | $9.1176 \times 10^{-1}$ | $9.0953 \times 10^{-1}$ | $9.1195 \times 10^{-1}$ | $9.1195 \times 10^{-1}$ |
| | | Std | $1.4631 \times 10^{-3}$ | $1.6446 \times 10^{-3}$ | $8.0795 \times 10^{-4}$ | $3.2240 \times 10^{-3}$ | $3.3876 \times 10^{-16}$ | $3.8122 \times 10^{-4}$ | $8.4913 \times 10^{-4}$ | $2.9322 \times 10^{-3}$ | $7.6148 \times 10^{-4}$ | $4.2149 \times 10^{-4}$ |
| | 10 | Ave | $9.3244 \times 10^{-1}$ | $9.3252 \times 10^{-1}$ | $9.3369 \times 10^{-1}$ | $9.3039 \times 10^{-1}$ | $9.3357 \times 10^{-1}$ | $9.3399 \times 10^{-1}$ | $9.3384 \times 10^{-1}$ | $9.2926 \times 10^{-1}$ | $9.3423 \times 10^{-1}$ | $9.3395 \times 10^{-1}$ |
| | | Std | $1.5894 \times 10^{-3}$ | $3.2795 \times 10^{-3}$ | $9.8854 \times 10^{-4}$ | $2.5713 \times 10^{-3}$ | $2.6843 \times 10^{-4}$ | $4.3312 \times 10^{-4}$ | $9.1003 \times 10^{-4}$ | $3.1956 \times 10^{-3}$ | $1.1932 \times 10^{-3}$ | $7.3509 \times 10^{-4}$ |

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

| Image | Threshold | Metric | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| baboon | 4 | Ave | $7.8185 \times 10^{-1}$ | $7.8106 \times 10^{-1}$ | $7.8186 \times 10^{-1}$ | $7.8027 \times 10^{-1}$ | $7.8186 \times 10^{-1}$ | $7.8186 \times 10^{-1}$ | $7.8186 \times 10^{-1}$ | $7.8148 \times 10^{-1}$ | $7.8174 \times 10^{-1}$ | $7.8174 \times 10^{-1}$ |
| | | Std | $3.1523 \times 10^{-5}$ | $2.6359 \times 10^{-3}$ | $1.1292 \times 10^{-16}$ | $3.9493 \times 10^{-3}$ | $1.1292 \times 10^{-16}$ | $1.1292 \times 10^{-16}$ | $1.1292 \times 10^{-16}$ | $1.1906 \times 10^{-3}$ | $6.5272 \times 10^{-4}$ | $6.5272 \times 10^{-4}$ |
| | 6 | Ave | $8.5968 \times 10^{-1}$ | $8.3046 \times 10^{-1}$ | $8.5985 \times 10^{-1}$ | $8.5722 \times 10^{-1}$ | $8.5910 \times 10^{-1}$ | $8.5901 \times 10^{-1}$ | $8.5902 \times 10^{-1}$ | $8.5686 \times 10^{-1}$ | $8.5963 \times 10^{-1}$ | $8.5930 \times 10^{-1}$ |
| | | Std | $2.4962 \times 10^{-3}$ | $1.5693 \times 10^{-1}$ | $2.1350 \times 10^{-3}$ | $7.3448 \times 10^{-3}$ | $5.6460 \times 10^{-16}$ | $1.2366 \times 10^{-3}$ | $9.3989 \times 10^{-4}$ | $7.0364 \times 10^{-3}$ | $1.6212 \times 10^{-3}$ | $8.1059 \times 10^{-4}$ |
| | 8 | Ave | $9.0503 \times 10^{-1}$ | $6.6244 \times 10^{-1}$ | $9.0515 \times 10^{-1}$ | $7.2106 \times 10^{-1}$ | $9.0706 \times 10^{-1}$ | $9.0546 \times 10^{-1}$ | $8.9841 \times 10^{-1}$ | $8.9824 \times 10^{-1}$ | $9.0612 \times 10^{-1}$ | $9.0535 \times 10^{-1}$ |
| | | Std | $5.5109 \times 10^{-3}$ | $4.0625 \times 10^{-1}$ | $3.2042 \times 10^{-3}$ | $3.6321 \times 10^{-1}$ | $5.5322 \times 10^{-4}$ | $3.7134 \times 10^{-3}$ | $3.1993 \times 10^{-2}$ | $8.2449 \times 10^{-3}$ | $3.9570 \times 10^{-3}$ | $4.2691 \times 10^{-3}$ |
| | 10 | Ave | $9.3122 \times 10^{-1}$ | $3.7263 \times 10^{-1}$ | $9.3093 \times 10^{-1}$ | $5.3090 \times 10^{-1}$ | $9.3177 \times 10^{-1}$ | $9.2144 \times 10^{-1}$ | $8.7057 \times 10^{-1}$ | $9.2021 \times 10^{-1}$ | $9.0099 \times 10^{-1}$ | $9.3149 \times 10^{-1}$ |
| | | Std | $4.9990 \times 10^{-3}$ | $4.6402 \times 10^{-1}$ | $3.0462 \times 10^{-3}$ | $4.6242 \times 10^{-1}$ | $7.7762 \times 10^{-4}$ | $5.4565 \times 10^{-2}$ | $1.4609 \times 10^{-1}$ | $8.3495 \times 10^{-3}$ | $1.3361 \times 10^{-1}$ | $2.6996 \times 10^{-3}$ |
| bank | 4 | Ave | $7.4549 \times 10^{-1}$ | $7.4525 \times 10^{-1}$ | $7.4554 \times 10^{-1}$ | $7.4482 \times 10^{-1}$ | $7.4554 \times 10^{-1}$ | $7.4506 \times 10^{-1}$ | $7.4551 \times 10^{-1}$ | $7.4487 \times 10^{-1}$ | $7.4550 \times 10^{-1}$ | $7.4554 \times 10^{-1}$ |
| | | Std | $2.0501 \times 10^{-4}$ | $6.1294 \times 10^{-4}$ | $1.3683 \times 10^{-5}$ | $9.4303 \times 10^{-4}$ | $2.2584 \times 10^{-16}$ | $1.4940 \times 10^{-3}$ | $1.9667 \times 10^{-4}$ | $1.5980 \times 10^{-3}$ | $1.9658 \times 10^{-4}$ | $1.3683 \times 10^{-5}$ |
| | 6 | Ave | $8.1464 \times 10^{-1}$ | $8.1541 \times 10^{-1}$ | $8.1490 \times 10^{-1}$ | $8.1409 \times 10^{-1}$ | $8.1456 \times 10^{-1}$ | $8.1494 \times 10^{-1}$ | $8.1468 \times 10^{-1}$ | $8.1395 \times 10^{-1}$ | $8.1429 \times 10^{-1}$ | $8.1456 \times 10^{-1}$ |
| | | Std | $1.0313 \times 10^{-3}$ | $2.8049 \times 10^{-3}$ | $7.2130 \times 10^{-4}$ | $3.8865 \times 10^{-3}$ | $1.1292 \times 10^{-16}$ | $5.9255 \times 10^{-4}$ | $4.8826 \times 10^{-4}$ | $3.1549 \times 10^{-3}$ | $6.6096 \times 10^{-4}$ | $1.1292 \times 10^{-16}$ |
| | 8 | Ave | $8.5602 \times 10^{-1}$ | $8.5592 \times 10^{-1}$ | $8.5631 \times 10^{-1}$ | $8.5455 \times 10^{-1}$ | $8.5621 \times 10^{-1}$ | $8.5760 \times 10^{-1}$ | $8.5708 \times 10^{-1}$ | $8.5669 \times 10^{-1}$ | $8.5699 \times 10^{-1}$ | $8.5663 \times 10^{-1}$ |
| | | Std | $2.2378 \times 10^{-3}$ | $4.9149 \times 10^{-3}$ | $1.6701 \times 10^{-3}$ | $6.1437 \times 10^{-3}$ | $1.2856 \times 10^{-3}$ | $3.4487 \times 10^{-3}$ | $4.2876 \times 10^{-3}$ | $5.3954 \times 10^{-3}$ | $3.4577 \times 10^{-3}$ | $1.8104 \times 10^{-3}$ |
| | 10 | Ave | $8.8543 \times 10^{-1}$ | $8.8429 \times 10^{-1}$ | $8.8525 \times 10^{-1}$ | $8.8227 \times 10^{-1}$ | $8.8579 \times 10^{-1}$ | $8.8572 \times 10^{-1}$ | $8.8568 \times 10^{-1}$ | $8.7981 \times 10^{-1}$ | $8.8474 \times 10^{-1}$ | $8.8595 \times 10^{-1}$ |
| | | Std | $2.3913 \times 10^{-3}$ | $4.3030 \times 10^{-3}$ | $1.5559 \times 10^{-3}$ | $4.7059 \times 10^{-3}$ | $3.4508 \times 10^{-4}$ | $3.1034 \times 10^{-3}$ | $2.9192 \times 10^{-3}$ | $5.1919 \times 10^{-3}$ | $4.3061 \times 10^{-3}$ | $1.3422 \times 10^{-3}$ |
| barbara | 4 | Ave | $6.5794 \times 10^{-1}$ | $6.5793 \times 10^{-1}$ | $6.5794 \times 10^{-1}$ | $6.5800 \times 10^{-1}$ | $6.5794 \times 10^{-1}$ | $6.5794 \times 10^{-1}$ | $6.5794 \times 10^{-1}$ | $6.5797 \times 10^{-1}$ | $6.5794 \times 10^{-1}$ | $6.5794 \times 10^{-1}$ |
| | | Std | $0.0000 \times 10^{00}$ | $5.9756 \times 10^{-5}$ | $0.0000 \times 10^{00}$ | $6.2162 \times 10^{-4}$ | $0.0000 \times 10^{00}$ | $0.0000 \times 10^{00}$ | $0.0000 \times 10^{00}$ | $5.4503 \times 10^{-4}$ | $0.0000 \times 10^{00}$ | $0.0000 \times 10^{00}$ |
| | 6 | Ave | $7.3958 \times 10^{-1}$ | $7.4054 \times 10^{-1}$ | $7.3965 \times 10^{-1}$ | $7.3843 \times 10^{-1}$ | $7.3936 \times 10^{-1}$ | $7.3945 \times 10^{-1}$ | $7.3986 \times 10^{-1}$ | $7.3887 \times 10^{-1}$ | $7.3963 \times 10^{-1}$ | $7.3936 \times 10^{-1}$ |
| | | Std | $1.2771 \times 10^{-3}$ | $2.4713 \times 10^{-3}$ | $9.2488 \times 10^{-4}$ | $6.0913 \times 10^{-3}$ | $5.6460 \times 10^{-16}$ | $4.2145 \times 10^{-4}$ | $8.4074 \times 10^{-4}$ | $4.8124 \times 10^{-3}$ | $7.4454 \times 10^{-4}$ | $5.6460 \times 10^{-16}$ |
| | 8 | Ave | $7.9937 \times 10^{-1}$ | $8.0116 \times 10^{-1}$ | $7.9966 \times 10^{-1}$ | $7.9523 \times 10^{-1}$ | $8.0132 \times 10^{-1}$ | $8.0050 \times 10^{-1}$ | $8.0136 \times 10^{-1}$ | $7.9293 \times 10^{-1}$ | $7.9969 \times 10^{-1}$ | $8.0097 \times 10^{-1}$ |
| | | Std | $3.6565 \times 10^{-3}$ | $7.0809 \times 10^{-3}$ | $2.6070 \times 10^{-3}$ | $9.6245 \times 10^{-3}$ | $5.6460 \times 10^{-16}$ | $1.7136 \times 10^{-3}$ | $3.3587 \times 10^{-3}$ | $8.5137 \times 10^{-3}$ | $2.9981 \times 10^{-3}$ | $1.2984 \times 10^{-3}$ |
| | 10 | Ave | $8.3800 \times 10^{-1}$ | $8.3735 \times 10^{-1}$ | $8.3810 \times 10^{-1}$ | $8.3542 \times 10^{-1}$ | $8.3855 \times 10^{-1}$ | $8.3839 \times 10^{-1}$ | $8.3757 \times 10^{-1}$ | $8.2864 \times 10^{-1}$ | $8.3660 \times 10^{-1}$ | $8.3870 \times 10^{-1}$ |
| | | Std | $5.3098 \times 10^{-3}$ | $6.4311 \times 10^{-3}$ | $4.3157 \times 10^{-3}$ | $9.0808 \times 10^{-3}$ | $1.5144 \times 10^{-3}$ | $3.2208 \times 10^{-3}$ | $5.4654 \times 10^{-3}$ | $1.2515 \times 10^{-2}$ | $6.9074 \times 10^{-3}$ | $1.9804 \times 10^{-3}$ |
| boat | 4 | Ave | $7.0743 \times 10^{-1}$ | $7.0730 \times 10^{-1}$ | $7.0743 \times 10^{-1}$ | $7.0660 \times 10^{-1}$ | $7.0743 \times 10^{-1}$ | $7.0743 \times 10^{-1}$ | $7.0743 \times 10^{-1}$ | $7.0714 \times 10^{-1}$ | $7.0743 \times 10^{-1}$ | $7.0743 \times 10^{-1}$ |
| | | Std | $1.2738 \times 10^{-5}$ | $5.2107 \times 10^{-4}$ | $3.3876 \times 10^{-16}$ | $1.8333 \times 10^{-3}$ | $3.3876 \times 10^{-16}$ | $3.3876 \times 10^{-16}$ | $3.3876 \times 10^{-16}$ | $6.6292 \times 10^{-4}$ | $3.3876 \times 10^{-16}$ | $3.3876 \times 10^{-16}$ |
| | 6 | Ave | $8.0498 \times 10^{-1}$ | $8.0527 \times 10^{-1}$ | $8.0552 \times 10^{-1}$ | $8.0286 \times 10^{-1}$ | $8.0628 \times 10^{-1}$ | $8.0525 \times 10^{-1}$ | $8.0535 \times 10^{-1}$ | $8.0416 \times 10^{-1}$ | $8.0506 \times 10^{-1}$ | $8.0609 \times 10^{-1}$ |
| | | Std | $1.9007 \times 10^{-3}$ | $2.1358 \times 10^{-3}$ | $1.1789 \times 10^{-3}$ | $6.7234 \times 10^{-3}$ | $5.6460 \times 10^{-16}$ | $1.5983 \times 10^{-3}$ | $1.7772 \times 10^{-3}$ | $3.0217 \times 10^{-3}$ | $1.3594 \times 10^{-3}$ | $4.0263 \times 10^{-4}$ |
| | 8 | Ave | $8.5846 \times 10^{-1}$ | $8.5916 \times 10^{-1}$ | $8.5930 \times 10^{-1}$ | $8.5935 \times 10^{-1}$ | $8.6035 \times 10^{-1}$ | $8.5990 \times 10^{-1}$ | $8.6039 \times 10^{-1}$ | $8.5396 \times 10^{-1}$ | $8.5976 \times 10^{-1}$ | $8.5936 \times 10^{-1}$ |
| | | Std | $3.1528 \times 10^{-3}$ | $4.7921 \times 10^{-3}$ | $1.7430 \times 10^{-3}$ | $8.3239 \times 10^{-3}$ | $1.4461 \times 10^{-3}$ | $1.6842 \times 10^{-3}$ | $2.1554 \times 10^{-3}$ | $8.4787 \times 10^{-3}$ | $3.0585 \times 10^{-3}$ | $1.6997 \times 10^{-3}$ |
| | 10 | Ave | $8.9548 \times 10^{-1}$ | $8.9411 \times 10^{-1}$ | $8.9473 \times 10^{-1}$ | $8.9241 \times 10^{-1}$ | $8.9764 \times 10^{-1}$ | $8.9724 \times 10^{-1}$ | $8.9519 \times 10^{-1}$ | $8.8677 \times 10^{-1}$ | $8.9337 \times 10^{-1}$ | $8.9528 \times 10^{-1}$ |
| | | Std | $5.6167 \times 10^{-3}$ | $6.5833 \times 10^{-3}$ | $3.2530 \times 10^{-3}$ | $8.0224 \times 10^{-3}$ | $7.2536 \times 10^{-4}$ | $2.0843 \times 10^{-3}$ | $3.0822 \times 10^{-3}$ | $7.8028 \times 10^{-3}$ | $5.2980 \times 10^{-3}$ | $3.2774 \times 10^{-3}$ |
| brain | 4 | Ave | $3.7684 \times 10^{-1}$ | $3.7677 \times 10^{-1}$ | $3.7674 \times 10^{-1}$ | $3.7676 \times 10^{-1}$ | $3.7673 \times 10^{-1}$ | $3.7674 \times 10^{-1}$ | $3.7669 \times 10^{-1}$ | $3.7697 \times 10^{-1}$ | $3.7668 \times 10^{-1}$ | $3.7679 \times 10^{-1}$ |
| | | Std | $3.4561 \times 10^{-4}$ | $3.3560 \times 10^{-4}$ | $3.9361 \times 10^{-4}$ | $5.0207 \times 10^{-4}$ | $3.6593 \times 10^{-4}$ | $3.5373 \times 10^{-4}$ | $3.7657 \times 10^{-4}$ | $4.5989 \times 10^{-4}$ | $3.3902 \times 10^{-4}$ | $2.9604 \times 10^{-4}$ |
| | 6 | Ave | $6.5865 \times 10^{-1}$ | $4.9226 \times 10^{-1}$ | $4.5444 \times 10^{-1}$ | $6.9139 \times 10^{-1}$ | $4.3883 \times 10^{-1}$ | $5.0737 \times 10^{-1}$ | $6.1940 \times 10^{-1}$ | $5.8723 \times 10^{-1}$ | $5.4933 \times 10^{-1}$ | $5.8427 \times 10^{-1}$ |
| | | Std | $5.4732 \times 10^{-2}$ | $1.3098 \times 10^{-1}$ | $1.0197 \times 10^{-1}$ | $5.7404 \times 10^{-2}$ | $8.8712 \times 10^{-2}$ | $1.2349 \times 10^{-1}$ | $8.7389 \times 10^{-2}$ | $1.1123 \times 10^{-1}$ | $1.4390 \times 10^{-1}$ | $1.1513 \times 10^{-1}$ |
| | 8 | Ave | $7.6612 \times 10^{-1}$ | $7.2601 \times 10^{-1}$ | $7.6325 \times 10^{-1}$ | $7.4042 \times 10^{-1}$ | $7.6843 \times 10^{-1}$ | $7.4909 \times 10^{-1}$ | $7.6147 \times 10^{-1}$ | $6.9011 \times 10^{-1}$ | $7.4929 \times 10^{-1}$ | $7.6283 \times 10^{-1}$ |
| | | Std | $3.1178 \times 10^{-2}$ | $7.8746 \times 10^{-2}$ | $2.6351 \times 10^{-2}$ | $5.4074 \times 10^{-2}$ | $2.5880 \times 10^{-2}$ | $6.7023 \times 10^{-2}$ | $2.6416 \times 10^{-2}$ | $6.5521 \times 10^{-2}$ | $3.6423 \times 10^{-2}$ | $3.1330 \times 10^{-2}$ |
| | 10 | Ave | $7.6754 \times 10^{-1}$ | $7.7545 \times 10^{-1}$ | $7.7465 \times 10^{-1}$ | $7.7925 \times 10^{-1}$ | $7.7124 \times 10^{-1}$ | $7.7659 \times 10^{-1}$ | $7.8460 \times 10^{-1}$ | $7.4420 \times 10^{-1}$ | $7.7966 \times 10^{-1}$ | $7.9650 \times 10^{-1}$ |
| | | Std | $2.8464 \times 10^{-2}$ | $5.0717 \times 10^{-2}$ | $2.9104 \times 10^{-2}$ | $5.6951 \times 10^{-2}$ | $2.6333 \times 10^{-2}$ | $3.0514 \times 10^{-2}$ | $3.1401 \times 10^{-2}$ | $6.2877 \times 10^{-2}$ | $3.1633 \times 10^{-2}$ | $3.6746 \times 10^{-2}$ |

**Table 14.** *Cont.*

| Image | Threshold | Metric | BHO | NDO | GJA | SSLO | LEE | SAO | RBMO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| bridge | 4 | Ave | $6.9311 \times 10^{-1}$ | $6.9389 \times 10^{-1}$ | $6.9311 \times 10^{-1}$ | $6.9384 \times 10^{-1}$ | $6.9311 \times 10^{-1}$ | $6.9311 \times 10^{-1}$ | $6.9311 \times 10^{-1}$ | $6.9333 \times 10^{-1}$ | $6.9311 \times 10^{-1}$ | $6.9311 \times 10^{-1}$ |
| | | Std | $6.8714 \times 10^{-6}$ | $2.1949 \times 10^{-3}$ | $1.1292 \times 10^{-16}$ | $2.9091 \times 10^{-3}$ | $1.1292 \times 10^{-16}$ | $1.1292 \times 10^{-16}$ | $1.1292 \times 10^{-16}$ | $3.1094 \times 10^{-3}$ | $1.1292 \times 10^{-16}$ | $1.1292 \times 10^{-16}$ |
| | 6 | Ave | $8.1543 \times 10^{-1}$ | $8.1598 \times 10^{-1}$ | $8.1564 \times 10^{-1}$ | $8.1535 \times 10^{-1}$ | $8.1527 \times 10^{-1}$ | $8.1480 \times 10^{-1}$ | $8.1499 \times 10^{-1}$ | $8.1325 \times 10^{-1}$ | $8.1539 \times 10^{-1}$ | $8.1521 \times 10^{-1}$ |
| | | Std | $2.7597 \times 10^{-3}$ | $4.9376 \times 10^{-3}$ | $1.1678 \times 10^{-3}$ | $7.7628 \times 10^{-3}$ | $3.3876 \times 10^{-16}$ | $9.5752 \times 10^{-4}$ | $1.8773 \times 10^{-3}$ | $8.5453 \times 10^{-3}$ | $1.1698 \times 10^{-3}$ | $8.0551 \times 10^{-4}$ |
| | 8 | Ave | $8.7742 \times 10^{-1}$ | $8.7462 \times 10^{-1}$ | $8.7817 \times 10^{-1}$ | $8.7393 \times 10^{-1}$ | $8.7827 \times 10^{-1}$ | $8.7813 \times 10^{-1}$ | $8.7677 \times 10^{-1}$ | $8.7333 \times 10^{-1}$ | $8.7760 \times 10^{-1}$ | $8.7826 \times 10^{-1}$ |
| | | Std | $3.8828 \times 10^{-3}$ | $6.8789 \times 10^{-3}$ | $2.2062 \times 10^{-3}$ | $1.1602 \times 10^{-2}$ | $1.2072 \times 10^{-3}$ | $1.9286 \times 10^{-3}$ | $4.5428 \times 10^{-3}$ | $1.0762 \times 10^{-2}$ | $3.7647 \times 10^{-3}$ | $2.0696 \times 10^{-3}$ |
| | 10 | Ave | $9.1302 \times 10^{-1}$ | $9.1079 \times 10^{-1}$ | $9.1343 \times 10^{-1}$ | $9.1276 \times 10^{-1}$ | $9.1556 \times 10^{-1}$ | $9.1558 \times 10^{-1}$ | $9.1392 \times 10^{-1}$ | $9.0773 \times 10^{-1}$ | $9.1386 \times 10^{-1}$ | $9.1584 \times 10^{-1}$ |
| | | Std | $5.8384 \times 10^{-3}$ | $9.4939 \times 10^{-3}$ | $4.2564 \times 10^{-3}$ | $6.8804 \times 10^{-3}$ | $2.4738 \times 10^{-3}$ | $7.6182 \times 10^{-3}$ | $6.4535 \times 10^{-3}$ | $9.3464 \times 10^{-3}$ | $7.2528 \times 10^{-3}$ | $5.3954 \times 10^{-3}$ |
| camera | 4 | Ave | $7.5084 \times 10^{-1}$ | $7.2713 \times 10^{-1}$ | $7.2125 \times 10^{-1}$ | $7.5844 \times 10^{-1}$ | $7.3304 \times 10^{-1}$ | $6.9808 \times 10^{-1}$ | $7.5913 \times 10^{-1}$ | $7.5152 \times 10^{-1}$ | $7.4483 \times 10^{-1}$ | $7.5847 \times 10^{-1}$ |
| | | Std | $2.1361 \times 10^{-2}$ | $3.7741 \times 10^{-2}$ | $3.5908 \times 10^{-2}$ | $5.1520 \times 10^{-3}$ | $3.4685 \times 10^{-2}$ | $3.6565 \times 10^{-2}$ | $5.3832 \times 10^{-4}$ | $1.3903 \times 10^{-2}$ | $2.8790 \times 10^{-2}$ | $2.0595 \times 10^{-3}$ |
| | 6 | Ave | $8.0320 \times 10^{-1}$ | $8.0354 \times 10^{-1}$ | $8.0382 \times 10^{-1}$ | $7.9953 \times 10^{-1}$ | $8.0543 \times 10^{-1}$ | $8.0490 \times 10^{-1}$ | $8.0455 \times 10^{-1}$ | $8.0132 \times 10^{-1}$ | $8.0421 \times 10^{-1}$ | $8.0456 \times 10^{-1}$ |
| | | Std | $2.6642 \times 10^{-3}$ | $4.3932 \times 10^{-3}$ | $2.0994 \times 10^{-3}$ | $1.0309 \times 10^{-2}$ | $1.1292 \times 10^{-16}$ | $1.0898 \times 10^{-3}$ | $1.3317 \times 10^{-3}$ | $5.3853 \times 10^{-3}$ | $1.7152 \times 10^{-3}$ | $1.6905 \times 10^{-3}$ |
| | 8 | Ave | $8.3094 \times 10^{-1}$ | $8.3263 \times 10^{-1}$ | $8.3453 \times 10^{-1}$ | $8.2704 \times 10^{-1}$ | $8.3668 \times 10^{-1}$ | $8.3532 \times 10^{-1}$ | $8.3372 \times 10^{-1}$ | $8.2422 \times 10^{-1}$ | $8.3549 \times 10^{-1}$ | $8.3463 \times 10^{-1}$ |
| | | Std | $5.0436 \times 10^{-3}$ | $4.5118 \times 10^{-3}$ | $3.8335 \times 10^{-3}$ | $1.2769 \times 10^{-2}$ | $3.1533 \times 10^{-3}$ | $3.2502 \times 10^{-3}$ | $3.8801 \times 10^{-3}$ | $1.2113 \times 10^{-2}$ | $4.6657 \times 10^{-3}$ | $3.5981 \times 10^{-3}$ |
| | 10 | Ave | $8.6095 \times 10^{-1}$ | $8.5745 \times 10^{-1}$ | $8.6033 \times 10^{-1}$ | $8.5849 \times 10^{-1}$ | $8.6226 \times 10^{-1}$ | $8.6026 \times 10^{-1}$ | $8.5886 \times 10^{-1}$ | $8.4937 \times 10^{-1}$ | $8.6065 \times 10^{-1}$ | $8.5987 \times 10^{-1}$ |
| | | Std | $6.0346 \times 10^{-3}$ | $9.8191 \times 10^{-3}$ | $5.7288 \times 10^{-3}$ | $1.1529 \times 10^{-2}$ | $3.4242 \times 10^{-3}$ | $4.2943 \times 10^{-3}$ | $6.3991 \times 10^{-3}$ | $1.3199 \times 10^{-2}$ | $6.1487 \times 10^{-3}$ | $5.4403 \times 10^{-3}$ |
| cell | 4 | Ave | $7.2796 \times 10^{-1}$ | $7.2795 \times 10^{-1}$ | $7.2795 \times 10^{-1}$ | $7.2856 \times 10^{-1}$ | $7.2795 \times 10^{-1}$ | $7.2795 \times 10^{-1}$ | $7.2795 \times 10^{-1}$ | $7.2795 \times 10^{-1}$ | $7.2795 \times 10^{-1}$ | $7.2795 \times 10^{-1}$ |
| | | Std | $3.1833 \times 10^{-5}$ | $2.2584 \times 10^{-16}$ | $2.2584 \times 10^{-16}$ | $9.9055 \times 10^{-4}$ | $2.2584 \times 10^{-16}$ | $2.2584 \times 10^{-16}$ | $2.2584 \times 10^{-16}$ | $2.2908 \times 10^{-5}$ | $2.2584 \times 10^{-16}$ | $2.2584 \times 10^{-16}$ |
| | 6 | Ave | $7.3140 \times 10^{-1}$ | $7.2937 \times 10^{-1}$ | $7.2630 \times 10^{-1}$ | $7.3443 \times 10^{-1}$ | $7.3062 \times 10^{-1}$ | $7.2351 \times 10^{-1}$ | $7.2319 \times 10^{-1}$ | $7.2436 \times 10^{-1}$ | $7.2283 \times 10^{-1}$ | $7.2592 \times 10^{-1}$ |
| | | Std | $2.0467 \times 10^{-2}$ | $1.9922 \times 10^{-2}$ | $2.1215 \times 10^{-2}$ | $1.9707 \times 10^{-2}$ | $1.6421 \times 10^{-2}$ | $2.1134 \times 10^{-2}$ | $2.2424 \times 10^{-2}$ | $2.4677 \times 10^{-2}$ | $2.1467 \times 10^{-2}$ | $2.0974 \times 10^{-2}$ |
| | 8 | Ave | $7.7368 \times 10^{-1}$ | $7.7089 \times 10^{-1}$ | $7.7527 \times 10^{-1}$ | $7.6453 \times 10^{-1}$ | $7.7464 \times 10^{-1}$ | $7.7258 \times 10^{-1}$ | $7.7239 \times 10^{-1}$ | $7.7089 \times 10^{-1}$ | $7.7513 \times 10^{-1}$ | $7.7495 \times 10^{-1}$ |
| | | Std | $9.7507 \times 10^{-3}$ | $1.2523 \times 10^{-2}$ | $5.4677 \times 10^{-3}$ | $1.5869 \times 10^{-2}$ | $7.6730 \times 10^{-4}$ | $7.8557 \times 10^{-3}$ | $9.2329 \times 10^{-3}$ | $1.4845 \times 10^{-2}$ | $4.7834 \times 10^{-3}$ | $2.6755 \times 10^{-3}$ |
| | 10 | Ave | $8.0444 \times 10^{-1}$ | $8.0296 \times 10^{-1}$ | $8.0596 \times 10^{-1}$ | $8.0411 \times 10^{-1}$ | $8.1015 \times 10^{-1}$ | $8.0610 \times 10^{-1}$ | $8.0479 \times 10^{-1}$ | $7.9724 \times 10^{-1}$ | $8.0680 \times 10^{-1}$ | $8.0915 \times 10^{-1}$ |
| | | Std | $6.8750 \times 10^{-3}$ | $9.3244 \times 10^{-3}$ | $7.9107 \times 10^{-3}$ | $1.0840 \times 10^{-2}$ | $8.0015 \times 10^{-5}$ | $7.2430 \times 10^{-3}$ | $6.7964 \times 10^{-3}$ | $1.0737 \times 10^{-2}$ | $6.5355 \times 10^{-3}$ | $3.5475 \times 10^{-3}$ |
| columbia | 4 | Ave | $7.1971 \times 10^{-1}$ | $7.1987 \times 10^{-1}$ | $7.1971 \times 10^{-1}$ | $7.1955 \times 10^{-1}$ | $7.1971 \times 10^{-1}$ | $7.1978 \times 10^{-1}$ | $7.1974 \times 10^{-1}$ | $7.1921 \times 10^{-1}$ | $7.1971 \times 10^{-1}$ | $7.1971 \times 10^{-1}$ |
| | | Std | $1.1292 \times 10^{-16}$ | $2.4193 \times 10^{-3}$ | $1.1292 \times 10^{-16}$ | $3.6586 \times 10^{-3}$ | $1.1292 \times 10^{-16}$ | $1.6996 \times 10^{-4}$ | $1.1376 \times 10^{-4}$ | $2.2743 \times 10^{-3}$ | $1.1292 \times 10^{-16}$ | $1.1292 \times 10^{-16}$ |
| | 6 | Ave | $7.9306 \times 10^{-1}$ | $7.9332 \times 10^{-1}$ | $7.9348 \times 10^{-1}$ | $7.9570 \times 10^{-1}$ | $7.9375 \times 10^{-1}$ | $7.9431 \times 10^{-1}$ | $7.9372 \times 10^{-1}$ | $7.9030 \times 10^{-1}$ | $7.9359 \times 10^{-1}$ | $7.9377 \times 10^{-1}$ |
| | | Std | $1.8474 \times 10^{-3}$ | $4.4889 \times 10^{-3}$ | $1.5345 \times 10^{-3}$ | $8.1408 \times 10^{-3}$ | $3.3876 \times 10^{-16}$ | $1.4197 \times 10^{-3}$ | $1.8446 \times 10^{-3}$ | $6.3202 \times 10^{-3}$ | $2.1375 \times 10^{-3}$ | $8.2384 \times 10^{-5}$ |
| | 8 | Ave | $8.3752 \times 10^{-1}$ | $8.3449 \times 10^{-1}$ | $8.3709 \times 10^{-1}$ | $8.3449 \times 10^{-1}$ | $8.3738 \times 10^{-1}$ | $8.3658 \times 10^{-1}$ | $8.3666 \times 10^{-1}$ | $8.3439 \times 10^{-1}$ | $8.3669 \times 10^{-1}$ | $8.3766 \times 10^{-1}$ |
| | | Std | $2.9757 \times 10^{-3}$ | $7.5153 \times 10^{-3}$ | $2.0112 \times 10^{-3}$ | $8.6325 \times 10^{-3}$ | $1.1565 \times 10^{-3}$ | $2.4140 \times 10^{-3}$ | $3.4237 \times 10^{-3}$ | $7.3768 \times 10^{-3}$ | $2.7579 \times 10^{-3}$ | $1.1175 \times 10^{-3}$ |
| | 10 | Ave | $8.6142 \times 10^{-1}$ | $8.6015 \times 10^{-1}$ | $8.6045 \times 10^{-1}$ | $8.6532 \times 10^{-1}$ | $8.5949 \times 10^{-1}$ | $8.6136 \times 10^{-1}$ | $8.5906 \times 10^{-1}$ | $8.5420 \times 10^{-1}$ | $8.6038 \times 10^{-1}$ | $8.6105 \times 10^{-1}$ |
| | | Std | $4.8179 \times 10^{-3}$ | $6.0141 \times 10^{-3}$ | $4.1734 \times 10^{-3}$ | $1.2912 \times 10^{-2}$ | $9.3868 \times 10^{-4}$ | $4.9533 \times 10^{-3}$ | $3.7042 \times 10^{-3}$ | $6.3466 \times 10^{-3}$ | $5.2010 \times 10^{-3}$ | $2.7178 \times 10^{-3}$ |
| couple | 4 | Ave | $7.2971 \times 10^{-1}$ | $7.2885 \times 10^{-1}$ | $7.2991 \times 10^{-1}$ | $7.2893 \times 10^{-1}$ | $7.2991 \times 10^{-1}$ | $7.2918 \times 10^{-1}$ | $7.2970 \times 10^{-1}$ | $7.2893 \times 10^{-1}$ | $7.2918 \times 10^{-1}$ | $7.2991 \times 10^{-1}$ |
| | | Std | $1.2007 \times 10^{-3}$ | $2.5780 \times 10^{-3}$ | $3.3876 \times 10^{-16}$ | $2.8759 \times 10^{-3}$ | $3.3876 \times 10^{-16}$ | $1.3475 \times 10^{-3}$ | $7.9470 \times 10^{-4}$ | $3.3411 \times 10^{-3}$ | $1.3475 \times 10^{-3}$ | $3.3876 \times 10^{-16}$ |
| | 6 | Ave | $8.3226 \times 10^{-1}$ | $8.3280 \times 10^{-1}$ | $8.3314 \times 10^{-1}$ | $8.3099 \times 10^{-1}$ | $8.3313 \times 10^{-1}$ | $8.3343 \times 10^{-1}$ | $8.3323 \times 10^{-1}$ | $8.3053 \times 10^{-1}$ | $8.3319 \times 10^{-1}$ | $8.3313 \times 10^{-1}$ |
| | | Std | $1.5334 \times 10^{-3}$ | $1.2932 \times 10^{-3}$ | $7.4539 \times 10^{-4}$ | $4.3889 \times 10^{-3}$ | $2.0419 \times 10^{-4}$ | $7.3568 \times 10^{-4}$ | $9.8343 \times 10^{-4}$ | $4.5662 \times 10^{-3}$ | $6.1716 \times 10^{-4}$ | $2.0419 \times 10^{-4}$ |
| | 8 | Ave | $8.7659 \times 10^{-1}$ | $8.7639 \times 10^{-1}$ | $8.7699 \times 10^{-1}$ | $8.7215 \times 10^{-1}$ | $8.7783 \times 10^{-1}$ | $8.7766 \times 10^{-1}$ | $8.7734 \times 10^{-1}$ | $8.7503 \times 10^{-1}$ | $8.7760 \times 10^{-1}$ | $8.7778 \times 10^{-1}$ |
| | | Std | $1.5501 \times 10^{-3}$ | $3.5710 \times 10^{-3}$ | $7.5826 \times 10^{-4}$ | $4.7311 \times 10^{-3}$ | $3.3876 \times 10^{-16}$ | $5.6560 \times 10^{-4}$ | $8.6225 \times 10^{-4}$ | $3.8435 \times 10^{-3}$ | $5.0269 \times 10^{-4}$ | $3.0561 \times 10^{-4}$ |
| | 10 | Ave | $9.0309 \times 10^{-1}$ | $9.0306 \times 10^{-1}$ | $9.0372 \times 10^{-1}$ | $9.0017 \times 10^{-1}$ | $9.0398 \times 10^{-1}$ | $9.0429 \times 10^{-1}$ | $9.0488 \times 10^{-1}$ | $9.0000 \times 10^{-1}$ | $9.0460 \times 10^{-1}$ | $9.0378 \times 10^{-1}$ |
| | | Std | $1.6416 \times 10^{-3}$ | $3.7762 \times 10^{-3}$ | $7.0856 \times 10^{-4}$ | $3.5116 \times 10^{-3}$ | $1.6428 \times 10^{-4}$ | $9.0869 \times 10^{-4}$ | $1.5499 \times 10^{-3}$ | $4.1125 \times 10^{-3}$ | $1.4444 \times 10^{-3}$ | $4.4981 \times 10^{-4}$ |

**Table 15.** Comparison of running times of various algorithms.

| Image/Algorithm | GJA | SSLO | ESC | SBOA | MISBOA |
|---|---|---|---|---|---|
| baboon | 26.09 | 22.05 | 40.74 | 22.71 | 23.77 |
| bank | 26.20 | 24.49 | 40.80 | 22.15 | 22.85 |
| barbara | 26.50 | 24.78 | 41.32 | 22.46 | 23.05 |
| boat | 26.89 | 24.99 | 41.95 | 22.72 | 23.51 |
| brain | 26.19 | 24.54 | 40.88 | 22.21 | 22.90 |
| bridge | 44.81 | 41.15 | 73.94 | 36.27 | 38.05 |
| camera | 26.12 | 24.41 | 40.72 | 22.07 | 22.68 |
| cell | 84.37 | 76.88 | 152.04 | 65.68 | 69.91 |
| columbia | 26.26 | 24.52 | 40.93 | 22.14 | 22.82 |
| couple | 26.82 | 25.04 | 41.45 | 22.73 | 23.47 |

## 5. Conclusions

In this paper, a Multi-Strategy Improved Secretary Bird Optimization Algorithm (MISBOA) is proposed to overcome the shortcomings of the original SBOA in terms of insufficient population diversity, weak local exploitation ability, and premature convergence. Three effective improvement strategies, namely chaotic elite initialization, adaptive spiral Lévy flight, and dynamic neighborhood-guided mutation, are incorporated into the original SBOA framework. These strategies significantly improve the balance between global exploration and local exploitation, enhance the ability to escape local optima, and strengthen the convergence accuracy and robustness of the algorithm.

To verify the performance of MISBOA, extensive experiments are conducted on the IEEE CEC2014, CEC2017, and CEC2020 benchmark suites. Compared with several advanced metaheuristic algorithms and representative CEC competition-winning algorithms, MISBOA achieves superior optimization accuracy, faster convergence speed, and stronger stability on most benchmark functions. In particular, MISBOA demonstrates highly competitive performance on complex multimodal, hybrid, and composition functions, indicating that the proposed strategies can effectively improve both exploration capability and exploitation performance. Furthermore, the convergence curves, boxplot analysis, statistical tests, strategy effectiveness analysis, and parameter sensitivity analysis consistently demonstrate that MISBOA possesses strong robustness, stable search behavior, and reliable optimization capability.

Furthermore, MISBOA is successfully applied to Otsu-based multilevel threshold image segmentation. Experimental results on multiple benchmark artistic images show that MISBOA can obtain higher objective fitness values and superior image quality metrics, including PSNR, FSIM, and SSIM. The segmented images also exhibit clearer object boundaries, richer detail preservation, and better visual effects. These findings confirm that MISBOA has strong applicability and effectiveness in solving complex artistic image segmentation problems.

Although MISBOA demonstrates promising optimization performance, several limitations still exist. First, the introduction of multiple enhancement strategies inevitably increases the computational complexity and computational overhead of the algorithm compared with the original SBOA. In particular, the dynamic neighborhood-guided mutation strategy requires additional neighborhood interaction calculations, which may increase the computational burden when solving high-dimensional optimization problems. Second, some control parameters in MISBOA are still empirically determined, and different optimization problems may require different parameter settings to achieve optimal performance. Therefore, the adaptive parameter adjustment capability of the algorithm still requires further investigation. Third, the current work mainly focuses on continuous numerical optimization and Otsu-based multilevel threshold image segmentation prob-

lems, while the performance of MISBOA on other challenging optimization scenarios, such as constrained optimization, multi-objective optimization, dynamic optimization, and discrete combinatorial optimization problems, has not yet been fully explored. In addition, although MISBOA exhibits strong optimization performance experimentally, deeper theoretical analysis and strict mathematical convergence proofs still remain insufficient.

Therefore, future work may focus on reducing the computational burden through adaptive neighborhood selection, parallel computing, or lightweight update mechanisms. In addition, MISBOA can be further extended to solve more real-world optimization problems, such as feature selection, engineering design, cloud task scheduling, wireless sensor network deployment, and UAV path planning. Future studies may also investigate adaptive parameter control mechanisms and theoretical convergence analysis to further improve the interpretability, scalability, and practical applicability of the proposed algorithm.

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

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

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

## References

1. Xu, W.; Fu, Y.-L.; Zhu, D. ResNet and Its Application to Medical Image Processing: Research Progress and Challenges. *Comput. Methods Programs Biomed.* **2023**, *240*, 107660. [CrossRef]
2. Li, J.; Bhatti, U.A.; Nawaz, S.A.; Huang, M.; Ahmad, R.M.; Ghadi, Y.Y. Remote-Sensing Image Classification: A Comprehensive Review and Applications. In *Deep Learning for Multimedia Processing Applications*; CRC Press: Boca Raton, FL, USA, 2024; pp. 18–47.
3. Ge, D.; Yao, X.; Xiang, W.; Chen, Y. Vehicle Detection and Tracking Based on Video Image Processing in Intelligent Transportation System. *Neural Comput. Appl.* **2023**, *35*, 2197–2209.
4. Soltani Firouz, M.; Sardari, H. Defect Detection in Fruit and Vegetables by Using Machine Vision Systems and Image Processing. *Food Eng. Rev.* **2022**, *14*, 353–379. [CrossRef]
5. Ruby, E.; Amirthayogam, G.; Sasi, G.; Chitra, T.; Choubey, A.; Gopalakrishnan, S. Advanced Image Processing Techniques for Automated Detection of Healthy and Infected Leaves in Agricultural Systems. *Mesopotamian J. Comput. Sci.* **2024**, *4*, 44–52. [CrossRef] [PubMed]
6. Shakrani, K.V.; Kanyangarara, N.M.; Parowa, P.T.; Gupta, V.; Kumar, R. A Deep Learning Model for Face Recognition in Presence of Mask. *Acta Inf. Malays* **2022**, *6*, 43–46. [CrossRef]
7. Burger, W.; Burge, M.J. *Digital Image Processing: An Algorithmic Introduction*; Springer Nature: Berlin/Heidelberg, Germany, 2022.
8. Archana, R.; Jeevaraj, P.E. Deep Learning Models for Digital Image Processing: A Review. *Artif. Intell. Rev.* **2024**, *57*, 11. [CrossRef]
9. Li, M.; Jiang, Y.; Zhang, Y.; Zhu, H. Medical Image Analysis Using Deep Learning Algorithms. *Front. Public Health* **2023**, *11*, 1273253. [CrossRef]
10. Jurado, J.M.; López, A.; Pádua, L.; Sousa, J.J. Remote Sensing Image Fusion on 3D Scenarios: A Review of Applications for Agriculture and Forestry. *Int. J. Appl. Earth Obs. Geoinf.* **2022**, *112*, 102856. [CrossRef]
11. Prunella, M.; Scardigno, R.M.; Buongiorno, D.; Brunetti, A.; Longo, N.; Carli, R.; Dotoli, M.; Bevilacqua, V. Deep Learning for Automatic Vision-Based Recognition of Industrial Surface Defects: A Survey. *IEEE Access* **2023**, *11*, 43370–43423. [CrossRef]
12. Cohen, N.; Newman, Y.; Shamir, A. Semantic Segmentation in Art Paintings. *Comput. Graph. Forum* **2022**, *41*, 261–275. [CrossRef]
13. Yuan, S.; Zhao, W.; Gao, S.; Xia, S.; Hang, B.; Qu, H. An Adaptive Threshold-Based Quantum Image Segmentation Algorithm and Its Simulation: S. Yuan et Al. *Quantum Inf. Process.* **2022**, *21*, 359. [CrossRef]

14. Preethi, P.; Mamatha, H.R. Region-Based Convolutional Neural Network for Segmenting Text in Epigraphical Images. *Artif. Intell. Appl.* **2023**, *1*, 103–111. [CrossRef]
15. Sasmal, B.; Dhal, K.G. A Survey on the Utilization of Superpixel Image for Clustering Based Image Segmentation. *Multimed. Tools Appl.* **2023**, *82*, 35493–35555. [CrossRef]
16. Sun, R.; Lei, T.; Chen, Q.; Wang, Z.; Du, X.; Zhao, W.; Nandi, A.K. Survey of Image Edge Detection. *Front. Signal Process.* **2022**, *2*, 826967. [CrossRef]
17. Gao, Y.; Jiang, Y.; Peng, Y.; Yuan, F.; Zhang, X.; Wang, J. Medical Image Segmentation: A Comprehensive Review of Deep Learning-Based Methods. *Tomography* **2025**, *11*, 52. [CrossRef]
18. Goo, H.W.; Abdul Latiff, H. A Step-by-Step Guide for Cardiac Computed Tomography Ventricular Volumetry Using Three-Dimensional Threshold-Based Segmentation for Congenital Heart Disease. *Pediatr. Radiol.* **2025**, *55*, 2699–2703. [CrossRef]
19. Nagendram, S.; Singh, A.; Harish Babu, G.; Joshi, R.; Pande, S.D.; Ahammad, S.H.; Dhabliya, D.; Bisht, A. Stochastic Gradient Descent Optimisation for Convolutional Neural Network for Medical Image Segmentation. *Open Life Sci.* **2023**, *18*, 20220665. [CrossRef]
20. Mahajan, V.; Parulekar, S.; Magar, K.; Nagrale, S.; Ghule, G. Dynamic Programming Approach for Solving Variational Problems in Computer Vision. In *Information and Communication Systems*; CRC Press: Boca Raton, FL, USA, 2026; pp. 96–99.
21. Liu, Q.; Li, N.; Jia, H.; Qi, Q.; Abualigah, L. A Chimp-Inspired Remora Optimization Algorithm for Multilevel Thresholding Image Segmentation Using Cross Entropy. *Artif. Intell. Rev.* **2023**, *56*, 159–216. [CrossRef]
22. Ma, G.; Yue, X. An Improved Whale Optimization Algorithm Based on Multilevel Threshold Image Segmentation Using the Otsu Method. *Eng. Appl. Artif. Intell.* **2022**, *113*, 104960. [CrossRef]
23. Abualigah, L.; Habash, M.; Hanandeh, E.S.; Hussein, A.M.; Shinwan, M.A.; Zitar, R.A.; Jia, H. Improved Reptile Search Algorithm by Salp Swarm Algorithm for Medical Image Segmentation. *J. Bionic Eng.* **2023**, *20*, 1766–1790. [CrossRef]
24. Yu, M.; Zhang, J.; Yu, L.; Yang, H.; Lu, H.; Wei, X.; Xu, J. Alzheimer's Disease Brain Image Segmentation Using Multi-Feature Fusion in 3D Rényi Entropy Model and Quantum Hybrid Optimization. *Artif. Intell. Rev.* **2026**, *59*, 95. [CrossRef]
25. Fu, Y.; Liu, D.; Chen, J.; He, L. Secretary Bird Optimization Algorithm: A New Metaheuristic for Solving Global Optimization Problems. *Artif. Intell. Rev.* **2024**, *57*, 123. [CrossRef]
26. Meng, Q.; Kuang, X.; Yu, Z.; He, M.; Cui, H. Augmented Secretary Bird Optimization Algorithm for Wireless Sensor Network Deployment and Engineering Problem. *PLoS ONE* **2025**, *20*, e0329705. [CrossRef]
27. Qin, S.; Liu, J.; Bai, X.; Hu, G. A Multi-Strategy Improvement Secretary Bird Optimization Algorithm for Engineering Optimization Problems. *Biomimetics* **2024**, *9*, 478. [CrossRef]
28. Liao, Y.; Xing, C.; Guo, Q. An Enhanced Secretary Bird Optimization Algorithm for Target Localization Based on TDOA. *Phys. Commun.* **2025**, *73*, 102853. [CrossRef]
29. Liang, J.J.; Qu, B.Y.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization*; Technical Report; Computational Intelligence Laboratory, Zhengzhou University: Zhengzhou, China; Nanyang Technological University: Singapore, 2013; Volume 635, p. 2014.
30. Wu, G.; Mallipeddi, R.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for the CEC 2017 Competition on Constrained Real-Parameter Optimization*; Technical Report; National University of Defense Technology: Changsha, China; Kyungpook National University: Daegu, Republic of Korea; Nanyang Technological University: Singapore, 2017; Volume 9, p. 2017.
31. Mohamed, A.W.; Hadi, A.A.; Mohamed, A.K.; Awad, N.H. Evaluating the Performance of Adaptive Gainingsharing Knowledge Based Algorithm on CEC 2020 Benchmark Problems. In *2020 IEEE Congress on Evolutionary Computation (CEC)*; IEEE: Piscataway, NJ, USA, 2020; pp. 1–8.
32. Yu, M.; Yang, H.; Zhang, J.; Ouyang, K.; Fu, S.; Tan, P.; Jiang, F.; Xu, J. Bounty Hunter Optimizer: A Novel Metaheuristic with an Application to Multi-UAV Mobile Edge Computing and Path Planning. *Knowl.-Based Syst.* **2026**, *341*, 115836. [CrossRef]
33. Xiao, W.; Ouyang, K.; Lian, J.J.; Gu, S.; Liu, Y.; Wei, D.; Sha, X.; Wang, J.; Fu, S.; Qiu, M. Newton Downhill Optimizer for Global Optimization. 2025; *preprint*. [CrossRef] [PubMed]
34. Zhang, K.; Zhao, H.; Li, X.; Fu, C.; Jin, J. Gekko Japonicus Algorithm: A Novel Nature-Inspired Algorithm for Engineering Problems and Path Planning. *J. Bionic Eng.* **2026**, *23*, 431–471. [CrossRef]
35. Ye, J.; Sunat, K.; Chiewchanwattana, S. Stochastic Social Learning Optimization: Combining Social Learning and Bucket Theory for Efficient Optimization. *Knowl.-Based Syst.* **2026**, *341*, 115767. [CrossRef]
36. Chen, H.; Ahmadianfar, I.; Heidari, A.A.; Kordani, M.; Samadi Koucheksaraee, A.; Liang, G. LEE: A Physics-Inspired Optimizer Based on LangEvin Equation. *Neurocomputing* **2026**, *666*, 132288. [CrossRef]
37. Deng, L.; Liu, S. Snow Ablation Optimizer: A Novel Metaheuristic Technique for Numerical Optimization and Engineering Design. *Expert Syst. Appl.* **2023**, *225*, 120069. [CrossRef]
38. Fu, S.; Li, K.; Huang, H.; Ma, C.; Fan, Q.; Zhu, Y. Red-Billed Blue Magpie Optimizer: A Novel Metaheuristic Algorithm for 2D/3D UAV Path Planning and Engineering Design Problems. *Artif. Intell. Rev.* **2024**, *57*, 134.

39. Ouyang, K.; Fu, S.; Chen, Y.; Cai, Q.; Heidari, A.A.; Chen, H. Escape: An Optimization Method Based on Crowd Evacuation Behaviors. *Artif. Intell. Rev.* **2024**, *58*, 19. [CrossRef]
40. Woolson, R.F. Wilcoxon Signed-rank Test. In *Wiley Encyclopedia of Clinical Trials*; Wiley: Hoboken, NJ, USA, 2007; pp. 1–3.
41. Sheldon, M.R.; Fillyaw, M.J.; Thompson, W.D. The Use and Interpretation of the Friedman Test in the Analysis of Ordinal-scale Data in Repeated Measures Designs. *Physiother. Res. Int.* **1996**, *1*, 221–228. [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.