electronics

MDPI

Article

Improved African Vulture Optimization Algorithm Based on Random Opposition-Based Learning Strategy

Xingsheng Kuang [1], Junfa Hou [2], Xiaotong Liu [1], Chengming Lin [1], Zhu Wang [1,*] and Tianlei Wang [3,4,*]

[1] School of Mechanical and Automation Engineering, Wuyi University, Jiangmen 529020, China; 2112204155@wyu.edu.cn (X.K.); 2112204107@wyu.edu.cn (X.L.); 2112304011@wyu.edu.cn (C.L.)
[2] School of Rail Transit, Wuyi University, Jiangmen 529020, China; 2112204087@wyu.edu.cn
[3] School of Electronic and Information Engineering, Wuyi University, Jiangmen 529020, China
[4] Jiangmen Key Laboratory of Kejie Semiconductor Bonding Technology and Control System, Jiangmen 529020, China
* Correspondence: wharry@wyu.edu.cn (Z.W.); 12116349@bjtu.edu.cn (T.W.)

Abstract: This paper proposes an improved African vulture optimization algorithm (IROAVOA), which integrates the random opposition-based learning strategy and disturbance factor to solve problems such as the relatively weak global search capability and the poor ability to balance exploration and exploitation stages. IROAVOA is divided into two parts. Firstly, the random opposition-based learning strategy is introduced in the population initialization stage to improve the diversity of the population, enabling the algorithm to more comprehensively explore the potential solution space and improve the convergence speed of the algorithm. Secondly, the disturbance factor is introduced at the exploration stage to increase the randomness of the algorithm, effectively avoiding falling into the local optimal solution and allowing a better balance of the exploration and exploitation stages. To verify the effectiveness of the proposed algorithm, comprehensive testing was conducted using the 23 benchmark test functions, the CEC2019 test suite, and two engineering optimization problems. The algorithm was compared with seven state-of-the-art metaheuristic algorithms in benchmark test experiments and compared with five algorithms in engineering optimization experiments. The experimental results indicate that IROAVOA achieved better mean and optimal values in all test functions and achieved significant improvement in convergence speed. It can also solve engineering optimization problems better than the other five algorithms.

Keywords: African vulture optimization algorithm; random opposition-based learning; perturbation operator; metaheuristic algorithm

Citation: Kuang, X.; Hou, J.; Liu, X.; Lin, C.; Wang, Z.; Wang, T. Improved African Vulture Optimization Algorithm Based on Random Opposition-Based Learning Strategy. *Electronics* **2024**, *13*, 3329. https://doi.org/10.3390/electronics13163329

Academic Editor: Maciej Ławryńczuk

Received: 13 July 2024
Revised: 11 August 2024
Accepted: 12 August 2024
Published: 22 August 2024

1. Introduction

Metaheuristic algorithms are one of the effective methods for solving many real-world engineering optimization problems and are widely used in various fields, such as economics, engineering, politics, and management [1]. Most metaheuristic algorithms are inspired by the survival of the fittest theory of evolutionary algorithms, the collective command of swarm particles, the behavior of biologically inspired algorithms, and the logical behavior of natural physical algorithms. Compared with traditional optimization algorithms, metaheuristic algorithms have a simple structure, low computational complexity, do not require gradient information, and have a strong ability to escape local optima [2]. Metaheuristic algorithms are divided into three main types: those based on biological evolution, those based on physical laws, and those based on population behavior [3]. Algorithms based on the biological evolution process simulate the evolutionary process of organisms in nature and are widely used in the field of artificial intelligence (AI) to solve highly complex optimization problems [4]. Representative algorithms include genetic algorithm (GA) [5], differential evolution (DE) [6], and partial differential equations (PDE) [7]. Physics-based algorithms simulate the physical laws of nature. Representative algorithms include the

Gravity Search Algorithm (GSA) [8] and the Multi-Verse Optimization (MVO) [9]. Group-based algorithms simulate the biological behavior of group life. Representative algorithms include Particle Swarm Optimization (PSO) [10], Gray Wolf Optimizer (GWO) [11], Whale Optimization Algorithm (WOA) [12], Moth-Flame Optimization (MFO) [13], Tunicate Swar Algorithm (TSA) [14], and Harris Hawk Algorithm (HHO) [15].

The African vulture optimization algorithm [16] is one of the metaheuristic algorithms that is inspired by the competition and navigation behaviors of the African vultures. This algorithm has received widespread attention from researchers due to its simple structure and implementation method, as well as its excellent performance in finding optimal solutions [17]. Haimid et al. proposed an improved African vulture optimization algorithm to solve the minimization problem of fuel cell SOFC empirical voltage and current curves [18]. Zhou et al. proposed an improved African vulture optimization algorithm to solve the dual resource constraint flexible job scheduling problem with machine and worker constraints [19]. In order to solve the problem of rolling bearing defect diagnosis, Govind et al. introduced the improved African vulture optimization algorithm to optimize TVF-WMD filter parameters [20]. Singh et al. introduced the African vulture optimization algorithm to optimize the solution to the TSP shortest path problem [21]. Ahmed et al. introduced the African vulture optimization algorithm to accurately predict the location parameters of various solar photovoltaic units [22].

Although the African vulture optimization algorithm has good search performance, it still has some shortcomings, such as a tendency to fall into local optima, limited population diversity, and insufficient exploration capabilities in multimodal problems Wang et al. cited the representative vulture selection strategy and the rotating flight strategy. Zheng et al. proposed three strategies, selection accumulation mechanism, representative vulture selection strategy and rotating flight strategy to improved the African vulture optimization algorithm, which better balanced the balance between local search and global search [23]. Liu et al. introduced quasi-oppositional learning and differential evolution operators to improve the population diversity and enhancement algorithm of the algorithm. For the exploration ability [24], Hao et al. introduced tent chaos mapping and a time-varying mechanism optimization algorithm to obtain the optimal solution and improve the convergence speed [25].

However, although most of the existing improved AVOA have improved optimization performance and achieved satisfactory results in solving specific engineering problems, they are not suitable for most optimization problems and still have some limitations and uncertainties. The details are as follows:

1. During the exploration phase, the algorithm's reliance solely on updates from the optimal position diminishes population diversity, leading to slower convergence in the initial stages.
2. With a poor ability to balance exploration and exploitation, the AVOA is sensitive to local optimal solutions, and it is difficult to obtain ideal solutions.

In order to solve the above problems, this paper proposes an improved African vulture optimization algorithm. The first part adds a random opposition-based learning strategy in the initialization population stage of the algorithm to increase the diversity of the population, allowing the algorithm to search the space more broadly in the early stages. The second part introduces a disturbance factor in the exploration phase to increase the randomness of the algorithm and improve the algorithm's ability to escape from the local optimum, thus balancing the exploration and exploitation of the algorithm. In order to evaluate the effectiveness of the algorithm, this article conducted simulation experiments using 23 benchmark test functions and the CEC2019 test suite. IROAVOA performance results are compared with basic AVOA, ROAVOA, IAOVA, and seven other swarm intelligence algorithms. The results show that the proposed IROAVOA has better solution accuracy and convergence accuracy.

The remainder of this article is organized as follows. The original AVOA is briefly introduced in Section 2. IROAVOA is introduced in Section 3. In Section 4, the performance

of IROAVOA is analyzed using classic benchmark test functions. Section 5 discusses the results, summarizes the study, and suggests possible future research areas.

2. Original African Vultures Optimization Algorithm

AVOA is a new metaheuristic algorithm proposed by Mirjalili et al. [16] in 2021. This approach mimics the competitive and navigational behavior of African vultures. Known for their unique physical characteristics, African vultures are regarded as intelligent and strong creatures. One of the distinguishing characteristics of African vultures is that they take appropriate actions in different situations depending on the current level of hunger (hunger rate), which is also a characteristic of the AVOA. The solution process of the African vulture optimization algorithm is shown in Figure 1. The mathematical model of the hunger rate is shown in Equation (1).

$$F_t(t) = (2 \times rand + 1) \times z \times \left(1 - \frac{t}{T}\right) + dt \tag{1}$$

$$dt = h \times \left(\sin^w\left(\frac{\pi}{2} \times \frac{t}{T}\right) + \cos\left(\frac{\pi}{2} \times \frac{t}{T}\right) - 1\right) \tag{2}$$

where $F_t(t)$ is the vulture hunger rate of the i-th vulture in the t-th iteration. dt shows a fixed parameter set before the algorithm works. t represents the current number of iterations. T represents the maximum number of iterations, and x$rand$ indicates a random number between 0 and 1. h represents a random number between -2 and 2. z is a random number between -1 and 1. w is a fixed value that is set to 2.5 in AVOA. When the value drops below zero, it means that the vulture is in a hungry state. When the z increases to zero, it means that the vulture is in a satiated state. In order to show the key characteristics of the vulture, the first- or second-best vulture is selected as the leader vulture, and its mathematical equation is shown in Equation (3).

$$R_i(t) = \begin{cases} BestVulture1 & if\ p > rand \\ BestVulture2 & Otherwise \end{cases} \tag{3}$$

where $R_i(t)$ is a randomly selected vulture. *BestVulture1* and *BestVulture2* represent the first- and second-best vulture, respectively. p is a constant, which is set to 0.8.

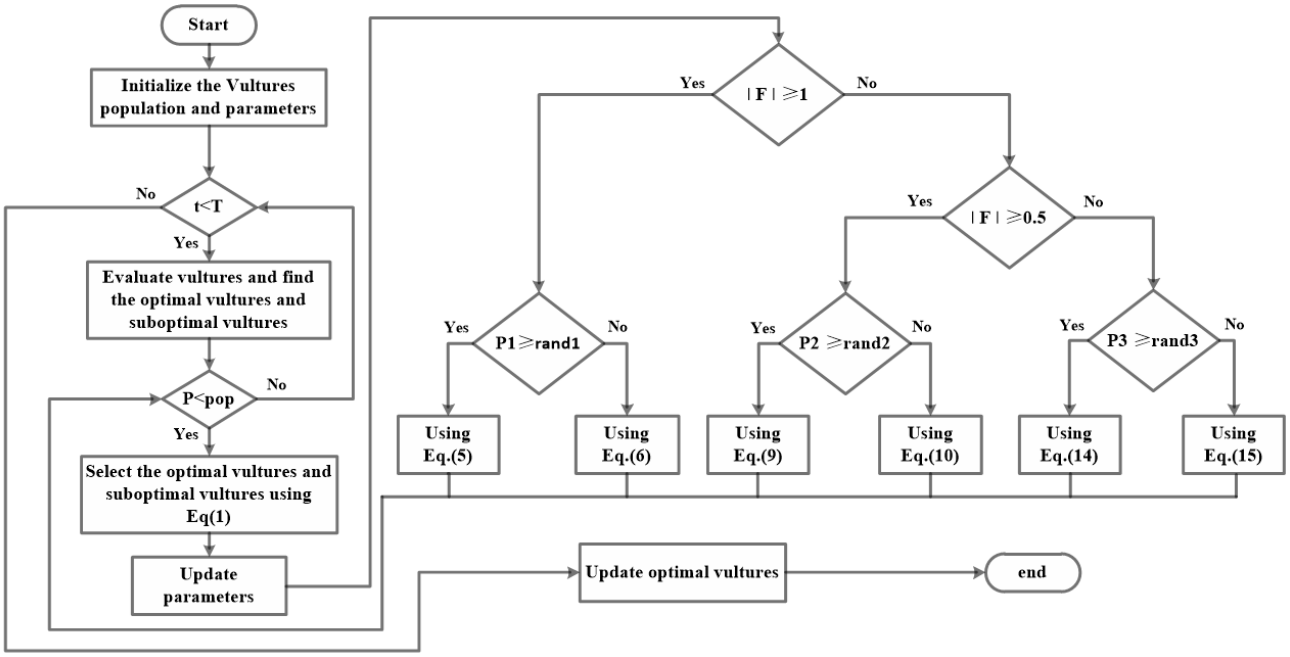

Figure 1. African vulture optimization algorithm flowchart.

2.1. Exploration Phase

When $|F_t(t)| \geq 1$, vultures hunt for food in different areas. AVOA has entered the exploration phase. Based on the movement of vultures to protect foods, it is divided into two strategies. The mathematical model is as follows:

$$P_i(t+1) = \begin{cases} Eq.(5)\ if\ p_1 \geq rand_{p1} \\ Eq.(6)\ if\ p_1 < rand_{p1} \end{cases} \tag{4}$$

$$P_i(t+1) = R_i(t) - D_i(t) \times F_i(t) \tag{5}$$

$$P_i(t+1) = R_i(t) - F_i(t) + rand \times ((ub - lb) \times rand \times lb) \tag{6}$$

$$D_i(t) = |X \times R_i(t) - P_i(t)| \tag{7}$$

where $P_t(t+1)$ is the new position for the next iteration, and p_1 is set to 0.6. $rand_{p_1}$ is a random number between 1 and 0. $D_i(t)$ indicates the distance between the current vulture and the selected leader vulture, and X is a random number between -2 and 2. ub and lb represent upper and lower limits.

2.2. Exploitation Phase

When $|F_t(t)| < 1$, vultures hunt for food in different areas. AVOA has entered the exploration phase. Based on the movement of vultures to protect foods, it is divided into two strategies. The mathematical model is as follows:

$$P_i(t+1) = \begin{cases} Eq.(9)\ if\ p_2 \geq rand_{p_2} \\ Eq.(10)\ if\ p_2 < rand_{p_2} \end{cases} \tag{8}$$

$$P_i(t+1) = D_i(t) - (F_i(t) + rand) - d_i(t) \tag{9}$$

$$P_i(t+1) = R_i(t) - (S_1 - S_2) \tag{10}$$

$$d_i(t) = R_i(t) - P_i(t) \tag{11}$$

$$\begin{cases} S_1 = R_i(t) \times \left(\frac{rand \times P_i(t)}{2 \times \pi}\right) \times \cos(P_i(t)) \\ S_2 = R_i(t) \times \left(\frac{rand \times P_i(t)}{2 \times \pi}\right) \times \sin(P_i(t)) \end{cases} \tag{12}$$

where p_2 is set to 0.4. $R_i(t)$ is one of the best vultures. $rand_{p_2}$ is a random number between 1 and 0. In the second stage, the value of F is less than 1. This stage simulates the vulture's accumulation of food and fierce competition for food:

$$P_i(t+1) = \begin{cases} Eq.(14)\ if\ p_3 \geq rand_{p_3} \\ Eq.(15)\ if\ p_3 < rand_{p_3} \end{cases} \tag{13}$$

$$P_i(t+1) = \frac{A_1 + A_2}{2} \tag{14}$$

$$P_i(t+1) = R_i(t) - |d_i(t)| \times F_i(t) \times Levy(d) \tag{15}$$

$$\begin{cases} A_1 = BestVulture1(t) - \frac{BestVulture1(t) - P_i(t)^2}{BestVulture1(t)^2 - P_i(t)^2} \times F_i(t) \\ A_2 = BestVulture1(t) - \frac{BestVulture2(t) - P_i(t)^2}{BestVulture2(t) - P_i(t)^2} \times F_i(t) \end{cases} \tag{16}$$

$$Levy(d) = 0.01 \times \frac{u}{|v|^{\frac{1}{\beta}}}, u \sim (0, \sigma_u^2), v \sim (0, \sigma_v^2) \tag{17}$$

$$\sigma_u = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}}\right)^{\frac{1}{\beta}} \tag{18}$$

where p_3 is set to 0.4. $rand_{p_3}$ is a random number between 1 and 0. u and v are random numbers that follow a Gaussian distribution. σ_v and β are set to 1 and 1.5. Γ is the standard gamma function.

3. Improved African Vulture Optimization Algorithm

3.1. Perturbation Operator

During the process of competition and navigation, the African vulture group mainly uses the position information of the optimal vulture, gradually approaches them, and updates its own position according to Equation (4). During the algorithm-solving process, a new optimal solution is generated around the optimal solution. However, as the iteration proceeds, the population diversity gradually decreases, which may cause the algorithm to fall into a local optimum. In order to overcome this shortcoming, this paper introduces an interference factor w in the exploration stage. The changing trend of this interference factor is shown in Figure 2. In the initial stage of the algorithm, providing a random number can promote a wider search and increase the exploration solution space possibility. In the later stages of the algorithm, the introduction of random numbers helps to escape from the local optimal solution and avoids falling into the dilemma of local search, making it more likely to find the global optimal solution. The definition of perturbation operator is shown in Formula (19).

$$w = (0.5 \times (0.1 + rand)) \times randn(0, 1) \tag{19}$$

where $rand$ is a random number between 1 and 0. $randn(0, 1)$ is a random number obeying the normal distribution with mean 0 and standard deviation 1. The vulture position update in the exploration phase is defined as follows:

$$P_i(t + 1) = R_i(t) \times (1 - w) - D_i(t) \times F_i(t) \times w \tag{20}$$

$$P_i(t + 1) = R_i(t) \times w - F_i(t) + rand \times ((ub - lb) \times rand \times lb) \tag{21}$$

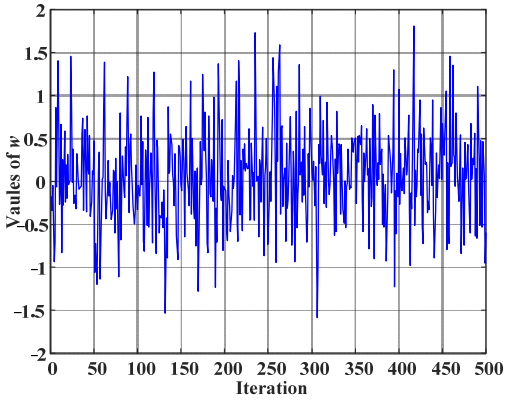

Figure 2. w change trend chart.

3.2. Random Opposition-Based Learning

The bibliography on motion ecology provides a wealth of insights into the complex patterns and dynamics of animal movement [26–28]. Meyer et al. predicted the movement of springboks within the next hour with a certainty of approximately 20%, while the remaining 80% of the movement is stochastic in nature, stemming from unaccounted factors in the modeling algorithm and individual behavioral traits of the springboks [29]. Stochastic is of great significance in ecological and animal behavioral research. Opposition-based learning is a method based on estimation and opposition estimation principles proposed by Tizhoosh et al. [30]. It is inspired by the concept of opposition in the real world and has been widely used in optimization algorithms, reinforcement learning, artificial neural networks, and fuzzy systems [31]. Optimization usually starts with a candidate solution, and the initial population and parameters are chosen based on randomness. If the initial candidate solution is close to the optimal solution, the algorithm will converge

quickly. Conversely, the initial candidate solution may be far away from the candidate solution, in which case the algorithm will take longer to converge, or in the worst case, the algorithm may not converge at all. Opposition learning causes candidate solutions to generate opposite points, which can improve the convergence speed of the algorithm under a certain probability. Therefore, the opposite point of each candidate solution can be further explored. If it is found to be beneficial, we can consider using the opposite point as a candidate solution for the next iteration.

Definition 1. *Assume that x is a real number defined in the interval* $[L_1, L_2]$*, and its opposite point* x_{op} *is defined as shown in Equations (22) and (23).*

$$x_{op} = L_1 + L_2 - x \tag{22}$$

If $L_1 = 0$ *and* $L_2 = 0$*, then:*

$$x_{op} = 1 - x \tag{23}$$

Similarly, when going beyond two dimensions, we can define opposite positions.

Definition 2. *Assume that* $p(x_1, \ldots, x_D)$ *is a point on the D-dimensional coordinate system, where each* x_i *is a real number in the interval* $[L_{1i}, L_{2i}]$*, and the definition of the opposite point* $\widetilde{p}$ *of* p *is shown in Equation (24).*

$$x_{iop} = L_{1i} + L_{2i} - x_i \ \forall i \in [1, D] \tag{24}$$

where x_{iop} *is the coordinate of* $\widetilde{p}$*.*

The fixed distance between the reverse solution generated by the opposition-based learning strategy and the current solution limits its randomness. In order to enhance the diversity of the population and improve its ability to avoid falling into the local optimal solution, Long W et al. proposed the random opposition-based learning strategy [32], and its one-dimensional solution space is shown in Figure 3. This strategy is proposed to further expand the search space and improve the randomness of the algorithm and the population's ability to avoid local optimality. Its definition is shown in Equation (25).

$$x_{iop} = L_{1i} + L_{2i} - (rand \times x_i) \ \forall i \in [1, D] \tag{25}$$

where *rand* is a random number between 1 and 0.

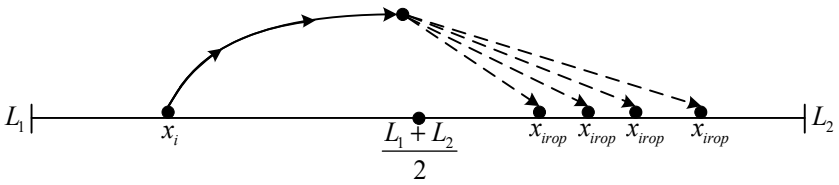

Figure 3. ROBL one-dimensional solution space.

3.3. Summary of the Proposed Method

In the African vulture optimization algorithm, the transition between exploration and exploitation depends on the vulture's hunger rate F. In the early stages of exploration, the algorithm converges slowly due to the lack of diversity in the population. As the number of iterations increases, the value of the hunger rate F gradually decreases, keeping the algorithm in the exploitation stage. However, this algorithm has shortcomings, such as easily falling into local optimality and imbalance in exploration and exploitation capabilities. To solve these problems, this paper first introduces a random opposition-based learning strategy during population initialization to improve the diversity of the initial population. Secondly, interference factors are introduced in the exploration phase of each population, allowing the algorithm to explore more extensively in the search space, effectively improv-

ing the ability to avoid falling into local optima, improving the global search capability of the algorithm, and better balancing the algorithm's explore and exploit. The solution process of the improved African Vulture optimization algorithm is shown in Figure 4.

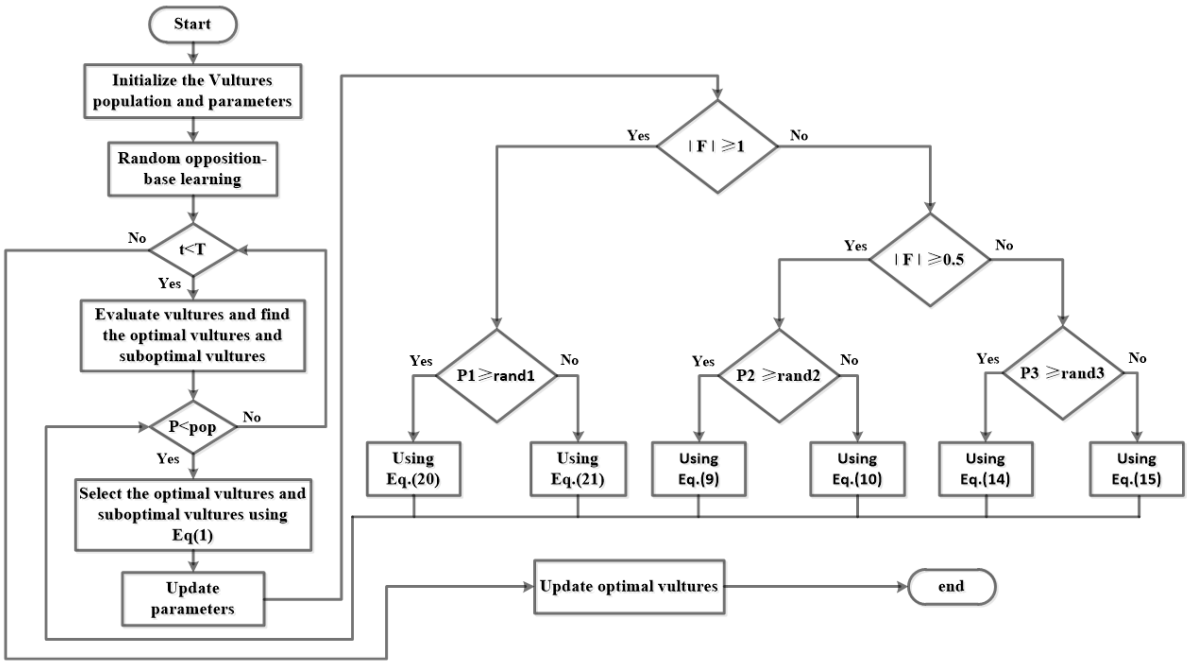

Figure 4. Improved African vulture optimization algorithm flowchart.

3.4. Computational Complexity

The computational complexity of the AVOA hinges primarily on three key steps: initialization, fitness evaluation, and the update of vulture position vectors. In the initialization phase, assigning initial states to N vultures incurs a computational cost of O(N). As the algorithm progresses to search for optimal positions and update the position vectors of all vultures, the complexity arises from two main components: O(T × N), stemming from the multiplication of the number of iterations and vultures, and O(T × N × D), which encapsulates the impact of iterations, vultures, and the dimensionality of the problem. By integrating these factors, the overall computational complexity of the AVOA can be succinctly expressed as O(N × T × (1 + D)), effectively highlighting the interplay between algorithm performance and the number of vultures, iterations, and the complexity of the problem domain. Turning to the computational complexity of IROAVOA, consider the worst situation, as each vulture updates the position using the random opposition study perturbation factor throughout the iteration and then generates two candidate positions, updating the computational complexity of all vulture positions to O (2 × T × 2 × N × D). Therefore, the overall assumed complexity of IROAVOA is O(N × 2 × T × 2 × (1 + D)).

Compared with the original AVOA, the introduction of the random adversarial learning and perturbation factor increases the computational complexity to a certain extent. However, these additional time costs can improve the search performance of the algorithm, so the additional computational complexity is acceptable.

4. Discussion

4.1. Experimental Design and Parameter Setting

In order to verify that the IROAVOA has better optimization performance and effectiveness, a comparative experiment was first conducted between AVOA and the three algorithms proposed in this article: IROAVOA, ROAVOA, and IAVOA. Then, IROAVOA was compared with other classic algorithms PSO and GWO, and seven types of MVO, WOA, MFO, TSA, and HHO were used for comparative experiments. The experiment

used 23 benchmark test functions (F1–F23) [33] and the CEC2019 [34] test suite, as shown in Tables 1–4. These benchmark functions are variations of classical functions in terms of shifted, rotated, expanded, and combined. The benchmark test functions of Table 1 are unimodal benchmark functions with only one optimal solution and are often used to verify the local exploitation stage of the algorithm. The benchmark test functions of Table 2 are multimodal benchmark functions. The benchmark test functions of Table 3 are fixed-dimensions multimodal. Multimodal benchmark functions are used to test the global exploration capabilities of the algorithm. Dim represents the dimension of the function, Range indicates the range of the function search space, and f_{min} represents the theoretical optimal value of the function. The mathematical expressions and function characteristics of the classic benchmark test functions and the CEC2019 test suite are shown in Table 4.

Table 1. Unimodal benchmark functions.

Number	Name	Benchmark	Dim	Range	$f_{\min}$
F_1	Sphere	$F1 = \sum_{i=1}^{d} x_i^2$	30	$[-100, 100]$	0
F_2	Schwefel'problem2.22	$F2 = \sum_{i=1}^{d} \lvert x_i \rvert + \prod_{i=1}^{d} \lvert x_i \rvert$	30	$[-10, 10]$	0
F_3	Schwefel'problem1.2	$F3 = \sum_{i=1}^{d} \left(\sum_{j=1}^{i} x_j \right)^2$	30	$[-100, 100]$	0
F_4	Schwefel'problem2.21	$F4 = \max\{\lvert x_i \rvert, 1 \le i \le d\}$	30	$[-100, 100]$	0
F_5	Rosenbrock	$F5 = \sum_{i=1}^{d-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right]$	30	$[-30, 30]$	0
F_6	Step	$F6 = \sum_{i=1}^{d} (\lvert x_i + 0.5 \rvert)^2$	30	$[-100, 100]$	0
F_7	Noise	$F7 = \sum_{i=1}^{d} i x_i^4 + random[0, 1)$	30	$[-1.28, 1.28]$	0

Table 2. Multimodal benchmark functions.

Number	Name	Benchmark	Dim	Range	$f_{\min}$
F_8	Generalized Schwfel's problem	$F8 = \sum_{i=1}^{d} -x_i \sin\left(\sqrt{\lvert x_i \rvert} \right)$	30	$[-500, 500]$	−12,569.5
F_9	Rastrigin	$F9 = 10d + \sum_{i=1}^{d} \left[x_i^2 - 10\cos(2\pi x_i) \right]$	30	$[-5.12, 5.12]$	0
F_{10}	Ackley	$F10 = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2} \right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d} \cos(2\pi x_i) \right) + 20 + \exp(1)$	30	$[-32, 32]$	0
F_{11}	Griewank	$F11 = \sum_{i=1}^{d} \frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	30	$[-600, 600]$	0
F_{12}	Generalized penalized function 1	$F12 = \frac{\pi}{d}\left\{ 10\sin^2(\pi y_1) + \sum_{i=1}^{d-1} (y_i - 1)^2 \left[1 + 10\sin^2(\pi y_{i+1}) \right] + (y_d - 1)^2 \right\} + \sum_{i=1}^{d} \mu(x_i, 10, 100, 4)$ $where\ y_i = 1 + \frac{1}{4}(x_i + 1),$ $\mu(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \le a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$	30	$[-50, 50]$	0
F_{13}	Generalized penalized function 2	$F13 = 0.1\left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{d} (x_i - 1)^2 \left[1 + \sin^2(3\pi x_i + 1) \right] + (x_d - 1)^2 \left[1 + \sin^2(2\pi x_d) \right] \right\} + \sum_{i=1}^{d} \mu(x_i, 5, 100, 4)$ $where\ \mu(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \le a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$	30	$[-50, 50]$	0

Table 3. Fixed-dimensions multimodal benchmark functions.

Number	Name	Benchmark	Dim	Range	$f_{\min}$
F_{14}	Shekel's foxholes function	$F14 = (\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} \left(x_i - a_{ij}\right)^6})^{-1}$	2	[−65.536, 65.536]	1
F_{15}	Kowalik's function	$F15 = \sum_{i=1}^{11} \left[a_i - \frac{x_i(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	[−5, 5]	0.0003
F_{17}	Branin	$F17 = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(x_1) + 10$	2	[−5, 10] [10, 15]	0.3788
F_{18}	Goldstein–Price function	$F18 = \left[1 + (x_1 + x_2 + 1)^2 \cdot (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \times \left[30 + (2x_1 - 3x_2)^2 \cdot (18 - 32x_1 + 12x_1^2 + 48x_2^2 - 36x_1x_2 + 27x_2^2)\right]$	2	[−2, 2]	3
F_{19}	Hartmann 1	$F19 = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2)$	3	[0, 1]	−3.86
F_{20}	Hartmann 2	$F20 = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2)$	6	[0, 1]	−3.32
F_{21}	Shekel 1	$F21 = -\sum_{i=1}^{5} \left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	4	[0, 10]D	−10.1532
F_{22}	Shekel 2	$F22 = -\sum_{i=1}^{7} \left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	4	[0, 10]D	−10.4028
F_{23}	Shekel 3	$F23 = -\sum_{i=1}^{10} \left[(x - a_i)(x - a_i)^T + c_i\right]^{-1}$	4	[0, 10]D	−10.5363

Table 4. Descriptions of the CEC2019 test suite.

Number	Name	D	Range	F_{min}
CEC-01	Storn's Chebyshev Polynomial Fitting Problem	9	[−8192, 8192]	1
CEC-02	Inverse Hilbert Matrix Problem	16	[−16,384, 16,384]	1
CEC-03	Lennard-Jones Minimum Energy Cluster	18	[−4, 4]	1
CEC-04	Rastrigin's Function	10	[−100, 100]	1
CEC-05	Griewangk's Function	10	[−100, 100]	1
CEC-06	Weierstrass Function	10	[−100, 100]	1
CEC-07	Modified Schwefel's Function	10	[−100, 100]	1
CEC-08	Expanded Schaffer's F6 Function	10	[−100, 100]	1
CEC-09	Happy Cat Function	10	[−100, 100]	1
CEC-10	Ackley Function	10	[−100, 100]	1

The experimental environment for the optimization test in this article is the Window11 operating system, Intel(R) Core(TM) i5–11155G7 CPU @2.50 GHz, 16 GB memory, and MATLAB 2021a. In the algorithm optimization of this article, the parameter settings of each algorithm are consistent with the references. The population size is 30, and the maximum number of iterations is 500. Each optimization algorithm will be run independently on each benchmark function 30 times, and the test results will be compared with the function and the optimal value of the function. The results mainly include the best value (Best), the mean (Mean), and the standard deviation (Std). The average value can verify the optimization ability of the algorithm, and the standard deviation can verify the stability of the algorithm. The parameter settings of all comparison algorithms are shown in Table 5.

Table 5. Algorithm parameter setting.

Algorithm	Parameter	Value	Algorithm	Parameter	Value
IROAVOA	L_1	0.8	WOA	a	$[0,2]$
	L_2	0.2		b	2
	W	2.5	MFO	a	$[-2,-1]$
	P_1	0.6		b	1
	P_2	0.4	TSA	$P_{\min}$	1
	P_3	0.6		$P_{\max}$	4
GWO	α	$[0,2]$	HHO	E_1	$[0,2]$
	C_1	2		E_0	$[-1,1]$
PSO	C_2	2	MVO	$WEP_{\max}$	1
	$w_{\min}$	0.2		$WEP_{\min}$	0.2
	$w_{\max}$	0.9			

4.2. Convergence Analysis

As shown in the convergence curve in Figures 5–7, the convergence speed and convergence accuracy of the three improved algorithms in the single-peak test function F1–F5 are significantly improved compared to the traditional AVOA. For F6 and F7, even if the algorithm falls into a local optimum, its convergence accuracy is better than AVOA. For multimodal functions F10, F12, and fixed-dimensional multimodal test function F15, IROAVOA has better convergence accuracy. For multimodal test functions F8–F9, F11, F13, and fixed-dimensional multimodal test functions F16–F23, the four algorithms have the same convergence accuracy, but the convergence speed of the three improved algorithms is faster than the traditional AVOA. Whether for unimodal or multimodal test functions, IROAVOA can provide better convergence.

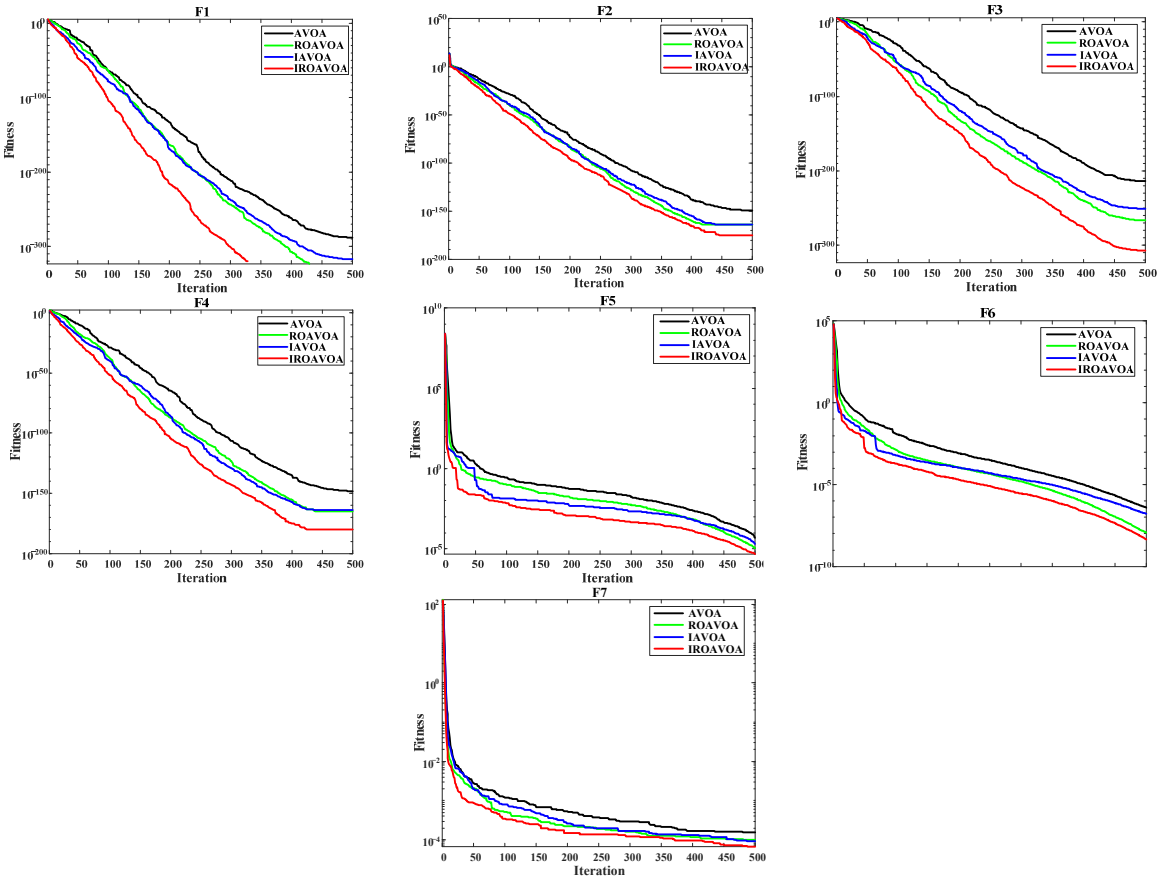

Figure 5. Convergence curve of unimodal optimization function.

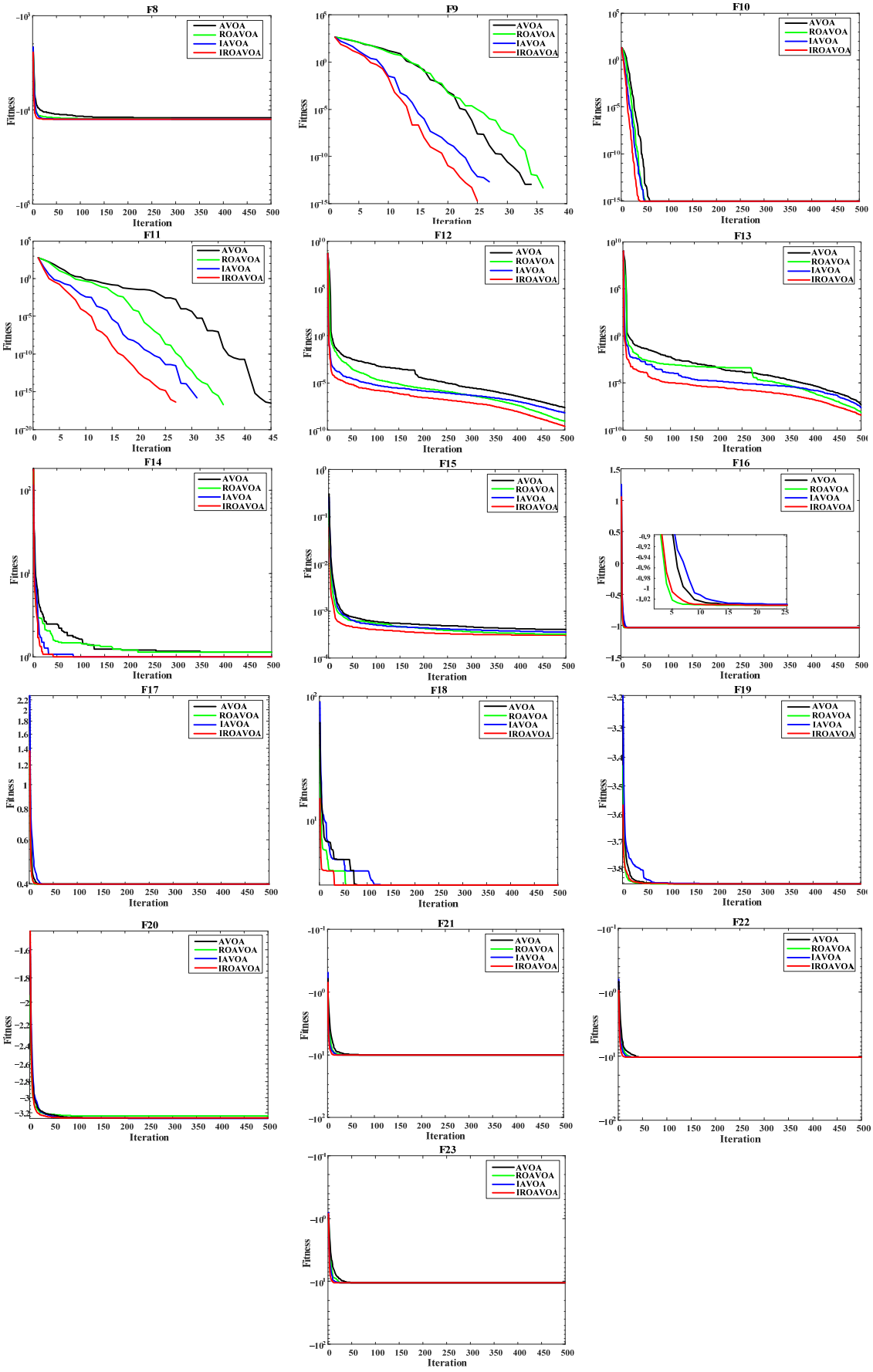

Figure 6. Convergence curve of multimodal optimization functions.

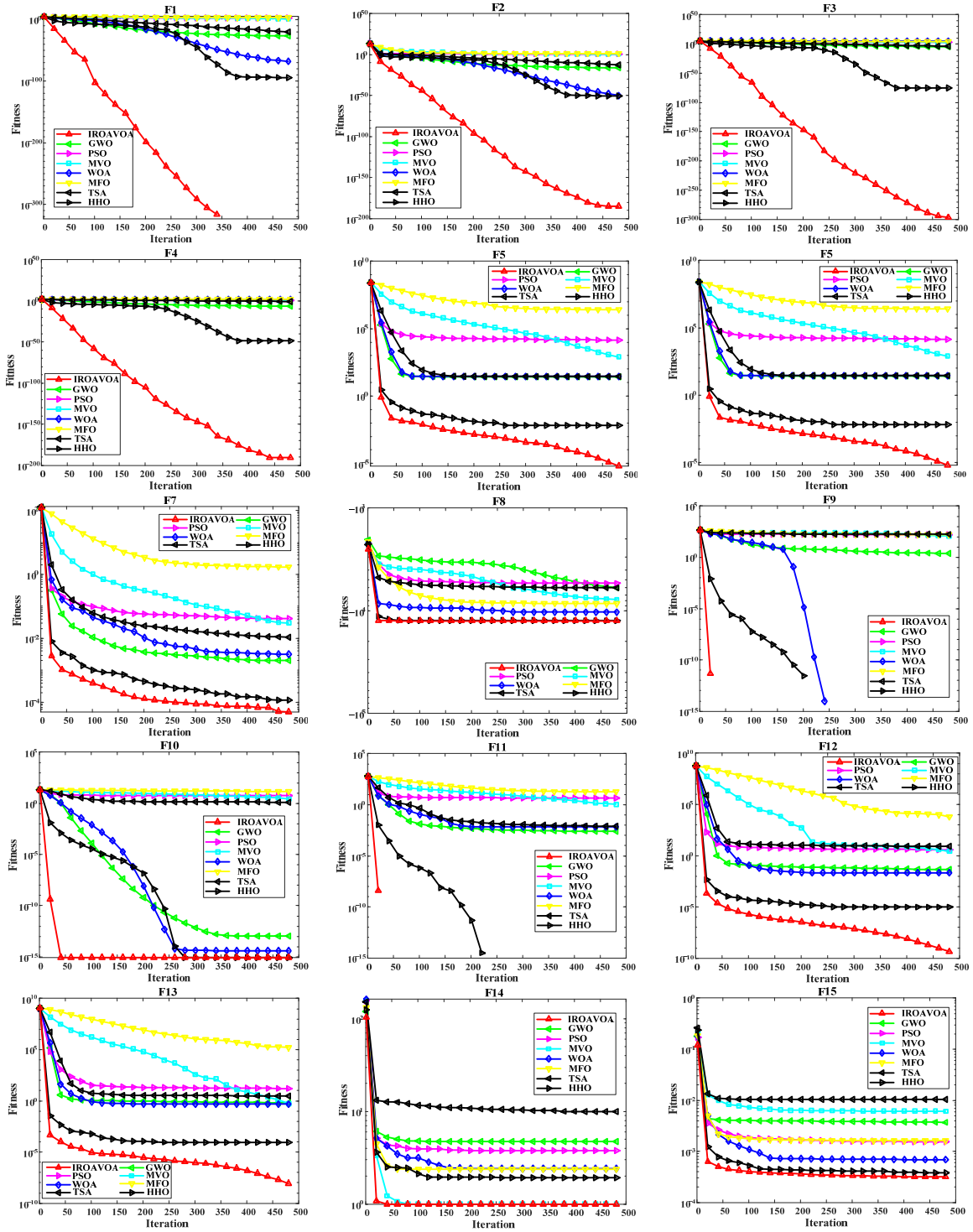

Figure 7. *Cont.*

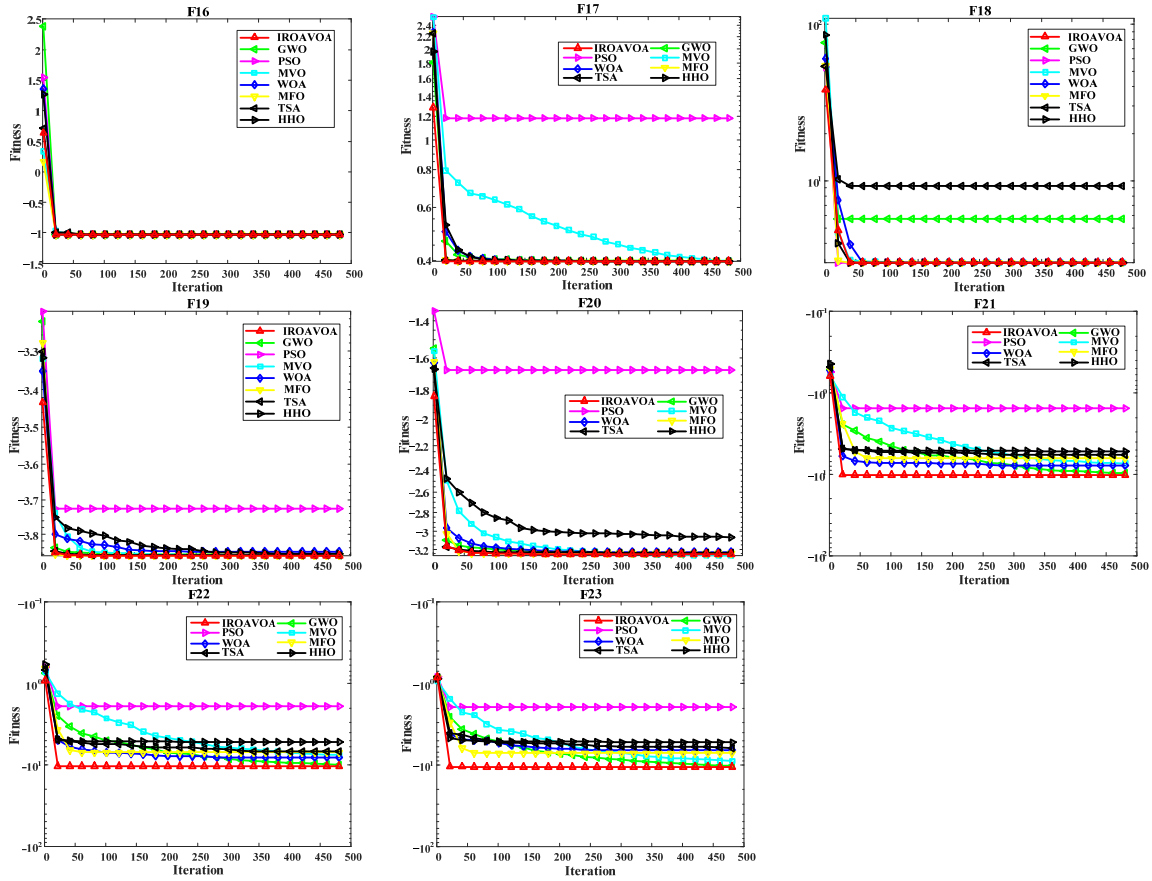

Figure 7. Convergence curves of 23 sets of benchmark test functions between the IROAVOA and other comparative optimization algorithms.

4.3. Analysis of 23 Groups of Benchmark Test Function Results

We conducted a comprehensive evaluation of the proposed improved algorithm based on the unimodal and multimodal test functions described previously. Tables A1–A3 list the results of the optimal value, mean, and standard deviation obtained by IROAVOA and other algorithms for each benchmark test function (F1–F23). In order to ensure the performance of fair comparison algorithms, the maximum number of iterations of all comparison algorithms is set to 500, and the population number is set to 30. The results show that the proposed IROAVOA exhibits excellent performance on most tested functions.

For the unimodal test functions (F1–F7), IROAVOA can effectively find the global optimal value. Compared with other comparison algorithms, the solution accuracy of the proposed improved algorithm is significantly improved. As shown in Tables A1–A3 and Figures 5 and 6, the mean and standard deviation of IROAVOA are small, and the convergence speed is faster than other algorithms, indicating that IROAVOA shows the best performance on these test functions and is better than other comparison algorithms. For the multimodal test function (F8–F23), IROAVOA is significantly better than other comparison algorithms in the test functions F12, F14–F15, and F17–F20. In the benchmark test functions F9–F11, the performance of the algorithm is similar to HHO but significantly better than GWO, PSO, MVO, WOA, MFO, and TSA. However, the performance of some benchmark test functions of IROAVOA is not as good as other optimization algorithms. Therefore, IROAVOA has certain limitations and requires further testing and application.

4.4. Comparison with Basic AVOA and Two Variants of AVOA

To evaluate the effectiveness of each component, the improved IROAVOA was compared with AVOA, ROAVOA improved based on random opposition learning strategy, and IAOVA improved based on perturbation factor. Under the same experimental design, 23 different types of test functions in Tables 1–3 were tested simultaneously, and the optimal values, mean values, and standard deviations were obtained as shown in Tables A1–A3. To demonstrate the dynamic convergence performance of IROAVOA, the average optimal fitness convergence curves of each test function are shown in Figures 5 and 6.

Based on the results, we can find that the performance of the three improved optimization algorithms is superior to traditional AVOA. For the unimodal test functions (F1–F7), IROAVOA outperforms other algorithms on most test functions. For F1 and F3, the four algorithms can achieve the same optimal fitness value, but IROAVOA has slightly better standard deviation than ROAVOA, IAVOA, and AVOA. For F5–F7, all four algorithms did not reach the theoretical optimal value, indicating that IROAVOA has local development potential. For the multimodal testing function (F8–F23), the three improved algorithms showed slight improvements in both the optimal and average values. For F10 and F13–F15, the average value of IROAVOA has reached the theoretical optimal value, and the performance of the other two improved algorithms has also been improved. The three improved algorithms have slightly improved the average values of most test functions, but the search performance has only significantly improved. The main reason is that the traditional AVOA algorithm itself has good search ability and can discover the theoretical optimal solution of multimodal test functions, so the room for improvement is relatively small. These results indicate that the performance of AVOA has been improved to a certain extent by applying random opposition-based learning and perturbation factor.

4.5. CEC2019 Test Suite Result Analysis

In order to further illustrate the superiority of the improved algorithm in this article, the IEEE CEC2019 test suite is used to evaluate the performance of IROAVOA in solving complex numerical problems. The CEC2019 suite contains 10 complex single-objective test functions. In order to ensure the fairness of the experiment, the proposed improved algorithm and the other eight comparison algorithms were run independently on each function 30 times. The maximum number of iterations is set to 500, and the number of populations is set to 30. Table 6 lists the optimal values, mean values, and standard deviation results of the test results. The last row of the table shows the Friedman test results.

Table 6. CEC2019 experimental results of IROAVOA and other algorithms.

Benchmark Function	Index	IROAVOA	GWO	PSO	MVO	WOA	MFO	TSA	HHO
F_1	Best	1.00×10^{0}	5.11×10^{1}	2.58×10^{10}	9.20×10^{3}	2.19×10^{3}	9.11×10^{4}	2.72×10^{1}	1.00×10^{0}
	Mean	1.67×10^{-2}	1.20×10^{3}	7.09×10^{10}	1.27×10^{4}	3.65×10^{3}	9.14×10^{4}	9.18×10^{2}	3.33×10^{-2}
	Std	1.29×10^{-1}	1.64×10^{3}	2.76×10^{10}	2.60×10^{3}	1.40×10^{3}	4.69×10^{2}	1.06×10^{3}	1.83×10^{-1}
F_2	Best	4.65×10^{0}	7.19×10^{0}	6.77×10^{4}	8.39×10^{1}	5.28×10^{1}	8.11×10^{1}	5.99×10^{0}	5.00×10^{0}
	Mean	7.75×10^{-2}	7.22×10^{0}	1.12×10^{5}	8.94×10^{1}	5.30×10^{1}	8.11×10^{1}	1.25×10^{1}	1.67×10^{-1}
	Std	6.00×10^{-1}	2.18×10^{-2}	1.53×10^{4}	3.61×10^{0}	3.67×10^{-1}	3.45×10^{-2}	4.55×10^{0}	9.13×10^{-1}
F_3	Best	5.53×10^{0}	1.28×10^{1}	1.33×10^{1}	1.30×10^{1}	1.13×10^{1}	1.11×10^{1}	1.37×10^{1}	8.02×10^{0}
	Mean	9.21×10^{-2}	1.39×10^{6}	1.37×10^{1}	3.89×10^{7}	1.00×10^{18}	3.09×10^{1}	1.37×10^{1}	2.67×10^{-1}
	Std	7.13×10^{-1}	4.79×10^{6}	7.14×10^{-6}	2.13×10^{8}	3.05×10^{18}	9.98×10^{1}	3.12×10^{-12}	1.46×10^{0}
F_4	Best	4.43×10^{1}	2.14×10^{1}	4.52×10^{5}	5.65×10^{1}	7.11×10^{1}	8.97×10^{1}	1.27×10^{2}	6.22×10^{1}
	Mean	7.39×10^{-1}	2.39×10^{1}	8.21×10^{5}	8.05×10^{1}	7.13×10^{1}	8.97×10^{1}	1.88×10^{2}	2.07×10^{0}
	Std	5.72×10^{0}	3.85×10^{-2}	2.64×10^{5}	1.33×10^{1}	3.26×10^{-1}	8.07×10^{-14}	2.43×10^{1}	1.14×10^{1}

Table 6. *Cont.*

Benchmark Function	Index	IROAVOA	GWO	PSO	MVO	WOA	MFO	TSA	HHO
F_5	Best	1.45×10^{0}	1.81×10^{0}	1.56×10^{6}	3.81×10^{0}	2.90×10^{0}	1.17×10^{2}	1.36×10^{2}	2.02×10^{0}
	Mean	2.41×10^{-2}	2.08×10^{0}	2.74×10^{6}	5.00×10^{0}	3.02×10^{0}	1.17×10^{2}	1.38×10^{2}	6.74×10^{-2}
	Std	1.87×10^{-1}	2.82×10^{-2}	8.93×10^{5}	6.26×10^{-1}	1.57×10^{-1}	7.70×10^{-5}	2.22×10^{0}	3.69×10^{-1}
F_6	Best	7.02×10^{0}	1.09×10^{1}	6.63×10^{0}	7.94×10^{0}	9.35×10^{0}	5.45×10^{0}	1.11×10^{1}	9.30×10^{0}
	Mean	1.17×10^{-1}	2.00×10^{1}	1.81×10^{1}	8.94×10^{0}	1.19×10^{1}	5.45×10^{0}	2.11×10^{1}	3.10×10^{-1}
	Std	9.09×10^{-1}	7.87×10^{-1}	4.23×10^{0}	5.19×10^{-1}	2.42×10^{0}	1.75×10^{-6}	2.69×10^{0}	1.70×10^{0}
F_7	Best	1.16×10^{3}	9.13×10^{2}	1.81×10^{5}	1.71×10^{3}	1.38×10^{3}	1.61×10^{3}	2.41×10^{3}	1.13×10^{3}
	Mean	1.93×10^{1}	1.43×10^{3}	3.11×10^{5}	2.21×10^{3}	1.40×10^{3}	1.61×10^{3}	4.69×10^{3}	3.75×10^{1}
	Std	1.50×10^{2}	7.81×10^{0}	1.05×10^{5}	2.89×10^{2}	3.53×10^{1}	5.69×10^{0}	6.40×10^{2}	2.06×10^{2}
F_8	Best	4.46×10^{0}	5.31×10^{0}	5.53×10^{0}	5.45×10^{0}	5.06×10^{0}	5.33×10^{0}	5.59×10^{0}	4.89×10^{0}
	Mean	7.44×10^{-2}	5.88×10^{0}	6.00×10^{0}	6.00×10^{0}	5.10×10^{0}	5.33×10^{0}	6.00×10^{0}	1.63×10^{-1}
	Std	5.76×10^{-1}	2.77×10^{-1}	1.80×10^{-1}	3.56×10^{-1}	8.22×10^{-2}	3.29×10^{-6}	6.85×10^{-3}	8.94×10^{-1}
F_9	Best	1.40×10^{0}	1.20×10^{0}	2.00×10^{4}	1.84×10^{0}	1.32×10^{0}	8.10×10^{0}	2.86×10^{0}	1.39×10^{0}
	Mean	2.45×10^{-2}	1.44×10^{0}	3.66×10^{4}	2.34×10^{0}	1.61×10^{0}	8.10×10^{0}	3.53×10^{0}	4.63×10^{-2}
	Std	1.90×10^{-1}	5.70×10^{-2}	1.32×10^{4}	2.14×10^{-1}	1.27×10^{-1}	1.54×10^{-5}	2.80×10^{-1}	2.54×10^{-1}
F_{10}	Best	2.11×10^{1}	2.15×10^{1}	2.15×10^{1}	2.15×10^{1}	2.14×10^{1}	2.11×10^{1}	2.15×10^{1}	2.14×10^{1}
	Mean	3.52×10^{-1}	2.27×10^{1}	2.27×10^{1}	2.26×10^{1}	2.24×10^{1}	2.11×10^{1}	2.27×10^{1}	7.12×10^{-1}
	Std	2.73×10^{0}	2.29×10^{-1}	2.30×10^{-1}	3.14×10^{-1}	4.22×10^{-1}	4.45×10^{-14}	2.34×10^{-1}	3.90×10^{0}

It can be seen from the data in Table 4 that for CEC2019, IROVAOA is better than the other eight algorithms in the 10 test functions and can effectively find the global optimal value. Although the standard deviation of the proposed algorithm is slightly inferior to other algorithms, they all have smaller standard deviations than the traditional AVOA, which shows that the improved IROAVOA has better stability. The above results show that the performance of the proposed IROAVOA is more competitive and can effectively solve various complex optimization problems.

4.6. Statistical Test Result Analysis

The mean and standard deviation are metrics to evaluate the overall stability of the algorithm, but they do not fully reflect the results of each run. Due to the randomness of the results of each run, it is not sufficient to rely only on fitness and standard values and standard deviations to evaluate algorithm performance. In order to verify whether there are differences between the proposed algorithm and other algorithms, this section uses the Wilcoxon rank sum test [35] and the Friedman ranking test [36].

The Friedman ranking test ranks IROAVOA and other algorithms based on the fitness obtained using Equation (26), where k is the number of algorithms, R_j is the average ranking of the *j*-th algorithm, and n is the number of test checks. The test is performed by assuming a distribution with $k - 1$ degrees of freedom. It first determines the ranking of each algorithm independently and then calculates the average ranking to arrive at a final ranking for each algorithm considered. Table 7 shows the Friedman ranking test result. According to the results, the IROAVOA is significantly different from other algorithms in most benchmark test functions. The overall ranking shows that the IROAVOA is better than other algorithms.

$$F_f = \frac{12n}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right] \tag{26}$$

Table 7. Friedman's statistical ranking results of the optimal value of 23 benchmark test functions of the IROAVOA and other algorithms.

Figure	Friedman Rank								
	IROAVOA	**GWO**	**PSO**	**MVO**	**WOA**	**MFO**	**TSA**	**HHO**	**AVOA**
F_1	1.18	5.00	8.77	7.17	4.00	8.07	6.00	3.00	1.82
F_2	1.00	5.00	8.03	7.10	3.47	8.87	6.00	3.53	2.00
F_3	1.00	4.27	7.03	6.00	8.97	8.00	4.73	3.00	2.00
F_4	1.00	4.00	7.13	6.07	8.07	8.70	5.03	3.00	2.00
F_5	1.03	4.23	8.90	7.13	5.00	7.97	5.77	2.93	2.03
F_6	1.00	4.93	8.90	6.10	4.30	7.20	7.57	2.97	2.03
F_7	1.80	4.27	7.60	7.37	4.67	9.00	5.80	2.13	2.37
F_8	1.00	7.67	8.43	5.70	3.97	5.07	7.83	2.60	2.73
F_9	2.48	5.00	7.93	6.33	2.55	7.57	8.17	2.48	2.48
F_{10}	2.10	5.00	8.10	6.33	3.70	8.80	6.77	2.10	2.10
F_{11}	3.10	3.90	8.77	7.13	3.22	8.10	4.58	3.10	3.10
F_{12}	1.00	4.87	7.17	6.30	4.13	8.37	8.17	2.97	2.03
F_{13}	1.03	5.73	8.43	4.20	5.07	8.53	7.03	3.00	1.97
F_{14}	1.90	7.00	6.77	3.93	6.03	3.35	8.47	4.70	2.85
F_{15}	1.43	5.10	6.27	7.07	5.97	7.67	4.87	3.50	3.13
F_{16}	2.33	6.07	8.83	7.67	4.73	1.10	7.40	4.27	2.60
F_{17}	2.05	4.87	9.00	4.80	6.70	1.95	7.83	5.80	2.00
F_{18}	3.27	6.93	8.70	5.10	6.33	1.03	7.03	2.73	3.87
F_{19}	2.03	5.80	8.93	4.00	7.37	1.27	5.97	6.77	2.87
F_{20}	3.20	4.57	9.00	4.47	5.40	3.60	4.37	7.63	2.77
F_{21}	1.48	3.97	8.93	5.17	5.47	3.92	6.50	7.17	2.40
F_{22}	1.40	3.97	9.00	4.13	6.07	4.55	6.77	6.73	2.38
F_{23}	1.68	4.23	8.70	4.77	5.90	3.40	6.83	6.90	2.58
Avg Rank	1.72	5.06	8.23	5.83	5.26	5.92	6.50	4.04	2.44
Overall Rank	1	4	9	6	5	7	8	3	2

The Wilcoxon rank sum test is a non-parametric statistical method with a significance level set at 5%. If the p value is greater than 0.05, it means rejecting the null hypothesis, that is, the performance of the IROAVOA is inferior to other comparison algorithms; if the p value is less than 0.05, it means accepting the null hypothesis, indicating that the performance of the IROAVOA is better than other algorithms. When the p value is NaN, it means that the performance of the IROAVOA and the comparison algorithm are equivalent. Table 6 lists the Wilcoxon rank sum test results for each benchmark test function. Table 8 shows Statistical results of Wilcoxon rank result. In order to describe the results more clearly, (W/T/L) symbols are added to the last row of the table to illustrate the IROAVOA in the number of wins, draws, and the number of failures. According to the results, IROAVOA is better than AVOA on 15 benchmark test functions and better than GWO, PSO, MVO, WOA, MFO, TSA, and HHO on 23 groups of benchmark test functions. This confirms that the proposed improved algorithm has significant superiority.

Table 8. Statistical results of Wilcoxon rank sum test of 23 benchmark test functions between IROAVOA and other algorithms.

Figure	IROAVOA *VS.*							
	GWO	**PSO**	**MVO**	**WOA**	**MFO**	**TSA**	**HHO**	**AVOA**
F_1	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	3.45×10^{-7}
F_2	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F_3	2.51×10^{-11}	2.51×10^{-11}	2.51×10^{-11}	2.51×10^{-11}	2.51×10^{-11}	2.51×10^{-11}	2.51×10^{-11}	2.51×10^{-11}
F_4	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F_5	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	4.50×10^{-11}	6.72×10^{-10}
F_6	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F_7	3.34×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	6.52×10^{-9}	3.02×10^{-11}	3.02×10^{-11}	1.67×10^{-1}	1.09×10^{-1}
F_8	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F_9	1.20×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	3.34×10^{-1}	1.21×10^{-12}	1.21×10^{-12}	NaN	NaN
F_{10}	1.17×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	9.84×10^{-10}	1.21×10^{-12}	1.21×10^{-12}	NaN	NaN
F_{11}	2.79×10^{-3}	1.21×10^{-12}	1.21×10^{-12}	3.34×10^{-1}	1.21×10^{-12}	1.27×10^{-5}	NaN	NaN
F_{12}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F_{13}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	8.10×10^{-10}
F_{14}	2.34×10^{-11}	2.34×10^{-11}	2.34×10^{-11}	2.34×10^{-11}	6.83×10^{-1}	2.34×10^{-11}	2.34×10^{-11}	8.89×10^{-2}
F_{15}	6.01×10^{-8}	3.02×10^{-11}	3.02×10^{-11}	8.15×10^{-11}	3.01×10^{-11}	1.61×10^{-6}	2.44×10^{-9}	7.60×10^{-7}
F_{16}	5.14×10^{-12}	5.14×10^{-12}	5.14×10^{-12}	5.14×10^{-12}	8.99×10^{-11}	5.14×10^{-12}	1.56×10^{-11}	2.08×10^{-2}
F_{17}	2.36×10^{-12}	2.36×10^{-12}	2.36×10^{-12}	2.36×10^{-12}	1.61×10^{-1}	2.36×10^{-12}	2.36×10^{-12}	5.70×10^{-1}
F_{18}	3.47×10^{-10}	3.34×10^{-11}	2.02×10^{-8}	1.03×10^{-6}	2.56×10^{-11}	5.49×10^{-11}	1.17×10^{-2}	2.06×10^{-1}
F_{19}	2.15×10^{-11}	2.15×10^{-11}	2.15×10^{-11}	2.15×10^{-11}	1.14×10^{-9}	2.15×10^{-11}	2.15×10^{-11}	5.77×10^{-8}
F_{20}	2.75×10^{-3}	3.02×10^{-11}	1.06×10^{-3}	5.83×10^{-3}	1.56×10^{-1}	1.67×10^{-1}	1.85×10^{-9}	2.34×10^{-1}
F_{21}	1.94×10^{-11}	1.94×10^{-11}	1.94×10^{-11}	1.94×10^{-11}	3.46×10^{-1}	1.94×10^{-11}	1.94×10^{-11}	1.24×10^{-9}
F_{22}	2.15×10^{-11}	2.15×10^{-11}	2.15×10^{-11}	2.15×10^{-11}	1.17×10^{-1}	2.15×10^{-11}	2.15×10^{-11}	5.18×10^{-11}
F_{23}	2.48×10^{-11}	2.48×10^{-11}	2.48×10^{-11}	2.48×10^{-11}	7.26×10^{-2}	2.48×10^{-11}	2.48×10^{-11}	1.53×10^{-9}
(W/T/L)	23/0/0	23/0/0	23/0/0	23/0/0	23/0/0	23/0/0	20/3/0	15/3/5

5. Engineering Design Problems

5.1. Pressure Vessel Design Problem

The pressure vessel design problem is designed to minimize vessel manufacturing costs, and Figure 8 shows the design for this problem. The four design variables are shell thickness Ts (s_3), head thickness Th (s_4), inner radius R (s_1), and container length L (s_2, excluding the head), where TS and Th are integers of 0.0625 times, and R and L are continuous variables. The specific mathematical model is shown in Equation (27).

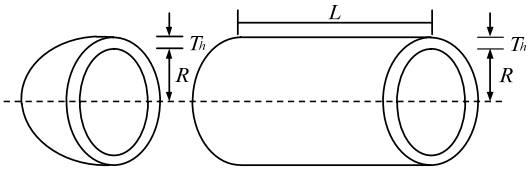

Figure 8. Pressure vessel design problem.

Compare the IROAVOA with AVOA and other optimization algorithms (such as GWO, WOA, HHO, and MVO). The population size and maximum number of iterations are set to 30 and 500, respectively. After running independently for 30 times, the optimal value is reached. Under the same experimental results, the results obtained are in Table 9. As can be seen from Table 9, IROAVOA has a smaller manufacturing cost and is more suitable compared with other algorithms. Therefore, IROAVOA can achieve optimal costs when solving such problems.

$$\begin{aligned}
&\min f(s) = 0.6224 s_1 s_3 s_4 + 1.7781 s_2 s_3^3 + 3.1661 s_1^2 s_4 + 19.84 s_1^2 s_3 \\
&g_1 = -s_1 + 0.0193 s_3 \leq 0, \\
&g_2 = -s_2 + 0.00954 s_3 \leq 0 \\
&g_3 = -\pi s_3^2 s_4 - \tfrac{4}{3}\pi s_3^3 + 1296000 \leq 0 \\
&g_4 = s_4 - 240 \leq 0 \\
&0 \leq s_1, s_2 \leq 99, \\
&0 \leq s_3, s_4 \leq 200.
\end{aligned} \tag{27}$$

Table 9. Comparison of pressure vessel design problem results of various algorithms.

Algorithm	T_s	T_h	L	R	$f(x)$	Worst	Mean	Std
IROAVOA	0.778271	0.384700	40.32492	199.9262	5885.50775	1.44×10^{11}	2.40×10^{9}	1.86×10^{10}
AVOA	0.780252	0.385680	40.42754	198.5041	5888.93379	5.89×10^{3}	1.96×10^{2}	1.08×10^{3}
GWO	0.779255	0.385775	40.37216	199.4523	5893.32353	1.82×10^{20}	2.06×10^{19}	4.56×10^{19}
WOA	0.851433	0.618424	43.22549	163.1540	6788.05596	1.66×10^{25}	6.19×10^{20}	2.17×10^{21}
MVO	0.866271	0.430271	44.86706	145.4594	6072.49998	1.37×10^{20}	1.70×10^{19}	3.68×10^{19}
HHO	0.947994	0.458334	47.89274	115.9952	6331.09005	6.33×10^{3}	2.11×10^{2}	1.16×10^{3}

5.2. Welded Beam Design Problem

The welded beam design problem is to reduce the cost in the welded beam design process. Figure 9 shows the design of the problem. The optimization problem can be described as finding the shear stress (s_1), bending stress (s_2), beam bending load (s_3), end deflection (s_4), and boundary condition constraints. The four design variables include the length of the beam (l), the height (t), thickness (b), and weld thickness (h), which minimize the design cost of manufacturing welded beams. The specific mathematical models are shown in Formulas (28) and (29).

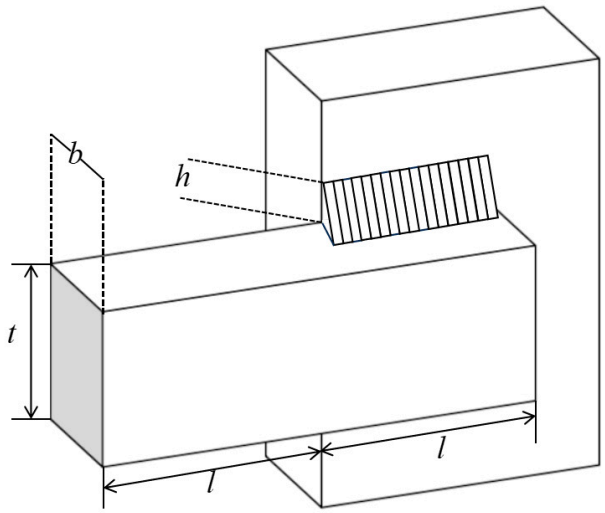

Figure 9. Welded beam design problem.

Compare the IROAVOA with AVOA and other optimization algorithms (such as GWO, WOA, HHO, and MVO). The population size and maximum number of iterations are set to 30 and 500, respectively. After running independently for 30 times, the optimal value is

obtained. It can be seen from Table 10 that IROAVOA achieves better results than other algorithms and can, therefore, be better applied.

$$\begin{aligned}
&\cos tf(s) = 1.1047s_1^2 s_2 + 0.04811 s_3 s_4 (14.0 + s_2)\\
&\mathbf{y}_1(s) = \tau(s) - 13600 \leq 0,\\
&\mathrm{y}_2(s) = \sigma(s) - 30000 \leq 0,\\
&\mathbf{y}_3(s) = \delta(s) - 0.25 \leq 0,\\
&\mathbf{y}_4 = s_1 - s_4 \leq 0,\\
&\mathbf{y}_5 = p - p_c \leq 0,\\
&\mathrm{y}_6 = 0.125 - s_1 \leq 0,\\
&\mathbf{y}_7 = 1.10471 s_1^2 + 0.04811 s_3 s_4 (14. + s_2) - 5.0 \leq 0,\\
&0.1 \leq s_1, s_4 \leq 2.0,\\
&0.1 \leq \mathrm{s}_2, s_3 \leq 10.0.
\end{aligned} \tag{28}$$

$$\begin{aligned}
&\tau = \sqrt{\tau_1 + 2\tau_1\tau_2(\tfrac{s_2}{2r}) + \tau_2^2}, \tau_1 = \tfrac{P_d}{s_1 s_2 \sqrt{2}},\\
&m = p_d(i + \tfrac{s_2}{2}), j = 2\left\{\sqrt{2} s_1 s_2 [\tfrac{s_2^2}{12} + \left(\tfrac{s_1 + s_3}{2}\right)^2]\right\},\\
&r = \sqrt{\tfrac{s_2^2}{4} + \left(\tfrac{s_1 + s_3}{2}\right)^2}, \sigma = \tfrac{6 p_d i}{s_4 s_3^2}, \delta = \tfrac{6 p_d i^3}{n s_3^2 s_4},\\
&p_c = \tfrac{4.013 n \sqrt{\frac{s_3^2 s_4^6}{36}}}{i^2}(1 - \tfrac{s_3}{2i}\sqrt{\tfrac{n}{4m}}),\\
&m = 12 \times 10^6, n = 30 \times 10^6,\\
&p_d = 6000 lb, i = 14 in, \tau_2 = \tfrac{mr}{j}.
\end{aligned} \tag{29}$$

Table 10. Comparison of welded beam design problem results of various algorithms.

Algorithm	τ (s_1)	θ (s_2)	P_c (s_3)	δ (s_4)	f (x)	Worst	Mean	Std
IROAVOA	0.20563	3.2552	9.0356	0.20578	1.6955	9.43×10^{14}	1.57×10^{13}	1.22×10^{14}
AVOA	0.20295	3.3033	9.0370	0.20573	1.6979	1.71×10^{17}	5.71×10^{15}	3.13×10^{16}
GWO	0.20538	3.2591	9.0412	0.20581	1.6969	9.05×10^{15}	7.41×10^{14}	2.28×10^{15}
WOA	0.17181	3.8612	9.4729	0.20365	1.7836	2.55×10^{30}	4.58×10^{16}	1.53×10^{17}
MVO	0.20330	3.3073	9.0390	0.20579	1.6999	7.95×10^{16}	9.41×10^{15}	1.77×10^{16}
HHO	0.19829	3.4453	8.9978	0.20751	1.7167	1.72×10^{00}	5.72×10^{-2}	3.13×10^{-1}

6. Conclusions

In view of the shortcomings of the algorithm, such as poor global search ability and poor ability to balance exploration and exploitation, this paper introduces an improved IROAVOA based on a combination of random opposition-based learning and disturbance factors. Opposition learning can generate opposite points for candidate solutions, and adding randomness can remove the fixed distance between the generated reverse solution and the current solution, further expanding the search space of the algorithm. Therefore, in the initial stage, random opposition-based learning can increase the initial generation of African vultures to enhance the population diversity and randomness of the algorithm and promote a wider search of the solution space, thereby enhancing the algorithm's ability to delve into a wider range of potential solutions. In the exploration stage, the perturbation operator helps African vultures avoid the dilemma of local search during navigation, improves the algorithm's ability to escape from local optima, and balances the exploration and exploitation stages. In order to verify the effectiveness of IROAVOA, simulations were conducted on 23 benchmark test functions and the IEEE CEC2019 test suite, and the exploration and exploitation capabilities, as well as convergence of the algorithm, were analyzed. The results show that IROAVOA outperforms traditional AVOA, two AVOA variants (IAVOA, ROAVOA), and seven other optimization algorithms. In ablation experiments, random opposition-based learning strategy and perturbation factor

effectively improved the exploitation of AVOA and its ability to balance exploration and exploitation since adding disturbance factors increases the time complexity of the algorithm. In subsequent research work, we will further reduce the time complexity of the algorithm and apply it to more practical engineering optimization problems.

Author Contributions: Conceptualization, Methodology, Software, Writing—Original Draft, J.H.; Visualization, Investigation, Writing—Reviewing, X.L.; Investigation, Validation, Funding Acquisition, T.W.; Supervision, C.L.; Funding Acquisition, Z.W.; Funding Acquisition, X.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Jiangmen Science and Technology Commissioner's scientific research cooperation project, grant number 2023760300180008278, the Jiangmen Science and Technology Plan Project under Grant 2022JC01021, and, in part, by the domestic development and industrialization of water quality online monitoring instruments and core accessories, grant number 2022ZAZX3034.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding authors.

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

Appendix A

Table A1. Results of unimodal benchmark functions.

Benchmark Function	Index	AVOA	ROAVOA	IAVOA	IROAVOA
F_1	Best	6.45×10^{-292}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	Mean	2.18×10^{-293}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	Std	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
F_2	Best	5.49×10^{-145}	5.04×10^{-173}	1.37×10^{-162}	2.86×10^{-192}
	Mean	1.86×10^{-146}	8.40×10^{-175}	4.55×10^{-164}	4.76×10^{-194}
	Std	1.02×10^{-145}	0.00×10^{0}	2.33×10^{-163}	0.00×10^{0}
F_3	Best	1.78×10^{-222}	1.06×10^{-280}	1.08×10^{-240}	5.74×10^{-284}
	Mean	6.02×10^{-224}	1.79×10^{-282}	3.84×10^{-242}	9.85×10^{-286}
	Std	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
F_4	Best	5.19×10^{-149}	5.32×10^{-166}	4.74×10^{-163}	7.36×10^{-193}
	Mean	1.75×10^{-150}	8.86×10^{-168}	1.59×10^{-164}	1.23×10^{-194}
	Std	9.60×10^{-150}	0.00×10^{0}	7.41×10^{-164}	0.00×10^{0}
F_5	Best	5.71×10^{-5}	1.06×10^{-5}	2.11×10^{-5}	3.49×10^{-6}
	Mean	3.46×10^{6}	2.89×10^{6}	1.35×10^{7}	1.19×10^{7}
	Std	1.90×10^{7}	2.24×10^{7}	7.39×10^{7}	9.21×10^{7}
F_6	Best	1.73×10^{-3}	8.48×10^{-9}	1.70×10^{-7}	5.36×10^{-9}
	Mean	5.98×10^{-5}	3.89×10^{-7}	9.64×10^{-7}	7.69×10^{-7}
	Std	3.27×10^{-4}	3.01×10^{-6}	5.28×10^{-6}	5.95×10^{-6}
F_7	Best	1.36×10^{-4}	1.06×10^{-4}	1.35×10^{-4}	6.17×10^{-5}
	Mean	1.76×10^{-2}	8.45×10^{-3}	1.87×10^{-2}	8.14×10^{-3}
	Std	9.64×10^{-2}	6.55×10^{-2}	1.03×10^{-1}	6.31×10^{-2}

Table A2. Results of multimodal benchmark functions.

Benchmark Function	Index	AVOA	ROAVOA	IAVOA	IROAVOA
F_8	Best	-1.23×10^4	-1.25×10^4	-1.26×10^4	-1.26×10^4
	Mean	-4.10×10^2	-2.09×10^2	-4.19×10^2	-2.09×10^2
	Std	2.25×10^3	1.62×10^3	2.29×10^3	1.62×10^3
F_9	Best	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0
	Mean	3.79×10^{-16}	1.26×10^{-16}	3.16×10^{-16}	1.58×10^{-16}
	Std	2.08×10^{-15}	9.78×10^{-16}	1.73×10^{-15}	1.22×10^{-15}
F_{10}	Best	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}
	Mean	1.01×10^{-16}	3.85×10^{-17}	1.01×10^{-16}	3.85×10^{-17}
	Std	5.51×10^{-16}	2.98×10^{-16}	5.51×10^{-16}	2.98×10^{-16}
F_{11}	Best	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0
	Mean	6.17×10^{-19}	4.32×10^{-19}	4.93×10^{-19}	2.47×10^{-19}
	Std	3.38×10^{-18}	3.34×10^{-18}	2.70×10^{-18}	1.91×10^{-18}
F_{12}	Best	3.01×10^{-8}	6.48×10^{-10}	8.90×10^{-9}	2.31×10^{-10}
	Mean	1.93×10^{-7}	1.26×10^{-8}	2.07×10^{-7}	2.78×10^{-8}
	Std	1.06×10^{-6}	9.78×10^{-8}	1.13×10^{-6}	2.15×10^{-7}
F_{13}	Best	7.48×10^{-8}	5.85×10^{-9}	3.51×10^{-8}	2.69×10^{-9}
	Mean	6.28×10^7	2.65×10^7	5.69×10^7	2.17×10^7
	Std	3.44×10^8	2.05×10^8	3.12×10^8	1.68×10^8

Table A3. Results of fixed-dimensions multimodal benchmark functions.

Benchmark Function	Index	AVOA	ROAVOA	IAVOA	IROAVOA
F_{14}	Best	1.10×10^0	9.98×10^{-1}	9.98×10^{-1}	9.98×10^{-1}
	Mean	3.66×10^{-2}	1.66×10^{-2}	3.33×10^{-2}	1.66×10^{-2}
	Std	2.00×10^{-1}	1.29×10^{-1}	1.82×10^{-1}	1.29×10^{-1}
F_{15}	Best	4.42×10^{-4}	3.58×10^{-4}	3.84×10^{-4}	3.13×10^{-4}
	Mean	1.48×10^{-5}	6.10×10^{-6}	1.29×10^{-5}	5.37×10^{-6}
	Std	8.11×10^{-5}	4.73×10^{-5}	7.07×10^{-5}	4.16×10^{-5}
F_{16}	Best	-1.03×10^0	-1.03×10^0	-1.03×10^0	-1.03×10^0
	Mean	-3.44×10^{-2}	-1.72×10^{-2}	-3.44×10^{-2}	-1.72×10^{-2}
	Std	1.88×10^{-1}	1.33×10^{-1}	1.88×10^{-1}	1.33×10^{-1}
F_{17}	Best	3.98×10^{-1}	3.98×10^{-1}	3.98×10^{-1}	3.98×10^{-1}
	Mean	1.33×10^{-2}	6.63×10^{-3}	1.33×10^{-2}	6.63×10^{-3}
	Std	7.27×10^{-2}	5.14×10^{-2}	7.27×10^{-2}	5.14×10^{-2}
F_{18}	Best	3.00×10^0	3.00×10^0	3.00×10^0	3.00×10^0
	Mean	1.00×10^{-1}	5.00×10^{-2}	1.00×10^{-1}	5.00×10^{-2}
	Std	5.48×10^{-1}	3.88×10^{-1}	5.48×10^{-1}	3.87×10^{-1}
F_{19}	Best	-3.86×10^0	-3.86×10^0	-3.86×10^0	-3.86×10^0
	Mean	-1.27×10^{-1}	-6.36×10^{-2}	-1.29×10^{-1}	-6.34×10^{-2}
	Std	6.96×10^{-1}	4.93×10^{-1}	7.05×10^{-1}	4.91×10^{-1}
F_{20}	Best	-3.29×10^0	-3.28×10^0	-3.28×10^0	-3.27×10^0
	Mean	-1.09×10^{-1}	-5.28×10^{-2}	-1.09×10^{-1}	-5.45×10^{-2}
	Std	5.99×10^{-1}	4.09×10^{-1}	5.98×10^{-1}	4.22×10^{-1}
F_{21}	Best	-1.02×10^1	-1.02×10^1	-1.02×10^1	-1.02×10^1
	Mean	-3.38×10^{-1}	-1.69×10^{-1}	-3.38×10^{-1}	-1.69×10^{-1}
	Std	1.85×10^0	1.31×10^0	1.85×10^0	1.31×10^0
F_{22}	Best	-1.04×10^1	-1.04×10^1	-1.04×10^1	-1.04×10^1
	Mean	-3.47×10^{-1}	-1.73×10^{-1}	-3.47×10^{-1}	-1.73×10^{-1}
	Std	1.90×10^0	1.34×10^0	1.90×10^0	1.34×10^0
F_{23}	Best	-1.05×10^1	-1.05×10^1	-1.05×10^1	-1.05×10^1
	Mean	-3.51×10^{-1}	-1.76×10^{-1}	-3.51×10^{-1}	-1.75×10^{-1}
	Std	1.92×10^0	1.36×10^0	1.92×10^0	1.36×10^0

Table A4. Unimodal benchmark function results of IROAVOA and other algorithms.

Function	Index	IROAVOA	GWO	PSO	MVO	WOA	MFO	TSA	HHO
F_1	Best	0.00×10^{0}	2.17×10^{-27}	3.79×10^{2}	1.34×10^{0}	1.70×10^{-73}	2.01×10^{3}	2.07×10^{-21}	6.35×10^{-95}
	Mean	0.00×10^{0}	2.17×10^{-27}	1.26×10^{3}	1.44×10^{0}	1.87×10^{-73}	2.01×10^{3}	2.29×10^{-21}	2.12×10^{-96}
	Std	0.00×10^{0}	6.03×10^{-31}	3.01×10^{2}	4.48×10^{-2}	6.07×10^{-75}	3.29×10^{0}	1.09×10^{-22}	1.16×10^{-95}
F_2	Best	2.29×10^{-188}	1.15×10^{-16}	8.62×10^{0}	4.56×10^{0}	4.93×10^{-51}	3.68×10^{1}	1.27×10^{-13}	1.90×10^{-50}
	Mean	3.81×10^{-190}	1.15×10^{-16}	1.59×10^{1}	4.58×10^{0}	5.05×10^{-51}	3.68×10^{1}	1.37×10^{-13}	6.35×10^{-52}
	Std	0.00×10^{0}	1.69×10^{-20}	2.10×10^{0}	1.00×10^{-2}	4.48×10^{-53}	2.69×10^{-2}	4.55×10^{-15}	3.48×10^{-51}
F_3	Best	6.02×10^{-293}	1.39×10^{-5}	1.31×10^{3}	2.23×10^{2}	4.18×10^{4}	2.06×10^{4}	1.45×10^{-4}	2.34×10^{-74}
	Mean	1.03×10^{-294}	1.39×10^{-5}	1.71×10^{4}	2.24×10^{2}	4.19×10^{4}	2.28×10^{4}	4.12×10^{-4}	7.82×10^{-76}
	Std	0.00×10^{0}	4.75×10^{-9}	1.74×10^{4}	5.11×10^{-1}	2.06×10^{2}	4.29×10^{3}	3.13×10^{-4}	4.28×10^{-75}
F_4	Best	7.98×10^{-183}	9.21×10^{-7}	8.22×10^{0}	2.11×10^{0}	5.10×10^{1}	7.00×10^{1}	3.71×10^{-1}	1.70×10^{-48}
	Mean	1.33×10^{-184}	9.22×10^{-7}	1.56×10^{1}	2.14×10^{0}	5.11×10^{1}	8.41×10^{1}	4.68×10^{-1}	5.67×10^{-50}
	Std	0.00×10^{0}	5.33×10^{-10}	2.28×10^{0}	1.32×10^{-2}	1.05×10^{-1}	1.22×10^{1}	5.37×10^{-2}	3.11×10^{-49}
F_5	Best	2.96×10^{-6}	2.68×10^{1}	1.41×10^{4}	3.96×10^{2}	2.81×10^{1}	1.63×10^{4}	2.87×10^{1}	1.28×10^{-2}
	Mean	8.36×10^{6}	2.68×10^{1}	1.29×10^{5}	3.97×10^{2}	2.81×10^{1}	3.43×10^{4}	3.03×10^{1}	4.28×10^{-4}
	Std	6.47×10^{7}	1.78×10^{-4}	5.57×10^{4}	6.94×10^{-1}	7.88×10^{-5}	8.64×10^{4}	1.87×10^{0}	2.34×10^{-3}
F_6	Best	3.63×10^{-9}	8.19×10^{-1}	3.98×10^{2}	1.25×10^{0}	4.12×10^{-1}	1.67×10^{3}	3.86×10^{0}	9.02×10^{-5}
	Mean	4.51×10^{-7}	8.19×10^{-1}	1.26×10^{3}	1.34×10^{0}	4.12×10^{-1}	1.67×10^{3}	3.97×10^{0}	3.01×10^{-6}
	Std	3.49×10^{-6}	2.09×10^{-5}	3.08×10^{2}	4.49×10^{-2}	1.53×10^{-4}	6.33×10^{0}	6.39×10^{-2}	1.65×10^{-5}
F_7	Best	6.72×10^{-5}	1.79×10^{-3}	3.83×10^{-2}	3.62×10^{-2}	3.43×10^{-3}	3.89×10^{0}	1.01×10^{-2}	1.33×10^{-4}
	Mean	8.64×10^{-3}	5.09×10^{-1}	6.29×10^{-1}	5.30×10^{-1}	5.06×10^{-1}	4.92×10^{0}	5.01×10^{-1}	4.43×10^{-6}
	Std	6.69×10^{-2}	2.88×10^{-1}	2.88×10^{-1}	2.82×10^{-1}	2.87×10^{-1}	1.93×10^{0}	2.85×10^{-1}	2.43×10^{-5}

Table A5. Fixed-dimension test function results of IROAVOA and other algorithms.

Function	Index	IROAVOA	GWO	PSO	MVO	WOA	MFO	TSA	HHO
F_8	Best	-1.26×10^{4}	-5.98×10^{3}	-5.35×10^{3}	-7.82×10^{3}	-9.90×10^{3}	-8.72×10^{3}	-5.64×10^{3}	-1.26×10^{4}
	Mean	-2.09×10^{2}	-5.88×10^{3}	-3.45×10^{3}	-7.82×10^{3}	-9.90×10^{3}	-8.72×10^{3}	-2.78×10^{3}	-4.19×10^{2}
	Std	1.62×10^{3}	1.68×10^{0}	1.00×10^{3}	8.29×10^{-1}	1.06×10^{1}	8.78×10^{-2}	7.47×10^{2}	2.29×10^{3}
F_9	Best	0.00×10^{0}	3.17×10^{0}	1.62×10^{2}	1.15×10^{2}	7.58×10^{-15}	1.58×10^{2}	1.87×10^{2}	0.00×10^{0}
	Mean	1.58×10^{-16}	3.17×10^{0}	3.17×10^{2}	1.15×10^{2}	8.91×10^{-15}	1.58×10^{2}	3.03×10^{2}	0.00×10^{0}
	Std	1.22×10^{-15}	5.95×10^{-4}	4.08×10^{1}	2.13×10^{-2}	3.22×10^{-15}	5.71×10^{-2}	3.10×10^{1}	0.00×10^{0}
F_{10}	Best	8.88×10^{-16}	1.03×10^{-13}	5.87×10^{0}	1.85×10^{0}	4.32×10^{-15}	1.39×10^{1}	2.30×10^{0}	8.88×10^{-16}
	Mean	4.24×10^{-17}	1.07×10^{-13}	8.46×10^{0}	1.87×10^{0}	4.32×10^{-15}	1.39×10^{1}	2.57×10^{0}	2.96×10^{-17}
	Std	3.29×10^{-16}	0.00×10^{0}	6.75×10^{-1}	6.92×10^{-3}	0.00×10^{0}	3.88×10^{-2}	7.48×10^{-2}	1.62×10^{-16}
F_{11}	Best	0.00×10^{0}	3.09×10^{-3}	4.59×10^{0}	8.55×10^{-1}	0.00×10^{0}	2.20×10^{1}	8.47×10^{-3}	0.00×10^{0}
	Mean	4.32×10^{-19}	3.09×10^{-3}	1.24×10^{1}	8.81×10^{-1}	1.48×10^{-18}	2.22×10^{1}	1.51×10^{-1}	0.00×10^{0}
	Std	3.34×10^{-18}	1.54×10^{-6}	2.62×10^{0}	1.18×10^{-2}	5.23×10^{-18}	2.21×10^{-1}	1.26×10^{-1}	0.00×10^{0}
F_{12}	Best	2.42×10^{-10}	5.05×10^{-2}	3.88×10^{0}	2.19×10^{0}	2.79×10^{-2}	9.79×10^{0}	8.08×10^{0}	5.81×10^{-6}
	Mean	5.14×10^{-8}	5.05×10^{-2}	6.31×10^{2}	2.20×10^{0}	2.79×10^{-2}	6.91×10^{5}	1.61×10^{3}	1.94×10^{-7}
	Std	3.98×10^{-7}	2.74×10^{-6}	2.59×10^{3}	3.71×10^{-3}	2.01×10^{-5}	3.45×10^{6}	4.86×10^{3}	1.06×10^{-6}
F_{13}	Best	3.02×10^{-9}	6.37×10^{-1}	1.70×10^{1}	1.76×10^{-1}	5.50×10^{-1}	4.62×10^{3}	3.17×10^{0}	8.71×10^{-5}
	Mean	1.55×10^{7}	6.37×10^{-1}	1.44×10^{4}	1.84×10^{-1}	5.50×10^{-1}	1.91×10^{6}	3.79×10^{0}	2.90×10^{-6}
	Std	1.20×10^{8}	5.20×10^{-5}	2.41×10^{4}	3.91×10^{-3}	1.37×10^{-3}	8.59×10^{6}	2.13×10^{-1}	1.59×10^{-5}

Table A6. Fixed-dimensions multimodal benchmark function results of IROAVOA and other algorithms.

Function	Index	IROAVOA	GWO	PSO	MVO	WOA	MFO	TSA	HHO
F_{14}	Best	9.98×10^{-1}	5.24×10^{0}	3.13×10^{0}	9.98×10^{-1}	2.73×10^{0}	2.19×10^{0}	8.43×10^{0}	1.62×10^{0}
	Mean	1.66×10^{-2}	5.24×10^{0}	3.55×10^{2}	9.98×10^{-1}	2.86×10^{0}	2.19×10^{0}	1.46×10^{2}	5.41×10^{-2}
	Std	1.29×10^{-1}	1.46×10^{-7}	1.77×10^{2}	2.29×10^{-9}	7.04×10^{-1}	8.28×10^{-16}	1.25×10^{2}	2.96×10^{-1}
F_{15}	Best	3.49×10^{-4}	3.77×10^{-3}	1.37×10^{-3}	8.11×10^{-3}	6.27×10^{-4}	1.83×10^{-3}	7.08×10^{-3}	3.89×10^{-4}
	Mean	5.87×10^{-6}	3.77×10^{-3}	4.44×10^{2}	8.12×10^{-3}	6.29×10^{-4}	1.83×10^{-3}	9.34×10^{1}	1.30×10^{-5}
	Std	4.54×10^{-5}	4.34×10^{-6}	2.39×10^{3}	1.21×10^{-5}	7.52×10^{-6}	7.75×10^{-7}	5.11×10^{2}	7.09×10^{-5}
F_{16}	Best	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}
	Mean	-1.72×10^{-2}	-1.03×10^{0}	-9.27×10^{-2}	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	-9.15×10^{-1}	-3.44×10^{-2}
	Std	1.33×10^{-1}	3.10×10^{-6}	1.06×10^{0}	8.62×10^{-6}	1.01×10^{-4}	6.37×10^{-16}	1.91×10^{-1}	1.88×10^{-1}
F_{17}	Best	3.98×10^{-1}	3.98×10^{-1}	1.10×10^{0}	3.98×10^{-1}	3.98×10^{-1}	3.98×10^{-1}	3.98×10^{-1}	3.98×10^{-1}
	Mean	6.63×10^{-3}	3.98×10^{-1}	2.14×10^{1}	3.98×10^{-1}	4.02×10^{-1}	3.98×10^{-1}	5.18×10^{0}	1.33×10^{-2}
	Std	5.14×10^{-2}	1.72×10^{-4}	4.48×10^{1}	5.54×10^{-5}	1.17×10^{-2}	4.27×10^{-16}	5.78×10^{0}	7.26×10^{-2}
F_{18}	Best	3.00×10^{0}	3.00×10^{0}	3.90×10^{0}	3.00×10^{0}	3.00×10^{0}	3.00×10^{0}	1.50×10^{1}	3.00×10^{0}
	Mean	5.00×10^{-2}	3.00×10^{0}	1.83×10^{1}	3.00×10^{0}	3.00×10^{0}	3.00×10^{0}	1.81×10^{2}	1.00×10^{-1}
	Std	3.87×10^{-1}	3.57×10^{-4}	2.02×10^{1}	8.06×10^{-5}	6.34×10^{-3}	6.52×10^{-15}	5.52×10^{2}	5.48×10^{-1}
F_{19}	Best	-3.86×10^{0}	-3.86×10^{0}	-3.71×10^{0}	-3.86×10^{0}	-3.85×10^{0}	-3.86×10^{0}	-3.86×10^{0}	-3.86×10^{0}
	Mean	-6.41×10^{-2}	-3.86×10^{0}	-2.29×10^{0}	-3.86×10^{0}	-3.85×10^{0}	-3.86×10^{0}	-3.05×10^{0}	-1.29×10^{-1}
	Std	4.96×10^{-1}	5.97×10^{-5}	7.57×10^{-1}	4.12×10^{-6}	1.84×10^{-3}	2.68×10^{-15}	6.62×10^{-1}	7.05×10^{-1}
F_{20}	Best	-3.27×10^{0}	-3.25×10^{0}	-1.96×10^{0}	-3.27×10^{0}	-3.19×10^{0}	-3.23×10^{0}	-3.22×10^{0}	-3.10×10^{0}
	Mean	-5.42×10^{-2}	-3.25×10^{0}	-3.12×10^{-1}	-3.27×10^{0}	-3.19×10^{0}	-3.23×10^{0}	-2.59×10^{0}	-1.03×10^{-1}
	Std	4.20×10^{-1}	2.16×10^{-5}	4.51×10^{-1}	6.73×10^{-6}	3.83×10^{-4}	1.89×10^{-15}	4.24×10^{-1}	5.67×10^{-1}
F_{21}	Best	-1.02×10^{1}	-9.48×10^{0}	-1.73×10^{0}	-7.96×10^{0}	-8.93×10^{0}	-6.46×10^{0}	-6.78×10^{0}	-5.36×10^{0}
	Mean	-1.69×10^{-1}	-9.47×10^{0}	-3.56×10^{-1}	-7.96×10^{0}	-8.88×10^{0}	-6.46×10^{0}	-1.52×10^{0}	-1.79×10^{-1}
	Std	1.31×10^{0}	3.26×10^{-3}	1.53×10^{-1}	8.92×10^{-4}	1.55×10^{-1}	3.12×10^{-15}	6.74×10^{-1}	9.78×10^{-1}
F_{22}	Best	-1.04×10^{1}	-1.00×10^{1}	-1.45×10^{0}	-7.99×10^{0}	-6.68×10^{0}	-6.82×10^{0}	-5.89×10^{0}	-5.25×10^{0}
	Mean	-1.73×10^{-1}	-1.00×10^{1}	-3.98×10^{-1}	-7.99×10^{0}	-6.66×10^{0}	-6.82×10^{0}	-1.50×10^{0}	-1.75×10^{-1}
	Std	1.34×10^{0}	3.24×10^{-3}	1.91×10^{-1}	1.09×10^{-3}	4.12×10^{-2}	1.73×10^{-15}	7.71×10^{-1}	9.59×10^{-1}
F_{23}	Best	-1.05×10^{1}	-1.01×10^{1}	-1.77×10^{0}	-9.10×10^{0}	-7.04×10^{0}	-7.21×10^{0}	-6.67×10^{0}	-5.12×10^{0}
	Mean	-1.75×10^{-1}	-1.01×10^{1}	-4.89×10^{-1}	-9.10×10^{0}	-7.02×10^{0}	-7.21×10^{0}	-1.81×10^{0}	-1.71×10^{-1}
	Std	1.36×10^{0}	3.00×10^{-3}	2.17×10^{-1}	1.14×10^{-3}	5.03×10^{-2}	1.93×10^{-15}	8.83×10^{-1}	9.35×10^{-1}

References

1. Kumar, V.; Chhabra, J.K.; Kumar, D. Parameter adaptive harmony search algorithm for unimodal and multimodal optimization problems. *J. Comput. Sci.* **2014**, *5*, 144–155. [CrossRef]
2. Xiao, Y.; Guo, Y.; Cui, H.; Wang, Y.; Li, J.; Zhang, Y. IHAOAVOA: An improved hybrid aquila optimizer and African vultures optimization algorithm for global optimization problems. *Math. Biosci. Eng.* **2022**, *19*, 10963–11017. [CrossRef]
3. Dhargupta, S.; Ghosh, M.; Mirjalili, S.; Sarkar, R. Selective opposition based grey wolf optimization. *Expert Syst. Appl.* **2020**, *151*, 113389. [CrossRef]
4. Pršić, D.; Nedić, N.; Stojanović, V. A nature inspired optimal control of pneumatic-driven parallel robot platform. *Proc. Inst. Mech. Eng. Part C J. Mech. Eng. Sci.* **2017**, *231*, 59–71.
5. Holland, J.H. Genetic algorithms. *Sci. Am.* **1992**, *267*, 66–73. [CrossRef]
6. Das, S.; Suganthan, P.N. Differential evolution: A survey of the state-of-the-art. *IEEE Trans. Evol. Comput.* **2010**, *15*, 4–31. [CrossRef]
7. Columbu, A.; Fuentes, R.D.; Frassu, S. Uniform-in-time boundedness in a class of local and nonlocal nonlinear attraction–repulsion chemotaxis models with logistics. *Nonlinear Anal. Real World Appl.* **2024**, *79*, 104135. [CrossRef]
8. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. GSA: A gravitational search algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]
9. Mirjalili, S.; Mirjalili, S.M.; Hatamlou, A. Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Comput. Appl.* **2016**, *27*, 495–513. [CrossRef]
10. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95-International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.
11. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
12. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
13. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl.-Based Syst.* **2015**, *89*, 228–249. [CrossRef]
14. Kaur, S.; Awasthi, L.K.; Sangal, A.L.; Dhiman, G. Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization. *Eng. Appl. Artif. Intell.* **2020**, *90*, 103541. [CrossRef]
15. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H. Harris hawks optimization: Algorithm and applications. *Future Gener. Comput. Syst.* **2019**, *97*, 849–872.

16. Abdollahzadeh, B.; Gharehchopogh, F.S.; Mirjalili, S. African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems. *Comput. Ind. Eng.* **2021**, *158*, 107408.
17. Salah, B.; Hasanien, H.M.; Ghali, F.M.A.; Alsayed, Y.M.; Abdel Aleem, S.H.E.; El-Shahat, A. African Vulture Optimization-Based Optimal Control Strategy for Voltage Control of Islanded DC Microgrids. *Sustainability* **2022**, *14*, 11800. [CrossRef]
18. Bagal, H.A.; Soltanabad, Y.N.; Dadjuo, M.; Wakil, K.; Zare, M.; Mohammed, A.S. SOFC model parameter identification by means of Modified African Vulture Optimization algorithm. *Energy Rep.* **2021**, *7*, 7251–7260.
19. He, Z.; Tang, B.; Luan, F. An improved African vulture optimization algorithm for dual-resource constrained multi-objective flexible job shop scheduling problems. *Sensors* **2022**, *23*, 90. [CrossRef] [PubMed]
20. Vashishtha, G.; Chauhan, S.; Kumar, A.; Kumar, R. An ameliorated African vulture optimization algorithm to diagnose the rolling bearing defects. *Meas. Sci. Technol.* **2022**, *33*, 075013.
21. Singh, N.; Houssein, E.H.; Mirjalili, S.; Cao, Y.; Selvachandran, G. An efficient improved African vultures optimization algorithm with dimension learning hunting for traveling salesman and large-scale optimization applications. *Int. J. Intell. Syst.* **2022**, *37*, 12367–12421.
22. Diab, A.A.Z.; Tolba, M.A.; El-Rifaie, A.M.; Denis, K.A. Photovoltaic parameter estimation using honey badger algorithm and African vulture optimization algorithm. *Energy Rep.* **2022**, *8*, 384–393.
23. Zheng, R.; Hussien, A.G.; Qaddoura, R.; Jia, H.; Abualigah, L.; Wang, S.; Saber, A. A multi-strategy enhanced African vultures optimization algorithm for global optimization problems. *J. Comput. Des. Eng.* **2023**, *10*, 329–356.
24. Liu, R.; Wang, T.; Zhou, J.; Hao, X.; Xu, Y.; Qiu, J. Improved African vulture optimization algorithm based on quasi-oppositional differential evolution operator. *IEEE Access* **2022**, *10*, 95197–95218.
25. Fan, J.; Li, Y.; Wang, T. An improved African vultures optimization algorithm based on tent chaotic mapping and time-varying mechanism. *PLoS ONE* **2021**, *16*, e0260725.
26. Toledo, S.; Shohami, D.; Schiffner, I.; Lourie, E.; Orchan, Y.; Bartan, Y.; Nathan, R. Cognitive map–based navigation in wild bats revealed by a new high-throughput tracking system. *Science* **2020**, *369*, 188–193.
27. Pearson, K. VII. Mathematical contributions to the theory of evolution.—III. Regression, heredity, and panmixia. *Philos. Trans. R. Soc. London. Ser. A Contain. Pap. A Math. Or Phys. Character* **1896**, *187*, 253–318.
28. Brownlee, J. XIV.—The Mathematical Theory of Random Migration and Epidemic Distribution. *Proc. R. Soc. Edinb.* **1912**, *31*, 262–289.
29. Meyer, P.G.; Cherstvy, A.G.; Seckler, H.; Hering, R.; Blaum, N.; Jeltsch, F.; Metzler, R. Directedeness, correlations, and daily cycles in springbok motion: From data via stochastic models to movement prediction. *Phys. Rev. Res.* **2023**, *5*, 043129.
30. Tizhoosh, H.R. Opposition-based learning: A new scheme for machine intelligence. In Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06), Vienna, Austria, 28–30 November 2005; Volume 1, pp. 695–701.
31. Mandavi, S.; Rahnamayan, S.; Deb, K. Opposition based learning: A literature review. *Swarm Evol. Comput.* **2018**, *39*, 1–23.
32. Long, W.; Jiao, J.; Liang, X.; Cai, S.; Xu, M. A random opposition-based learning grey wolf optimizer. *IEEE Access* **2019**, *7*, 113810–113825.
33. Yao, X.; Liu, Y.; Lin, G. Evolutionary programming made faster. *IEEE Trans. Evol. Comput.* **1999**, *3*, 82–102.
34. Price, K.V.; Awad, N.H.; Ali, M.Z.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for the 100-Digit Challenge Special Session and Competition on Single Objective Numerical Optimization*; Technical Report; Nanyang Technological University: Singapore, 2018.
35. García, S.; Fernández, A.; Luengo, J.; Herrera, F. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Inf. Sci.* **2010**, *180*, 2044–2064.
36. Theodorsson-Norheim, E. Friedman and Quade tests: BASIC computer program to perform nonparametric two-way analysis of variance and multiple comparisons on ranks of several related samples. *Comput. Biol. Med.* **1987**, *17*, 85–99. [PubMed]