Electronic Research Archive

http://www.aimspress.com/journal/era

ERA, 32(3): 2033–2074.
DOI: 10.3934/era.2024093
Received: 27 December 2023
Revised: 23 February 2024
Accepted: 28 February 2024
Published: 07 March 2024

Research article

IOOA: A multi-strategy fusion improved Osprey Optimization Algorithm for global optimization

Xiaodong Wen[1,†], Xiangdong Liu[2,†], Cunhui Yu[2], Haoning Gao[2], Jing Wang[2], Yongji Liang[1], Jiangli Yu[1,*] and Yan Bai[1,*]

[1] College of Electrical Engineering, Hebei University of Architecture, Zhangjiakou 075000, China

[2] Department of Energy Engineering, Hebei University of Architecture, Zhangjiakou 075000, China

† The authors contributed equally to this work.

* **Correspondence:** Email: yjl1245@hebiace.edu.cn, baiyan_jgxy@foxmail.com.

Abstract: With the widespread application of metaheuristic algorithms in engineering and scientific research, finding algorithms with efficient global search capabilities and precise local search performance has become a hot topic in research. The osprey optimization algorithm (OOA) was first proposed in 2023, characterized by its simple structure and strong optimization capability. However, practical tests have revealed that the OOA algorithm inevitably encounters common issues faced by metaheuristic algorithms, such as the tendency to fall into local optima and reduced population diversity in the later stages of the algorithm's iterations. To address these issues, a multi-strategy fusion improved osprey optimization algorithm is proposed (IOOA). First, the characteristics of various chaotic mappings were thoroughly explored, and the adoption of Circle chaotic mapping to replace pseudo-random numbers for population initialization improvement was proposed, increasing initial population diversity and improving the quality of initial solutions. Second, a dynamically adjustable elite guidance mechanism was proposed to dynamically adjust the position updating method according to different stages of the algorithm's iteration, ensuring the algorithm maintains good global search capabilities while significantly increasing the convergence speed of the algorithm. Lastly, a dynamic chaotic weight factor was designed and applied in the development stage of the original algorithm to enhance the algorithm's local search capability and improve the convergence accuracy of the algorithm. To fully verify the effectiveness and practical engineering applicability of the IOOA algorithm, simulation experiments were conducted using 21 benchmark test functions and the CEC-2022 benchmark functions, and the IOOA algorithm was applied to the LSTM power load forecasting problem as well as two engineering design problems. The experimental results show that the IOOA algorithm possesses outstanding global optimization performance in handling complex optimization problems and broad applicability in practical engineering applications.

Keywords: osprey optimization algorithm; Circle chaotic mapping; dynamic elite guidance mechanism; dynamic chaotic weight; multi-strategy fusion

1. Introduction

Global optimization problems are a crucial class of challenges in the fields of mathematics and computing, persisting in engineering, economics, natural sciences, and computing [1]. Traditional optimization methods often struggle to meet the solving demands when facing complex, multimodal, and discontinuous global optimization problems [2]. Metaheuristic algorithms, as flexible methods that disregard gradient information, demonstrate significant advantages in addressing these problems [3, 4].

The field of metaheuristic algorithm research is continuously evolving to cope with increasingly complex global optimization problems and practical application requirements. Based on the "no free lunch" theory [5], no single algorithm exists that can exhibit optimal performance on all problems in optimization and search. Consequently, strategic improvements to metaheuristic algorithms have become a focus of research for many scholars to enhance the overall performance of these algorithms.

Reference [6] proposes an improved prairie dog optimization algorithm (IPDOA), utilizing Tent chaotic mapping to initialize populations and enhance population diversity, along with a lens opposition-based learning strategy to enhance the algorithm's global search capabilities. Simulation results demonstrate that the improved prairie dog optimization algorithm offers superior optimization performance. Reference [7] proposes a modified beluga whale optimizer (OGGBWO) based on a random opposition-based learning strategy, adaptive Gauss variational operator, and elitist group genetic strategy, applied to 3D UAV path planning problems. Reference [8] presents an enhanced particle swarm optimization (PSO) algorithm designed for orderly electric vehicle charging strategy modeling, addressing the low optimization accuracy and slow convergence rate of the basic PSO algorithm through adjustments in the inertia weight index and learning factor. Reference [9] proposes a hybrid optimization algorithm that merges the hydrozoan algorithm (HA) with the sea turtle foraging algorithm (STFA) to tackle continuous optimization problems. Finally, reference [10] introduces a chaotic sparrow search algorithm (CSSA) for optimizing stochastic configuration network models. Employing logistic mapping, self-adaptive hyper-parameters, and a mutation operator to improve upon the basic SSA algorithm, simulation experiments showed that the CSSA algorithm exhibits a more robust global optimization capability compared to the basic SSA algorithm.

Various strategies for improving metaheuristic algorithms focus on several aspects: (a) Improved population initialization methods: Basic metaheuristic algorithms often generate initial populations randomly [11], leading to uneven initial population distribution and low individual quality. Scholars have proposed strategies such as chaotic mapping [12, 13] and Levy flights for population initialization improvement [14, 15], which can enhance initial population diversity and improve the quality of initial solutions. (b) Improved individual position update methods: Metaheuristic algorithms, when dealing with complex optimization problems with many local optima, are prone to getting stuck in local optima, causing search stagnation. Introducing strategies such as mutation [16, 17], disturbance, and adaptive adjustments [18, 19] during individual position updates can somewhat improve the algorithm's ability to escape local optima. (c) Strategies for combining two algorithms [20] aim to enhance the optimization capability by combining the advantages of two different algorithms. (d) Adding a weighting factor [21]: Inspired by PSO algorithms, numerous studies have shown that adding an inertia weight factor to the algorithm iteration process is beneficial in increasing the balance between exploration and exploitation.

The osprey optimization algorithm (OOA) was first proposed in 2023 [22], inspired by the hunting behavior of ospreys in nature. Its overall structure includes population initialization, global exploration,

and local development, offering advantages such as a simple structure and strong optimization capability. However, practical testing revealed that the random position update strategy during the exploration phase can increase ineffective search instances, affecting the algorithm's convergence speed. Additionally, during the development stage, calculating a new random position to guide individuals in finding the current optimal solution can enhance the algorithm's utilization in local search but may somewhat impact its optimization accuracy. To address these issues, this study proposes a multi-strategy fusion IOOA:

1) Adopt Circle chaotic mapping for population initialization improvement, which improves the initial population diversity and increases the robustness of the algorithm;

2) A dynamic elite guidance mechanism with adjustable ratio is proposed, which is applied to the exploration stage of the algorithm to dynamically adjust the individual position updating method for different stages of the algorithm iteration to improve the global convergence speed of the algorithm;

3) A dynamic chaotic weight factor is proposed, which is applied in the development stage of the algorithm to enhance the local search ability of the algorithm, avoid the algorithm from falling into the local optimum, and improve the algorithm's optimization accuracy.

The main structure of this paper is as follows. Section 2 introduces the mathematical model of the OOA algorithm, Section 3 describes in detail the IOOA algorithm proposed in this paper, Section 4 shows the simulation experiments and algorithm performance analysis, and Section 5 summarizes the work of this paper and the outlook of future research.

2. OOA

OOA is a population-based intelligent optimization algorithm inspired by the hunting behavior of osprey in nature. Similar to other intelligent optimization algorithms, it performs a random initialization population operation in the search space with the population initialization formula:

$$x_{i,j} = lb_j + r \cdot \left(ub_j - lb_j\right), \tag{1}$$

where $x_{i,j}$ is the individual, lb_j is the lower bound of the search, ub_j is the upper bound of the search, and r is a random number between [0, 1].

The first phase of the OOA is an exploratory phase, which is modeled by simulating the behavior of osprey locating and catching fish in nature. During the design process of the OOA algorithm, each individual in the population will consider other individuals with better positions as a school of fish, and the mathematical model of the target school of fish for each individual is as follows:

$$FP_i = \{X_k \mid k \in \{1, 2, \ldots, N\} \wedge F_k < F_i\} \cup \{X_{\text{best}}\}, \tag{2}$$

where FP_i is the set of fish for the ith eagle and X_{best} is the location of the best eagle.

Ospreys prey based on a stochastic detection mechanism, and by modeling the behavior of ospreys attacking fish (predation), the algorithm individual position update formula is as follows:

$$x_{i,j}^{NEW} = x_{i,j} + r_{i,j} \cdot \left(SF_{i,j} - I_{i,j} \cdot x_{i,j}\right), \tag{3}$$

where SF is the target fish selected by the individual, r is a random number between [0, 1], and the value of I is one of {1, 2}.

Boundary checking is performed for individuals with completed positional updating with the following equation:

$$x_{i,j}^{NEW} = \begin{cases} x_{i,j}^{NEW}, lb_j \le x_{i,j}^{NEW} \le ub_j \\ lb_j, x_{i,j}^{NEW} < lb_j \\ ub_j, x_{i,j}^{NEW} > ub_j. \end{cases} \tag{4}$$

If the updated individual position is better than the previous position, the previous position is replaced by the new position. The equation is as follows:

$$X_i = \begin{cases} X_i^{NEW}, F_i^{NEW} < F_i \\ X_i, \text{ else }. \end{cases} \tag{5}$$

Here X_i^{NEW} is the updated position, and F_i^{NEW} is the updated fitness value.

After catching a fish in nature, the osprey takes the fish to a safe location to feed and the development phase of the algorithm is modeled based on the above behavior. Each individual in the population calculates a new random location as a feeding area and the mathematical model of this behavior is as follows:

$$x_{i,j}^{NEW_2} = x_{i,j} + \frac{lb_j + r \cdot \left(ub_j - lb_j\right)}{t}. \tag{6}$$

Boundary checking for all individuals using the position update phase of the following equation:

$$x_{i,j}^{NEW_2} = \begin{cases} x_{i,j}^{NEW_2}, lb_j \le x_{i,j}^{NEW_2} \le ub_j \\ lb_j, x_{i,j}^{NEW_2} < lb_j \\ ub_j, x_{i,j}^{NEW_2} > ub_j. \end{cases} \tag{7}$$

Compare the quality of the updated individual with that of the original individual, and if the new position is superior, the new position is used to replace the original position.

$$X_i = \begin{cases} X_i^{NEW_2}, F_i^{NEW_2} < F_i \\ X_i, \text{ else }. \end{cases} \tag{8}$$

Here, $x_{i,j}^{NEW_2}$ is the new position of the ith individual, and $F_i^{NEW_2}$ is the updated individual fitness value.

The pseudo-code of the OOA algorithm is shown in Algorithm 1.

After analyzing the mathematical model of the OOA algorithm, it has been found that it possesses a simple structure and a reasonable mechanism, offering certain advantages. However, the actual testing process revealed that during the iterative process, the OOA algorithm updates individual positions randomly, which can lead to the detection of sub-optimal targets and result in invalid searches, impacting the algorithm's convergence speed. In the later stages of iteration, although individuals adopt a random perturbation strategy to enhance the algorithm's ability to escape local optima, this approach can result in less precise local searches and reduce the accuracy of finding the optimum.

Based on the above analysis, the OOA algorithm's optimization performance has significant room for improvement. Implementing a multi-strategy fusion approach to enhance the OOA algorithm can greatly improve its optimization performance, which has considerable research significance.

Algorithm 1 Osprey optimization algorithm

Input: population size: N, the maximum number of iterations: T, Dimension of the objective function: Dim, The boundary conditions of the variables: Ub and Lb.

Output: The optimal fitness value and the optimal position: F_b and X_b.

Define the initial population $i \Leftarrow 1, 2, ..., N$ and its related parameters
Population initialization by Eq (1)
while $t \leq T$ **do**
    for int i = 1 to N **do**
        Calculate individual fitness values
        Specify the target fish population for each individual from Eq (2)
        if Phase 1: Discovery Phase **then**
            Update the individual position using Eq (3)
            Checking the boundary conditions using Eq (4)
            Positional substitution according to Eq (5)
        end if
        if Phase 2: Development phase **then**
            Update the individual position using Eq (6)
            Checking the boundary conditions using Eq (7)
            Positional substitution according to Eq (8)
        end if
    end for
    Update the population optimal fitness value F_b and optimal position X_b
    $t = t + 1$
end while
return F_b and X_b

3. IOOA

Aiming at the above problems, in order to fully improve the performance of the OOA algorithm for optimization, this paper proposes a multi-strategy fusion IOOA on the basis of previous research, and the specific strategies are as follows:

1) Initialization: The IOOA algorithm adopts Circle chaotic mapping for the initialization of the algorithm population, which enhances the diversity of the initial individuals and improves the quality of the initial solution through the chaotic characteristics of Circle chaotic mapping.

2) Exploration phase: The IOOA algorithm carries out individual position updating through the dynamic elite guidance mechanism. In the early iterations of the algorithm, the elite guidance mechanism

occupies a relatively small proportion, and the algorithm adopts the position updating strategy of randomly detecting targets by individuals to fully traverse the solution space. In the late iteration of the algorithm, the elite guidance mechanism occupies a larger proportion, the individual position update method is changed to the elite guidance mechanism, and the individuals in the population follow the optimal individual position update, which reduces the number of ineffective searches of the algorithm and improves the algorithm's convergence speed.

3) Development phase: In the development phase of the IOOA algorithm, a dynamic chaotic weight factor strategy is added, in which the weight factor is defined by the Cubic chaotic mapping, and the weight coefficients are dynamically adjusted according to the changes in the number of iterations. This strategy can enhance the algorithm's local search ability and improve the optimization accuracy.

3.1. Circle chaos map

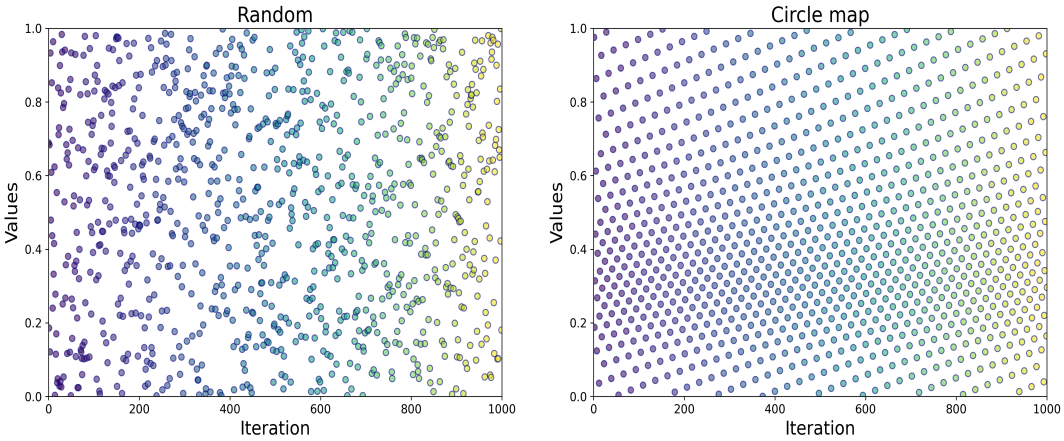
The traditional OOA algorithm uses pseudo-random numbers to initialize the population position in the population initialization stage, and this strategy leads to a certain degree to the low quality of the initial population individuals and insufficient traversal of the solution space, which in turn affects the algorithm's search quality. A large number of studies have shown that replacing pseudo-random numbers with chaotic sequences for population initialization improvement can increase the initial population diversity and improve the initial solution quality [23]. Meanwhile, chaotic mapping helps to reduce the randomness fluctuation of population initialization, which can increase the robustness of the algorithm [24].

Circle chaotic mapping is a typical representative of chaotic mapping, which has a simple mathematical structure with traversal and randomness. This study proposes to use Circle chaotic mapping for population initialization improvement and the mathematical expression of Circle chaotic mapping is as follows:

$$x_{i+1} = \mathrm{mod}\left(x_i + 0.2 - \left(\frac{0.5}{2\pi}\right)\sin\left(2\pi x_i\right), 1\right). \tag{9}$$

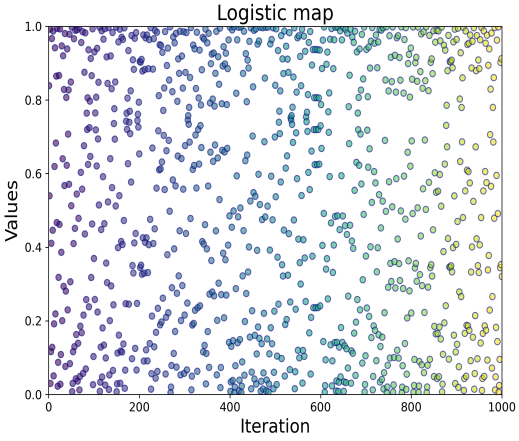
This study analyzes the chaotic characteristics of five types of chaos mappings commonly used for population initialization improvements in metaheuristic algorithms. The sample distribution of various chaos mappings within the solution space is shown in Figure 1.

As can be seen from Figure 1, the sample distribution of Circle chaotic mapping is more ergodic and homogeneous, and compared with the random number generator, the chaotic mapping has a certain degree of randomness while, at the same time, it has a certain degree of certainty. Randomness helps to improve the initial population diversity and avoid the algorithm falling into local optima. Determinism helps to improve the repeatability of the improved algorithm, which is very important for comparing the optimization performance of different algorithms.

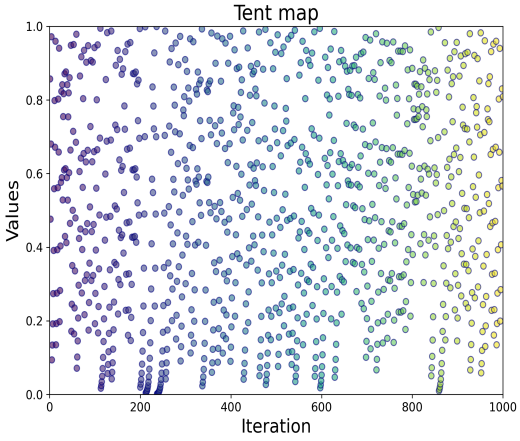


(a) Random

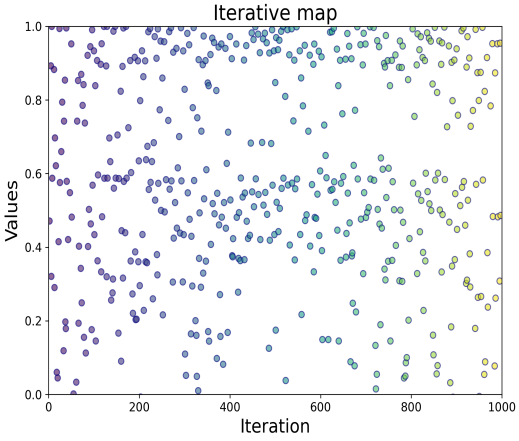
(b) Circle Map



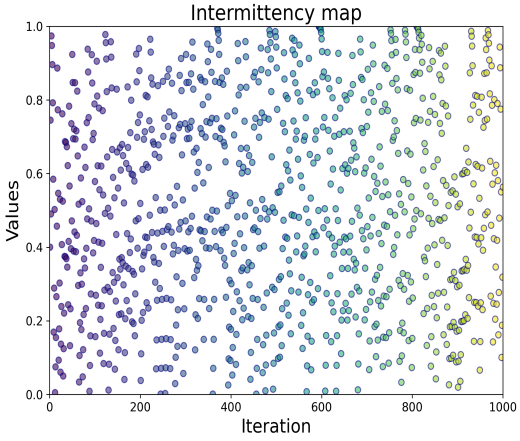
(c) Logistic Map



(d) Tent Map



(e) Iterative Map



(f) Intermittency Map

Figure 1. Sample distribution for 1000 iterations of each chaotic mapping.

3.2. Dynamic elite guidance mechanism with adjustable ratio

In the exploration phase of the OOA algorithm, the individual adopts random detection of attack targets for position updating, and the purpose of this strategy is to increase the individual's exploration of the search space and avoid the algorithm from falling into a local optimum at the beginning of the iteration. However, under the random search strategy, the target randomly selected by the individual may not be a better solution. Meanwhile, as the number of iterations increases, the random search strategy leads to an increase in the number of invalid searches of the algorithm to some extent. Based on the above analysis, this study proposes a dynamic elite guidance mechanism with adjustable ratio, which is applied to the exploration phase of the algorithm, and the position update formula is as follows:

$$x_{i,j}^{NEW} = x_{i,j} + \alpha \cdot r_{i,j} \cdot \left(X_{i,j}^{best} - X_{i,j}\right) + (1-\alpha) \cdot r_{i,j} \cdot \left(SF_{i,j} - I_{i,j} \cdot x_{i,j}\right), \alpha = \frac{t}{T}. \tag{10}$$

Among them, $X_{i,j}^{best}$ is the position of the individual with the optimal fitness value in the population. α is a dynamic adjustment factor to control the ratio between the elite bootstrapping mechanism and randomized exploration, and α increases linearly from 0 to 1 with the number of iterations.

By introducing a dynamic adjustment factor, the algorithm gradually shifts the individual position update method from random exploration to elite guidance during the exploration process as the number of iterations increases. At the beginning of the iteration, a smaller value of α makes the algorithm focus on exploration, increases the randomness of the algorithm to extensively explore the solution space, and avoids the algorithm from falling into a local optimum. As the number of iterations increases, the value of α gradually increases, the individual position update method focuses on elite guidance, and the algorithm can converge to the local optimal solution faster, reduce the number of invalid searches, and improve the convergence speed of the algorithm.

3.3. Dynamic chaotic weight factor

In the development phase of the OOA algorithm, for each individual that finds a locally optimal solution, a new randomized position is recalculated so that the individual's position in the search space changes slightly. This strategy can enhance the ability of the OOA algorithm to move away from local optimality to some extent. However, this approach does not fully utilize the position information of the global optimal solution, and the randomized position computation method is unable to perform a more adequate and accurate local search. For this reason, this study proposes a dynamic chaotic weighting factor defined by Cubic chaotic mapping, which is applied to the development stage of the OOA algorithm, and dynamically adjusts the weight coefficients through the change of iteration number so as to utilize the traversal nature of Cubic chaotic mapping to enhance the algorithm's local search ability. The mathematical expression of the dynamic chaotic weight factor is as follows:

$$\omega(t+1) = \frac{2.595\omega(t)\left(1-\omega(t)^2\right) \cdot (T-t)}{T}, t = 1, \ldots, T, \tag{11}$$

where $\omega(1)$ takes the value 0.3, t is the current number of iterations, and T is the maximum number of iterations.

The position of the introduced dynamic chaos weight factor is formulated as

$$x_{i,j}^{NEW_2} = \omega(t) \cdot x_{i,j} + \frac{lb_j + r \cdot \left(ub_j - lb_j\right)}{t}. \tag{12}$$

The introduction of the dynamic chaotic weight factor enables the individual to carry out a finer search in the neighborhood of the optimal solution when updating the position. As the number of iterations increases, the weight factor gradually becomes smaller, and the individual changes from the fine search at the beginning of the iteration to the rapid convergence to the optimal value. For the optimization problem with a large number of local optimal values, this strategy can sufficiently improve the algorithm's accuracy of searching for the optimal value and the ability to get rid of the local optimal value.

3.4. Overall structure of IOOA

Based on the in-depth analysis of the optimization mechanism of the OOA algorithm, this study proposes an enhanced OOA algorithm (IOOA) through the fusion of multiple strategies to enhance the optimization performance of the OOA algorithm, and the overall structure of IOOA is as follows:

Step 1: Parameter settings—number of populations N, number of iterations T, problem dimension D, boundary conditions lb, ub.

Step 2: Initialization—The chaotic sequence generated by Eq (9) replaces the random numbers in Eq (1) for population initialization.

Step 3: Record fitness values—Record all individual fitness values in the population, including the optimal fitness value and its location, and the worst fitness value and its location.

Step 4: Designation of target fish groups—Specify the target population of fish for each individual according to Eq (2).

Step 5: Exploration phase location update—Individual position update by Eq (10), boundary checking is performed by Eq (4) during the updating process, and positional replacement is performed by Eq (5) for the updated individual.

Step 6: Development phase location update—Individual position updating is performed by Eq (12), boundary checking is performed by Eq (7) during position updating, and position replacement is performed by Eq (8) for position updating completed individuals.

Step 7: Termination conditions—Determine whether the maximum number of iterations is reached, satisfy the condition, terminate the iteration, and record the optimal solution; otherwise, return to Step 3.

The pseudo-code of the IOOA algorithm is shown in Algorithm 2.

The flowchart of the IOOA algorithm is shown in Figure 2.

Algorithm 2 Improved osprey optimization algorithm

Input: population size: N, the maximum number of iterations: T, Dimension of the objective function: Dim, The boundary conditions of the variables: Ub and Lb.

Output: The optimal fitness value and the optimal position: F_b and X_b.

Define the initial population $i \Leftarrow 1, 2, ..., N$ and its related parameters
Population initialization by Eq (9)
while $t \leq T$ **do**
    for int i = 1 to N **do**
        Calculate individual fitness values
        Specify the target fish population for each individual from Eq (2)
        if Discovery Phase **then**
            Update the individual position using Eq (10)
            Checking the boundary conditions using Eq (4)
            if $F_i^{NEW} < F_i$ **then**
                $X_i = X_i^{NEW}$
            else
                $X_i = X_i$
            end if
        end if
        if Development phase **then**
            Update the individual position using Eq (12)
            Checking the boundary conditions using Eq (7)
            if $F_i^{NEW} < F_i$ **then**
                $X_i = X_i^{NEW}$
            else
                $X_i = X_i$
            end if
        end if
    end for
    Update the population optimal fitness value F_b and optimal position X_b
    $t = t + 1$
end while
return F_b and X_b

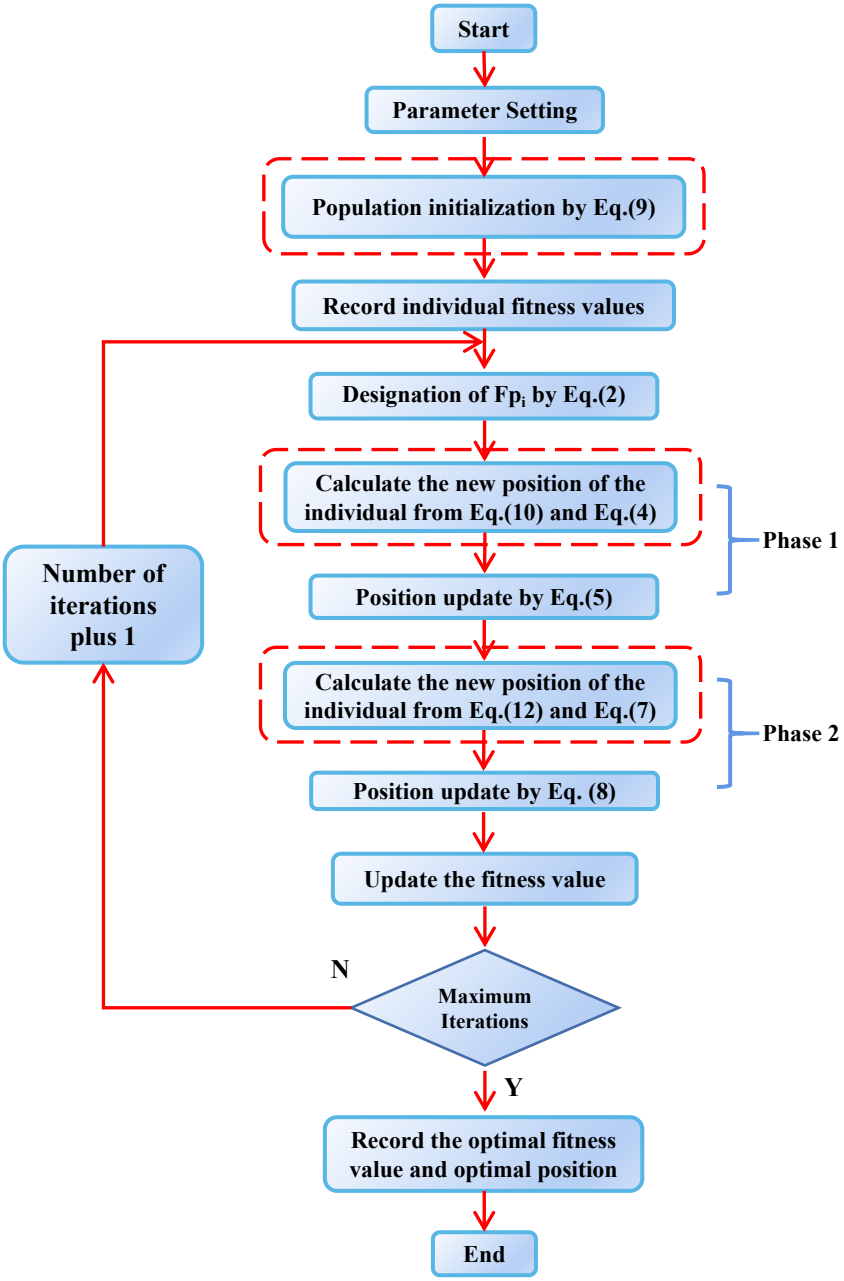


Figure 2. Flow chart of IOOA.

3.5. Time complexity analysis

Time complexity is an important index for evaluating the performance of algorithms, and the main content of this section is to analyze the time complexity of the IOOA algorithm.

Let the size of the population be N, and the dimension of the problem be D. The IOOA algorithm adopts Circle chaotic mapping for the population initialization operation, so the time complexity of the IOOA algorithm for assigning initial positions to the individuals of the population in all dimensions is

O(ND). The number of iterations is T. The IOOA algorithm individual position update is divided into two parts: the exploration phase and the development phase; and the time complexity of the exploration phase is O(TND), and the time complexity of the development phase is O(TND), and thus, the overall time complexity of the IOOA algorithm is O(ND(1+2T)).

The time complexity of the base OOA algorithm to generate the initial population of size N and dimension D is O(ND), and the time complexity of both the exploration phase and the development phase is O(TND), and thus the original algorithm time complexity is O(ND(1+2T)).

The mathematical model of the IOOA algorithm does not increase the loop nesting on the basis of the original algorithm. In summary, the time complexity of the IOOA algorithm is the same as that of the OOA algorithm and the improvement strategy proposed in this paper does not increase the time complexity of the algorithm.

4. Simulation experiments and performance analysis

In order to fully verify the optimization performance of the IOOA algorithm, the IOOA algorithm is tested with seven well-known algorithms on 21 benchmark test functions [25]. The comparison algorithms include: PSO algorithm [26], GWO algorithm [27], AO algorithm [28], WOA algorithm [29], GJO algorithm [30], DBO algorithm [31], OOA algorithm. The parameter settings of each algorithm are shown in Table 1. The test functions are shown in Table 2: F1–F5 are high-dimensional single-peak functions to test the convergence speed and optimization accuracy of the algorithms, F6–F11 are high-dimensional multi-peak functions to test the ability of the algorithms to jump out of the local optimum, and F12–F21 are fixed-dimensional test functions.

Table 1. Each algorithm parameter setting.

Algorithm	Parameters
PSO	$W_1 = 0.9; W_2 = 0.2; C_1, C_2 = 2; V_{min} = -5, V_{max} = 5$
GWO	α decreases linearly from 2 to 0; $r_1, r_2 \in [0, 1]$
AO	$\alpha = 0.1$; $\delta = 0.1$; $s = 0.01$; $u, v \in [0, 1]$
WOA	α decreased from 2 to 0; Spiral shape constant $b = 1$
GJO	E decreases linearly from 1.5 to 0
DBO	$k = 1$; $\lambda = 4$; $b = 0.3$; $S = 0.5$
OOA	$r \in [0, 1]$; $I \in \{1, 2\}$
IOOA	$r \in [0, 1]$; $I \in \{1, 2\}$; α increases linearly from 0 to 1

Experimental platform hardware environment: Windows-11 operating system PC, CPU: i7-8750H, RAM: 8 GB; software environment: PyCharm 2021.2.3.

Table 2. Test functions.

Function	Interval	Dimension	Min
$F_1(x) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]$	30/100	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[-10, 10]$	30/100	0
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} \chi_j\right)^2$	$[-100, 100]$	30/100	0
$F_4(x) = \max_i \{\lvert x_i \rvert, 1 \leqslant i \leqslant n\}$	$[-100, 100]$	30/100	0
$F_5(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	$[-100, 100]$	30/100	0
$F_6(x) = \sum_{i=1}^{n} -\left(x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)\right)$	$[-500, 500]$	30/100	$-418.9829d$
$F_7(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	$[-5.12, 5.12]$	30/100	0
$F_8(\boldsymbol{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + \mathrm{e}$	$[-32, 32]$	30/100	0
$F_9(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]$	30/100	0
$F_{10}(x) = \frac{\pi}{n}\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2 \left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$, $y_i = 1 + \frac{x_i + 1}{4}$; $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, x_i > a \\ 0, -a < x_i < a \\ k(-x_i - a)^m, x_i < -a \end{cases}$	$[-50, 50]$	30/100	0
$F_{11}(x) = 0.1\{\sin^2(3\pi x_i) + \sum_{i=1}^{D}(x_i - 1)^2\left[1 + \sin^2(3\pi x_i)\right] + (x_D - 1)^2\left[1 + \sin^2(2\pi x_D)\right]\} + \sum_{i-1}^{D} u(x_i, 5, 100, 4)$	$[-50, 50]$	30/100	0
$F_{12}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}\left(x_i - a_{ij}\right)^6}\right)^{-1}$	$[-65.5360, 65.5360]$	2	0.998004
$F_{13}(x) = \sum_{i=1}^{11}\left(a_i - \frac{x_1\left(b_i^2 + b_i x_2\right)}{b_i^2 + b_i x_3 + x_4}\right)^{-1}$	$[-5, 5]$	4	0.0003075
$F_{14}(x) = 4x_1^2 - 2.1x_1^4 + 1/3x_1^6 + x_1x_2 - 4x_2^2 + x_2^4$	$[-5, 5]$	2	-1.03163

Continued on next page

Function	Interval	Dimension	Min
$F_{15}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	$[-5, 5]$	2	0.398
$F_{16}(x) = [1 + (x_1 + x_2 + 1)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)] \times [30 + (2x_1 - 3x_2)^2 \times \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)]$	$[-5, 5]$	2	3
$F_{17}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	$[0, 1]$	3	−3.86
$F_{18}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	$[0, 1]$	6	−3.32
$F_{19}(x) = -\sum_{i=1}^{5}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	$[0, 10]$	4	−10.1532
$F_{20}(x) = -\sum_{i=1}^{7}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	$[0, 10]$	4	−10.4029
$F_{21}(x) = -\sum_{i=1}^{10}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	$[0, 10]$	4	−10.5364

4.1. Comparison with various well-known algorithms

To minimize random errors, the population size for all algorithms was set to 50, with a maximum iteration count of 500. Each algorithm ran independently 30 times, and evaluation metrics included the optimal value, standard deviation, and mean. The test results for each algorithm are presented in Tables 3–5 (with the optimal values bolded).

From the data in Tables 3 and 4, it can be seen that in the case of 30 dimensions, when facing high-dimensional unimodal functions F1–F5, IOOA consistently exhibits the highest optimization accuracy. It can stably locate the theoretical optimal value with optimization accuracy significantly higher than the compared algorithms, demonstrating a notable advantage. Especially in the case of function F5, the compared algorithms all experience varying degrees of search stagnation, while the proposed IOOA algorithm can precisely find the optimal value, fully proving the effectiveness of the improvement strategy in this paper and the superior performance of the IOOA algorithm. For functions F1 and F3, the OOA algorithm can also lock onto the optimal solution, demonstrating a certain inherent advantage of the algorithm. The use of averages to evaluate the algorithm's optimization robustness shows that the IOOA algorithm consistently and accurately locks onto the optimal solution, indicating strong robustness. When the dimension of the objective function increases from 30 to 100, on functions F2 and F4, the IOOA algorithm's optimization accuracy slightly decreases. However, compared to other algorithms, the IOOA algorithm still maintains the highest optimization accuracy and robustness.

Table 3. Results of 30-dimension simulation experiments.

Function	Metrics	PSO	GWO	AO	WOA	GJO	DBO	OOA	IOOA
	Mean	1.56E+00	3.76E-33	9.74E-46	2.26E-25	1.12E-113	1.16E-54	**0.00E+00**	**0.00E+00**
F1	Std	4.61E-01	6.89E-33	1.77E-45	5.49E-25	8.72E-114	3.10E-54	**0.00E+00**	**0.00E+00**
	Best	6.40E-01	1.08E-34	7.07E-49	8.38E-30	1.71E-114	2.12E-61	**0.00E+00**	**0.00E+00**
	Mean	1.52E+01	7.00E-20	1.31E-23	4.35E-17	9.48E-59	4.99E+00	7.63E-197	**0.00E+00**
F2	Std	1.85E+01	5.19E-20	1.21E-23	6.53E-18	4.93E-59	8.06E+00	**0.00E+00**	**0.00E+00**
	Best	4.48E+00	1.13E-20	1.76E-25	9.68E-19	3.74E-59	4.15E-33	1.29E-204	**0.00E+00**
	Mean	8.86E+01	2.64E-08	4.91E-45	5.77E-05	4.31E-99	8.02E+03	**0.00E+00**	**0.00E+00**
F3	Std	2.68E+01	4.31E-08	6.05E-45	1.20E-04	1.26E-98	9.76E+03	**0.00E+00**	**0.00E+00**
	Best	3.77E+01	3.09E-11	6.34E-48	6.03E-07	5.58E-102	3.46E-57	**0.00E+00**	**0.00E+00**
	Mean	2.40E+00	1.60E-08	1.08E-23	1.76E-05	4.55E-52	2.24E-26	4.53E-196	**0.00E+00**
F4	Std	1.19E+00	9.48E-09	1.18E-23	1.97E-05	2.93E-52	4.02E-26	**0.00E+00**	**0.00E+00**
	Best	1.07E+00	2.61E-09	6.19E-25	2.30E-07	1.60E-52	5.11E-29	1.72E-204	**0.00E+00**
	Mean	1.57E+00	5.09E-01	2.71E-04	1.04E+00	5.85E+00	4.41E-02	4.34E-02	**0.00E+00**
F5	Std	3.04E-01	3.52E-01	5.15E-04	5.20E-01	6.01E-01	5.85E-02	4.61E-02	**0.00E+00**
	Best	7.79E-01	2.86E-05	3.85E-06	2.51E-01	3.65E+00	1.58E-02	1.28E-02	**0.00E+00**
	Mean	-6.03E+03	-6.14E+03	-6.39E+03	-7.29E+03	-2.67E+03	-1.00E+04	-7.42E+03	**-1.25E+04**
F6	Std	1.05E+03	7.69E+02	4.73E+02	2.83E+02	4.96E+02	1.42E+03	1.64E+03	**1.64E+00**
	Best	-7.86E+03	-7.54E+03	-7.42E+03	-7.93E+03	-3.93E+03	-1.25E+04	-1.04E+04	**-1.25E+04**
	Mean	1.41E+02	1.94E+00	1.60E-03	1.89E-15	**0.00E+00**	1.42E+01	**0.00E+00**	**0.00E+00**
F7	Std	2.46E+01	3.23E+00	1.46E-03	1.02E-14	**0.00E+00**	3.29E+01	**0.00E+00**	**0.00E+00**
	Best	7.59E+01	5.68E-14	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Mean	3.18E+00	4.36E-14	2.81E-15	1.79E-13	3.99E-15	4.59E-01	**4.44E-16**	**4.44E-16**
F8	Std	4.94E-01	5.66E-15	1.67E-15	2.92E-13	0.00E+00	2.47E+00	**0.00E+00**	**0.00E+00**
	Best	1.87E+00	3.24E-14	4.44E-16	3.24E-14	3.99E-15	4.44E-16	**4.44E-16**	**4.44E-16**
	Mean	2.09E-01	2.09E-03	2.49E-08	1.48E-03	1.35E-03	**0.00E+00**	**0.00E+00**	**0.00E+00**
F9	Std	4.45E-02	4.93E-03	1.34E-07	5.56E-03	3.52E-03	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Best	1.41E-01	3.74E-05	4.41E-11	2.44E-05	5.49E-05	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Mean	2.42E+00	2.97E-02	2.40E-06	8.49E-02	1.03E+00	3.33E-03	2.79E-03	**3.38E-23**
F10	Std	1.33E+00	1.42E-02	3.68E-06	4.28E-02	2.41E-01	1.83E-03	1.20E-03	**7.99E-22**
	Best	5.59E-01	1.29E-02	5.35E-08	2.90E-02	6.59E-01	1.14E-03	3.88E-04	**8.90E-25**
	Mean	6.60E-01	3.76E-01	1.92E-05	1.40E+00	2.85E+00	5.45E-02	5.01E-02	**1.75E-17**
F11	Std	2.42E-01	1.86E-01	2.53E-05	3.48E-01	5.49E-02	2.87E-02	3.62E-02	**4.19E-17**
	Best	2.30E-01	3.59E-05	1.69E-06	4.67E-01	2.72E+00	2.41E-02	3.91E-03	**1.26E-19**

Table 4. Results of 100-dimension simulation experiments.

Function	Metrics	PSO	GWO	AO	WOA	GJO	DBO	OOA	IOOA
	Mean	1.17E+02	4.48E-15	9.78E-116	4.21E-16	6.22E-102	7.45e+00	**0.00E+00**	**0.00E+00**
F1	Std	1.63E+01	3.26e-15	5.23e-115	9.65E-16	3.31E-102	4.01e+01	**0.00E+00**	**0.00E+00**
	Best	8.77E+01	1.05E-15	2.55E-123	7.62E-19	2.22E-102	2.42E-60	**0.00E+00**	**0.00E+00**
	Mean	1.99E+02	1.53E-09	3.08E-58	9.46E-12	1.92E-52	1.53E+01	2.17E-194	**4.79E-290**
F2	Std	6.67E+01	4.81E-10	1.51E-57	8.80E-12	3.97E-53	2.04E+01	**0.00E+00**	**0.00E+00**
	Best	7.40E+01	6.38E-10	1.79E-62	5.73E-13	1.145E-52	4.86E-32	9.20E-203	**2.93E-302**
	Mean	1.68E+04	1.01E+02	4.84E-116	2.67E+01	3.94E-91	2.67E+05	**0.00E+00**	**0.00E+00**
F3	Std	9.26E+03	1.04E+02	2.46E-115	5.27E+01	4.92E-91	1.15E+05	**0.00E+00**	**0.00E+00**
	Best	6.67E+03	3.20E+00	2.62E-123	3.39E-02	2.59E-92	1.49E-53	**0.00E+00**	**0.00E+00**
	Mean	1.21E+01	1.53E-01	6.06E-60	3.34E-03	2.71E-48	3.54E-26	2.73E-197	**1.29E-289**
F4	Std	1.42E+00	1.65E-01	1.64E-59	2.37E-03	9.39E-49	5.93E-26	**0.00E+00**	**0.00E+00**
	Best	9.87E+00	1.63E-02	1.23E-63	7.10E-04	1.23E-48	2.17E-29	1.71E-203	**5.86E-308**
	Mean	1.14E+02	8.06E+00	6.39E-04	1.06E+01	2.32E+01	3.41E+02	1.44E-01	**0.00E+00**
F5	Std	1.87E+01	9.25E-01	1.37E-03	1.13E+00	6.90E-01	1.81E+03	6.63E-02	**0.00E+00**
	Best	8.77E+01	6.53E+00	8.12E-06	8.46E+00	2.04E+01	2.59E+00	1.39E-02	**0.00E+00**
	Mean	-1.45E+04	-1.65E+04	-1.29E+04	-2.38E+04	-5.01E+03	-3.23E+04	-2.51E+04	**-4.18E+04**
F6	Std	5.07E+03	2.14E+03	7.94E+02	7.98E+02	8.08E+02	4.80E+03	5.05E+03	**6.95E-01**
	Best	-2.47E+04	-2.03E+04	-1.44E+04	-2.58E+04	-7.08E+03	-3.79E+04	-3.73E+04	**-4.18E+04**
	Mean	1.97E+03	2.52E+01	**0.00E+00**	2.46E-01	**0.00E+00**	8.40E+03	**0.00E+00**	**0.00E+00**
F7	Std	2.16E+02	1.16E+01	**0.00E+00**	6.53E-01	**0.00E+00**	4.48E+04	**0.00E+00**	**0.00E+00**
	Best	1.61E+03	7.24E+00	**0.00E+00**	4.54E-13	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Mean	6.92E+00	7.63E-09	**4.44E-16**	1.50E-09	6.01E-15	3.31E+00	**4.44E-16**	**4.44E-16**
F8	Std	5.83E-01	3.09E-09	**0.00E+00**	1.58E-09	1.76E-15	5.15E+00	**0.00E+00**	**0.00E+00**
	Best	5.71E+00	2.93E-09	**4.44E-16**	5.84E-11	3.99E-15	**4.44E-16**	**4.44E-16**	**4.44E-16**
	Mean	1.03E+00	1.55E-03	**0.00E+00**	2.97E-15	3.34E-04	3.11E+00	**0.00E+00**	**0.00E+00**
F9	Std	2.62E-02	4.71E-03	**0.00E+00**	1.03E-14	1.79E-03	1.61E+01	**0.00E+00**	**0.00E+00**
	Best	9.68E-01	2.33E-15	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Mean	1.13E+01	2.12E-01	9.47E-07	3.06E-01	1.13E+00	7.32E-02	1.46E-03	**8.35E-10**
F10	Std	3.57E+00	4.19E-02	1.49E-06	7.89E-02	6.61E-02	2.56E-02	6.54E-04	**4.21E-09**
	Best	6.38E+00	1.50E-01	1.92E-09	1.98E-01	1.01E+00	2.73E-02	1.18E-04	**3.18E-33**
	Mean	2.65E+02	6.05E+00	3.34E-05	7.79E+00	9.91E+00	3.25E+00	9.02E-02	**3.31E-08**
F11	Std	8.98E+01	5.49E-01	4.36E-05	4.19E-01	4.73E-02	9.76E-01	3.61E-02	**1.49E-07**
	Best	1.68E+02	4.73E+00	4.08E-07	6.98E+00	9.82E+00	9.43E-01	1.79E-02	**1.45E-32**

Table 5. Results of fixed-dimension simulation experiments

Function	Metrics	PSO	GWO	AO	WOA	GJO	DBO	OOA	IOOA
	Mean	2.05E+00	3.22E+00	3.42E+00	1.15E+00	8.68E+00	1.03E+00	1.61E+00	**9.98E-01**
F12	Std	1.54E+00	2.95E+00	2.93E+00	3.32E-01	3.97E+00	1.78E-01	9.80E-01	**0.00E+00**
	Best	**9.98E-01**	**9.98E-01**	**9.98E-01**	**9.98E-01**	1.99E+00	**9.98E-01**	**9.98E-01**	**9.98E-01**
	Mean	1.34E-03	3.15E-03	1.12E-03	3.47E-04	1.85E-03	2.44E-03	1.74E-03	**3.24E-04**
F13	Std	4.11E-04	6.75E-03	3.83E-03	1.09E-04	5.11E-03	2.67E-03	1.12E-03	**1.97E-05**
	Best	6.93E-04	**3.07E-04**	3.08E-04	**3.07E-04**	3.11E-04	3.93E-04	3.88E-04	**3.07E-04**
	Mean	-1.031618	**-1.031628**	-1.031605	**-1.031628**	-1.031615	**-1.031628**	-1.031615	-1.031625
F14	Std	2.31E-05	**9.53E-09**	2.90E-05	2.62E-08	1.73E-05	1.80E-08	6.71E-05	9.38E-06
	Best	**-1.031628**	**-1.031628**	-1.031627	**-1.031628**	**-1.031628**	**-1.031628**	**-1.031628**	**-1.031628**
	Mean	**3.98E-01**	3.97E-01	3.97E-01	3.97E-01	3.97E-01	3.97E-01	3.97E-01	3.97E-01
F15	Std	3.09E-04	2.38E-07	1.07E-05	4.82E-07	4.01E-04	1.66E-07	9.66E-09	**1.56E-09**
	Best	**3.98E-01**	3.97E-01	3.97E-01	3.97E-01	3.97E-01	3.97E-01	3.97E-01	3.97E-01
	Mean	**3.00E+00**	**3.00E+00**	1.38E+01	**3.00E+00**	**3.00E+00**	**3.00E+00**	3.01E+00	**3.00E+00**
F16	Std	3.67E-04	6.31E-06	2.47E+01	**3.07E-07**	6.47E-06	8.93E-07	4.78E-02	1.12E-08
	Best	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**
	Mean	-3.85E+00	**-3.86E+00**	**-3.86E+00**	-3.85E+00	-3.85E+00	**-3.86E+00**	-3.82E+00	**-3.86E+00**
F17	Std	4.51E-03	2.55E-03	2.37E-04	2.35E-03	2.73E-03	**2.24E-04**	4.02E-02	8.75E-04
	Best	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**	**-3.86E+00**
	Mean	-3.06E+00	-3.25E+00	-3.24E+00	-2.92E+00	-3.07E+00	-3.28E+00	-2.84E+00	**-3.31E+00**
F18	Std	1.05E-01	7.49E-02	8.60E-02	3.06E-01	1.30E-01	5.75E-02	3.10E-01	**2.98E-02**
	Best	-3.27E+00	**-3.32E+00**	**-3.32E+00**	-3.17E+00	-3.28E+00	**-3.32E+00**	-3.27E+00	**-3.32E+00**
	Mean	-10.1477	-9.6475	-9.4693	-3.3299	-4.8815	-6.7734	9.9440	**-10.1521**
F19	Std	5.19E-03	1.51E+00	1.73E+00	1.76E+00	7.90E-01	2.37E+00	2.59E-01	**5.59E-03**
	Best	-10.1530	-10.1531	-10.1528	-5.0551	-6.4516	-10.1521	-10.1531	**-10.1532**
	Mean	-10.3934	-10.4024	-9.8679	-3.4682	-5.4512	-8.6352	-9.8388	**-10.4025**
F20	Std	7.67E-03	**2.21E-04**	1.59E+00	1.60E+00	1.39E+00	2.48E+00	7.21E-01	9.30E-04
	Best	-10.4023	**-10.4028**	-10.4026	-5.0876	-8.3486	-10.4014	-10.4026	**-10.4028**
	Mean	-10.5304	-9.9050	-8.5508	-3.2812	-5.0430	-7.2320	-9.9879	**-10.5359**
F21	Std	5.19E-03	1.93E+00	2.60E+00	1.75E+00	1.11E+00	2.72E+00	6.99E-01	**1.63E-03**
	Best	-10.5360	-10.5363	-10.5356	-6.4545	-8.5120	-10.5340	-10.5363	**-10.5364**

For high-dimensional multimodal functions F6–F11, the IOOA algorithm also demonstrates good optimization capabilities. In functions F10 and F11, although the IOOA algorithm does not converge to the theoretical optimal value, it has the highest optimization accuracy, with the compared algorithms exhibiting varying degrees of optimization error. In functions F7 and F9, both the IOOA algorithm and OOA algorithms can converge to the theoretical optimal value, proving a certain advantage of the algorithms. In the case of 30 dimensions, the IOOA algorithm shows good optimization robustness on all multimodal test functions. When the problem dimension increases from 30 to 100, for functions F10 and F11, the IOOA algorithm's average optimization value decreases, indicating unstable optimization performance. However, for other multimodal functions, the IOOA algorithm shows good optimization accuracy and robustness. A comprehensive analysis of all multimodal function test results demonstrates that the IOOA algorithm has higher convergence accuracy than other algorithms on all functions. The test results fully prove the significant advantage of IOOA algorithm in escaping local optima, validating the effectiveness of the improvement strategy in this paper.

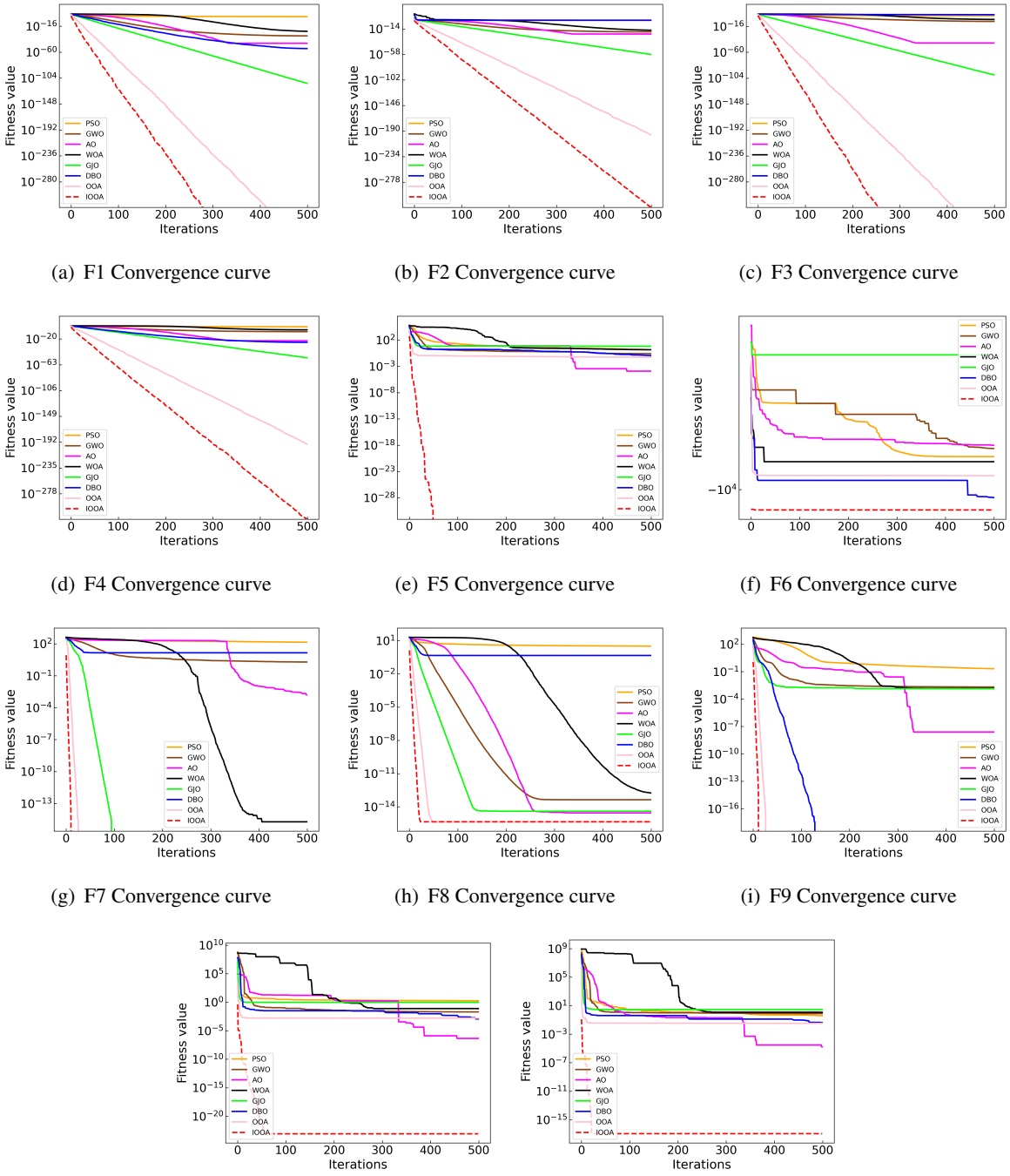
Fixed-dimensional test functions, also referred to as composite functions, comprise a main body constituted by multiple sub-functions, thereby exhibiting notable continuity. From the data presented in Table 5, when faced with fixed-dimensional test functions (F12–F21), the IOOA algorithm consistently demonstrates commendable optimization accuracy and robustness across most test functions. Specifically, in functions F12, F16 and F17, the IOOA algorithm reliably and accurately identifies the optimal solution across 30 runs, showcasing exceptional optimization robustness. In function F14, the IOOA algorithm's optimization performance is slightly lower than that of the GWO, WOA, and DBO algorithms. It can find the theoretical optimal value in multiple runs but has slightly lower robustness. For function F15, the IOOA algorithm's optimization performance is slightly lower than that of the PSO algorithm. For other test functions, the IOOA algorithm exhibits good optimization accuracy and robustness. In summary, the IOOA algorithm has a significant advantage over other algorithms in fixed-dimensional test functions, possessing the strongest overall optimization capability.

In order to show the convergence speed and optimization accuracy of each algorithm more clearly, the average adaptation convergence curve of the iterative process of each algorithm is plotted as shown in Figures 3–5, and the performance ranking of each algorithm for optimization is shown in Figure 6.

The average fitness change curves during the iterative process of various algorithms, as depicted in Figures 3–5, reveal that the IOOA algorithm exhibits the fastest convergence speed and does not experience search stagnation in functions F1–F5. Particularly, in function F5, the IOOA algorithm demonstrates an exceptionally rapid convergence speed. Monotonic functions primarily test the convergence speed and optimization accuracy of algorithms, and the experimental results above fully demonstrate the superiority of the IOOA algorithm in convergence speed.

For the high-dimensional multi-peaked function F6, the IOOA algorithm converged to the neighborhood of the optimal solution at the beginning of the iteration and did not fall into a local optimum. The IOOA algorithm employs Circle chaos mapping to generate a higher-quality initial population, laying a solid foundation for subsequent iterative optimization, to some extent reflecting the effectiveness of the population initialization improvement strategy proposed in this paper. For functions F10 and F11, the IOOA algorithm exhibits a strong ability to escape local optima, while the convergence curves of the compared algorithms show varying degrees of being trapped in local optima, resulting in lower optimization accuracy. The introduction of a dynamic chaotic weight factor during the development phase of the IOOA algorithm effectively enhances the algorithm's local search capability. Faced with

problems with multiple local optimal solutions, it strengthens the algorithm's ability to escape local optima, thereby improving the optimization accuracy of the algorithm.



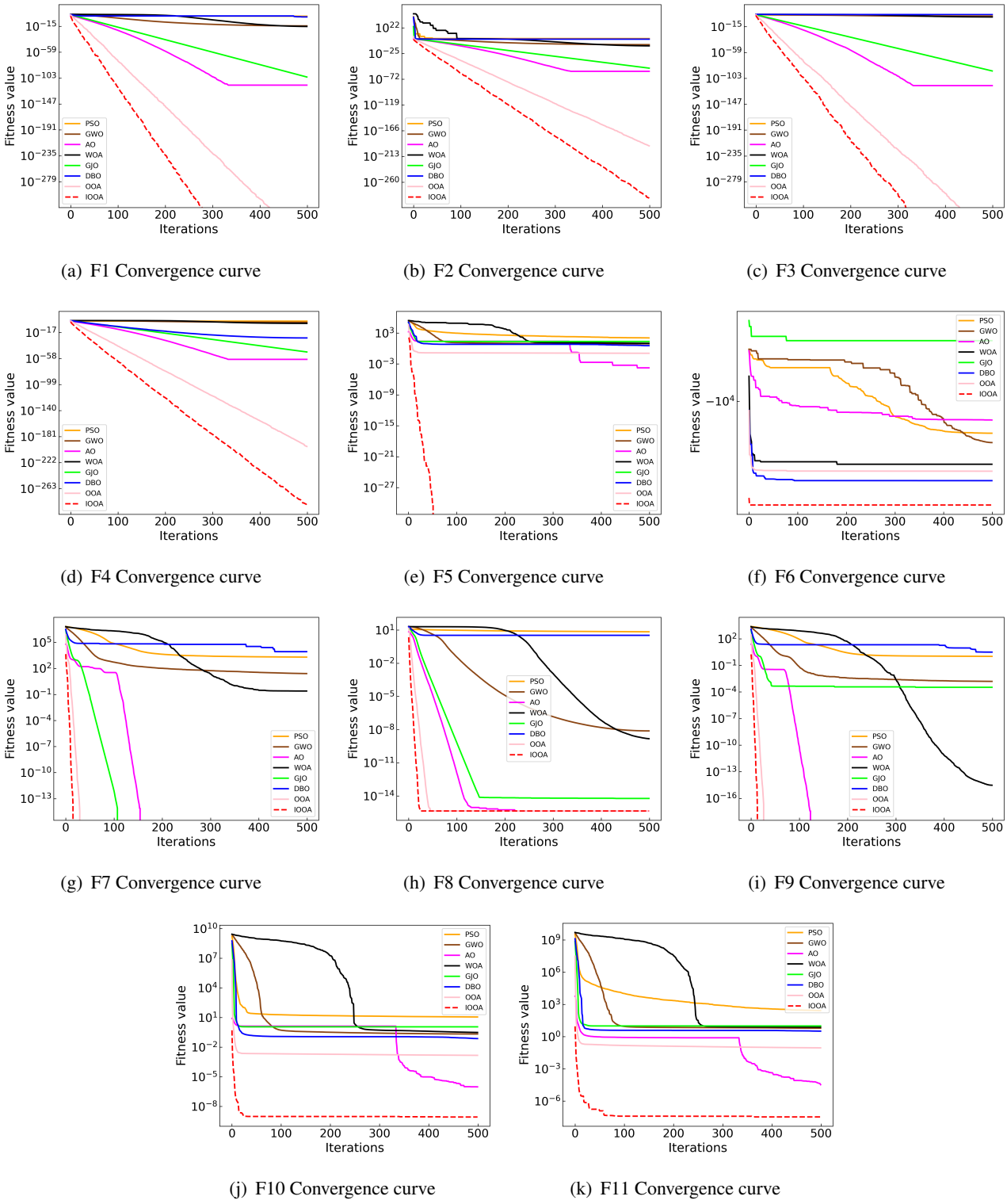
(a) F1 Convergence curve (b) F2 Convergence curve (c) F3 Convergence curve

(d) F4 Convergence curve (e) F5 Convergence curve (f) F6 Convergence curve

(g) F7 Convergence curve (h) F8 Convergence curve (i) F9 Convergence curve

(j) F10 Convergence curve (k) F11 Convergence curve

Figure 3. Convergence curves of all algorithms on 30-dimensional test functions.



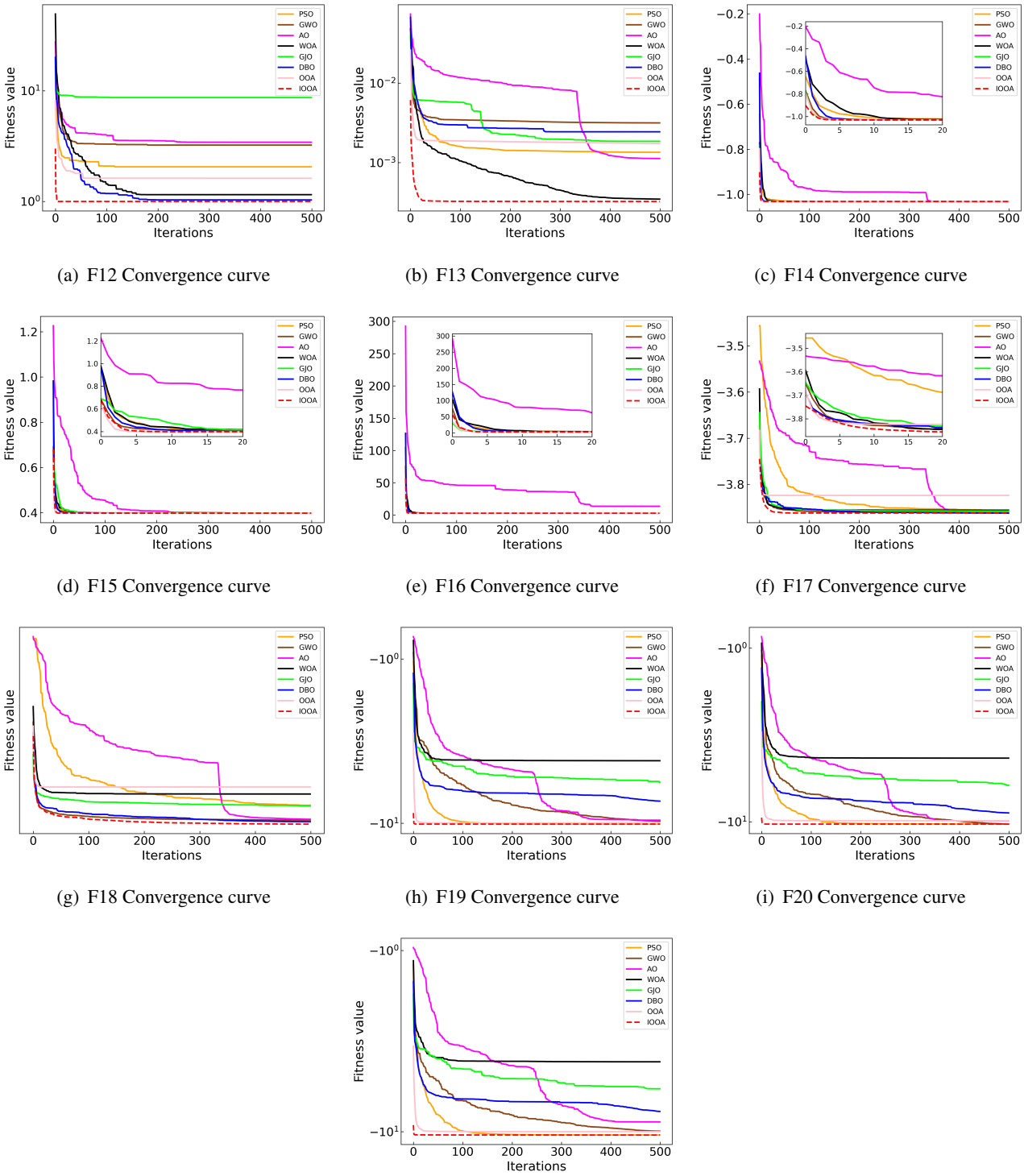
(a) F1 Convergence curve (b) F2 Convergence curve (c) F3 Convergence curve

(d) F4 Convergence curve (e) F5 Convergence curve (f) F6 Convergence curve

(g) F7 Convergence curve (h) F8 Convergence curve (i) F9 Convergence curve

(j) F10 Convergence curve (k) F11 Convergence curve

Figure 4. Convergence curves of all algorithms on 100-dimensional test functions.



(a) F12 Convergence curve (b) F13 Convergence curve (c) F14 Convergence curve

(d) F15 Convergence curve (e) F16 Convergence curve (f) F17 Convergence curve

(g) F18 Convergence curve (h) F19 Convergence curve (i) F20 Convergence curve

(j) F21 Convergence curve

Figure 5. Convergence curves of all algorithms on fixed-dimensional test functions.

The convergence curves of the average fitness values for fixed-dimensional test functions F12–F21 show that, on most test functions, IOOA has the fastest convergence speed and convergence accuracy.

In functions F15 and F16, the IOOA algorithm's convergence speed in the early stages of iteration is slightly lower than that of the OOA algorithm, but in function F16, the IOOA algorithm's convergence accuracy surpasses that of the OOA algorithm. On other functions, IOOA exhibits advantages in both convergence speed and accuracy. The above results demonstrate that when facing combinatorial optimization problems, the IOOA algorithm remains highly competitive.

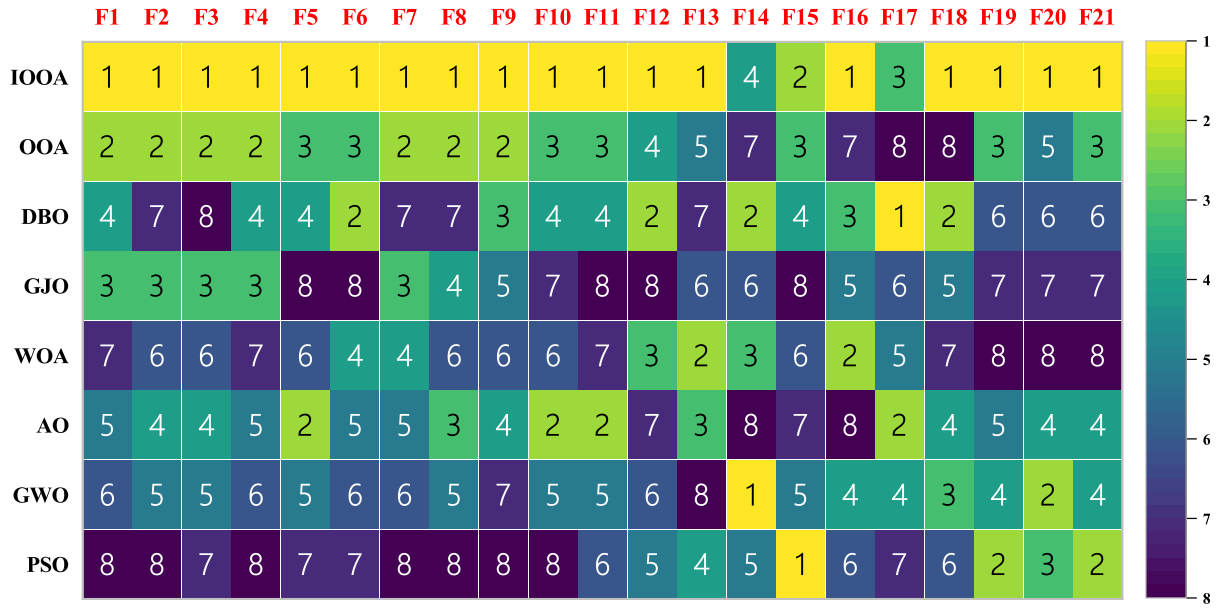
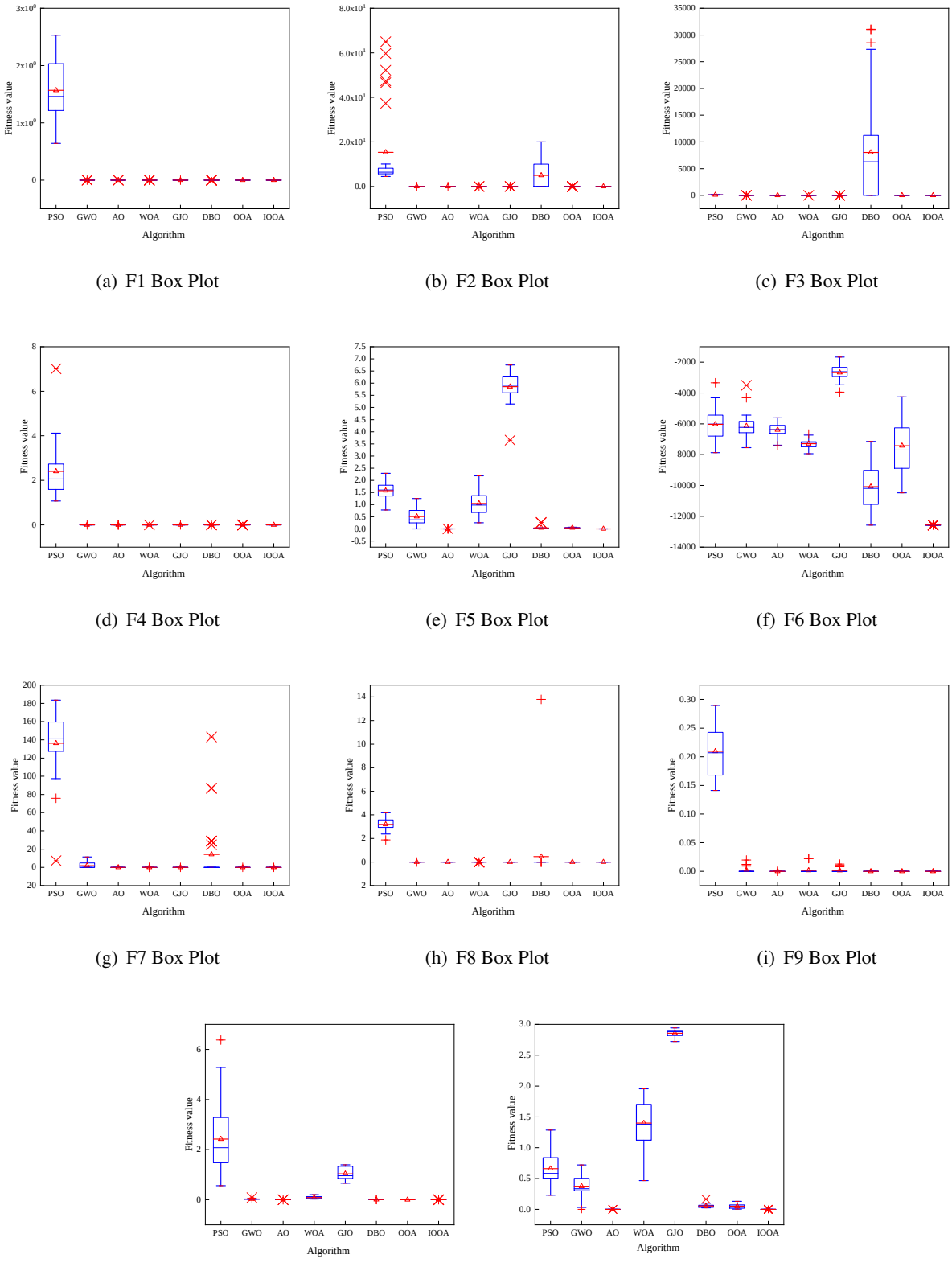


Figure 6. Comprehensive ranking of all algorithms.

The box plot is used to depict the central tendency and dispersion of a set or multiple sets of continuous data distributions. When optimizing algorithms for different problems, there is a certain degree of randomness, so multiple experiments are necessary to eliminate random errors. Drawing box plots by statistically analyzing the optimal values of each algorithm from multiple runs provides a more intuitive reflection of the robustness of each algorithm.

Figures 7–9 present box plots of the distribution of optimal values from multiple runs of each algorithm across all test functions. In both 30-dimensional and 100-dimensional test functions, IOOA exhibits the highest optimization stability, with almost no differences observed between its multiple runs. Particularly in the case of function F6, all comparative algorithms show varying degrees of optimization errors, while IOOA consistently and accurately finds the optimal solution over 30 runs. In the case of 100-dimensional test functions, for function F14 the optimization stability of IOOA is slightly lower than that of the GWO, WOA, and DBO algorithms. For function F17, the stability of IOOA is slightly lower than that of the AO and DBO algorithms. However, for other functions, IOOA consistently demonstrates the highest optimization stability.



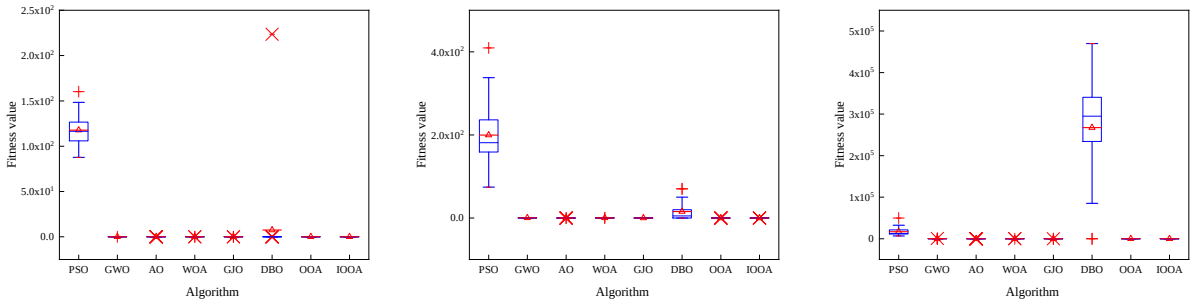
(a) F1 Box Plot (b) F2 Box Plot (c) F3 Box Plot

(d) F4 Box Plot (e) F5 Box Plot (f) F6 Box Plot

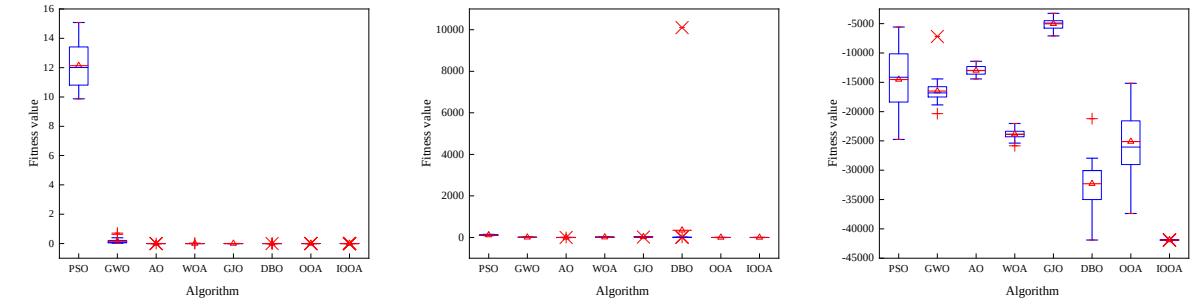
(g) F7 Box Plot (h) F8 Box Plot (i) F9 Box Plot

(j) F10 Box Plot (k) F11 Box Plot

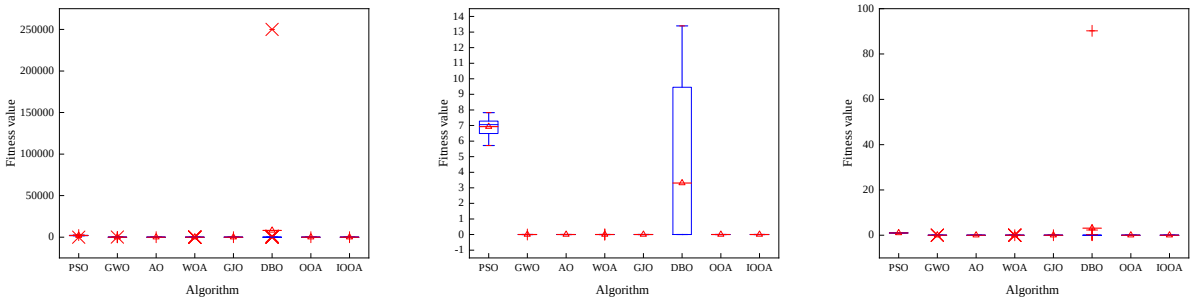
Figure 7. Box Plot of all algorithms on 30-dimensional test functions.



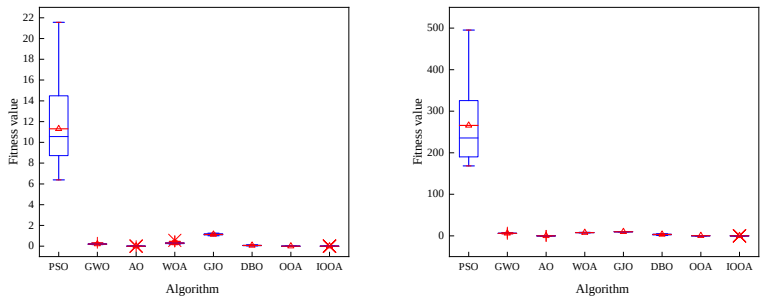
(a) F1 Box Plot (b) F2 Box Plot (c) F3 Box Plot



(d) F4 Box Plot (e) F5 Box Plot (f) F6 Box Plot

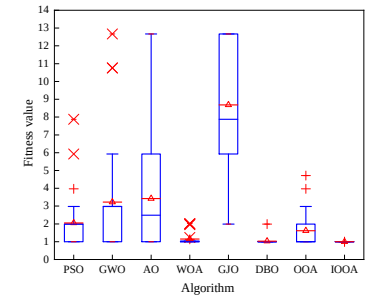


(g) F7 Box Plot (h) F8 Box Plot (i) F9 Box Plot

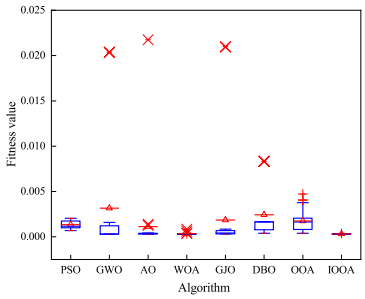


(j) F10 Box Plot (k) F11 Box Plot

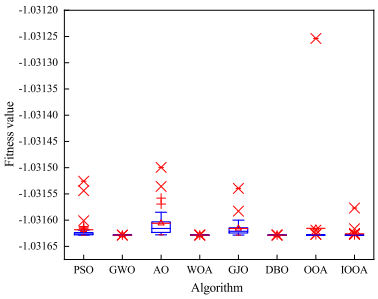
Figure 8. Box Plot of all algorithms on 100-dimensional test functions.



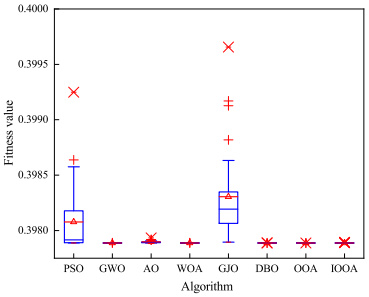
(a) F12 Box Plot



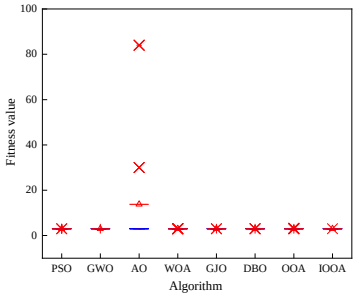
(b) F13 Box Plot



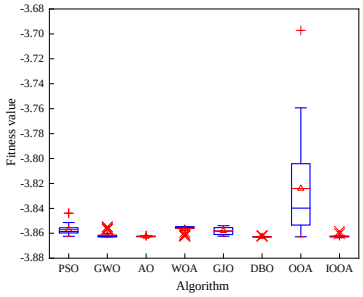
(c) F14 Box Plot



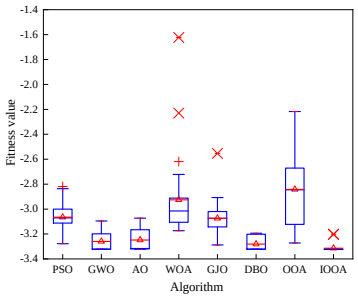
(d) F15 Box Plot



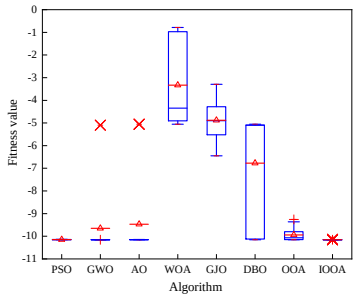
(e) F16 Box Plot



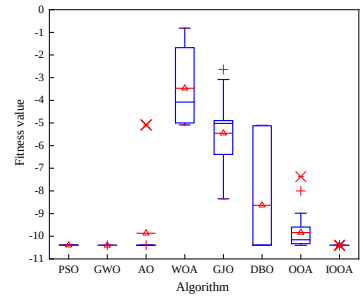
(f) F17 Box Plot



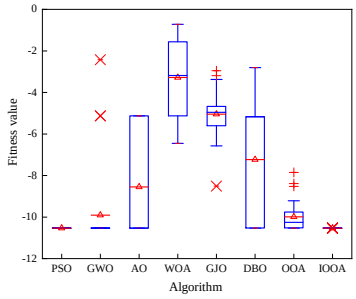
(g) F18 Box Plot



(h) F19 Box Plot



(i) F20 Box Plot



(j) F21 Box Plot

Figure 9. Box Plot of all algorithms on fixed-dimensional test functions.

The optimization performance evaluation metrics and convergence curves of each algorithm indicate that the IOOA algorithm exhibits a pronounced advantage in comprehensive optimization performance. The fusion strategy proposed in this paper significantly enhances the performance of the OOA algorithm. The utilization of Circle chaos mapping to generate the initial population contributes to an increased diversity in the initial population, thereby elevating the quality of initial solutions. The introduction of a dynamic elite guidance mechanism greatly accelerates the algorithm's convergence speed, preventing instances of ineffective searches. The dynamic chaotic weight strategy during the developmental phase enhances the algorithm's local search capability to a certain extent, avoiding entrapment in local optima and concurrently improving optimization accuracy. The results of the aforementioned simulation experiments comprehensively validate the effectiveness of the proposed improvement strategy and the superior performance of the IOOA algorithm. This robust empirical evidence establishes a solid theoretical foundation for the practical engineering application of the IOOA algorithm.

4.2. Wilcoxon rank-sum test

The Wilcoxon rank-sum test [32] is employed to validate whether there is a significant difference between two independent samples. A p-value greater than 0.05 indicates no significant difference between the compared algorithms, while a p-value less than 0.05 suggests a significant difference. N/A indicates that the Wilcoxon rank-sum test is not applicable for the two samples.

In this section, statistical analysis of optimization results for each algorithm is conducted using the Wilcoxon rank-sum test. At a significance level of $\alpha = 5\%$, all algorithms are independently run 30 times, and the Wilcoxon rank-sum test p-values are presented in Table 6.

The statistical results in Table 6 reveal that the IOOA algorithm's optimization results differ significantly from the PSO algorithm across all test functions. For functions F14 and F15, there is no significant difference between the IOOA algorithm and the GWO and WOA algorithms. Compared to the AO algorithm, there is no significant difference on function F17. The GJO algorithm converges to the theoretical optimum on function F7, making the Wilcoxon rank-sum test inapplicable for this case. When compared to the DBO algorithm, the Wilcoxon rank-sum test is not applicable for function F9, and there is no significant difference between the IOOA and DBO algorithms on function F15. On functions F1, F3, and F7–F9, the IOOA algorithm demonstrates the same optimization accuracy as the OOA algorithm, stabilizing at the theoretical optimum across multiple runs. Therefore, the Wilcoxon rank-sum test is not applicable for these functions.

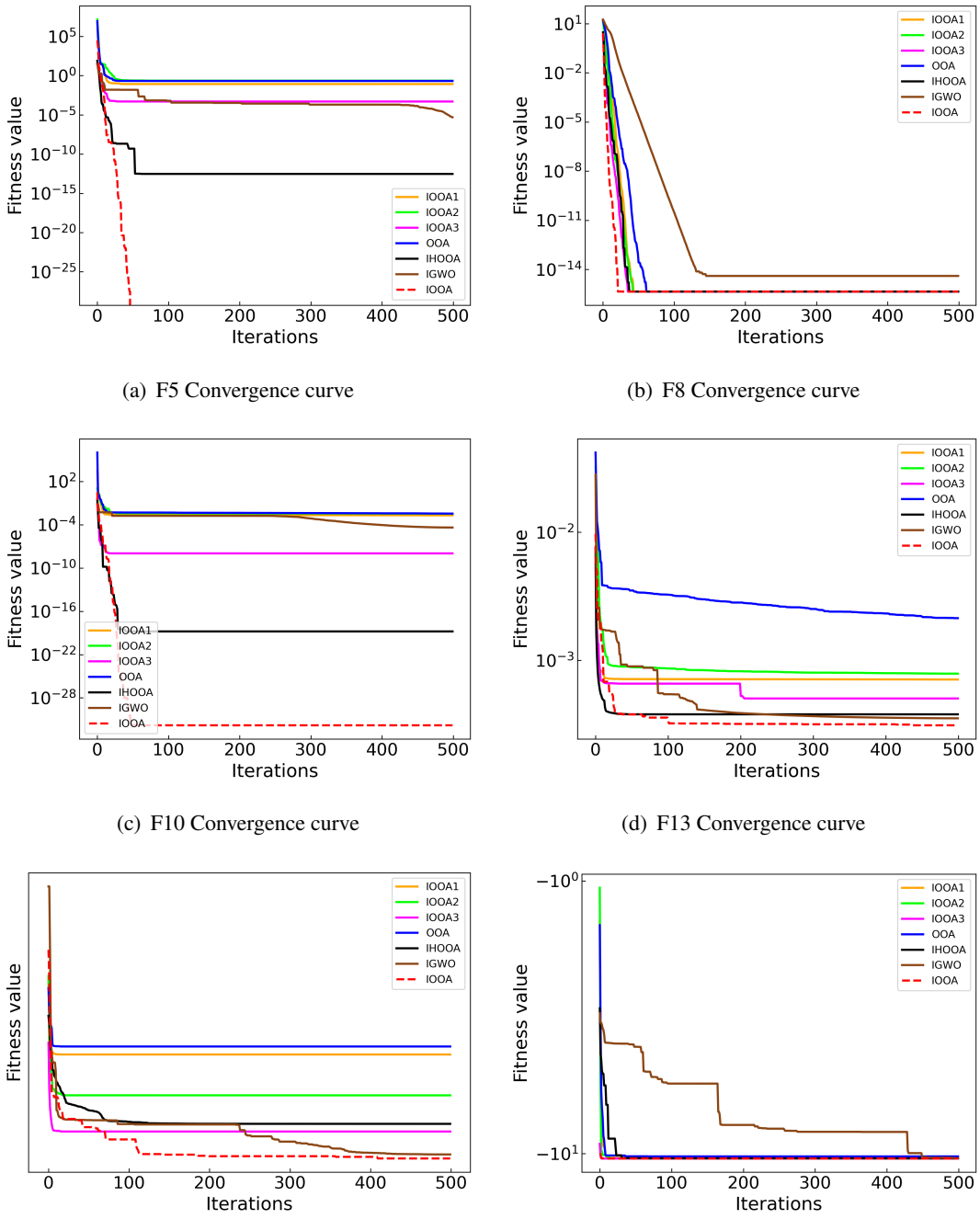
Table 6. Wilcoxon rank-sum test for the test functions.

Function	PSO	GWO	AO	WOA	GJO	DBO	OOA
F1	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	N/A
F2	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F3	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	N/A
F4	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F5	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F6	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	4.83E-10	2.87E-11
F7	2.87E-11	2.87E-11	1.26E-10	2.87E-11	N/A	2.87E-11	N/A
F8	2.87E-11	2.87E-11	9.19E-06	2.87E-11	2.87E-11	2.10E-09	N/A
F9	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	N/A	N/A
F10	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F12	5.65E-06	6.80E-08	9.13E-10	3.50E-11	2.87E-11	3.41E-03	5.41E-04
F13	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F14	2.40E-06	4.33E-01	7.43E-09	7.67E-01	2.95E-08	4.53E-03	2.95E-08
F15	9.67E-09	9.77E-02	3.80E-08	1.43E-01	2.87E-11	2.19E-01	1.54E-06
F16	4.27E-07	2.14E-03	3.17E-11	5.84E-10	5.20E-03	2.12E-09	2.12E-09
F17	4.00E-10	4.24E-03	5.84E-02	1.69E-10	1.12E-09	2.27E-02	9.31E-10
F18	7.76E-11	6.24E-10	7.73E-10	2.87E-11	5.22E-11	2.12E-09	5.22E-11
F19	7.73E-10	1.94E-09	4.00E-10	2.87E-11	2.87E-11	6.37E-11	1.53E-10
F20	2.05E-10	4.27E-07	9.31E-10	2.87E-11	2.87E-11	9.44E-11	4.28E-11
F21	3.31E-10	5.65E-06	1.69E-10	2.87E-11	2.87E-11	5.77E-11	6.81E-09

4.3. Analysis of the effectiveness of the improvement strategy for the IOOA algorithm

In this section, the effectiveness of the improvement strategy of the IOOA algorithm is analyzed through ablation experiments and comparison to improved versions of several well-known optimization algorithms. The improved OOA algorithm with only Circle chaotic mapping is named as IOOA1; the improved OOA algorithm with only dynamic elite guidance mechanism is named as IOOA2; and the improved OOA algorithm with only dynamic chaotic mapping is named as IOOA3. The improved optimization algorithms include the IHOOA algorithm [33] and the IGWO algorithm [34]. The experiment sets the population number to 50, the maximum iteration number to 500, and each algorithm runs independently 30 times. The test functions are: high-dimensional single-peak function (F5);

high-dimensional multi-peak functions (F8, F10); fixed-dimensional functions (F13, F18, F21). The change curve of the average fitness value during the iteration process of each algorithm is shown in Figure 10.



(a) F5 Convergence curve

(b) F8 Convergence curve

(c) F10 Convergence curve

(d) F13 Convergence curve

(e) F18 Convergence curve

(f) F21 Convergence curve

Figure 10. Experimental results of IOOA algorithm effectiveness analysis.

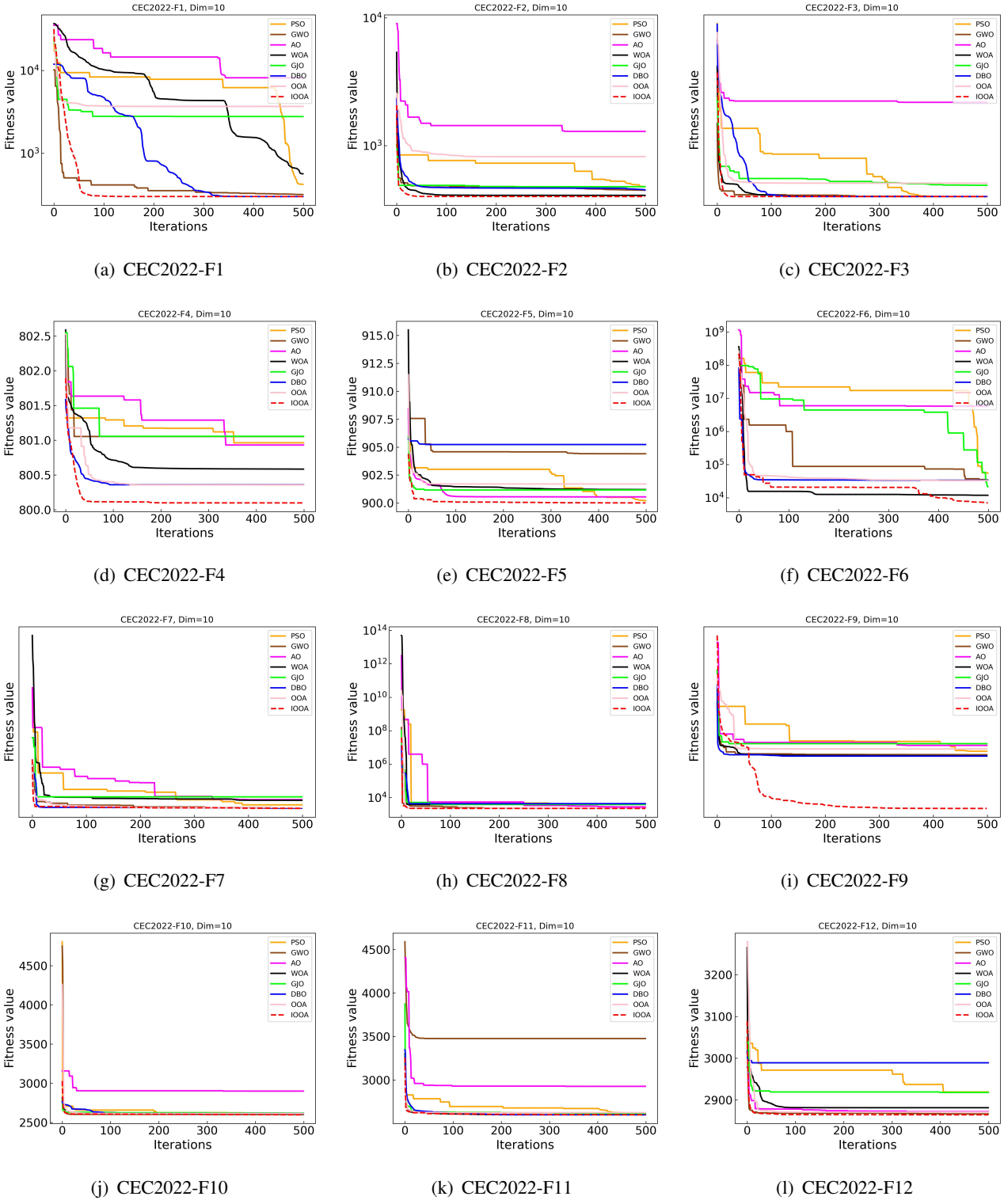
As can be seen from Figure 10, when each improvement strategy acts on the OOA algorithm alone, the performance of the algorithm is improved to a certain extent. When the three strategies act together in the OOA algorithm, the algorithm has the fastest convergence speed and the highest optimization accuracy, which has a significant advantage compared with other algorithms. Compared with the improved versions of other well-known algorithms, the IOOA algorithm also has a strong performance advantage. The above experimental results fully prove the effectiveness of the proposed improvement strategy. The strategy significantly improves the optimization performance of the OOA algorithm.

4.4. IOOA's performance on CEC-2022 benchmark functions

In order to further verify the optimization seeking performance of the IOOA algorithm, this section uses the CEC-2022 benchmark functions to test the performance of the IOOA algorithm. The parameter settings of each algorithm are shown in Table 1. There are 12 single-objective test functions with boundary constraints in the CEC-2022 benchmark functions, which are: the unimodal function (F1), multimodal function (F2–F5), hybrid function (F6–F8), and combined function (F9–F12). All the test functions solve minimization problems. The specific information of the CEC-2022 benchmark functions is shown in Table 7. The convergence curves of all algorithms on CEC-2022 benchmark functions are shown in Figure 11.

Table 7. CEC-2022 benchmark functions.

ID	Functions	Range	Optimum
F1	Shifted and full Rotated Zakharov Function	[−100, 100]	300
F2	Shifted and full Rotated Rosenbrock's Function	[−100, 100]	400
F3	Shifted and full Rotated Expanded Schaffer's f6 Function	[−100, 100]	600
F4	Shifted and full Rotated Non-Continuous Rastrigin's Function	[−100, 100]	800
F5	Shifted and full Rotated Levy Function	[−100, 100]	900
F6	Hybrid Function 1 (N = 3)	[−100, 100]	1800
F7	Hybrid Function 2 (N = 6)	[−100, 100]	2000
F8	Hybrid Function 3 (N = 5)	[−100, 100]	2200
F9	Composition Function 1 (N = 5)	[−100, 100]	2300
F10	Composition Function 2 (N = 4)	[−100, 100]	2400
F11	Composition Function 3 (N = 5)	[−100, 100]	2600
F12	Composition Function 4 (N = 6)	[−100, 100]	2700



(a) CEC2022-F1 (b) CEC2022-F2 (c) CEC2022-F3

(d) CEC2022-F4 (e) CEC2022-F5 (f) CEC2022-F6

(g) CEC2022-F7 (h) CEC2022-F8 (i) CEC2022-F9

(j) CEC2022-F10 (k) CEC2022-F11 (l) CEC2022-F12

Figure 11. Convergence curves of all algorithms on CEC-2022 benchmark functions.

As can be seen from Figure 11, the IOOA algorithm exhibits the highest comprehensive optimization performance on all CEC-2022 benchmark functions. On the unimodal function F1, the convergence speed of the IOOA algorithm is slightly lower than that of the GWO algorithm in the pre-iterative stage, mainly because the elite guidance mechanism of the IOOA algorithm in the pre-iterative stage accounts for a smaller proportion of the algorithm, and the algorithm mainly adopts a random search strategy to extensively explore the solution space. In the late iteration, the search strategy of the IOOA algorithm is changed to the elite guidance mechanism, which enables the algorithm to converge to the optimal solution quickly. At the same time, due to the introduction of dynamic chaotic weighting factors, the IOOA algorithm iteration process did not exhibit search stagnation, proving that the IOOA algorithm has a good ability to get out of local optima.

For multimodal functions F2–F5, the IOOA algorithm shows the fastest convergence speed with the highest optimization accuracy. Especially for functions F4 and F5, the addition of dynamic chaotic weighting factor strategy enables the IOOA algorithm to conduct a finer search near the current optimal solution, which further improves the algorithm's optimization accuracy.

For the hybrid functions F6–F8, the IOOA algorithm converges slightly slower than the WOA algorithm on function F6, but the IOOA algorithm has the highest optimization seeking accuracy. On functions F7 and F8, the IOOA algorithm shows good comprehensive optimization search performance.

For the combined functions F9–F12, the comparative algorithms all suffer from search stagnation to varying degrees on function F9, and the IOOA algorithm is able to avoid the influence of local optima during the iteration process, and so it has the highest optimization accuracy. The IOOA algorithm also has the strongest comprehensive optimization performance in the remaining combinatorial functions.

In summary, when facing the complex single-objective test function with boundary constraints, the unique optimization mechanism of the IOOA algorithm makes it show strong competitiveness. The dynamic elite guidance mechanism can greatly improve the convergence speed of the algorithm while ensuring that the algorithm has good global search capability. The dynamic chaotic weight factor strategy can prevent the algorithm from falling into local optimization and improve the optimization accuracy of the algorithm. The above simulation experiments fully prove the effectiveness of the improved strategy in this paper and the performance advantages of the IOOA algorithm.

4.5. IOOA applied to the LSTM power load forecasting problem

Accurate prediction of power load contributes to the development of rational power scheduling strategies by relevant professionals [35]. Due to the nonlinear characteristics of both the load and the various factors influencing it, the prediction of power load poses significant challenges. Thanks to the rapid advancement of artificial intelligence technology, load forecasting models based on recurrent neural networks (RNN) have garnered widespread attention. Such methods excel at extracting nonlinear relationships from historical data and have found extensive application in the field of power load forecasting [36].

LSTM is a special kind of recurrent neural network, which is characterized by its ability to cope with the gradient vanishing and gradient explosion problems during long sequence training [37]. Compared with the traditional recurrent neural network, LSTM adds three gates, which are the input gate, output gate, and forgetting gate, which makes LSTM able to selectively save the useful data, and it has higher prediction accuracy for time-dependent problems. The LSTM structure is shown in Figure 12.

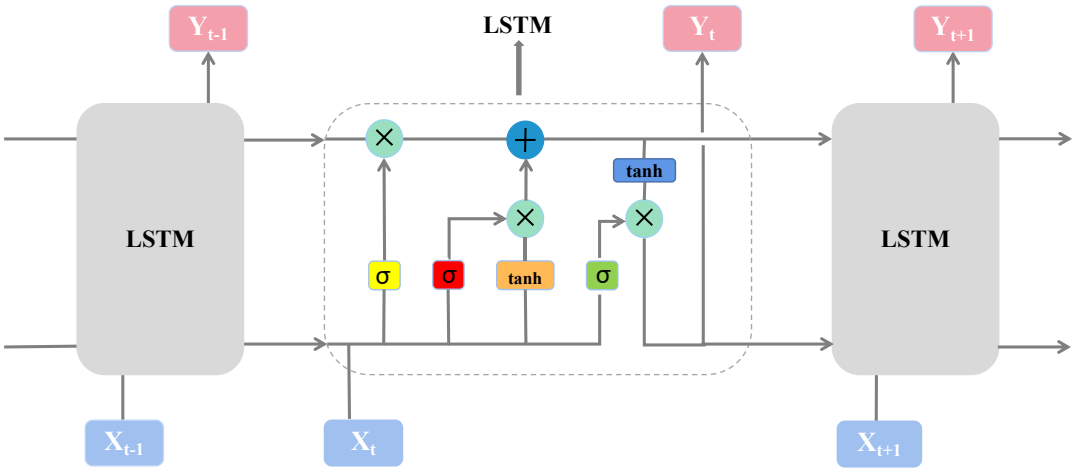


Figure 12. LSTM structure.

However, when facing complex time series tasks, it is difficult to achieve good prediction results when only using the basic LSTM model to deal with the problem, so a large number of researchers have improved the LSTM model to cope with complex prediction problems. [38] proposed an improved anti-noise adaptive long short-term memory neural network for robust remaining useful life prediction of lithium-ion batteries. Compared with other methods, the model has higher prediction accuracy and provides an effective method for the industrialized application of lithium-ion batteries. Reference [39] proposed an improved singular filter-Gaussian process regression-long and short-term memory model for lithium-ion battery remaining capacity estimation. The model was evaluated by multiple evaluation metrics. Experimental results showed that the method can achieve good prediction performance by using only a small number of datasets, which lays a theoretical foundation for the estimation of the remaining capacity of the battery's full lifecycle at extremely low temperatures.

In addition to the above methods, the performance and generalization ability of the LSTM model are influenced by hyperparameters, which are usually selected based on experience and have considerable uncertainty [40]. Metaheuristic algorithms have significant advantages over traditional methods when dealing with complex, nonlinear problems. Therefore, optimizing LSTM model hyperparameters through metaheuristic algorithms is an effective approach [41].

LSTM model hyperparameters include the number of neurons in the hidden layer, learning rate, training iterations, and so on. In this study, the IOOA algorithm is employed to search for LSTM hyperparameters. The fitness function is defined as the root mean square error between predicted values and actual values. Through the IOOA algorithm, a set of hyperparameters with the minimum prediction error is identified. This set of hyperparameters is then utilized to construct an LSTM model for power load prediction. The accuracy of the model's predictions is further used to evaluate the optimization

performance of the IOOA algorithm and its feasibility in practical engineering applications.

The experimental data consists of power load data for a specific region over a period of ten years, with a sampling interval of 1 day. The dataset comprises 3645 load samples, each with 3 features and 1 target. The first 70% of the data is designated as the training set, while the remaining data serves as the test set. Prior to analysis, the dataset undergoes preprocessing, involving empirical mode decomposition (EMD) of the original load data to mitigate the non-linear effects. All the algorithms mentioned earlier are employed for hyperparameter optimization of the LSTM model.

The performance evaluation metrics for the model include mean absolute percentage error (MAPE), root mean square error (RMSE), mean absolute error (MAE), and R-Squared (R^2). The fitting curves of the predicted values and actual values for each model are illustrated in Figure 13.

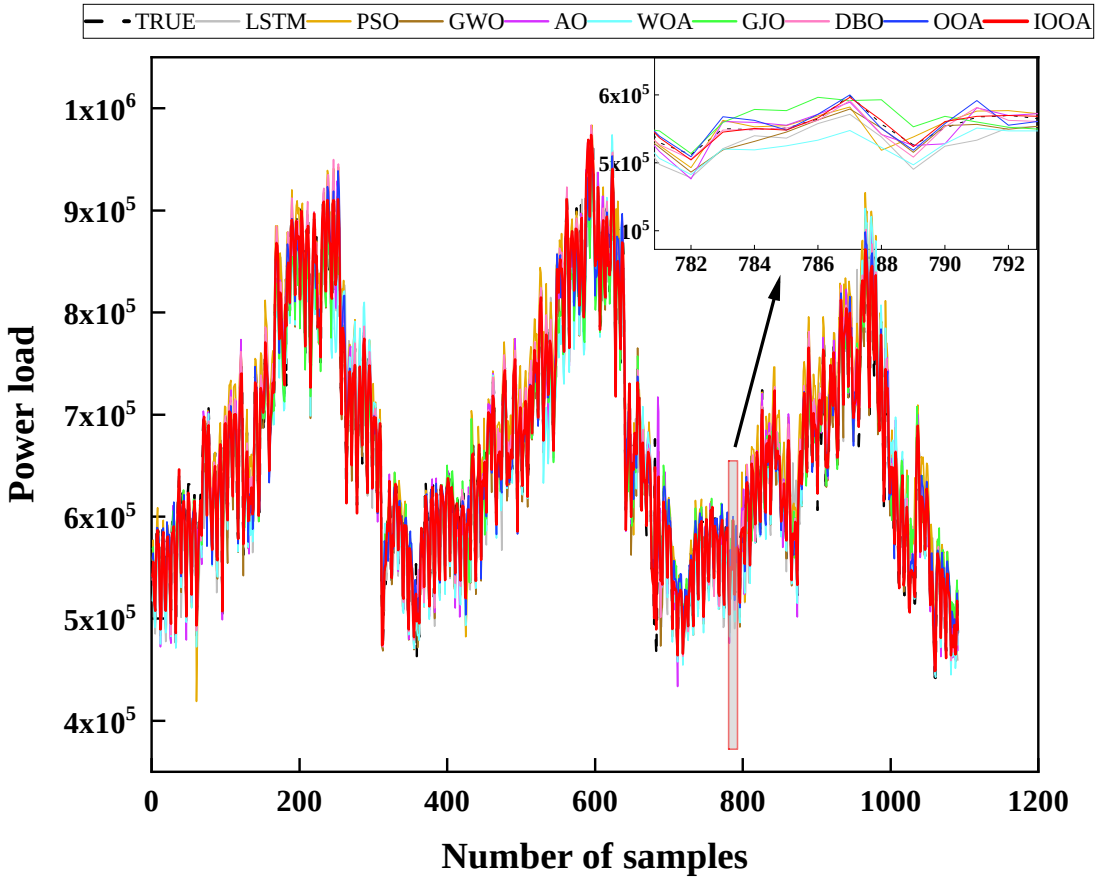


Figure 13. Prediction curves for each algorithm.

Figure 14 shows the prediction error curves for each model, and Table 8 gives the four performance evaluation metrics for each model.

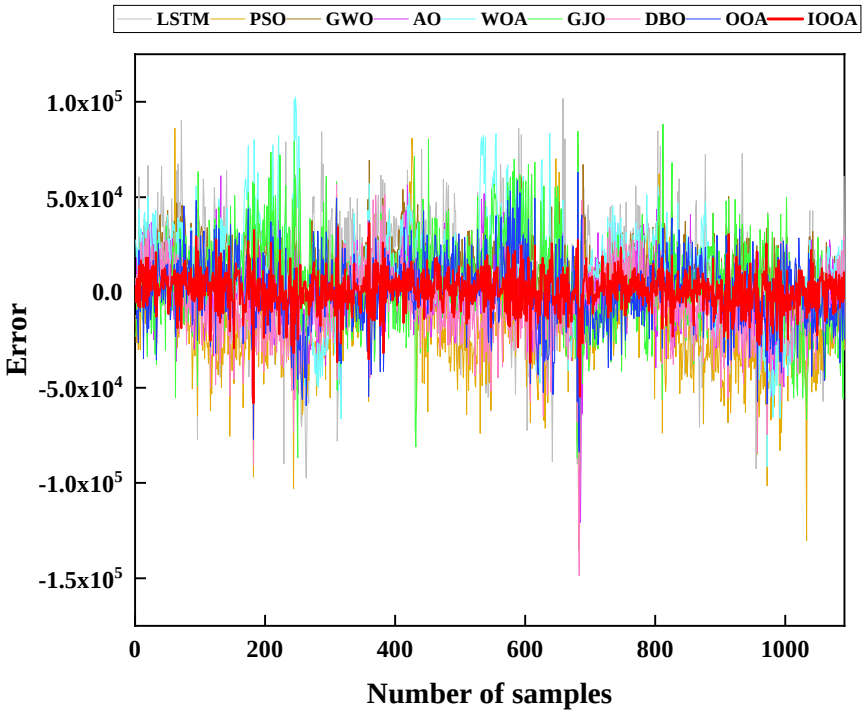


Figure 14. Prediction error curves for each model.

Table 8. Evaluation indicators of various model prediction results.

Model	MAPE	RMSE	MAE	R^2
LSTM	0.0388	32857.95	27036.39	0.9140
PSO-LSTM	0.0354	29912.61	24685.67	0.9291
GWO-LSTM	0.0309	25962.57	20405.28	0.9466
AO-LSTM	0.0227	20859.40	16589.29	0.9656
WOA-LSTM	0.0341	27674.84	23014.15	0.9327
GJO-LSTM	0.0268	23681.01	18450.37	0.9554
DBO-LSTM	0.0235	22869.75	17679.28	0.9640
OOA-LSTM	0.0201	18686.58	14268.64	0.9723
IOOA-LSTM	0.0119	10021.64	8421.41	0.9901

As can be seen from Figure 13, the IOOA-LSTM model prediction curve has the highest fit with the real load curve, the curve trend is basically the same, and the prediction value of the IOOA-LSTM model is closer to the real value compared to other models. Figure 14 visualizes the prediction errors of all models, and among all sample data, the IOOA-LSTM model has the smallest error range interval, proving that the model has the highest prediction accuracy. As can be seen from the evaluation metrics of each model in Table 8, the IOOA-LSTM model has an R^2 of 0.9901, which is higher than all the

compared models. Compared with the base LSTM model, the prediction accuracy of the IOOA-LSTM model is improved by 7.61%, which is a significant advantage.

The above experimental results fully prove that the IOOA algorithm can effectively find the hyperparameters of the LSTM model, and the LSTM model constructed by this set of hyperparameters has the highest prediction accuracy, which further verifies the effectiveness of the improvement strategy in this paper and the superior performance of the IOOA algorithm. For the model optimization problem in the field of machine learning, the IOOA algorithm proposed in this paper is able to find the optimal solution stably, reflecting its high reliability.

4.6. Engineering design problem

In order to fully verify the comprehensive optimization performance of the IOOA algorithm in the face of problems with complex constraints, two well-known engineering design problems in the structural field are used in this section to test the performance of the IOOA algorithm. They are the tension/compression spring design problem and the three-bar truss design problem, respectively [42, 43]. The IOOA algorithm is compared with other 11 famous optimization algorithms, and all algorithms have the same constraints to ensure the fairness of the experiment.

4.6.1. Tension/compression spring design problem

The objective of the tension/compression spring design problem is to minimize the weight of the spring while satisfying the constraints of minimum deflection, shear stress, vibration frequency, and ultimate outside diameter. The problem consists of three consecutive decision variables, namely, the spring coil diameter (x_1), the overall spring diameter (x_2), and the number of coils wound (x_3). The constraints are minimum deflection ($g_1(X)$), shear stress ($g_2(X)$), vibration frequency ($g_3(X)$), and limiting outer diameter ($g_4(X)$). The objective function of the problem is as follows:

$$
\begin{aligned}
f(X) &= (x_3 + 2)\, x_2 x_1^2, \\
g_1(X) &= 1 - \frac{x_2^3 x_3}{71785 x_1^4} \le 0, \\
g_2(X) &= \frac{4x_2^2 - x_1 x_2}{12566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5180 x_1^2} - 1 \le 0, \\
g_3(X) &= 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0, \\
g_4(X) &= \frac{x_1 + x_2}{1.5} - 1 \le 0, \\
0.05 \le x_1 \le 2, 0.25 \le x_2 \le 1.3, 2 \le x_3 \le 15.&
\end{aligned}
$$

The results of simulation experiments for the tension/compression spring design problem are shown in Table 9. The comparison algorithms include the PSO algorithm, GWO algorithm, AO algorithm, WOA algorithm, GJO algorithm, DBO algorithm, OOA algorithm, HHO algorithm [44], TSA algorithm [45], SMA algorithm [46], and SABO algorithm [47].

Table 9. Comparative results for the tension/compression spring problem.

Algorithms	x_1	x_2	x_3	f(X)
PSO	0.05828019	0.5352092	5.7625443	0.01411138
GWO	0.05300526	0.38917177	9.62539881	0.01271122
AO	0.05	0.31472432	14.41789636	0.01291778
WOA	0.0523544	0.37222075	10.48502312	0.01273787
GJO	0.05029085	0.32398347	13.50713292	0.01270669
DBO	0.05	0.31736567	14.04027594	0.01272658
OOA	0.05379799	0.40950601	8.75926347	0.0127519
HHO	0.05	0.31740887	14.03034291	0.01272043
TSA	0.05328826	0.39613607	9.48538621	0.01291972
SMA	0.055857	0.464672	7.13884204	0.013255
SABO	0.05301165	0.38778914	9.74747372	0.01280215
IOOA	0.05277867	0.38350141	9.87567011	0.0126865

The simulation experiment results show that the IOOA algorithm has the highest optimization accuracy in solving the tension/compression spring design problem. Under satisfying the constraints, the optimization result of the IOOA algorithm is f(X) = 0.0126865, and the optimal solution is X = [0.05277867, 0.38350141, 9.87567011]. The above experimental results show that the IOOA algorithm has good optimization performance and is extremely competitive in solving the tension/compression spring design problem.

4.6.2. Three-bar truss design problem

The three-bar truss is a common structural form widely used in bridges, buildings, and mechanical equipment. The purpose of the three-bar truss design problem is to minimize the volume of the three-bar truss by adjusting the cross-sectional area. The three-bar truss is subjected to stress (σ) constraints at each truss member. The design has three nonlinear continuous inequality constraints ($g_1(X)$, $g_2(X)$, and $g_3(X)$) and two continuous decision variables (x_1, x_2). The objective function of the problem is as follows:

$$
\begin{aligned}
f(X) &= \left(2\sqrt{2}x_1 + x_2\right) \times L, \\
g_1(X) &= \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \le 0, \\
g_2(X) &= \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \le 0, \\
g_3(X) &= \frac{1}{\sqrt{2}x_2 + s_1}P - \sigma \le 0, \\
0.001 \le x_1 \le 1, &0.001 \le x_2 \le 1, \\
L = 100\ \text{cm}, P = 2\text{KN/cm}^2, &\sigma = 2\text{KN/cm}^2.
\end{aligned}
$$

The results of simulation experiments for the three-bar truss design problem are shown in Table 10. The comparison algorithms include the GWO algorithm, AO algorithm, WOA algorithm, GJO algorithm, DBO algorithm, OOA algorithm, HHO algorithm, TSA algorithm, SMA algorithm, and SABO algorithm.

Table 10. Comparative results for the three-bar truss design problem.

Algorithms	x_1	x_2	f(X)
GWO	0.78916735	0.40687817	263.89804951
AO	0.7847076	0.41990285	263.9391
WOA	0.79453367	0.39281604	264.00952075
GJO	0.79180563	0.40157195	264.11364703
DBO	0.79220303	0.39226958	263.9958
OOA	0.78406522	0.42144665	263.91179684
HHO	0.79395286	0.39352164	263.9159446
TSA	0.77405138	0.45448349	264.38313985
SMA	0.79259872	0.39727822	263.9085954
SABO	0.78806251	0.41000207	263.89794479
IOOA	0.787675	0.41108447	263.8965

The optimal fitness value obtained by the IOOA algorithm when faced with the three-bar truss design problem is f(X) = 263.8965, and the optimal solution is X = [0.787675, 0.41108447]. This result outperforms the optimality finding results of other compared algorithms. The above results fully demonstrate that the IOOA algorithm is capable of handling engineering design problems with complex constraints.

5. Conclusions

This paper proposes a multi-strategy fusion improved osprey optimization algorithm (IOOA). Different from the traditional meta-heuristic algorithm improvement, this study is based on the detailed analysis of the improvement strategy; while retaining the performance advantages of the original algorithm, the improvement strategy proposed in this paper significantly improves the algorithm's optimization accuracy, convergence speed, and robustness. The algorithm performance is tested by a variety of performance evaluation indexes, and the following conclusions are drawn:

1) Chaotic mapping has traversal and randomness, and the IOOA algorithm improves the population initialization by replacing random numbers with Circle chaotic mapping, which reduces the random fluctuation of population initialization and improves the robustness of the algorithm. Compared with other algorithms, the initial population individuals of the IOOA algorithm have higher quality, and good initial individuals lay a solid foundation for algorithm optimization. Simulation experiments further verify the conclusion.

2) The original algorithm adopts a random position update strategy, which does not take into account

the position information of the current optimal individual, and this strategy may lead to ineffective searching in the algorithm to a certain extent. The elite guidance mechanism can utilize the individual position information of the current optimal solution to guide the population to approach the optimal individual quickly; however, it is not a good choice to use the elite guidance mechanism at the initial stage of the algorithm iteration. When the elite individuals fall into the local optimum, the algorithm will have a stagnant search. The IOOA algorithm has a faster convergence speed, mainly due to the adjustable ratio dynamic elite guidance mechanism, which makes the algorithm focus more on global exploration in the early stage of the search, and in the late stage of the search through the elite individuals to guide the population to converge quickly to avoid ineffective search, so it greatly improves the convergence speed of the algorithm.

3) In this paper, the traditional inertia weighting strategy is fused with Cubic chaotic mapping, and dynamic chaotic weighting factors are proposed. The weight factor is dynamically adjusted through the change of iteration number, and the traversal of Cubic chaotic mapping is utilized to enhance the local search capability of the algorithm, which greatly improves the optimization accuracy of the algorithm.

4) In terms of algorithm performance testing, the optimization accuracy, convergence speed, and robustness of the IOOA algorithm are extensively verified through 21 benchmark test functions in both 30-dimensional and 100-dimensional cases. Meanwhile, this paper comprehensively examines the effectiveness of the IOOA algorithm through CEC-2022 benchmark functions, ablation experiments, and engineering design problems in a real environment. Simulation experiment results demonstrate that the fusion strategy proposed in this paper markedly enhances the algorithm's optimization performance. Statistical results indicate a significant advantage of the IOOA algorithm over the comparison algorithms.

5) For the LSTM power load forecasting problem, the proposed IOOA-LSTM model has higher prediction accuracy compared with the traditional model due to the strong global optimization performance of the IOOA algorithm, which is able to accurately find the LSTM model hyperparameters. Meanwhile, the accurate power load prediction helps to specify the energy scheduling strategy and can improve the energy utilization rate.The proposed IOOA algorithm provides a proven method for the model optimization problem in the field of machine learning. For engineering design problems with complex constraints, the IOOA algorithm also has good optimization accuracy. Compared with other algorithms, the IOOA algorithm shows strong competitiveness.

Future research endeavors will focus on further improving the optimization performance of the IOOA algorithm and thoroughly exploring additional applications in the field of global optimization.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work was supported by Science and Technology Project of Hebei Education Department (QN2021050) and the Basic Scientific Research Business Fund Project of Universities in Hebei Province (No. 2022CXTD08)

Conflict of interest

The authors declare no confict of interest.

References

1. L. Abualigah, M. A. Elaziz, A. M. Khasawneh, M. Alshinwan, A. H. Gandomi, Meta-heuristic optimization algorithms for solving real-world mechanical engineering design problems: a comprehensive survey, applications, comparative analysis, and results, *Neural Comput. Appl.*, **34** (2022), 4081–4110. https://doi.org/10.1007/s00521-021-06747-4
2. Y. F. Cui, Z. Q. Geng, Q. X. Zhu, Y. M. Han, Review: Multi-objective optimization methods and application in energy saving, *Energy*, **125** (2017), 681–704. https://doi.org/10.1016/j.energy.2017.02.174
3. J. Tang, G. Liu, Q. Pan, A Review on Representative swarm intelligence algorithms for solving optimization problems:applications and trends, *IEEE/CAA J. Autom. Sin.*, **8** (2021), 1627–1643. https://doi.org/10.1109/JAS.2021.1004129
4. M. N. Omidvar, X. D. Li, X. Yao, A review of population-based metaheuristics for large-scale black-box global optimization-Part I, *IEEE Trans. Evol. Comput.*, **26** (2022), 802–822. https://doi.org/10.1109/TEVC.2021.3130838
5. H. David, G. William, No free lunch theorems for search, *Technical Report*, **122** (1995), 431–434.
6. H. J. Yu, Y. H. Wang, H. M. Jia, L. Abualigah, Modified prairie dog optimization algorithm for global optimization and constrained engineering problems, *Math. Biosci. Eng.*, **20** (2023), 19086–19132. https://doi.org/10.3934/mbe.2023844
7. C. Ye, W. T. Wang, S. P. Zhang, P. Shao, Optimizing 3D UAV path planning: A multi-strategy enhanced beluga whale optimizer, *Lect. Notes Artif. Intell.*, **14448** (2024), 42–54.
8. W. Y. Du, J. Ma, W. J. Yin, Orderly charging strategy of electric vehicle based on improved PSO algorithm, *Energy*, **271** (2022), 127088. https://doi.org/10.1016/j.energy.2023.127088
9. D. Tansui, A. Thammano, Hybrid nature-inspired optimization algorithm: hydrozoan and sea turtle foraging algorithms for solving continuous optimization problems, *IEEE Access*, **8** (2020), 65780–65800. https://doi.org/10.1109/ACCESS.2020.2984023
10. C. L. Zhang, S. F. Ding, A stochastic configuration network based on chaotic sparrow search algorithm, *Knowledge-Based Syst.*, **220** (2021), 106924. https://doi.org/10.1016/j.knosys.2021.106924
11. J. O. Agushaka, A. E. Ezugwu, Initialisation approaches for population-based metaheuristic algorithms: A comprehensive review, *Appl. Sci.*, **12** (2022), 896. https://doi.org/10.3390/app12020896
12. Z. M. Gao, J. Zhao, Y. J. Zhang, Review of chaotic mapping enabled nature-inspired algorithms, *Math. Biosci. Eng.*, **19** (2022), 8215–8258. https://doi.org/10.3934/mbe.2022383
13. G. Atali, L. Pehlvan, B. Grevn, H. L. Seker, Chaos in metaheuristic based artificial intelligence algorithms: A short review, *Turk. J. Electr. Eng. Comput. Sci.*, **29** (2021), 1354–1367. https://doi.org/10.3906/elk-2102-5

14. S. Ahmad, M. Sulaiman, P. Kumam, Z. Hussain, M. A. Jan, W. K. Mashwani, et al., A novel population initialization strategy for accelerating Levy flights based multi-verse optimizer, *J. Intell. Fuzzy Syst.*, **39** (2020), 1–17. https://doi.org/10.3233/JIFS-190112

15. W. A. Hussein, S. Sahran, S. N. H. S. Abdullah, Patch-Levy-based initialization algorithm for Bees Algorithm, *Appl. Soft Comput.*, **23** (2014), 104–121. https://doi.org/10.1016/j.asoc.2014.06.004

16. L. P. Chen, J. H. Gao, A. M. Lopes, Z. Q. Zhang, Z. B. Chu, R. C. Wu, Adaptive fractional-order genetic-particle swarm optimization Otsu algorithm for image segmentation, *Appl. Intell.*, **53** (2023), 26949–26966.

17. W. C. Huang, G. G. Zhang, Bearing fault-detection method based on improved grey wolf algorithm to optimize parameters of multistable stochastic resonance, *Sensors*, **23** (2023), 6529. https://doi.org/10.3390/s23146529

18. M. L. Zhao, H. A. Zhao, M. Zhao, Particle swarm optimization algorithm with adaptive two-population strategy, *IEEE Access*, **11** (2023), 62242–62260. https://doi.org/10.1109/ACCESS.2023.3287859

19. Y. Chun, X. Hua, Improved sine cosine algorithm for optimization problems based on self-adaptive weight and social strategy, *IEEE Access*, **11** (2023), 73053–73061. https://doi.org/10.1109/ACCESS.2023.3294993

20. K. Y. Zhong, Q. F. Luo, Y. Q. Zhou, M. Jiang, TLMPA: Teaching-learning-based marine predators algorithm, *AIMS Math.*, **6** (2021), 1395–1442. https://doi.org/10.3934/math.2021087

21. J. Wu, R. J. Nan, L. Chen, Improved salp swarm algorithm based on weight factor and adaptive mutation, *J. Exp. Theor. Artif. Intell.*, **31** (2019), 493–515. https://doi.org/10.1080/0952813X.2019.1572659

22. M. Dehghani, P. Trojovsky, Osprey optimization algorithm: A new bio-inspired metaheuristic algorithm for solving engineering optimization problems, *Front. Mech. Eng.*, **8** (2023), 1126450. https://doi.org/10.3389/fmech.2022.1126450

23. S. B .Aydemir, A novel arithmetic optimization algorithm based on chaotic maps for global optimization, *Evol. Intell.*, **16** (2022), 981–996. https://doi.org/10.1007/s12065-022-00711-4

24. T. Y. Wu, H. N. Li, S. C. Chu, CPPE: An improved phasmatodea population evolution algorithm with chaotic maps, *Mathmatics*, **11** (2023), 1977. https://doi.org/10.3390/math11091977

25. M. Jamil, X. S. Yang, H. J. Zepernick, Test functions for global optimization: A comprehensive survey, *Swarm intell. Bio-Inspired Comput.*, (2013), 193–222. https://doi.org/10.1016/B978-0-12-405163-8.00008-9

26. J. C. Bansal, P. K. Singh, N. R. Pal, Particle swarm optimization, *Evol. Swarm Intell. Algorithms*, (2019), 11–23.

27. S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, *Adv. Eng. Software*, **69** (2014), 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007

28. L. Abualigah, D. Yousri, M. A. Elaziz, A. A. Ewees, M. A. A. Al-qaness, A. H. Gandomi, Aquila optimizer: A novel meta-heuristic optimization algorithm, *Comput. Ind. Eng.*, **157** (2021), 107250. https://doi.org/10.1016/j.cie.2021.107250

29. S. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Software*, **95** (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

30. N. Chopra, M. M. Ansari, Golden jackal optimization: A novel nature-inspired optimizer for engineering applications, *Expert Syst. Appl.*, **198** (2022), 116934. https://doi.org/10.1016/j.eswa.2022.116934

31. J. K. Xue, B. Shen, Dung beetle optimizer: A new meta-heuristic algorithm for global optimization, *J. Supercomput.*, **79** (2023), 7305–7336. https://doi.org/10.1007/s11227-022-04959-6

32. J. Derrac, S. Garcia, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, *Swarm Evol. Comput.*, **1** (2011), 3–18. https://doi.org/10.1016/j.swevo.2011.02.002

33. Y. Zhang, P. T. Liu, Research on reactive power optimization based on hybrid osprey optimization algorithm, *Energies*, **16** (2023), 7101. https://doi.org/10.3390/en16207101

34. M. H. Nadimi-Shahraki, S. Taghian, S. Mirjalili, An improved grey wolf optimizer for solving engineering problems, *Expert Syst. Appl.*, **166** (2021), 113917. https://doi.org/10.1016/j.eswa.2020.113917

35. S. R. Khuntia, J. L. Rueda, M. A. M. M. van der Meijden, Forecasting the load of electrical power systems in mid- and long-term horizons: A review, *IET Gener. Transm. Distrib.*, **10** (2016), 3971–3977. https://doi.org/10.1049/iet-gtd.2016.0340

36. M. Abumohsen, A. Y. Owda, M. Owda, Electrical load forecasting using LSTM, GRU, and RNN algorithms, *Energies*, **16** (2023), 2283. https://doi.org/10.3390/en16052283

37. Y. Yu, X. S. Si, C. H. Hu, J. X. Zhang, A review of recurrent neural networks: LSTM cells and network architectures, *Neural Comput.*, **31** (2019), 1235–1270.

38. S. L. Wang, Y. C. Fan, S. Y. Jin, P. Takyi-Aninakwa, C. Fernandez, Improved anti-noise adaptive long short-term memory neural network modeling for the robust remaining useful life prediction of lithium-ion batteries, *Reliab. Eng. Syst. Saf.*, **230** (2022), 108920. ttps://doi.org/10.1016/j.ress.2022.108920

39. S. L. Wang, F. Wu, P. Takyi-Aninakwa, C. Fernandez, D. Stroe, Q. Huang, Improved singular filtering-Gaussian process regression-long short-term memory model for whole-life-cycle remaining capacity estimation of lithium-ion batteries adaptive to fast aging and multi-current variations, *Energy*, **284** (2023), 128677. https://doi.org/10.1016/j.energy.2023.128677

40. Y. S. Sun, Y. T. Cheng, T. Liu, Q. Huang, J. N. Guo, W. L. Jin, Research on signal detection of OFDM systems based on the LSTM network optimized by the improved chameleon swarm algorithm, *Mathmatics*, **11** (2023), 1989. https://doi.org/10.3390/math11091989

41. N. Bacanin, L. Jovanovic, M. Zivkovic, V. Kandasamy, M. Antonijevic, M. Deveci, et al., Multivariate energy forecasting via metaheuristic tuned long-short term memory and gated recurrent unit neural networks, *Inf. Sci.*, **642** (2023), 119122. https://doi.org/10.1016/j.ins.2023.119122

42. A. Tzanetos, M. Blondin, A qualitative systematic review of metaheuristics applied to tension/compression spring design problem: Current situation, recommendations, and research direction, *Eng. Appl. Artif. Intell.*, **118** (2022), 105521. https://doi.org/10.1016/j.engappai.2022.105521

43. L. Abualigah, M. A. Elaziz, A. Khasawneh, M. Alshinwan, R. Ibrahim, M. A. A. Al-qaness, et al., Meta-heuristic optimization algorithms for solving real-world mechanical engineering design problems: a comprehensive survey, applications, comparative analysis, and results, *Neural Comput. Appl.*, **34** (2022), 4081–4110. https://doi.org/10.1007/s00521-021-06747-4

44. A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. L. Chen, Harris hawks optimization: Algorithm and applications, *Future Gener. Comput. Syst.*, **97** (2019), 849–872. https://doi.org/10.1016/j.future.2019.02.028

45. S. Kaur, L. K. Awasthi, A. L. Sangal, G. Dhiman, Tunicate swarm algorithm: A new bio-inspired based metaheuristic paradigm for global optimization, *Eng. Appl. Artif. Intell.*, **90** (2020), 103541. https://doi.org/10.1016/j.engappai.2020.103541

46. S. M. Li, H. L. Chen, M. J. Wang, A. A. Heidari, S. Mirjalili, Slime mould algorithm: A new method for stochastic optimization, *Future Gener. Comput. Syst.*, **111** (2020), 300–323. https://doi.org/10.1016/j.future.2020.03.055

47. P. Trojovsky, M. Dehghani, Subtraction-average-based optimizer: A new swarm-inspired metaheuristic algorithm for solving optimization problems, *Biomimetics*, **8** (2023), 149. https://doi.org/10.3390/biomimetics8020149

AIMS Press