Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2023.026019

ARTICLE

Enhanced Harris Hawks Optimization Integrated with Coot Bird Optimization for Solving Continuous Numerical Optimization Problems

Hao Cui, Yanling Guo*, Yaning Xiao, Yangwei Wang*, Jian Li, Yapeng Zhang and Haoyu Zhang

College of Mechanical and Electrical Engineering, Northeast Forestry University, Harbin, 150040, China

*Corresponding Authors: Yanling Guo. Email: nefugyl@hotmail.com; Yangwei Wang. Email: wang.yangwei@nefu.edu.cn

Received: 10 August 2022 Accepted: 30 January 2023 Published: 28 June 2023

ABSTRACT

Harris Hawks Optimization (HHO) is a novel meta-heuristic algorithm that imitates the predation characteristics of Harris Hawk and combines Lévy flight to solve complex multidimensional problems. Nevertheless, the basic HHO algorithm still has certain limitations, including the tendency to fall into the local optima and poor convergence accuracy. Coot Bird Optimization (CBO) is another new swarm-based optimization algorithm. CBO originates from the regular and irregular motion of a bird called Coot on the water's surface. Although the framework of CBO is slightly complicated, it has outstanding exploration potential and excellent capability to avoid falling into local optimal solutions. This paper proposes a novel enhanced hybrid algorithm based on the basic HHO and CBO named Enhanced Harris Hawks Optimization Integrated with Coot Bird Optimization (EHHOCBO). EHHOCBO can provide higher-quality solutions for numerical optimization problems. It first embeds the leadership mechanism of CBO into the population initialization process of HHO. This way can take full advantage of the valuable solution information to provide a good foundation for the global search of the hybrid algorithm. Secondly, the Ensemble Mutation Strategy (EMS) is introduced to generate the mutant candidate positions for consideration, further improving the hybrid algorithm's exploration trend and population diversity. To further reduce the likelihood of falling into the local optima and speed up the convergence, Refracted Opposition-Based Learning (ROBL) is adopted to update the current optimal solution in the swarm. Using 23 classical benchmark functions and the IEEE CEC2017 test suite, the performance of the proposed EHHOCBO is comprehensively evaluated and compared with eight other basic meta-heuristic algorithms and six improved variants. Experimental results show that EHHOCBO can achieve better solution accuracy, faster convergence speed, and a more robust ability to jump out of local optima than other advanced optimizers in most test cases. Finally, EHHOCBO is applied to address four engineering design problems. Our findings indicate that the proposed method also provides satisfactory performance regarding the convergence accuracy of the optimal global solution.

KEYWORDS

Harris hawks optimization; coot bird optimization; hybrid; ensemble mutation strategy; refracted opposition-based learning

1 Introduction

The optimization process refers to determining the best solution for a given problem. Optimization problems are widespread in the contemporary military, engineering, and management fields. Optimization methods have become an essential computational tool for technicians in different specialties [1–4]. With the evolution of human society and technology, more complex optimization problems are emerging. However, mathematical-programming optimization algorithms cannot effectively solve these problems, such as the Gradient Method and Newton's Method. These conventional methods need help to locate the optimal global solution in large-scale, high-dimensional, and sub-optimal search domains [4–6]. To accomplish more powerful optimization tools, in recent decades many scholars have embarked on the research of meta-heuristic algorithms (MAs) [7]. MAs are a combination of stochastic algorithms and local search algorithms. The design of MAs is mainly inspired by a series of random phenomena in nature [8,9]. For MAs, they are concept-simple, require few parameter settings, and do not have any special requirements for the objective function. Consequently, MAs are not limited to specific problems and can be used in many applications [10,11]. MAs usually includes four categories [12–14]: evolutionary-based algorithms, physics-based algorithms, swarm-based algorithms (social behaviors of organisms in nature), and human-based algorithms. Some famous algorithms are Genetic Algorithm (GA) [15], Differential Evolution (DE) [16], Evolution Strategy (ES) [17], Simulated Annealing (SA) [18], Multi-Verse Optimizer (MVO) [19], Atom Search Optimization (ASO) [20], Arithmetic Optimization Algorithm (AOA) [21], Particle Swarm Optimization (PSO) [22], Sine Cosine Algorithm (SCA) [23], Sooty Tern Optimization Algorithm (STOA) [24], Chimp Optimization Algorithm (ChOA) [25], Aquila Optimizer (AO) [26], Dragonfly Algorithm (DA) [27], Butterfly Optimization Algorithm (BOA) [28], Slime Mould Algorithm (SMA) [29], Whale Optimization Algorithm (WOA) [30], Grey Wolf Optimizer (GWO) [31], Seagull Optimization Algorithm (SOA) [32], Gorilla Troops Optimizer (GTO) [33], Search Group Algorithm (SGA) [34], etc. Although hundreds of MAs have been proposed since the Artificial Intelligence (AI) age, the No Free Lunch (NFL) [35] theorem states that there is no single algorithm capable of tackling all optimization problems. Hence, further innovation in MAs is indispensable. At the same time, most MAs still need help with slow convergence, difficulty getting rid of local optimization, and imbalance between exploration and exploitation stages [36]. In addition to developing new MAs, several researchers try to boost the performance of existing algorithms by incorporating various search operators and achieving promising results. Nguyen et al. [13] proposed an improved Slime Mould Algorithm and applied it to control stepped hydropower plants. Debnath et al. [37] hybridized the Dragonfly Algorithm with Differential Evolution to better solve the problem of maximizing secondary user throughput in energy-harvesting cognitive radio networks. Ziyu et al. [38] proposed an improved Particle Swarm Optimization algorithm named TACPSO by introducing acceleration factors and random speeds. Li et al. [39] combined Differential Evolution and Hunger Games Search into a new DECEHGS algorithm, validated on the CEC2014 test suite, CEC2017 test suite, and four engineering problems. Jia et al. [40] constructed an improved Condor Search Algorithm based on the Lévy flight and simulated annealing mechanisms. All the above-improved variants are proven to outperform the basic algorithm to some extent, which indicates that it is feasible to enhance the performance of algorithms by introducing some additional search strategies.

The Harris Hawks Optimization (HHO) simulates the cooperative behavior and chasing style of Harris hawks in nature [41]. HHO is easy to implement with few parameters and performs well in solving many optimization problems. However, the basic HHO algorithm has several limitations, including the tendency to fall into the local optima and poor convergence accuracy. Its insufficient exploration phase causes this. Research on the improvement of HHO has been in full swing. For

example, Fan et al. [42] proposed a quasi-reflective Harris Hawks Algorithm (QRHHO), enhancing the native algorithm's exploration capability and convergence speed. Dokeroglu et al. [43] presented an island parallel HHO (IP-HHO) version of the algorithm for optimizing continuous multi-dimensional problems. Wang et al. [44] hybridized the exploration phase of Aquila Optimizer and the exploitation phase of Harris Hawks Optimization to preserve the powerful search abilities of each in both algorithms. They presented a novel improved optimizer, namely IHAOHHO. Another algorithm discussed in this paper is Coot Bird Optimization (CBO), proposed by Naruei et al. [45] in 2021. The CBO algorithm mimics four movement patterns of coot birds on the water surface, including random movement, chain movement, location adjustment according to the group leaders, and leader movement. In this algorithm, the Coot leaders continuously lead the population toward the goal. The leaders sometimes have to leave their current positions to find better target areas, which gives the algorithm have strong global ability to explore different parts of the search space. But the overall framework of CBO is relatively complex compared to other algorithms, and its exploitation stage needs to be revised, often leading to premature convergence. Considering this, mixing HHO and CBO can compensate for the lack of HHO exploration capability and thus improve its global search performance. The hybrid algorithm can effectively use the solution information in the search space to maintain population diversity later in the search. Currently, some improved variants of the CBO-based algorithm have achieved promising performance. Huang et al. [46] developed a COOTCLCO algorithm that provided satisfactory solutions for solving the uneven distribution and low coverage problems of randomly deployed Wireless Sensor Network (WSN) nodes.

Motivated by the NFL theory and fully considering the characteristics of the basic HHO and CBO algorithms: HHO has good exploitation properties, but its exploration stage is insufficient, and conversely, CBO owns excellent exploration capability benefiting from the leader mechanism, but the algorithm itself is complex and lacks exploitation ability. Therefore, this paper attempts to combine the basic HHO and CBO algorithms by taking advantage of each and proposes a novel hybrid meta-heuristic algorithm with better all-around search performance for global optimization, called EHHOCBO. First, integrate the leadership mechanism of CBO into the population initialization stage of HHO. In the leader movement, the optimal individual guides the whole population to explore all regions of the search space as much as possible in a circular envelope so that the algorithm has good randomness and rich population diversity. It can make up for the lack of helpful information exchange between the optimal individual and other individuals in the swarm, thus laying a good foundation for the global search of HHO. Then, during the iterative computation, Ensemble Mutation Strategy (EMS) [47] is introduced to generate the variant candidate positions of the current individual further to enhance the algorithm exploration capability and convergence accuracy. Moreover, to extend the search range and avoid the algorithm from falling into the local optima in the later search phase, Refracted Opposition-Based Learning (ROBL) [48] is used to evaluate the fitness value of the inverse position of the current optimal solution and perform the update. To assess the performance of the proposed EHHOCBO, we used 23 classical benchmark functions and the IEEE CEC2017 test suite for testing and applied the proposed algorithm to address four engineering problems. Experimental results demonstrate that EHHOCBO performs better than competitors concerning solution accuracy, convergence speed, stability, and local optima avoidance.

This paper is organized as follows: Section 2 briefly introduces the basic HHO and CBO algorithm and two improved search strategies. In Section 3, the proposed hybrid EHHOCBO optimizer is described in detail. In Section 4, we evaluate the performance of EHHOCBO through a series of simulation experiments and analyze the results obtained. Based on this, the proposed method addresses

four real-world engineering design problems to highlight its practicality in Section 5. Finally, a conclusion of this paper is provided in Section 6.

2 Related Works

2.1 Harris Hawks Optimization (HHO)

Harris Hawks Optimization (HHO) was proposed by Heidari et al. [41] in 2019, which mimics the predatory behavior of the Harris Hawk, a species of raptor living in southern Arizona, U.S.A. The HHO algorithm consists of three main components: the exploration phase, the exploitation phase, and the transition from exploration to exploitation. Fig. 1 shows the different search processes of the basic HHO algorithm.

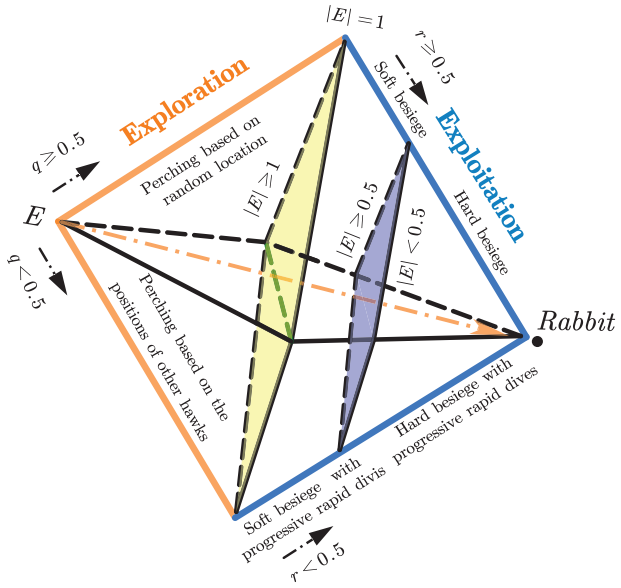

Figure 1: Different phases of HHO

2.1.1 Exploration Phase

During this phase, the Harris hawks randomly perch in certain places. They detect the prey with keen vision and then choose between two strategies with the same probability of undertaking hunting activities. The formula for the position update of Harris hawk in this phase is as follows.

$$X(t+1) = \begin{cases} X_{rand}(t) - r_1 \cdot |X_{rand}(t) - 2r_2 X(t)|, q \geq 0.5 \\ \left(X_{prey}(t) - X_m(t)\right) - r_3 \left(LB + r_4 (UB - LB)\right), q < 0.5 \end{cases} \quad (1)$$

where $X(t)$ and $X(t+1)$ represent the position vector of the Harris hawk in the current and next iterations. t represents the current number of iterations. X_{prey} is the prey position, which is also regarded as the optimal solution. $X_{rand}(t)$ is the position vector of the random individual in the current population. r_1, r_2, r_3, r_4, and q are random numbers between [0, 1]. LB and UB are the lower and upper bounds of variables. $X_m(t)$ represents the average position of all hawks in the population, which is calculated as follows:

$$X_m(t) = \frac{1}{N}\sum_{i=1}^{N} X_i(t) \tag{2}$$

where N is the population size. $X_i(t)$ is the current position vector of the i-th hawk.

2.1.2 Transition from Exploration to Exploitation

In the HHO algorithm, the prey's escape energy E drives the algorithm to switch between the global exploration and local exploitation phases. The prey's energy decreases gradually throughout the escaping process, which can be simulated as in Eq. (3).

$$E = 2E_0\left(\frac{T-t}{T}\right) \tag{3}$$

where E_0 is a random number between $[-1, 1]$, representing the initial state of the escaping energy of the prey. T is the maximum number of iterations. When $|E| \geq 1$, the hawk will continue searching for the location of prey in the target area, defined as the exploration phase. In the case of $|E| < 1$, the hawk will start to hunt the prey found in the earlier step and enter the exploitation stage.

2.1.3 Exploitation Phase

Four possible strategies exist in the exploitation phase, including soft besiege, hard besiege, soft besiege with progressive rapid dives, and hard besiege with progressive rapid dives, to simulate the attack process of Harris hawk on its prey. r represents the probability of whether the prey can escape the danger before the Harris hawk attack, which is a random number between [0,1]. If $r < 0.5$ means that the prey has successfully run through the the dangerous situation, $r \geq 0.5$ denotes the case of unsuccessful escape. Different combinations of r-value and escape energy E correspond to different predation strategies. When $|E| < 0.5$, a hard siege is performed. Otherwise, a soft siege is conducted.

- *Soft Besiege*

The soft besiege is performed when $r \geq 0.5$ and $|E| \geq 0.5$. At this stage, the position of Harris hawk is updated as follows:

$$X(t+1) = \Delta X(t) - E\left|JX_{prey}(t) - X(t)\right| \tag{4}$$

$$\Delta X(t) = X_{prey}(t) - X(t) \tag{5}$$

$$J = 2(1 - r_5) \tag{6}$$

where $\Delta X(t)$ denotes the distance between the hawk's position and the prey. r_5 is the random number between [0, 1]. J indicates the random jump intensity of the prey.

- *Hard Besiege*

The hawk will take a hard besiege when $r \geq 0.5$ and $|E| < 0.5$. The mathematical formula for this behavior is as follows:

$$X(t+1) = X_{prey}(t) - E|\Delta X(t)| \tag{7}$$

- *Soft Besiege with Progressive Rapid Dives*

When $r < 0.5$ and $|E| \geq 0.5$, the hawk will take a soft besiege with progressive rapid dives. Lévy flight is integrated into the HHO algorithm, and the mathematical model of the above behavior is as follows:

$$Y = X_{prey}(t) - E\left|JX_{prey}(t) - X(t)\right| \tag{8}$$

$$Z = Y + S \times \mathrm{LF}(D) \tag{9}$$

$$X(t+1) = \begin{cases} Y, if\ F(Y) < F(X(t)) \\ Z, if\ F(Z) < F(X(t)) \end{cases} \tag{10}$$

where D is the dimension of problem. S is a random vector whose size is $1 \times D$. $F(\cdot)$ is the objective function. Only the better position between Y and Z is selected as the next position. $\mathrm{LF}(\cdot)$ is the Lévy flight function, which is calculated as follows:

$$\mathrm{LF}(x) = 0.01 \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}}, \sigma = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma(1+\beta) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}} \right)^{1/\beta} \tag{11}$$

where u and v are two random numbers between $[0, 1]$. β is a constant with a fixed value of 1.5. $\Gamma(\cdot)$ is the gamma function.

- *Hard Besiege with Progressive Rapid Dives*

When $r < 0.5$ and $|E| < 0.5$, the Harris hawk will perform a hard siege to get close to the prey and then make a surprise attack. The mathematical model of this behavior is simulated as follows:

$$Y = X_{prey}(t) - E\left|JX_{prey}(t) - X_m(t)\right| \tag{12}$$

$$Z = Y + S \times \mathrm{LF}(D) \tag{13}$$

$$X(t+1) = \begin{cases} Y, & if\ F(Y) < F(X(t)) \\ Z, & if\ F(Z) < F(X(t)) \end{cases} \tag{14}$$

where $X_m(t)$ is calculated by Eq. (2). Note that only the better position between Y and Z is selected as the next position. The pseudo-code of the basic HHO algorithm is shown in Algorithm 1.

Algorithm 1: Pseudo-Code of Harris Hawks Optimization (HHO)

1. Initialize the population size N and the maximum iterations T
2. Initialize the random position of the population $X_i(i = 1, 2, \cdots, N)$
3. **While** $t \leq T$
4. Calculate the fitness value of each hawk
5. Save X_{prey} as the best position
6. **For** each hawk X_i
7. Calculate the E by Eq. (3)
8. **If** $|E| \geq 1$
9. Update the hawk's position by Eq. (1)
10. **End If**
11. **If** $|E| < 1$

(Continued)

Algorithm 1: (Continued)

```
12.    If r ≥ 0.5&&|E| ≥ 0.5
13.      Update the hawk's position by Eq. (4)
14.    Else if r ≥ 0.5&& |E| < 0.5
15.      Update the hawk's position by Eq. (7)
16.    Else if r < 0.5&& |E| ≥ 0.5
17.      Update the hawk's position by Eq. (10)
18.    Else if r < 0.5&& |E| < 0.5
19.      Update the hawk's position by Eq. (14)
20.   End If
21.  End For
22.  t = t + 1
23. End While
```

2.2 Coot Bird Optimization(CBO)

Coot Bird Optimization (CBO) is a bio-inspired, population-based, and gradient-free optimization technique developed by Naruei et al. [45] in 2021, which mimics the collective behaviors of American Coots (a small water bird) on the water surface. In the CBO, there are four different irregular and regular movements implemented: (1) Random movement, (2) Chain movement, (3) Adjusting the position based on the group leaders, and (4) Leader movement. The mathematical model of the algorithm is presented below.

Usually, the Coots live in a group and make a chain structure to move toward the target area (food). In front of the group are a few coots, also known as group leaders, who guide the direction and take charge of the whole flock. Therefore, according to the habits of Coots, the initial population is divided into two parts: the leader Coots and the follower Coots. If N is the population size, the number of Coot leaders is calculated as a percentage of the total population equal to L, and the remaining members $(N - L)$ are considered Coot followers. It is noted that the leaders are all selected from the population randomly. Then the mentioned four movements are implemented.

2.2.1 Random Movement

In this stage, the random position Q is defined using Eq. (15). The Coot followers are moved towards this random position to explore various parts of the search domain.

$$Q = rand\,(1, D)\,.*\,(UB - LB) + LB \tag{15}$$

where D is the dimension of problem. LB and UB are the lower and upper bounds of variables. The random movement gives the algorithm better global search performance and strengthens the algorithm's capability to escape from the local optima. The Coot's new position is updated as follows:

$$X_i\,(t+1) = X_i\,(t) + A \times r_6 \times (Q - X_i\,(t)) \tag{16}$$

where $X_i(t+1)$ denotes the position of the i-th follower in the next iteration t, r_6 is a random number in the range of [0, 1], and the parameter A is calculated according to Eq. (17).

$$A = 1 - \frac{t}{T} \tag{17}$$

where t represents the number of current iterations and T is the maximum iterations.

2.2.2 Chain Movement

In the Salp Swarm Algorithm (SSA), the average position of two individuals is used to perform chain movements. The same approach is employed in CBO. The new position of the Coot follower is calculated as follows:

$$X_i(t+1) = \frac{1}{2} \times (X_{i-1}(t) + X_i(t)) \tag{18}$$

where $X_{i-1}(t)$ is the position of the $(i-1)$-th follower in the current iteration t.

2.2.3 Adjust the Position Based on the Group Leaders

Generally, the whole group is led by some of the group leaders in the front, and all the remaining coot followers need to adjust their position based on the leaders and move towards them. However, a severe issue that must be addressed is that each Coot should update its position according to which leader. Eq. (19) is designed to select the leader as follows:

$$k = 1 + (i \ MOD \ L) \tag{19}$$

where i is the index number of the current follower, L denotes the number of leaders, and k stands for the leader's index number.

The next position of the Coot follower based on the selected leader k is calculated using Eq. (20).

$$X_i(t+1) = LeaderX_k(t) + 2 \times r_7 \times \cos(2R\pi) \times (LeaderX_k(t) - X_i(t)) \tag{20}$$

where $LeaderX_k(t)$ is the position of the selected leader, r_7 is a random number within the interval $[0, 1]$, and R denotes a random number within the interval $[-1, 1]$.

2.2.4 Leader Movement

The group must be oriented towards the optimal area, so in some instances, the leaders have to leave the current optimal position to search for a better one. The formula for updating the leader position is given as follows:

$$LeaderX_i(t+1) = \begin{cases} B \times r_8 \times \cos(2R\pi) \times (gBest(t) - LeaderX_i(t)) + gBest(t), r_9 < 0.5 \\ B \times r_8 \times \cos(2R\pi) \times (gBest(t) - LeaderX_i(t)) - gBest(t), r_9 \geq 0.5 \end{cases} \tag{21}$$

In Eq. (21), $gBest$ denotes the current optimal position, r_8 and r_9 are random numbers within the interval $[0, 1]$, and R is a random number within the interval $[-1, 1]$. $B \times r_8$ generates a more significant stochastic movement to help the algorithm eliminate the local optimal solutions. And $\cos(2R\pi)$ is designed to search for the best individual with different radius to obtain a superior position. The value of B is calculated using Eq. (22).

$$B = 2 - t \times \left(\frac{1}{T}\right) \tag{22}$$

where t represents the number of current iterations, and T denotes the maximum. The pseudo-code of Coot Bird Optimization (CBO) is summarized in Algorithm 2.

Algorithm 2: Pseudo-Code of Coot Bird Optimization (CBO)

1. Initialize the population size N, the maximum iterations T, and the number of leaders L
2. Initialize the positions of all coots
3. Select the leaders from the population randomly

(Continued)

Algorithm 2: (Continued)

4. Estimate the fitness of coot followers and leaders
5. Save the best follower or leader as the global optimum ($gBest$)
6. **While** $t \leq T$
7. Calculate the parameters of A and B using Eqs. (17) and (22)
8. **If** $rand < 0.5$
9. R, r_7, and r_8 denote random vectors along the problem dimension
10. **Else**
11. R, r_7, and r_8 denote random numbers
12. **End If**
13. **For** $i = 1$ to $(N - L)$
14. Calculate the value of k using Eq. (19)
15. **If** $rand < 0.5$
16. Update the position of the follower according to Eq. (20)
17. **Else**
18. **If** $rand < 0.5 \&\& i \sim= 1$
19. Update the position of the follower according to Eq. (18)
20. **Else**
21. Update the position of the follower according to Eq. (16)
22. **End If**
23. **End If**
24. Estimate the fitness of follower $F(X)$
25. **If** the fitness of the follower $F(X)$ < the fitness of the leader(k) fitness $F(LeaderX_k)$
26. $Temp = LeaderX_k$; $LeaderX_k = X$; $X = Temp$
27. **End If**
28. **End For**
29. **For** $i = 1$ to L
30. Update the position of the leader using Eq. (21)
31. Estimate the fitness of each leader $F(LeaderX_k)$
32. **If** the fitness of the leader $F(LeaderX_k)$ < the fitness of the global optimum $F(gBest)$
33. $Temp = gBest$; $gBest = LeaderX_k$; $LeaderX_k = Temp$
34. **End If**
35. **End For**
36. $t = t + 1$
37. **End While**
38. **Return** $gBest$

2.3 Ensemble Mutation Strategy (EMS)

The set variation strategy is a new improvement mechanism proposed by Zheng et al. [8,47], which can generate diverse individuals to improve the global search capability of the hybrid algorithm. The mathematical formula of EMS is as follows:

$$V_{i1} = \begin{cases} X_{R1} + F_1 \times (X_{R2} - X_{R3}), r_{10} < C_1 \\ X_i, r_{10} \geq C_1 \end{cases} \tag{23}$$

$$V_{i2} = \begin{cases} X_{R4} + F_2 \times (X_{R5} - X_{R6}) + F_2 \times (X_{R7} - X_{R8}), r_{11} < C_2 \\ X_i, r_{11} \geq C_2 \end{cases} \tag{24}$$

$$V_{i3} = \begin{cases} X_i + F_3 \times (X_{R9} - X_i) + F_3 \times (X_{R10} - X_{R11}), r_{12} < C_3 \\ X_i, r_{12} \geq C_3 \end{cases} \tag{25}$$

where V_{i1}, V_{i2}, and V_{i3} are the newly generated mutant positions of the ith search agent. $R1 \sim R11$ are different integer exponents in the range $[1, N]$. F_1, F_2, and F_3 represent scale factors with values of 1.0, 0.8, and 1.0, respectively. $r_{10} \sim r_{12}$ are random numbers in the range of [0,1]. In addition, the parameters C_1, C_2, and C_3 are equal to 0.1, 0.2, and 0.9, which denote the crossover rate.

After the mutant candidate positions V_{i1}, V_{i2}, and V_{i3} are generated, the best position V_i with the lowest fitness value will be selected to compare with the fitness of the original post X_i, and then the better one will be saved as the new X_i to participate in the following iterative calculation. These processes can be described using Eq. (26).

$$X_i = \begin{cases} V_i, & if\ F(V_i) < F(X_i) \\ X_i, & otherwise \end{cases} \tag{26}$$

where $F(\cdot)$ is the cost function.

2.4 Refracted Opposition-Based Learning

Refracted Opposition-based Learning [48] is an improved variant of Opposition-based Learning (OBL) [49]. ROBL introduces the principle of refraction of light to generate dynamic inverse solutions based on OBL, which solves the limitation that OBL can easily fall into local optimum in later iterations. Currently, ROBL has been used to effectively improve the optimization performance of the Grey Wolf Optimizer (GWO) [31]. The schematic diagram of ROBL is shown in Fig. 2, and its mathematical model is formulated as follows:

$$X_{i,j}^* = \frac{a_j + b_j}{2} + \frac{a_j + b_j}{2z\eta} - \frac{X_{i,j}}{z\eta}, j = 1, 2, \ldots, D \tag{27}$$

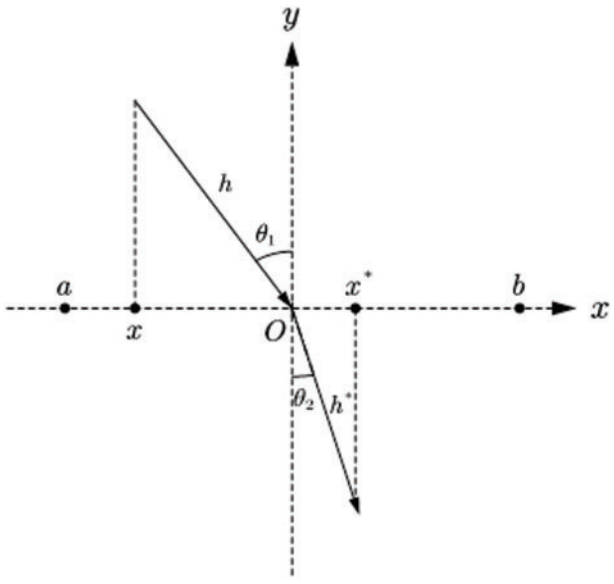

Figure 2: Refracted opposition-based learning

where $X_{i,j}$ is the location of the ith search agent in the jth dimension. $X^*_{i,j}$ is the inverse solution of $X_{i,j}$. a_j and b_j are the upper and lower limits in the jth dimension. D deontes the dimension of problem. η is the refraction index, which can be calculated using Eq. (28). z indicates the scale factor, which can be calculated using Eq. (29).

$$z = \frac{h}{h^*} \tag{28}$$

$$\eta = \frac{\sin\theta_1}{\sin\theta_2} \tag{29}$$

where h and h^* are the lengths of the incident light and the refracted light. θ_1 and θ_2 are the angle of incidence and refraction, respectively.

3 The Proposed EHHOCBO Algorithm

This section integrates the basic HHO and CBO algorithms into a new swarm intelligence algorithm called EHHOCBO. EHHOCBO takes HHO as the core framework and is hybridized with the powerful exploratory leadership mechanism of hybrid CBO. In addition, ensemble mutation strategy (EMS) and refracted opposition-based learning (ROBL) are introduced into the preliminary hybrid algorithm to enhance its search performance further.

According to Section 2, the basic HHO algorithm consists of three main steps: the exploration phase, the transition from exploration to exploitation, and the exploitation phase. In the exploration phase, Harris hawk detects prey in the search space with two equal probabilities. After detecting prey, the hawk transitions between global and local search based on the decay of prey energy. In the exploitation phase, the energy of the prey and the probability of the prey escaping determine the four different predation methods that the Harris hawk will adopt, namely soft besiege, hard besiege, soft besiege with progressive rapid dives, and soft besiege with progressive rapid dives. When the prey escapes successfully, the Lévy flight is introduced into the algorithm to ensure that the algorithm can jump out of the local optima. In the basic algorithm, the Lévy flight is only used when the position has a better fitness value, which leads to the algorithm being unable to jump out of the local optimum well. The low population diversity and simple search method in the exploration phase weaken the global search capability of HHO. This leads to a longer time to obtain the optimal global solution and increases the possibility of getting stuck in local optima. These can be improved to explore better and exploit the HHO algorithm [50]. For the CBO algorithm, the population is divided into a few randomly chosen Coot leaders and Coot followers. The leaders can make full use of the information in the search space and have a strong tendency to explore, thus effectively leading the followers to the target area. CBO mainly consists of individual random movement, chain movement, and optimal individual-guided movement. The CBO algorithm will choose a random position in the search space as a reference, which is also a an excellent way to help the algorithm jump out of the local optimum. Chain movement allows the algorithm to quickly improve the accuracy of the algorithm when the population is concentrated. In the leader movement of the optimal individual, the position of the follower will change with the position of the corresponding leader. This allows the leader to lead the followers closer and closer to the optimal region. In this paper, the optimal individual movement mechanism of CBO is integrated into the population initialization of HHO to take full advantage of the strengths of both algorithms. In the CBO optimal individual-led movement, $B \times r_8$ generates a large amount of randomness. This process can be used to initialize the population to increase the diversity of the position and make up for the lack of information exchange with other individuals,

thus laying a good foundation for global search. First, the population initialization formula Eq. (30) of the hybrid algorithm is derived from Eq. (21). After the positions of all search agents are generated by using Eq. (30), the fitness value of each new position is compared with that of the original position. The candicate position with the highest fitness value is selected as the updated position for the hybrid algorithm.

$$Xnew_i(t+1) = \begin{cases} B \times r_8 \times \cos(2R\pi) \times \left(X_{prey}(t) - X_i(t)\right) + X_{prey}(t), r_9 < 0.5 \\ B \times r_8 \times \cos(2R\pi) \times \left(X_{prey}(t) - X_i(t)\right) - X_{prey}(t), r_9 \geq 0.5 \end{cases} \tag{30}$$

where $Xnew_i(t+1)$ is the position of the ith search agent in the $(t+1)$th iteration. $X_i(t)$ denotes the position in the iteration. $X_{prey}(t)$ is the current optimal solution, and the remaining parameters are defined in Eq. (21).

Secondly, although the basic HHO algorithm introduces the Lévy flight to ensure that it can jump out of the local optimum, it still has limitations that can cause the algorithm to fall into a local optimum in later iterations. The EMS has multiple mutation operators, which can generate three differentcandidate positions according to Eqs. (23)–(25). Then, the mutant positions with the lowest fitness is selected for comparison with the fitness of the original positions. The position with the highest fitness is selected as the new position. In this way, the global search ability of the algorithm can be improved. At the end of the algorithm, ROBL is integrated into the HHO. The inverse solution of the current optimum is generated by Eq. (27). This can expand the search range, improve the convergence speed, and reduce the possibility of falling into the local optima. The introduction of EMS and ROBL strategies makes up for the shortcomings of the basic HHO algorithm in the exploitation stage. The flowchart and pseudo-code of the proposed EHHOCBO are shown in Fig. 3 and Algorithm 3, respectively.

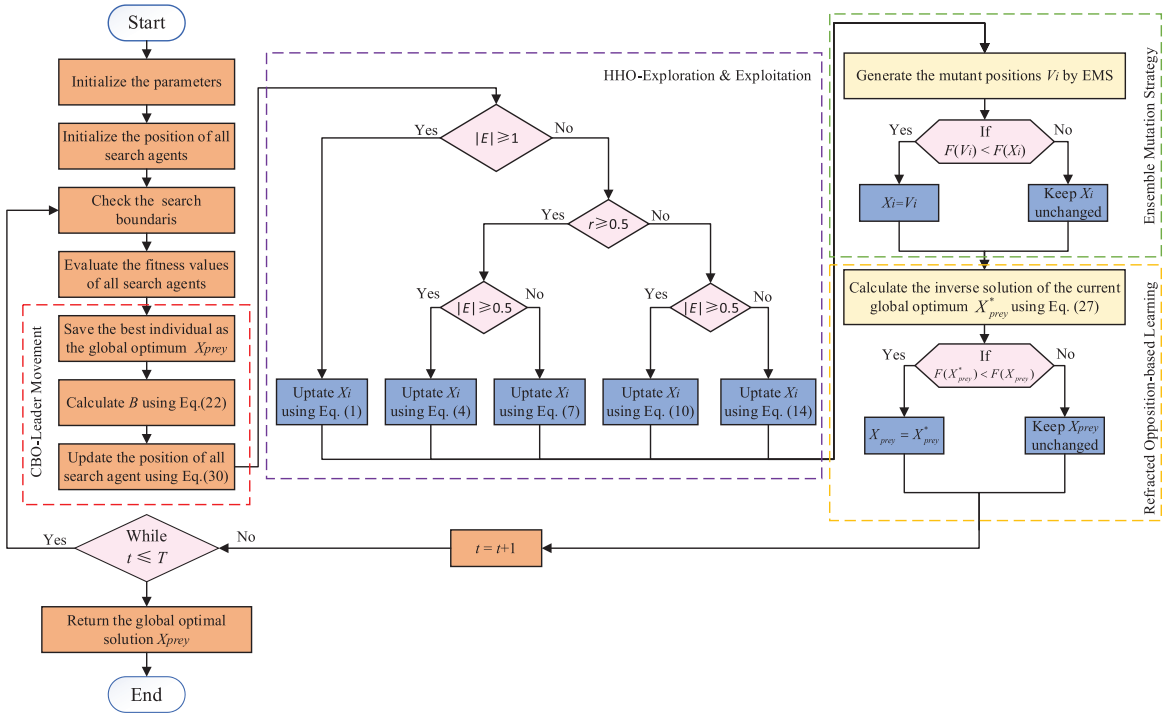

Figure 3: Flow chart of the proposed EHHOCBO algorithm

Algorithm 3: Pseudo-Code of the Proposed EHHOCBO

Initialize the population size N and the maximum iterations T
Initialize the position of each search agent $X_i(i = 1, 2, \cdots, N)$
While $t \leq T$
  Calculate the fitness values of each hawk
  Set X_{prey} as the position of the current optimal solution
  Calculate the value of B by Eq. (22)
  For each hawk X_i
    Update the position of Hawks by Eq. (30)
    If $F(Xnew_i) < F(X_i)$
      $X_i = Xnew_i$
      If the fitness of the Hawks $F(X_i)$ is better than the fitness of the global optimum $F(X_{prey})$
        $X_{prey} = X_i; bestF = F(X_i)$
      End If
  End For
  Calculate E by Eq. (3)
  For each hawk X_i
    If $|E| \geq 1$
      Update the hawk's position by Eq. (1)
    End If
    If $|E| < 1$
      If $r \geq 0.5 \&\& |E| \geq 0.5$
        Update the hawk's position by Eq. (4)
      Else if $r \geq 0.5 \&\& |E| < 0.5$
        Update the hawk's position by Eq. (7)
      Else if $r < 0.5 \&\& |E| \geq 0.5$
        Update the hawk's position by Eq. (10)
      Else if $r < 0.5 \&\& |E| < 0.5$
        Update the hawk's position by Eq. (14)
      End If
    End If
    Calculate the new mutant locations V_{i1}, V_{i2}, V_{i3} by Eqs. (23)–(25)
    Set V_i as the best trial vector with the lowest fitness from V_{i1}, V_{i2}, V_{i3}
    If $F(V_i) < F(X_i)$
      $X_i = V_i$
    End If
  End For
  Implement ROBLto calculate the inverse solution of the global optimum X^*_{prey}
  If the fitness of the inverse solution $F\left(X^*_{prey}\right)$ is better than the fitness of the global optimum $f(X_{prey})$
    $X_{prey} = X^*_{prey}; bestF = F(X^*_{prey})$
  End If
  $t = t + 1$
End While
Return X_{prey}

4 Experimental Results and Discussions

In this section, the effectiveness and feasibility of the proposed EHHOCBO is validated on two sets of optimization problems. Firstly, the performance of the algorithm in solving simple numerical problems is verified by 23 classical benchmark functions [51]. The algorithm's performance in solving complex numerical optimization problems is evaluated using the IEEE CEC2017 benchmark functions [52]. All experiments were conducted on Windows 10 operating system with a computer hardware configuration of Intel(R) Core(TM) i5-7300HQ CPU@2.50 GHz and 8 GB RAM, and the tool used was MATLAB2020a software.

4.1 Experiment 1: Classical Benchmark Functions

In this section, 23 classical benchmark functions are used to evaluate the performance of EHHOCBO. The functions F1–F7 are unimodal and are used to assess the development capability of the algorithm. F8–F13 are multimodal, which are used to test the exploration capability of the algorithm and its ability to escape from local optimum solutions. F14–F23 are fixed-dimension multimodalities, a combination of unimodal and multimodal, used to evaluate the stability of an algorithm between exploration and exploitation [51]. The population size and maximum iterations for all algorithms were set to 30 and 500, respectively, for a fair comparison. In the proposed EHHOCBO, we set the refraction index $z = 100$ and $k = 1000$. All other parameter settings of the algorithm were kept the same as in the original paper, as shown in Table 1. All experiments should be tested independently 30 to reduce randomness. Detailed descriptions of the 23 functions are listed in Tables 2–4, and Fig. 4 visualizes the search space for the 23 benchmark functions. The following algorithms are adopted for comparison tests:

- Harris Hawks Optimization (HHO) [41]
- Coot Bird Optimization(CBO) [45]
- Arithmetic Optimization Algorithm (AOA) [21]
- Whale Optimization Algorithm (WOA) [30]
- Grey Wolf Optimizer (GWO) [31]
- Particle Swarm Optimization (PSO) [22]
- Sooty Tern Optimization Algorithm (STOA) [24]
- Chimp Optimization Algorithm (ChOA) [25]

Table 1: Parameter settings of EHHOCBO and comparison algorithms

Algorithm	Parameter setting
HHO [41]	$E_0 \in [-1, 1]$
CBO [45]	—
AOA [21]	$\alpha = 5; \mu = 0.5; Min = 0.1; Max = 1$
WOA [30]	$b = 1; a_1 = [2, 0]; a_2 = [-2, -1]$
GWO [31]	$a = [2, 0]$
PSO [22]	$c_1 = 2; c_2 = 2; \omega_{min} = 0.4; \omega_{max} = 0.9$
STOA [24]	$C_f = 2; C_B = [2, 0]; u = v = 1$
ChOA [25]	$f = [2.5, 0]; m = Gauss\ chaotic$

(Continued)

Table 1 (continued)

Algorithm	Parameter setting
EHHOCBO	$E_0 \in [-1, 1]$; $z = 100, \eta = 1000$; $c_1 = 0.1$; $c_2 = 0.2$; $c_3 = 0.9$; $F_1 = 1.0$; $F_2 = 0.8$; $F_3 = 1.0$

Table 2: Unimodal benchmark functions

Function	Dimension (D)	Range	$F_{\min}$
$F_1(x) = \sum_{i=1}^{D} x_i^2$	30	[−100,100]	0
$F_2(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	30	[−10,10]	0
$F_3(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{D} x_j \right)^2$	30	[−100,100]	0
$F_4(x) = \max_i \{ \lvert x_i \rvert, 1 \le i \le D \}$	30	[−100,100]	0
$F_5(x) = \sum_{i=1}^{D-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right]$	30	[−30,30]	0
$F_6(x) = \sum_{i=1}^{D} (\lvert x_i + 0.5 \rvert)^2$	30	[−100,100]	0
$F_7(x) = \sum_{i=1}^{D} i x_i^4 + random[0, 1)$	30	[−1.28,1.28]	0

Table 3: Multimodal benchmark functions

Function	Dimension (D)	Range	$F_{\min}$
$F_8(x) = \sum_{i=1}^{D} -x_i \sin\left(\sqrt{\lvert x_i \rvert} \right)$	30	[−500,500]	$-418.9829 \times D$
$F_9(x) = \sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	30	[−5.12,5.12]	0

(Continued)

Table 3 (continued)

Function	Dimension (D)	Range	$F_{\min}$
$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{D}x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e$	30	[−32,32]	0
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{D}x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[−600,600]	0
$F_{12}(x) = \frac{\pi}{D}\left\{10\sin(\pi y_1) + \sum_{i=1}^{D-1}(y_i-1)^2 [1+10\sin^2(\pi y_{i+1})] + (y_n-1)^2\right\} + \sum_{i=1}^{D}u(x_i,10,100,4)$, $y_i = 1+\frac{x_i+1}{4}$, $u(x_i,a,k,m) = \begin{cases} k(x_i-a)^m x_i > a \\ 0 - a < x_i < a \\ k(-x_i-a)^m x_i < -a \end{cases}$	30	[−50,50]	0
$F_{13}(x) = 0.1\left\{\sin^2(3\pi x_i) + \sum_{i=1}^{D}(x_i-1)^2 \left[1+\sin^2(3\pi x_i+1)\right] + (x_D-1)^2 \left[1+\sin^2(2\pi x_D)\right]\right\} + \sum_{i=1}^{D}u(x_i,5,100,4)$	30	[−50,50]	0

Table 4: Fix-dimension multimodal benchmark functions

Function	Dimension (D)	Range	$F_{\min}$
$F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\left(j + \sum_{i=1}^{n}(x_i - a_{ij})^6\right)^{-1}\right)^{-1}$	2	[−65,65]	0.998
$F_{15}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right]^2$	4	[−5,5]	0.00030

(Continued)

Table 4 (continued)

Function	Dimension (D)	Range	$F_{\min}$
$F_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5,5]$	-1.0316
$F_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	$[-5,5]$	0.398
$F_{18}(x) = \left[1 + (x_1 + x_2 + 1)^2 \times \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right] \times \left[30 + (2x_1 - 3x_2)^2 \times \left(18 - 32x_2 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right]$	2	$[-2,2]$	3
$F_{19}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	3	$[-1,2]$	-3.8628
$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	6	$[0,1]$	-3.32
$F_{21}(x) = -\sum_{i=1}^{5}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0,10]$	-10.1532
$F_{22}(x) = -\sum_{i=1}^{7}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0,10]$	-10.4028
$F_{23}(x) = -\sum_{i=1}^{10}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0,10]$	-10.5363

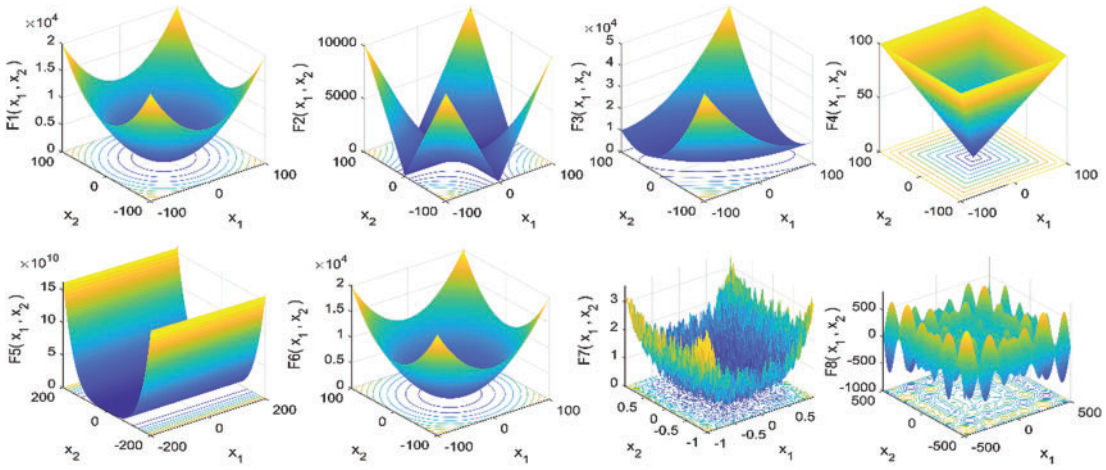

Figure 4: (Continued)

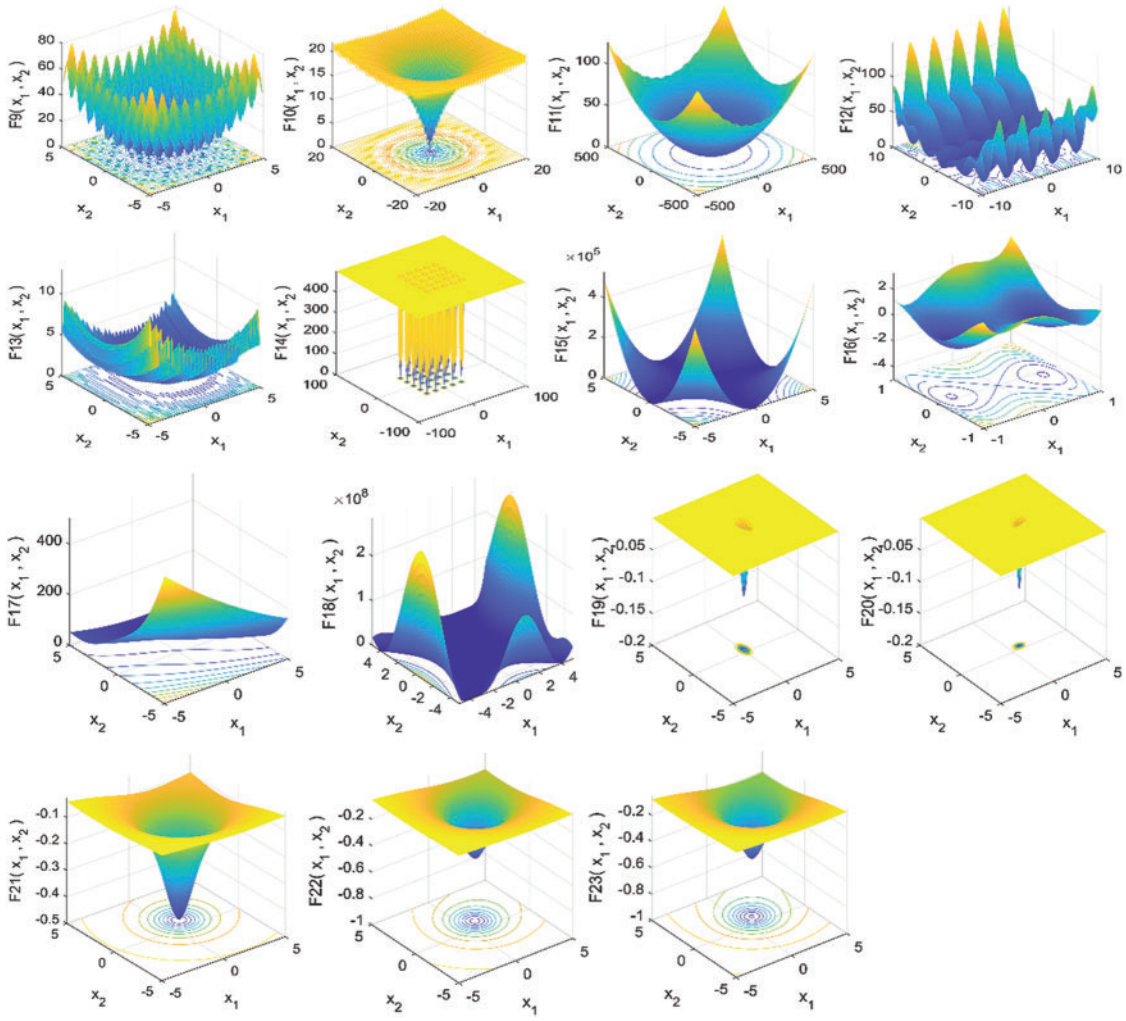

Figure 4: 3D view of the search space for 23 benchmark functions

In order to evaluate the performance of the algorithm more intuitively, this paper introduces the Mean fitness (Mean) and Standard deviation (Std) evaluation indicators. The difference between the mean fitness and the target value represents the convergence accuracy of the algorithm. The standard deviation evaluates the degree of fluctuation of the experimental results. The size of the Std depends on how stable the algorithm is. The following formula calculates them:

$$\text{Mean} = \frac{1}{times}\sum_{i=1}^{times} A_i \tag{31}$$

$$\text{Std} = \sqrt{\frac{1}{times-1}\sum_{i=1}^{times}(A_i - \text{Mean})^2} \tag{32}$$

where *times* is the number of experiments. A_i is the result for each experiment.

4.1.1 *Effectiveness of the Introduced Strategy*

This paper first integrates the original HHO and CBO algorithms into a new intelligent algorithm named EHHOCBO. The algorithm takes HHO as the core framework, introduces the strong exploratory leadership mechanism of CBO, and then improves the algorithm with EMS and ROBL strategies. To investigate the impact of each improved component on the algorithm, this paper designs a comparison experiment between EHHOCBO and three other derived algorithms (EHHOCBO1, EHHOCBO2, EHHOCBO3). EHHOCBO1 only introduced the leadership mechanism of CBO into the algorithm. EHHOCBO2 introduces the leadership mechanism of the CBO algorithm and the EMS strategy into the algorithm. EHHOCBO3 introduces the leadership mechanism of the CBO algorithm and ROBL strategy. The performance of these four algorithms is tested using 23 standard test functions under the same parameter settings, and the sizes of Mean and Std are listed in Table 5.

Table 5: Results of EHHOCBO1, EHHOCBO2, EHHOCBO3, and EHHOCBO on 23 benchmark functions

F_n	Measure	EHHOCBO	EHHOCBO1	EHHOCBO2	EHHOCBO3
F_1	Mean	**0.00E+00**	9.05E−298	4.84E−316	**0.00E+00**
	Std	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F_2	Mean	**0.00E+00**	1.15E−145	4.96E−157	**0.00E+00**
	Std	**0.00E+00**	6.28E−145	2.71E−156	**0.00E+00**
F_3	Mean	**0.00E+00**	2.11E−246	4.89E−252	**0.00E+00**
	Std	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F_4	Mean	**0.00E+00**	2.44E−150	2.07E−152	**0.00E+00**
	Std	**0.00E+00**	1.10E−149	1.13E−151	**0.00E+00**
F_5	Mean	**6.73E−05**	7.24E−04	8.09E−05	9.65E−04
	Std	**1.03E−05**	7.36E−04	8.05E−05	1.25E−03
F_6	Mean	**2.12E−09**	6.13E−08	2.14E−09	9.53E−08
	Std	**2.52E−09**	4.88E−08	2.85E−09	1.14E−07
F_7	Mean	**1.50E−04**	1.61E−04	2.04E−04	2.50E−04
	Std	**1.56E−04**	1.62E−04	2.88E−04	2.46E−04
F_8	Mean	**−1.26E+04**	−1.18E+04	−1.23E+04	−1.22E+04
	Std	**3.69E−05**	1.14E+03	6.36E+02	8.31E+02
F_9	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F_{10}	Mean	**8.88E−16**	**8.88E−16**	**8.88E−16**	**8.88E−16**
	Std	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F_{11}	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F_{12}	Mean	**3.73E−10**	1.14E−08	4.84E−10	1.36E−08
	Std	8.19E−10	2.03E−08	**7.50E−10**	1.37E−08
F_{13}	Mean	**5.32E−09**	1.31E−07	7.20E−09	7.33E−04
	Std	**6.90E−09**	2.08E−07	1.21E−08	2.79E−03
F_{14}	Mean	**9.98E−01**	1.10E+00	**9.98E−01**	1.06E+00
	Std	**2.84E−16**	4.00E−01	2.90E−16	2.52E−01

(Continued)

Table 5 (continued)

F_n	Measure	EHHOCBO	EHHOCBO1	EHHOCBO2	EHHOCBO3
F_{15}	Mean	**3.07E−04**	3.69E−04	3.38E−04	3.38E−04
	Std	**1.69E−18**	2.32E−04	1.67E−04	1.67E−04
F_{16}	Mean	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**
	Std	**4.18E−16**	5.89E−16	**4.14E−16**	**4.18E−16**
F_{17}	Mean	**3.98E−01**	**3.98E−01**	**3.98E−01**	**3.98E−01**
	Std	**3.24E−16**	1.51E−15	5.42E−16	1.49E−15
F_{18}	Mean	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**
	Std	**7.09E−14**	2.01E−11	1.01E−13	3.78E−12
F_{19}	Mean	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**
	Std	**2.59E−15**	4.50E−15	3.76E−15	3.30E−15
F_{20}	Mean	**−3.30E+00**	−3.27E+00	−3.27E+00	−3.29E+00
	Std	**5.11E−02**	6.03E−02	6.05E−02	5.35E−02
F_{21}	Mean	**−1.02E+01**	−5.23E+00	−9.98E+00	−5.31E+00
	Std	**4.13E−15**	9.31E−01	9.31E−01	1.39E+00
F_{22}	Mean	**−1.03E+01**	−5.36E+00	−1.02E+01	−6.05E+00
	Std	**1.35E−01**	1.43E+00	9.70E−01	2.16E+00
F_{23}	Mean	**−1.05E+01**	−5.40E+00	−1.04E+01	−5.40E+00
	Std	**3.86E−15**	1.48E+00	4.35E−15	1.48E+00

The experimental results show that in comparing between EHHOCBO and the remaining three derived algorithms, EHHOCBO has better simulation results on the test functions F1–F23. This verifies the effectiveness of the improved method in this paper. The theoretical minimum of the test functions was achieved for the test functions F1–F4, F8, F9, F11, F14, F16–F19, and F23. The remaining three derived algorithms also reached theoretical minima in some test functions, but the overall performance shows a different advantage than the EHHOCBO algorithm. In terms of standard deviation, the simulation results of EHHOCBO also obtained better results. This shows that EHHOCBO has strong stability in solving all test functions. The results show that the improvements in this paper significantly improve the performance of the original HHO algorithm. The improved method in this paper enables the EHHOCBO algorithm to obtain strong exploration and utilization capabilities, laying a foundation for the ability to provide high-quality solutions. After validation, EHHOCBO was selected as the final version for further comparison and discussion.

4.1.2 Comparison between EHHOCBO and Other Optimization Algorithms

This section evaluates the proposed algorithm's exploration and development capabilities using standard test functions. The above eight well-known algorithms are compared mainly from numerical analysis, box plot, convergence curve, and Wilcoxon test. After 30 independent runs of the experiment. The Mean and Std of the obtained results are recorded in Table 6 below.

Table 6: Results of 23 benchmark functions

F_n	Measure	GWO	WOA	PSO	AOA	HHO	ChOA	STOA	CBO	EHHOCBO
F_1	Mean	2.22E−27	7.92E−72	2.63E+00	9.73E−48	5.75E−98	6.49E−06	4.99E−07	1.20E−22	**0.00E+00**
	Std	5.43E−27	4.22E−71	1.15E+00	4.80E−47	2.87E−97	1.28E−05	6.71E−07	6.59E−22	**0.00E+00**
F_2	Mean	8.23E−17	4.46E−51	1.44E−49	**0.00E+00**	1.51E−50	3.39E−05	1.26E−05	2.01E−14	**0.00E+00**
	Std	6.13E−17	1.30E−50	7.53E−49	**0.00E+00**	5.19E−50	4.15E−05	2.54E−05	6.47E−14	**0.00E+00**
F_3	Mean	1.43E−05	4.25E+04	1.95E+02	4.04E−03	3.50E−72	9.74E+01	1.29E−01	3.96E−26	**0.00E+00**
	Std	3.47E−05	1.23E+04	6.70E+01	7.64E−03	1.79E−71	1.57E+02	2.52E−01	1.92E−26	**0.00E+00**
F_4	Mean	1.08E−06	5.26E+01	1.97E+00	3.02E−02	3.14E−49	2.87E−01	5.03E−02	2.10E−13	**0.00E+00**
	Std	1.26E−06	2.68E+01	2.31E−01	1.77E−02	1.47E−48	3.48E−01	8.79E−02	9.37E−13	**0.00E+00**
F_5	Mean	2.70E+01	2.80E+01	1.12E+03	2.85E+01	8.34E−03	2.89E+01	2.84E+01	4.50E+01	**6.73E−05**
	Std	8.74E−01	5.51E−01	6.36E+02	3.06E−01	1.36E−02	1.26E−01	4.05E−01	3.20E+01	**1.03E−05**
F_6	Mean	7.66E−01	4.69E−01	2.42E+00	3.27E+00	1.02E−04	3.72E+00	2.56E+00	1.96E−01	**2.12E−09**
	Std	3.53E−01	2.86E−01	1.21E+00	2.36E−01	1.16E−04	4.53E−01	4.64E−01	1.22E−01	**2.52E−09**
F_7	Mean	2.19E−03	2.54E−03	1.92E+01	9.03E−05	1.28E−04	2.09E−03	6.91E−03	5.75E−03	**1.50E−04**
	Std	1.22E−03	2.76E−03	1.04E+01	8.86E−05	1.01E−04	2.80E−03	5.71E−03	5.22E−03	**1.56E−04**
F_8	Mean	−6.02E+03	−1.08E+04	−6.22E+03	−5.43E+03	−1.24E+04	−5.74E+03	−5.91E+03	−7.40E+03	**−1.26E+04**
	Std	1.02E+03	1.73E+03	1.50E+03	4.29E+02	5.03E+02	7.42E+01	5.13E+02	7.23E+02	**3.69E−05**
F_9	Mean	1.90E+00	1.75E+00	1.65E+02	0.00E+00	0.00E+00	1.66E+01	1.12E+01	1.73E−04	**0.00E+00**
	Std	2.74E+00	7.09E+00	2.44E+01	0.00E+00	0.00E+00	2.01E+01	1.64E+01	9.48E−04	**0.00E+00**
F_{10}	Mean	1.00E−13	4.56E−15	2.62E+00	8.88E−16	8.88E−16	2.00E+01	2.00E+01	2.76E−09	**8.88E−16**
	Std	1.47E−14	2.38E−15	4.84E−01	0.00E+00	0.00E+00	1.20E−03	1.20E−03	1.40E−08	**0.00E+00**
F_{11}	Mean	4.27E−03	6.89E−03	1.22E−01	1.76E−01	0.00E+00	2.57E−02	2.59E−02	1.26E−16	**0.00E+00**
	Std	1.06E−02	2.77E−02	3.73E−02	1.34E−01	0.00E+00	6.67E−02	4.07E−02	2.75E−16	**0.00E+00**
F_{12}	Mean	4.27E−02	2.20E−02	6.26E−02	5.10E+06	1.02E−05	5.41E−01	2.41E−01	3.74E−01	**3.73E−10**
	Std	2.01E−02	1.08E−02	5.55E−02	4.63E+07	1.58E−05	2.24E−01	1.61E−01	8.27E−01	**8.19E−10**
F_{13}	Mean	6.82E−01	5.70E−01	5.65E−01	2.83E+00	9.94E−05	2.72E+00	1.89E+00	4.10E−01	**5.32E−09**
	Std	2.49E−01	2.52E−01	2.77E−01	1.02E−01	1.54E−04	1.27E−01	2.51E−01	5.26E−01	**6.90E−09**
F_{14}	Mean	4.29E+00	3.48E+00	2.88E+00	9.80E+00	1.72E+00	1.32E+00	1.23E+00	**9.98E−01**	**9.98E−01**
	Std	4.20E+00	3.66E+00	2.01E+00	4.28E+00	1.97E+00	1.78E+00	5.64E−01	6.60E−16	**2.84E−16**
F_{15}	Mean	3.14E−03	8.30E−04	8.44E−04	1.18E−02	3.74E−04	1.30E−03	1.09E−03	1.32E−03	**3.07E−04**
	Std	6.88E−03	5.59E−04	1.59E−04	1.76E−02	1.71E−04	5.13E−05	3.35E−04	3.60E−03	**1.69E−18**
F_{16}	Mean	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**
	Std	2.62E−08	3.12E−09	4.46E−16	1.24E−07	8.85E−10	1.46E−05	3.45E−06	3.23E−12	**4.18E−16**
F_{17}	Mean	**3.98E−01**	**3.98E−01**	**3.98E−01**	4.14E−01	**3.98E−01**	5.54E−01	**3.98E−01**	**3.98E−01**	**3.98E−01**
	Std	3.11E−06	3.57E−05	0.00E+00	1.54E−02	2.30E−05	8.47E−01	4.95E−04	8.84E−07	**3.24E−16**
F_{18}	Mean	**3.00E+00**	**3.00E+00**	**3.00E+00**	7.58E+00	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**
	Std	4.97E−05	8.07E−05	7.71E−15	1.02E+01	2.83E−07	2.95E−04	9.94E−05	1.16E−12	**7.09E−14**
F_{19}	Mean	**−3.86E+00**	−3.85E+00	−3.86E+00	−3.85E+00	**−3.86E+00**	−3.85E+00	−3.85E+00	**−3.86E+00**	**−3.86E+00**
	Std	1.81E−03	1.34E−02	2.92E−15	4.03E−03	5.45E−03	1.57E−03	1.16E−03	1.46E−12	**2.59E−15**
F_{20}	Mean	−3.22E+00	−3.23E+00	−3.25E+00	−3.07E+00	−3.08E+00	−2.63E+00	−2.91E+00	−3.29E+00	**−3.30E+00**
	Std	1.32E−01	9.13E−02	5.99E−02	1.06E−01	1.36E−01	4.84E−01	4.48E−01	5.35E−02	**5.11E−02**
F_{21}	Mean	−9.14E+00	−8.50E+00	−7.06E+00	−3.46E+00	−5.05E+00	−1.91E+00	−3.09E+00	−7.98E+00	**−1.02E+01**
	Std	2.06E+00	2.52E+00	3.27E+00	1.47E+00	4.62E−03	1.85E+00	3.77E+00	3.20E+00	**4.13E−15**
F_{22}	Mean	−1.01E+01	−8.05E+00	−9.46E+00	−4.28E+00	−5.42E+00	−3.61E+00	−4.71E+00	−9.49E+00	**−1.03E+01**
	Std	1.40E+00	3.20E+00	2.47E+00	1.54E+00	1.30E+00	1.97E+00	4.47E+00	2.38E+00	**1.35E−01**
F_{23}	Mean	−1.01E+01	−7.14E+00	−9.71E+00	−3.76E+00	−5.64E+00	−4.25E+00	−7.71E+00	−9.88E+00	**−1.05E+01**
	Std	1.75E+00	3.55E+00	2.60E+00	1.51E+00	1.56E+00	1.59E+00	3.71E+00	2.03E+00	**3.86E−15**

The proposed EHHOCBO experiment showed better mean fitness and standard deviation. The EHHOCBO has an overwhelming advantage in solving the unimodal test function problem. In the test functions F1–F4, the EHHOCBO can acquire the theoretically global optimal solution compared with the original HHO algorithm and CBO algorithm. In the multi-dimensional test functions

F8–F13, the EHHOCBO algorithm achieves better simulation results than the other algorithms. It is worth mentioning that in functions F9 and F11, EHHOCBO obtains the ideal optimal value. The experimental results of the unimodal and multimodal functions fully demonstrate that the EHHOCBO has a better exploitation and exploration capability and a higher possibility of jumping out of the local optimum solution. This shows that the introduced improvement strategy greatly improves the relationship between exploration and exploitation in the HHO algorithm. The EHHOCBO integrates the leadership mechanism of the CBO algorithm into the population initialization of the HHO algorithm. In this way, the stable exploration and development capabilities of the two algorithms are extracted so that the algorithm can easily jump out of the local optimum. For the fixed-dimension benchmark functions F14–F23, the EHHOCBO algorithm obtains better solutions and smaller deviations. Especially in the functions F14, F16–F19, EHHOCBO obtained the theoretically optimal Mean and Std. The smaller Std indicates that the EHHOCBO has higher stability, which can ensure that the algorithm can obtain more accurate results in the application process. The fixed dimension test function results show that the proposed EHHOCBO has good stability between exploration and exploitation due to the introduction of the EMS improvement strategy.

Boxplots are drawn in this paper to visualize the distribution of the data. It can well describe the consistency between data. Fig. 5 depicts the boxplot of EHHOCBO against eight other comparison algorithms over 15 representative benchmark functions. In this figure, serial numbers 1–9 correspond to the algorithms in the table in order, specifically 1-GWO, 2-WOA, 3-PSO, 4-AOA, 5-HHO, 6-ChOA, 7-STOA, 8-CBO, and 9-EHHOCBO. As can be seen from Fig. 5, the EHHOCBO algorithm has better consistency than other original algorithms. No outliers were generated during the iteration. The obtained median, maximum and minimum values are more concentrated than other comparison algorithms and only slightly worse than the HHO algorithm in function F6. The above confirms the high stability of EHHOCBO.

In order to study the convergence behavior of the EHHOCBO algorithm, this paper compares the convergence curve of the EHHOCBO algorithm with the convergence curves of the other eight comparison algorithms on some representative test functions. The results are shown in Fig. 6. According to the convergence curve graph, we can see that the EHHOCBO algorithm has a better convergence performance. In the test functions F1, F3, and F4, the EHHOCBO algorithm can quickly converge to the global optimum, and its convergence curve shows the fastest decay trend. At the same time, other algorithms have obvious lag and slow convergence speed. This is because the algorithm introduces an improved strategy so that the algorithm have better randomness and population diversity. Among the test functions F5 and F6, the convergence of the EHHOCBO algorithm is also the best, and its convergence curve is consistent with that of HHO at the beginning of the iteration. But in the later stage, the convergence trend of HHO could be clearer, and the EHHOCBO algorithm still maintains good convergence until the end. This shows that the introduction of the improved strategy effectively reduces the possibility of the algorithm falling into the local optimum. In the multi-dimensional test functions F8–F12, the EHHOCBO algorithm also maintains good convergence. In test functions F9–F11, EHHOCBO converges to the optimal global solution only after several iterations. In F8, the convergence of EHHOCBO is slightly worse than that of HHO in the early stage. But in the later iterations, the HHO algorithm gradually falls into the local optimum, while the EHHOCBO algorithm still maintains good convergence performance. In F10–F12, although the EHHOCBO algorithm does not reach the theoretical optimum, its convergence performance and final accuracy are the best among all algorithms. The EHHOCBO algorithm also shows strong performance in solving fixed-dimensional testing problems. EHHOCBO converges rapidly in the early stages of testing functions F17, F18, F20, F22, and F23. And make a quick transitions between exploration and exploitation.

Finally, the optimal value was determined. The convergence accuracy and operational efficiency of the EHHOCBO algorithm are also improved to some extent compared to HHO and CBO.

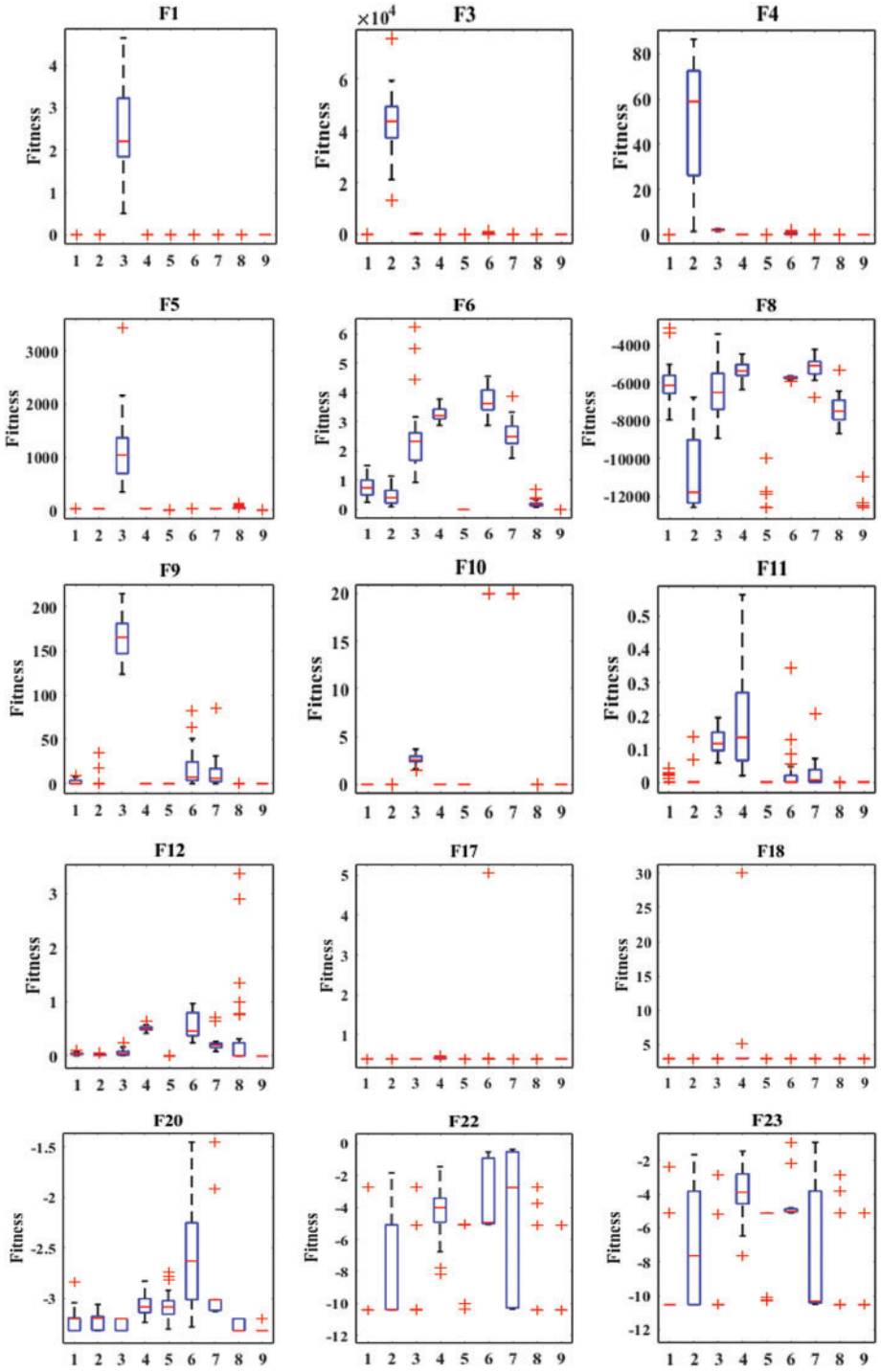

Figure 5: Boxplot analysis on 23 benchmark functions

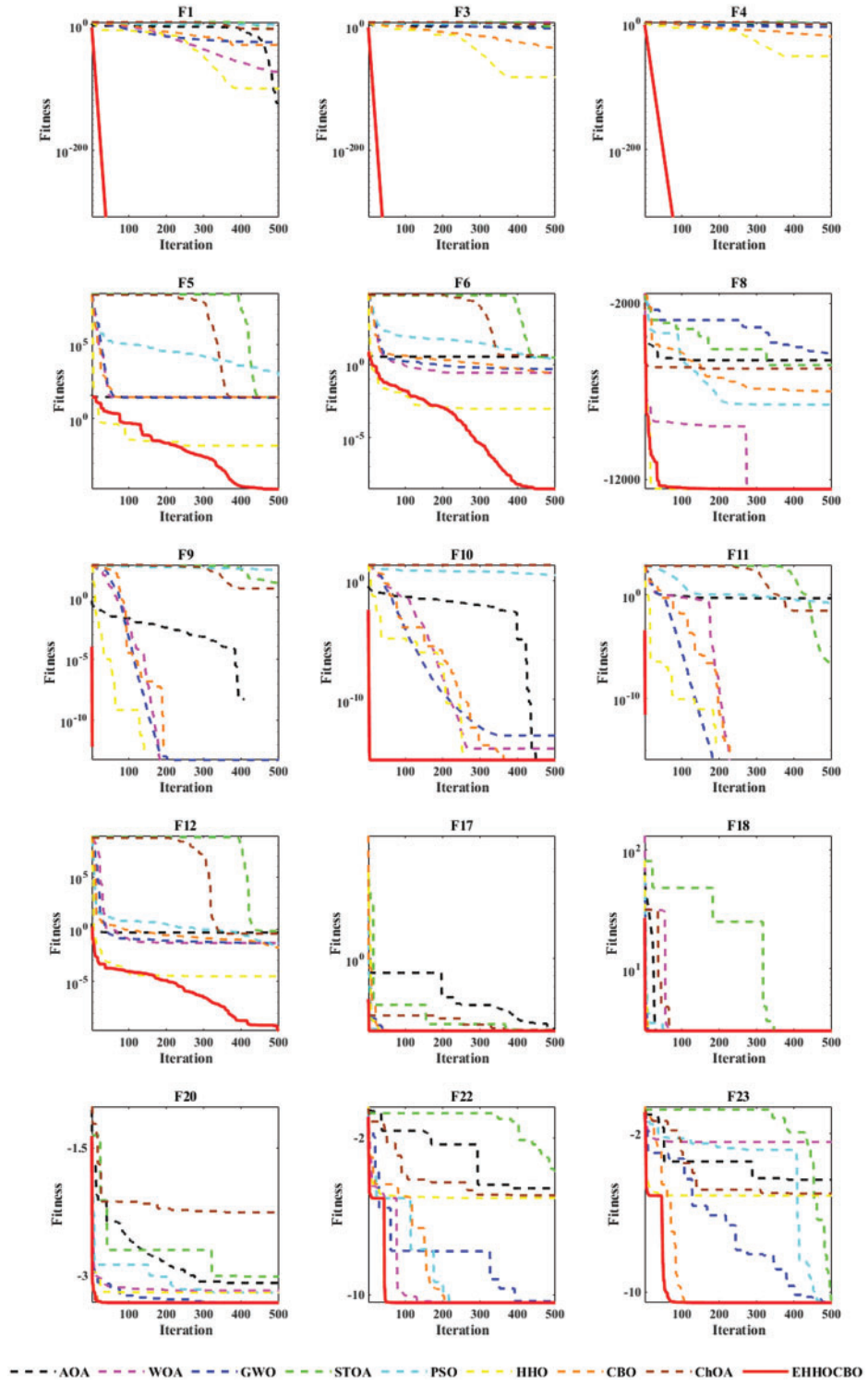

Figure 6: Convergence curves of different algorithms on fifteen benchmark functions

Table 7: Comparison results of the average computation time for different algorithms (unit: s)

F_n	GWO	WOA	PSO	AOA	HHO	ChOA	STOA	CBO	EHHOCBO
F_1	3.26E−01	**9.67E−02**	1.59E−01	1.80E−01	2.35E−01	2.78E+00	2.77E−01	1.60E−01	7.46E−01
F_2	3.61E−01	**1.07E−01**	1.52E−01	1.86E−01	2.09E−01	2.72E+00	2.57E−01	1.60E−01	7.24E−01
F_3	5.04E−01	3.34E−01	3.51E−01	4.16E−01	7.88E−01	2.68E+00	4.45E−01	4.48E−01	**2.81E−01**
F_4	4.01E−01	**1.15E−01**	1.64E−01	2.22E−01	3.08E−01	2.98E+00	2.83E−01	1.76E−01	7.67E−01
F_5	4.37E−01	1.52E−01	1.85E−01	2.85E−01	4.68E−01	2.70E+00	2.90E−01	2.30E−01	1.24E−01
F_6	3.82E−01	**1.01E−01**	1.42E−01	1.97E−01	3.13E−01	2.61E+00	2.56E−01	1.69E−01	1.92E−01
F_7	4.85E−01	**2.39E−01**	2.56E−01	3.91E−01	6.58E−01	2.82E+00	3.41E−01	3.37E−01	2.29E+00
F_8	4.49E−01	**1.78E−01**	2.21E−01	2.53E−01	5.16E−01	2.84E+00	3.25E−01	2.47E−01	1.32E+00
F_9	4.25E−01	**1.27E−01**	1.91E−01	2.21E−01	4.124E−01	2.87E+00	2.97E−01	2.37E−01	1.05E+00
F_{10}	4.14E−01	**1.39E−01**	1.92E−01	2.35E−01	4.40E−01	2.87E+00	3.36E−01	2.39E−01	1.12E+00
F_{11}	3.95E−01	**1.43E−01**	1.88E−01	2.35E−01	4.31E−01	2.38E+00	2.90E−01	2.23E−01	1.11E+00
F_{12}	4.27E−01	3.74E−01	4.20E−01	4.56E−01	**1.06E−01**	2.30E+00	4.56E−01	4.91E−01	3.33E−01
F_{13}	4.77E−01	3.83E−01	4.31E−01	4.47E−01	**1.14E−01**	2.37E+00	4.59E−01	5.07E−01	2.49E+00
F_{14}	5.52E−01	5.75E−01	5.26E−01	5.96E−01	1.55E+00	6.55E−01	**5.10E−01**	7.78E−01	1.34E+00
F_{15}	1.41E−01	9.28E−02	**8.16E−02**	1.11E−01	3.05E−01	6.74E+00	1.16E−01	1.71E−01	1.05E+00
F_{16}	1.12E−01	9.31E−02	**6.97E−02**	1.10E−01	3.27E−01	4.17E−01	9.66E−02	1.54E−01	1.04E+00
F_{17}	1.11E−01	9.49E−02	**5.42E−02**	9.47E−02	3.17E−01	4.67E+00	9.97E−02	1.58E−01	1.03E+00
F_{18}	9.94E−02	7.73E−02	**4.62E−02**	9.09E−02	2.73E−01	4.37E−01	8.69E−02	1.55E−01	9.59E−02
F_{19}	1.25E−01	**1.18E−01**	7.93E−01	1.23E−01	3.42E−01	5.34E−01	1.22E−01	1.73E−01	1.06E+00
F_{20}	1.57E−01	**1.08E−01**	9.24E−01	1.33E−01	3.54E−01	8.73E−01	1.36E−01	1.67E−01	9.56E−01
F_{21}	1.43E−01	1.20E−01	**9.94E−02**	1.30E−01	3.86E−01	6.05E−01	1.29E−01	1.84E−01	1.19E+00
F_{22}	1.67E−01	1.40E−01	**1.15E−01**	1.42E−01	4.10E−01	6.19E−01	1.49E−01	1.90E−01	1.19E+00
F_{23}	2.01E−01	**1.60E−01**	1.35E−01	1.78E−01	4.89E−01	5.76E−01	1.65E−01	2.23E−01	1.28E+00
Total	7.29E+00	4.07E+00	5.90E+00	5.43E+00	9.75E+00	5.10E+01	5.92E+00	5.98E+00	2.28E+01

Table 7 reports the average computation time for each algorithm. The total running time for each algorithm is calculated and ranked as follows: ChOA(51 s) > EHHOCBO(22.8 s) > HHO(9.75 s) > GWO(7.29 s) > CBO(5.98 s) > STOA(5.92 s) > PSO(5.90 s) > AOA(5.43 s) > WOA(4.07 s). Table 7 shows that the EHHOCBO algorithm takes more time than the original HHO algorithm. The main reason is that the improved approach adds steps and extra time to the HHO algorithm. Overall, our proposed algorithm is acceptable due to the performance improvements.

To better evaluate the correlation, the Wilcoxon rank sum test was designed, and the significance level was set at 0.05 [53]. The *p*-values are recorded in Table 8. In the table, the parts where EHHOCBO performs better are highlighted in bold. NaN indicates the same performance as the comparison algorithm. The rest is the part where EHHOCBO has poor performance. As can be seen from Table 8, EHHOCBO performs worse than some algorithms only on individual functions. EHHOCBO performs slightly worse than the PSO algorithm on F18 and slightly worse than the HHO algorithm on F7. In other cases, EHHOCBO achieved the same or better results. Based on statistical theory, EHHOCBO has better performance.

Table 8: *p*-values of the Wilcoxon rank-sum test

F_n	EHHOCBO *vs*. GWO	EHHOCBO *vs*. WOA	EHHOCBO *vs*. PSO	EHHOCBO *vs*. AOA	EHHOCBO *vs*. HHO	EHHOCBO *vs*. ChOA	EHHOCBO *vs*. STOA	EHHOCBO *vs*. CBO
F_1	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**
F_2	**1.21E−12**	**1.21E−12**	**1.21E−12**	NaN	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**
F_3	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**
F_4	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**	**1.21E−12**
F_5	**3.02E−11**	**3.02E−11**	**3.02E−11**	**3.02E−11**	**1.61E−10**	**3.02E−11**	**3.02E−11**	**3.02E−11**
F_6	**3.02E−11**	**3.02E−11**	**3.02E−11**	**3.02E−10**	**4.08E−11**	**3.02E−11**	**3.02E−11**	**3.02E−11**
F_7	**1.07E−09**	**6.52E−09**	**3.02E−11**	**3.18E−01**	3.71E−01	**1.01E−08**	**6.07E−11**	**9.92E−11**
F_8	**3.02E−11**	**3.82E−09**	**3.02E−11**	**3.02E−11**	**6.53E−07**	**3.02E−11**	**3.02E−11**	**3.02E−11**
F_9	**1.12E−12**	**1.61E−01**	**1.21E−12**	NaN	NaN	**1.21E−12**	**1.21E−12**	**1.10E−02**
F_{10}	**1.11E−12**	**9.16E−09**	**1.21E−12**	NaN	NaN	**1.21E−12**	**1.21E−12**	**5.26E−09**
F_{11}	**1.37E−03**	**3.34E−01**	**1.21E−12**	**1.21E−12**	NaN	**1.21E−12**	**1.21E−12**	**4.19E−02**
F_{12}	**3.02E−11**	**3.02E−11**	**3.02E−11**	**3.02E−11**	**3.02E−11**	**3.02E−11**	**3.02E−11**	**3.02E−11**
F_{13}	**3.02E−11**	**3.02E−11**	**3.02E−11**	**3.02E−11**	**3.34E−11**	**3.02E−11**	**3.02E−11**	**3.02E−11**
F_{14}	**2.33E−11**	**2.45E−11**	**2.89E−06**	**2.33E−11**	**2.33E−11**	**2.33E−11**	**2.33E−11**	**4.28E−02**
F_{15}	**3.00E−11**	**3.00E−11**	**3.00E−11**	**3.00E−11**	**3.00E−11**	**3.00E−11**	**3.00E−11**	**3.00E−11**
F_{16}	**1.89E−11**	**1.89E−11**	**6.33E−05**	**1.89E−11**	**1.89E−11**	**1.89E−11**	**1.89E−11**	**8.23E−06**
F_{17}	**1.21E−12**	**1.21E−12**	NaN	**1.21E−12**	**4.57E−12**	**1.21E−12**	**1.21E−12**	**1.93E−10**
F_{18}	**3.00E−11**	**3.00E−11**	8.59E−01	**3.00E−11**	**1.68E−09**	**3.00E−11**	**3.00E−11**	**4.84E−03**
F_{19}	**1.71E−11**	**1.71E−11**	**2.23E−03**	**1.71E−11**	**1.71E−11**	**1.71E−11**	**1.71E−11**	**1.71E−11**
F_{20}	**1.22E−03**	**2.53E−05**	**2.13E−03**	**8.49E−11**	**1.94E−09**	**2.47E−10**	**8.49E−11**	**5.14E−05**
F_{21}	**1.59E−11**	**1.59E−11**	**1.59E−11**	**1.59E−11**	**1.59E−11**	**1.59E−11**	**1.59E−11**	**1.59E−11**
F_{22}	**1.95E−11**	**1.95E−11**	**1.95E−11**	**1.95E−11**	**1.95E−11**	**1.95E−11**	**1.95E−11**	**1.95E−11**
F_{23}	**2.38E−11**	**2.38E−11**	**2.38E−11**	**2.38E−11**	**2.38E−11**	**2.38E−11**	**2.38E−11**	**2.38E−11**
+/=/−	23/0/0	23/0/0	21/1/1	20/3/0	19/3/1	23/0/0	23/0/0	23/0/0

In addition, the performance of some optimization algorithms gradually deteriorates as the dimension of the problem expands. In this paper, EHHOCBO and HHO algorithms are simulated in different dimensions to evaluate the impact of scalability on the EHHOCBO algorithm. The experimental tools are F1–F13 of the standard test functions, and the results are recorded in Table 9. The experimental results show that the simulation accuracy of both the original HHO and EHHOCBO decreases with the increase in the number of iterations. However, the simulation results of EHHOCBO are consistently more accurate than HHO. The accuracy of EHHOCBO will remain relatively high and will always be kept in a relatively precise state. It is worth mentioning that EHHOCBO reaches the global optimal solution in F1–F4, F9, and F11. In conclusion, the EHHOCBO algorithm also performs well in solving high-dimensional problems.

Table 9: Fitness values of HHO and EHHOCBO in different dimensions on 13 test functions

F_n	100		500		1000	
	HHO	EHHOCBO	HHO	EHHOCBO	HHO	EHHOCBO
F_1	1.13E−94	**0.00E+00**	2.91E−93	**0.00E+00**	8.12E−94	**0.00E+00**
F_2	3.53E−50	**0.00E+00**	3.09E−50	**0.00E+00**	7.31E−49	**0.00E+00**

(Continued)

Table 9 (continued)

F_n	100		500		1000	
	HHO	EHHOCBO	HHO	EHHOCBO	HHO	EHHOCBO
F_3	1.30E−61	**0.00E+00**	1.08E−37	**0.00E+00**	2.25E−21	**0.00E+00**
F_4	3.35E−48	**0.00E+00**	8.23E−49	**0.00E+00**	5.52E−48	**0.00E+00**
F_5	5.35E−02	**1.52E−02**	1.84E−01	**9.26E−02**	5.37E−01	**2.02E−01**
F_6	3.96E−04	**4.44E−05**	1.79E−03	**2.77E−04**	2.31E−02	**1.34E−02**
F_7	1.76E−04	**1.56E−04**	1.86E−04	**1.66E−04**	1.60E−04	**8.67E−05**
F_8	−41896.6	**−41840.5**	−209468.7	**−209476.7**	−418972.3	**−418976.7**
F_9	0.00E+00	**0.00E+00**	0.00E+00	**0.00E+00**	0.00E+00	**0.00E+00**
F_{10}	8.88E−16	**8.88E−16**	8.88E−16	**8.88E−16**	8.88E−16	**8.88E−16**
F_{11}	0.00E+00	**0.00E+00**	0.00E+00	**0.00E+00**	0.00E+00	**0.00E+00**
F_{12}	4.10E−06	**3.33E−07**	2.05E−06	**7.28E−07**	1.92E−05	**2.16E−06**
F_{13}	9.37E−05	**1.62E−05**	3.48E−03	**2.83E−03**	1.03E−02	**1.01E−03**

4.2 Experiment 2: IEEE CEC2017 Test Functions

The benchmark function test proves that EHHOCBO performs well in solving simple problems, but is insufficient to prove its superior performance. To further exploit its performance, this paper applies the EHHOCBO algorithm to solve 29 test functions in IEEE CEC2017 [50,51] to evaluate its performance in solving complex numerical problems. In this section, comparative experiments are conducted between EHHOCBO and six well-known hybrid algorithms based on IEEE CEC2017 test functions. The six algorithms are the Differential Squirrel Search Algorithm (DSSA) [54], Equilibrium Slime Mould Algorithm (ESMA) [55], Grey Wolf Optimizer Based on Aquila Exploration Method (AGWO) [56], Teaching-Learning-Based Optimization Algorithm with Reinforcement Learning Strategy (RLTLBO) [57], Leader Harris Hawks optimization (LHHO) [58] and Hybrid Sine-Cosine Harris Hawks Optimization (SCHHO) [59]. IEEE CEC2017 test functions include 29 functions, which are widely used for performance testing and evaluation of intelligent bee swarm algorithms. Among them, CEC-1 and CEC-3 are single-peaked functions, F4–F10 are simple multi-peaked functions, CEC-11 to CEC-20 are hybrid functions, and CEC-21 to CEC-30 are combined functions. Table 10 lists these functions' names, dimensions, target values, and search ranges. The proposed EHHOCBO algorithm was run independently of the other comparison algorithms 30 times. The population size and maximum iterations were set to 30 and 500. The mean and standard deviation obtained were recorded in Table 11.

Table 10: IEEE CEC2017 test function

Function	Name	Dim	Range	$F_{\min}$
	Unimodal functions			
CEC-01	Shifted and Rotated Bent Cigar Function	10	[−100,100]	100
CEC-03	Shifted and Rotated Zakharov Function	10	[−100,100]	300
	Simple multimodal functions			
CEC-04	Shifted and Rotated Rosenbrock's Function	10	[−100,100]	400

(Continued)

Table 10 (continued)

Function	Name	Dim	Range	$F_{\min}$
CEC-05	Shifted and Rotated Rastrigin's Function	10	[−100,100]	500
CEC-06	Shifted and Rotated Expanded Scaffer's F6 Function	10	[−100,100]	600
CEC-07	Shifted and Rotated Lunacek Bi_Rastrigin Function	10	[−100,100]	700
CEC-08	Shifted and Rotated Non-Continuous Rastrigin's Function	10	[−100,100]	800
CEC-09	Shifted and Rotated Levy Function	10	[−100,100]	900
CEC-10	Shifted and Rotated Schwefel's Function	10	[−100,100]	1000
	Hybrid functions			
CEC-11	Hybrid Function 1 ($N = 3$)	10	[−100,100]	1100
CEC-12	Hybrid Function 2 ($N = 3$)	10	[−100,100]	1200
CEC-13	Hybrid Function 3 ($N = 3$)	10	[−100,100]	1300
CEC-14	Hybrid Function 4 ($N = 4$)	10	[−100,100]	1400
CEC-15	Hybrid Function 5 ($N = 4$)	10	[−100,100]	1500
CEC-16	Hybrid Function 6 ($N = 4$)	10	[−100,100]	1600
CEC-17	Hybrid Function 6 ($N = 5$)	10	[−100,100]	1700
CEC-18	Hybrid Function 6 ($N = 5$)	10	[−100,100]	1800
CEC-19	Hybrid Function 6 ($N = 5$)	10	[−100,100]	1900
CEC-20	Hybrid Function 6 ($N = 6$)	10	[−100,100]	2000
	Composition functions			
CEC-21	Composition Function 1 ($N = 3$)	10	[−100,100]	2100
CEC-22	Composition Function 2 ($N = 3$)	10	[−100,100]	2200
CEC-23	Composition Function 3 ($N = 4$)	10	[−100,100]	2300
CEC-24	Composition Function 4 ($N = 4$)	10	[−100,100]	2400
CEC-25	Composition Function 5 ($N = 5$)	10	[−100,100]	2500
CEC-26	Composition Function 6 ($N = 5$)	10	[−100,100]	2600
CEC-27	Composition Function 7 ($N = 6$)	10	[−100,100]	2700
CEC-28	Composition Function 8 ($N = 6$)	10	[−100,100]	2800
CEC-29	Composition Function 9 ($N = 3$)	10	[−100,100]	2900
CEC-30	Composition Function 10 ($N = 3$)	10	[−100,100]	3000

Table 11: Results of IEEE CEC2017 test function

F_n	Measure	DSSA	ESMA	AGWO	RLTLBO	SCHHO	LHHO	EHHOCBO
CEC-1	Mean	8.60E+09	6.98E+03	2.39E+08	**2.52E+03**	7.72E+03	3.66E+09	2.96E+03
	Std	1.78E+09	4.51E+03	1.41E+08	2.81E+03	3.31E+01	2.09E+09	**2.78E+03**
CEC-3	Mean	8.53E+03	3.08E+02	1.91E+03	**3.00E+02**	5.69E+03	8.99E+03	**3.00E+02**
	Std	1.83E+03	1.94E+01	1.76E+03	2.31E−02	1.20E+03	2.84E+03	**1.73E−06**
CEC-4	Mean	9.62E+02	4.16E+02	4.28E+02	4.08E+02	4.35E+02	6.88E+02	**4.05E+02**
	Std	2.23E+02	2.73E+01	2.60E+01	1.18E+01	5.42E+01	2.29E+02	**1.44E+00**
CEC-5	Mean	5.98E+02	5.20E+02	5.39E+02	5.16E+02	**5.12E+02**	5.64E+02	5.27E+02
	Std	1.29E+01	6.81E+00	5.55E+00	**6.33E+00**	1.29E+01	1.82E+01	9.75E+00

(Continued)

Table 11 (continued)

F_n	Measure	DSSA	ESMA	AGWO	RLTLBO	SCHHO	LHHO	EHHOCBO
CEC-6	Mean	6.51E+02	**6.00E+02**	6.10E+02	6.01E+02	6.58E+02	6.37E+02	6.03E+02
	Std	8.37E+00	**3.14E−01**	2.97E+00	9.69E−01	8.66E+01	9.68E+00	9.13E−01
CEC-7	Mean	8.24E+02	7.30E+02	7.57E+02	7.35E+02	1.73E+01	7.85E+02	**7.27E+02**
	Std	1.71E+01	6.78E+00	7.28E+00	1.12E+01	8.63E+00	1.56E+01	**1.24E+00**
CEC-8	Mean	8.20E+02	8.23E+02	8.21E+02	**8.21E+02**	1.02E+03	8.35E+02	**8.21E+02**
	Std	1.09E+01	1.00E+01	1.07E+01	7.36E+01	2.03E+01	9.80E+00	**7.31E+00**
CEC-9	Mean	1.67E+03	9.19E+02	9.33E+02	**9.04E+02**	1.21E+04	1.38E+03	9.14E+02
	Std	2.03E+02	6.58E+01	2.36E+01	**1.48E+00**	2.25E+02	2.21E+02	1.92E+01
CEC-10	Mean	2.63E+03	1.68E+03	2.17E+03	1.74E+03	3.60E+03	2.34E+03	**1.56E+03**
	Std	**1.32E+02**	3.10E+02	3.24E+02	3.48E+02	1.65E+02	3.21E+02	2.02E+02
CEC-11	Mean	2.05E+03	1.18E+03	1.16E+03	**1.12E+03**	1.65E+04	1.69E+03	1.14E+03
	Std	5.83E+01	9.78E+01	2.54E+01	2.43E+03	7.35E+02	7.40E+02	**1.98E+01**
CEC-12	Mean	1.77E+08	6.46E+05	3.01E+06	1.69E+04	7.69E+05	1.06E+07	**1.25E+04**
	Std	8.78E+07	6.54E+05	3.28E+06	2.45E+04	8.34E+05	1.62E+07	**1.05E+04**
CEC-13	Mean	5.51E+06	9.94E+03	2.22E+04	4.34E+03	3.94E+05	1.65E+04	**1.82E+03**
	Std	6.20E+06	1.21E+04	1.25E+04	2.76E+03	3.73E+05	1.14E+04	**5.80E+02**
CEC-14	Mean	3.19E+03	5.07E+03	3.55E+03	**1.47E+03**	5.37E+05	1.62E+03	1.49E+03
	Std	3.89E+03	5.57E+03	1.84E+03	**2.11E+01**	3.91E+03	1.73E+02	2.77E+01
CEC-15	Mean	2.18E+04	9.63E+03	4.48E+03	1.61E+03	1.15E+05	7.77E+03	**1.59E+03**
	Std	5.00E+04	7.99E+03	2.65E+03	5.80E+01	6.57E+03	4.65E+03	**6.35E+01**
CEC-16	Mean	2.13E+03	1.73E+03	1.74E+03	**1.68E+03**	3.24E+03	1.97E+03	1.72E+03
	Std	9.23E+01	9.33E+01	1.05E+02	**8.52E+01**	1.65E+02	1.53E+02	1.09E+02
CEC-17	Mean	1.88E+03	1.77E+03	1.78E+03	**1.75E+03**	3.12E+03	1.77E+03	**1.75E+03**
	Std	6.86E+01	4.70E+01	2.99E+01	**1.03E+01**	2.39E+02	1.84E+01	2.07E+01
CEC-18	Mean	1.29E+07	2.78E+04	7.96E+04	6.61E+03	2.87E+04	1.69E+04	**5.28E+03**
	Std	1.15E+07	1.26E+04	6.55E+04	3.00E+06	2.34E+03	1.03E+04	**6.89E+03**
CEC-19	Mean	2.54E+05	1.32E+04	1.09E+04	2.00E+05	5.68E+05	3.43E+05	**1.96E+03**
	Std	3.15E+05	1.22E+04	6.34E+03	1.04E+02	3.43E+05	1.72E+05	**5.97E+01**
CEC-20	Mean	2.26E+03	2.04E+03	2.14E+03	**2.03E+03**	2.45E+03	2.18E+03	**2.13E+03**
	Std	5.33E+01	3.63E+01	4.64E+01	**1.06E+01**	2.71E+02	7.09E+01	5.64E+01
CEC-21	Mean	2.30E+03	2.31E+03	2.32E+03	2.25E+03	1.60E+03	2.35E+03	**2.18E+03**
	Std	2.53E+01	4.77E+01	4.31E+01	5.06E+01	5.80E+01	4.68E+01	**2.49E+01**
CEC-22	Mean	2.74E+03	2.45E+03	2.33E+03	**2.30E+03**	2.69E+03	2.68E+03	**2.30E+03**
	Std	1.99E+02	3.58E+02	7.51E+00	1.37E+01	6.04E+02	3.83E+02	**3.78E+00**
CEC-23	Mean	2.70E+03	2.62E+03	2.64E+03	2.62E+03	**2.05E+03**	2.68E+03	2.61E+03
	Std	1.40E+01	8.91E+01	5.97E+00	6.12E+00	7.25E+01	3.00E+01	**1.26E+00**
CEC-24	Mean	2.84E+03	2.76E+03	2.77E+03	**2.72E+03**	5.70E+03	2.83E+03	**2.72E+03**
	Std	3.62E+01	1.13E+01	7.85E+00	**6.94E+01**	2.65E+02	5.76E+01	1.00E+02
CEC-25	Mean	3.27E+03	**2.93E+03**	2.94E+03	**2.93E+03**	1.63E+03	3.12E+03	**2.93E+03**
	Std	8.80E+01	2.68E+01	1.41E+01	2.23E+01	2.26E+01	1.42E+02	**1.87E+01**
CEC-26	Mean	3.78E+03	3.06E+03	3.23E+03	2.94E+03	5.95E+03	3.85E+03	**2.93E+03**

(Continued)

Table 11 (continued)

F_n	Measure	DSSA	ESMA	AGWO	RLTLBO	SCHHO	LHHO	EHHOCBO
	Std	2.15E+02	3.10E+02	4.27E+02	9.82E+01	2.07E+03	4.11E+02	**7.92E+01**
CEC-27	Mean	3.18E+03	**3.10E+03**	**3.10E+03**	**3.10E+03**	3.27E+03	3.22E+03	**3.10E+03**
	Std	2.51E+01	1.79E+01	**1.73E+00**	4.07E+00	1.56E+02	7.55E+01	1.44E+01
CEC-28	Mean	3.31E+03	3.43E+03	3.36E+03	3.24E+03	**3.13E+03**	3.56E+03	3.27E+03
	Std	5.15E+01	1.76E+02	8.53E+01	**1.15E+02**	1.18E+02	1.59E+02	1.35E+02
CEC-29	Mean	3.45E+03	3.22E+03	3.24E+03	**3.19E+03**	5.78E+03	3.41E+03	3.22E+03
	Std	6.20E+01	7.69E+01	6.44E+01	**2.11E+01**	4.57E+02	1.36E+02	3.71E+01
CEC-30	Mean	1.74E+06	3.85E+05	6.20E+05	5.07E+05	2.19E+05	4.64E+06	**2.83E+05**
	Std	1.75E+06	4.81E+06	6.82E+05	1.74E+05	7.25E+04	4.44E+06	**1.52E+05**

The results show that the performance of the EHHOCBO algorithm is significantly better than that of DSSA, ESMA, AGWO, LHHO, and SCHHO algorithms and is comparable to the performance of the RLTLBO algorithm. For unimodal functions CEC-1 and CEC-3, EHHOCBO gives better simulation results, but in CEC-1, the standard deviation of the algorithm is slightly worse than RLTLBO. For multimodal functions, EHHOCBO achieves the best standard deviation in the test functions CEC-4, CEC-7, CEC8, and CEC-10, and the simulation results in other multimodal functions are also slightly worse than the best results. The EHHOCBO algorithm also performs well in mixed and composite functions. Among the 20 test functions from CEC-11 to CEC-30, the EHHOCBO algorithm performs better than or equal to other algorithms in 15 of them. In the remaining six functions, the simulation results of EHHOCBO are slightly worse than the optimal results. The above experimental results show that EHHOCBO also has better advantages in solving various complex optimization problems. Fig. 7 is the radar ranking chart of the five algorithms on the CEC2017 test function. The size of the range enclosed by each curve in the figure represents algorithm's performance. The smaller the range, the better the performance. The figure shows that the EHHOCBO has better performance.

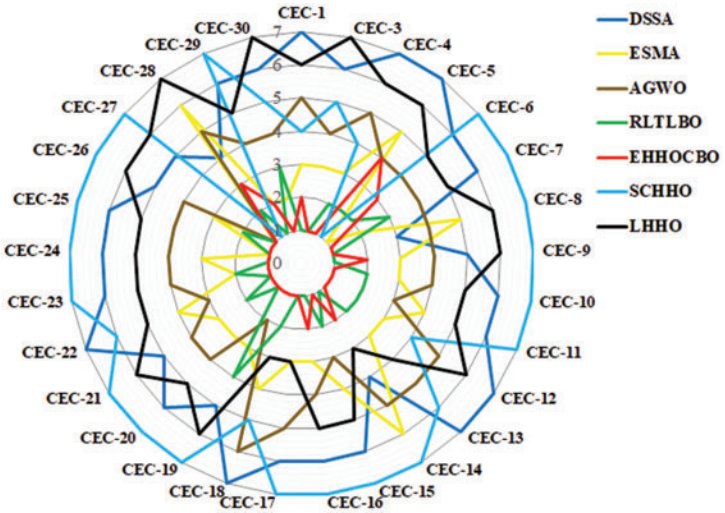

Figure 7: Ranking on IEEE CEC2017 test function

5 EHHOCBO for Addressing Engineering Problems

This section uses EHHOCBO to solve four common engineering problems in the structure field. Engineering problem testing validates the applicability and black-box nature of meta-heuristic algorithms in real-world constrained optimization. The four problems used are the cantilever beam design problem, speed reducer design problem, welded beam design problem, and rolling element bearing design problem. This paper introduces the death penalty function [60] to deal with those non-feasible candidate solutions under equality and inequality constraints. The maximum number of iterations and population size are still set to 500 and 30, respectively. In this experiment, each algorithm runs independently 30 times. The detailed experimental results and discussions are as follows.

5.1 Cantilever Beam Design Problem

The cantilever beam design problem is a structural engineering design problem that is related to the weight optimization of a cantilever beam with square cross-section. One end of the cantilever beam is rigidly supported, and the vertical force acts on the free node of the cantilever. The beam comprises five hollow blocks with constant thickness, and its height is a decision variable. The structure of the cantilever beam is shown in Fig. 8, and the mathematical formula of the problem is as follows:

Consider: $\vec{x} = [x_1, x_2, x_3, x_4, x_5]$

Minimize: $f\left(\vec{x}\right) = 0.0624\left(x_1 + x_2 + x_3 + x_4 + x_5\right)$

Subject to: $g\left(\vec{x}\right) = \frac{61}{{x_1}^3} + \frac{37}{{x_2}^3} + \frac{19}{{x_3}^3} + \frac{7}{{x_4}^3} + \frac{1}{{x_5}^3} - 1 \leq 0$

Variable range: $0.01 \leq x_i \leq 100, i = 1, 2, \cdots, 5$

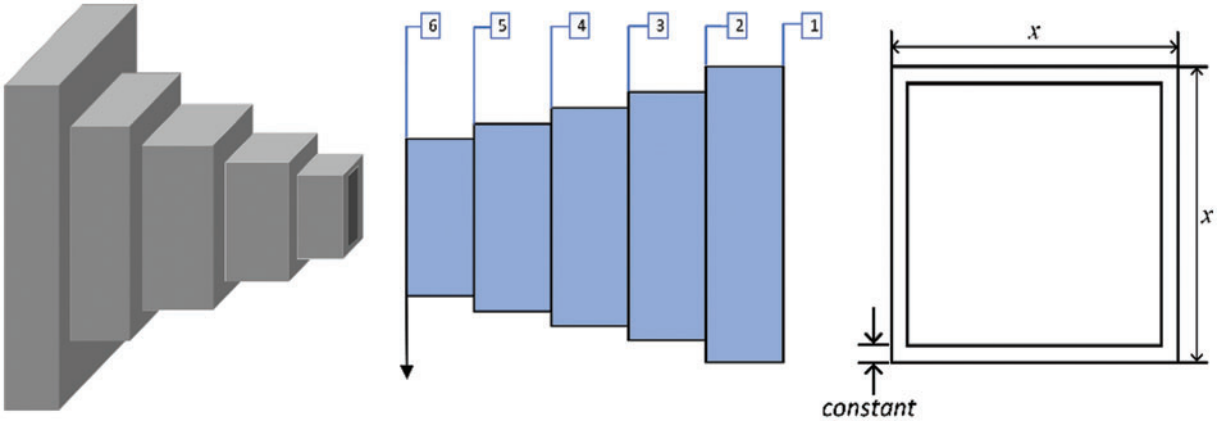

Figure 8: Schematic diagram of cantilever beam design problem

The optimization results of EHHOCBO and other algorithms for the cantilever beam design problem are presented in Table 12. The results show that EHHOCBO proposed in this paper achieves the best design. Its performance is significantly improved compared with the basic CBO and HHO algorithms. Therefore, we believe that the EHHOCBO algorithm has good potential for solving such problems.

Table 12: Results of the cantilever beam design problem

Algorithm	Optimum variables					Minimum cost
	x_1	x_2	x_3	x_4	x_5	
HHO	5.8340	5.4193	4.6158	3.5883	2.0527	1.3422
WOA	5.4947	6.2341	4.5780	2.9461	5.0408	1.5159
GWO	6.0252	5.3617	4.4035	3.5094	2.1795	1.3403
CBO	5.5809	6.6886	5.0170	2.8448	2.4897	1.4115
STOA	6.7764	5.6809	4.4343	2.9944	2.2305	1.3801
AOA	4.7413	4.8541	7.7214	9.3955	9.1411	2.2373
ChOA	6.5512	5.4661	4.1481	3.5411	1.9854	1.3536
AGWO	6.0395	5.1586	4.4599	3.5354	2.3121	1.3419
RLTLBO	5.9646	5.3115	4.6256	3.467	2.1152	1.3406
ESMA	5.975	5.3064	4.4528	3.5805	2.164	1.3403
LHHO	6.2389	5.2433	4.3853	3.3666	2.2891	1.343
SCHHO	5.8484	6.6103	3.8281	3.646	2.3151	1.3883
EHHOCBO	6.0013	5.2993	4.5250	3.5151	2.1340	**1.3400**

5.2 Speed Reducer Design Problem

The objective of this optimization problem is to minimize the weight of the reduction gear subject to 11 constraints. The decision variables in this problem are the width of the tooth face x_1, the modulus of the gear x_2, the number of teeth x_3 in the pinion, the length of the first shaft x_4, and the diameter x_5, the length of the second shaft x_6, and the diameter x_7. Fig. 9 is the schematic diagram of the reducer, and the mathematical formula is shown as follows:

Consider: $\vec{x} = [x_1, x_2, x_3, x_4, x_5, x_6, x_7]$

Minimize: $f\left(\vec{x}\right) = 0.7854x_1{x_2}^2\left(3.3333{x_3}^2 + 14.9334x_3 - 43.0934\right) - 1.508x_1\left({x_6}^2 + x_7^2\right) + 7.4777\left({x_6}^3 + {x_7}^3\right)$

$$g_1\left(\vec{x}\right) = \frac{27}{x_1{x_2}^2x_3} - 1 \le 0, g_2\left(\vec{x}\right) = \frac{397.5}{x_1{x_2}^2{x_3}^2} - 1 \le 0$$

$$g_3\left(\vec{x}\right) = \frac{1.93{x_4}^3}{x_2x_3{x_6}^4} - 1 \le 0, g_4\left(\vec{x}\right) = \frac{1.93{x_5}^3}{x_2x_3{x_7}^4} - 1 \le 0$$

Subject to: $g_5\left(\vec{x}\right) = \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9 \times 10^6}}{110.0{x_6}^3} - 1 \le 0$

$$g_6\left(\vec{x}\right) = \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 157.5 \times 10^6}}{85.0{x_6}^3} - 1 \le 0$$

$$g_7\left(\vec{x}\right) = \frac{x_2x_3}{40} - 1 \le 0, g_8\left(\vec{x}\right) = \frac{5x_2}{x_1} - 1 \le 0$$

$$g_9\left(\vec{x}\right) = \frac{x_1}{12x_2} - 1 \le 0, g_{10}\left(\vec{x}\right) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0$$
$$g_{11}\left(\vec{x}\right) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0$$

Variable range: $2.6 \le x_1 \le 3.6, 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28, 7.3 \le x_4 \le 8.3, 7.8 \le x_5 \le 8.3, 2.9 \le x_6 \le 3.9, 5.0 \le x_7 \le 5.5$

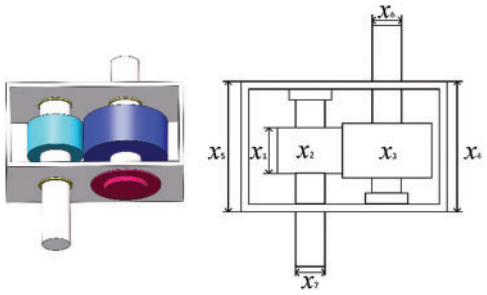

Figure 9: Schematic diagram of speed reducer design problem

The results of this experiment are recorded in Table 13. The results show that the minimum weight of EHHOCBO is slightly worse than that of RLTLBO and ESMA algorithms. Although EHHOCBO could have better design results, it achieves relatively good performance. Therefore, it is reasonable to believe that the proposed hybrid technique is suitable for solving the speed reducer design problem.

Table 13: Results of the cantilever beam design problem

Algorithms	Optimum variables							Minimum weight
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	
HHO	3.5171	0.7020	23.0301	7.3000	7.9853	3.3486	5.2861	4233.5583
WOA	3.5000	0.7000	17.0000	8.0694	8.0694	3.7917	5.3208	3160.7315
GWO	3.5035	0.7000	17.0000	7.8834	7.9377	3.3642	5.2878	3010.1887
CBO	3.5009	0.7000	17.0000	7.4292	8.3000	3.3518	5.2869	3009.3681
STOA	3.6000	0.7000	17.0000	8.3000	8.3000	3.3694	5.5000	3202.1389
AOA	3.5023	0.7000	17.0000	7.3000	8.3000	3.5203	5.5000	3195.8477
ChOA	3.6000	0.7000	17.0000	8.0581	8.3000	3.7853	5.3064	3193.0844
AGWO	3.5034	0.7000	17.0000	7.3038	7.8173	3.4473	5.3612	3070.6211
RLTLBO	3.5432	0.7063	15.3372	7.3760	7.7711	3.4832	5.3148	**2812.3482**
ESMA	3.5000	0.7000	17.0000	7.3000	7.7155	3.3502	5.2867	2994.4839
LHHO	3.5000	0.7000	17.0000	7.7287	7.7422	3.4822	5.2867	3035.8028
SCHHO	3.6000	0.7000	17.0000	7.3000	8.3000	3.3621	5.5000	3191.3621
EHHOCBO	3.5000	0.7000	17.0000	7.7064	7.7659	3.3510	5.2867	2999.3758

5.3 Welded Beam Design Problem

The purpose of the design problem of the welded beam is to reduce the manufacturing cost of the design as much as possible under the constraint of meeting the beam's shear stress, bending stress, bending load, end deviation, and boundary conditions. The problem can be described as an optimization design problem with four decision variables. The four decision variables are length l, height t, thickness b, and weld thickness h of the beam bar. Fig. 10 is the schematic diagram of the welded beam. The mathematical representation of the problem is as follows:

Consider: $\vec{x} = [x_1, x_2, x_3, x_4] = [h, l, t, b]$

Minimize: $f\left(\vec{x}\right) = 1.10471x_1^2x_2 + 0.04811x_3x_4\left(14 + x_2\right)$

$$g_1\left(\vec{x}\right) = \tau\left(\vec{x}\right) - \tau_{\max} \le 0$$

$$g_2\left(\vec{x}\right) = \sigma - \sigma_{\max} \le 0$$

$$g_3\left(\vec{x}\right) = \delta - \delta_{\max} \le 0$$

Subject to: $g_4\left(\vec{x}\right) = x_1 - x_4 \le 0$

$$g_5\left(\vec{x}\right) = P - P_C\left(\vec{x}\right) \le 0$$

$$g_6\left(\vec{x}\right) = 0.125 - x_1 \le 0$$

$$g_7\left(\vec{x}\right) = 1.10471x_1^2 + 0.04811x_3x_4\left(14 + x_2\right) - 5 \le 0$$

Variable range: $0.1 \le x_1, x_4 \le 2, 0.1 \le x_2, x_3 \le 10$

where $f\left(\vec{x}\right)$ represents the cost of manufacturing welded beams, and other variables are defined as follows:

$$\tau\left(\vec{x}\right) = \sqrt{\left(\tau'\right)^2 + 2\tau'\tau''\frac{x_2}{2R} + \left(\tau''\right)^2}, \tau' = \frac{P}{\sqrt{2}x_1x_2}, \tau'' = \frac{MR}{J}, M = P\left(L + \frac{x_2}{2}\right)$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}, \sigma\left(\vec{x}\right) = \frac{6PL}{Ex_3^2x_4}$$

$$\delta\left(\vec{x}\right) = \frac{6PL^3}{Ex_3^2x_4}, P_C\left(\vec{x}\right) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$$

$P = 6000\,\text{lb}, L = 14\,\text{in}, E = 30 \times 10^6\,\text{psi}, G = 12 \times 10^6\,\text{psi},$

$\delta_{\max} = 0.25\,\text{in}, \tau_{\max} = 13600\,\text{psi}, \sigma_{\max} = 30000\,\text{psi}.$

The results of the experiment are shown in Table 14. EHHOCBO obtaines the lowest manufacturing cost of 1.7256 under the corresponding parameters $h = 0.2057$, $l = 3.4698$, $t = 9.0438$, and $b = 0.2057$. Since EHHOCBO has achieved the most satisfactory results in all competitive algorithms, it has a good application prospect in designing welded beams.

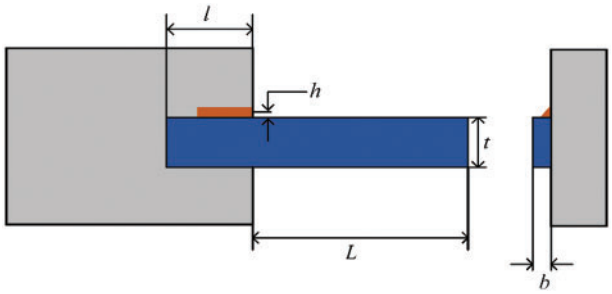

Figure 10: Schematic diagram of the welded beam design problem

Table 14: Results of the welded beam design problem

Algorithm	Optimum variables				Minimum cost
	$h\,(x_1)$	$l\,(x_2)$	$t\,(x_3)$	$b\,(x_4)$	
HHO	0.2062	3.6249	8.9799	0.2083	1.7566
WOA	0.4092	4.1027	6.2098	0.4357	3.1151
GWO	0.2053	3.4862	9.0483	0.2057	1.7281
CBO	0.2676	2.8620	7.8765	0.2764	1.9923
STOA	0.1652	4.9035	9.0370	0.2081	1.8580
AOA	0.1950	5.1639	10.0000	0.2022	2.0810
ChOA	0.1840	3.6292	10.0000	0.2066	1.8877
AGWO	0.1895	3.8867	9.0386	0.2074	1.7673
RLTLBO	0.21733	3.7545	8.7271	0.22084	1.8422
ESMA	0.2511	2.9794	8.1776	0.2512	1.8856
LHHO	0.1254	8.0461	8.9965	0.20757	2.1204
SCHHO	0.19879	4.1188	9.2621	0.2066	1.8477
EHHOCBO	0.2057	3.4698	9.0436	0.2057	**1.7256**

5.4 Rolling Element Bearing Design Problem

The objective of the rolling element bearing design problem is to find the maximum dynamic load capacity of a bearing subject to more than ten decision parameters. The decision parameters are pitch diameter D_m, ball diameter D_b, number of balls Z, inner and outer raceway radius of curvature coefficients f_i and f_o, $K_{d\min}$, $K_{d\max}$, δ, e, and ζ. Fig. 11 shows schematic diagram of the rolling element bearing. The mathematical expression of the problem is as follows:

$$\text{Maximize: } C_d = \begin{cases} f_c Z^{2/3} D_b^{1.8}, \textit{if } D_b \leq 25.4\,\text{mm} \\ 3.647 f_c Z^{2/3} D_b^{1.4}, \textit{else} \end{cases}$$

Subject to:

$$g_1\left(\vec{x}\right) = \frac{\phi_0}{2\sin^{-1}\left(D_b/D_m\right)} - Z + 1 \le 0$$

$$g_2\left(\vec{x}\right) = 2D_b - K_{d\min}\left(D-d\right) > 0$$

$$g_3\left(\vec{x}\right) = K_{d\max}\left(D-d\right) - 2D_b \ge 0$$

$$g_4\left(\vec{x}\right) = \zeta B_w - D_b \le 0$$

$$g_5\left(\vec{x}\right) = D_m - 0.5\left(D+d\right) \ge 0$$

$$g_6\left(\vec{x}\right) = \left(0.5+e\right)\left(D+d\right) - D_m \ge 0$$

$$g_7\left(\vec{x}\right) = 0.5\left(D - D_m - D_b\right) - \delta D_b \ge 0$$

$$g_8\left(\vec{x}\right) = f_i \ge 0.515$$

$$g_9\left(\vec{x}\right) = f_o \ge 0.515$$

where

$$f_c = 37.91\left[1 + \left\{1.04\left(\frac{1-\gamma}{1+\gamma}\right)^{1.72}\left(\frac{f_i\left(2f_o-1\right)}{f_o\left(2f_i-1\right)}\right)^{0.41}\right\}^{10/3}\right]^{-0.3} \times \left[\frac{\gamma^{0.3}\left(1-\gamma\right)^{1.39}}{\left(1+\gamma\right)^{1/3}}\right]\left[\frac{2f_i}{2f_i-1}\right]^{0.41}$$

$$x = \left[\left\{\left(D-d\right)/2 - 3\left(T/4\right)\right\}^2 + \left\{D/2 - T/4 - D_b\right\}^2 - \left\{d/2 + T/4\right\}^2\right]$$

$$y = 2\left\{\left(D-d\right)/2 - 3\left(T/4\right)\right\}\left\{D/2 - T/4 - D_b\right\}$$

$$\phi_0 = 2\prod - \cos^{-1}\left(\frac{x}{y}\right), \gamma = \frac{D_b}{D_m}, f_i = \frac{r_i}{D_b}, f_o = \frac{r_o}{D_b}, T = D - d - 2D_b$$

$$D = 160, d = 90, B_w = 30, r_i = r_o = 11.033, 0.5\left(D+d\right) \le D_m \le 0.6\left(D+d\right)$$

$$0.15\left(D-d\right) \le D_b \le 0.45\left(D-d\right), 4 \le Z \le 50, 0.515 \le f_i \text{ and } f_o \le 0.6$$

$$0.4 \le K_{d\min} \le 0.5, 0.6 \le K_{d\max} \le 0.7, 0.3 \le \delta \le 0.4, 0.02 \le e \le 0.1, 0.6 \le \zeta \le 0.85.$$

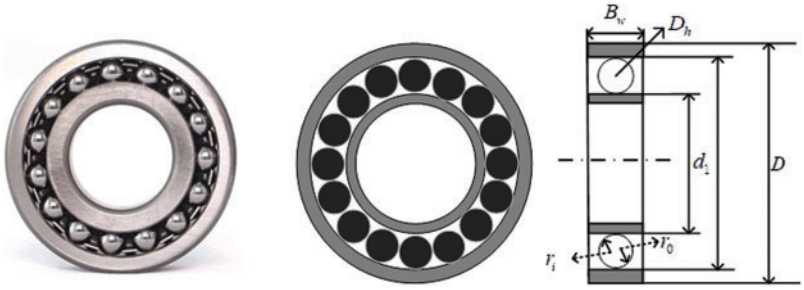

Figure 11: Schematic diagram of rolling element bearing design problem

The results of this experiment are reported in Table 15. It can be seen from this table that compared with other classical algorithms, the proposed EHHOCBO provides the best fitness value of 85549.

Table 15: Results for rolling element bearing design problem

Algorithm	HHO	WOA	GWO	CBO	STOA	AOA	ChOA	AGWO	RLTLBO	ESMA	LHHO	SCHHO	EHHOCBO
	125.0	125.0	125.5	125.7	125.0	125.0	125.0	125.0	125.6	125.7	125.6	125.0	125.7
D_b	21.0	21.0	21.4	21.4	21.0	20.2	21.0	21.3	21.4	21.4	21.4	21.1	21.4
Z	11.5	11.5	10.8	10.8	9.5	11.0	11.2	11.1	10.6	11.2	11.0	10.1	11.2
f_i	0.515	0.515	0.515	0.515	0.515	0.515	0.515	0.515	0.515	0.515	0.515	0.515	0.515
f_o	0.515	0.515	0.599	0.518	0.515	0.600	0.596	0.515	0.553	0.515	0.515	0.600	0.515
$K_{d\,\min}$	0.400	0.400	0.476	0.455	0.400	0.400	0.440	0.439	0.404	0.448	0.500	0.400	0.479
$K_{d\,\max}$	0.600	0.600	0.622	0.676	0.700	0.600	0.700	0.666	0.650	0.651	0.700	0.700	0.652
δ	0.300	0.300	0.306	0.300	0.300	0.300	0.300	0.316	0.617	0.300	0.303	0.300	0.300
e	0.048	0.048	0.080	0.067	0.020	0.020	0.020	0.029	0.297	0.079	0.040	0.020	0.075
ζ	0.600	0.600	0.607	0.600	0.600	0.600	0.653	0.619	0.726	0.654	0.611	0.611	0.605
Optimum cost	82533	82552	85162	85520	77760	77138	82445	84334	85306	85548	85405	78349	**85549**

The above experiments show that EHHOCBO has advantages in solving practical engineering problems. This is attributed to the combination of HHO and CBO, as well as the introduction of the ensemble mutation strategy and refracted opposition-based learning, which improves the proposed method's searchability significantly.

6 Conclusion and Future Directions

Considering the respective characteristics of the HHO algorithm and CBO algorithm, a new improved hybrid algorithm named EHHOCBO is proposed in this paper. First, the leader mechanism of CBO is introduced in the initialization phase of HHO to provide a good basis for global search and enhance the exploration ability of the algorithm. Then, an ensemble mutation strategy is employed to increase the population diversity and further boost the exploration trend. Finally, refracted opposition-based learning is used to update the optimal solution to expand the search range and avoid the algorithm falling into the local optima. In order to fully evaluate the performance of the proposed algorithm, EHHOCBO was compared with the basic HHO, CBO algorithm, and other meta-heuristic algorithms based on classical benchmark functions and the IEEE CEC2017 test suite. Wilcoxon's rank sum test verified the significance of the experimental results. Through a series of numerical statistics, it is verified that the EHHOCBO algorithm has significantly improved in terms of accuracy, convergence speed, stability, and avoidance of falling into local optimum solutions. In addition, four engineering problems were used to verify the applicability of the EHHOCBO algorithm. Experiments show that the algorithm can effectively provide competitive solutions for these real-life engineering problems.

Although the EHHOCBO algorithm proposed in this paper has been significantly improved over the basic HHO and CBO algorithms, there is still room for further improvement in its performance on IEEE CEC2017 test functions. The EHHOCBO algorithm still suffers from the major limitation of excessive computation time and needs to be improved. It is believed that this situation can be alleviated by introducing several parallel mechanisms, such as master-slave models, cellular models and coordination strategies. In future work, we aim to further improve the performance of the EHHOCBO algorithm by developing new improvement strategies and parallelism mechanisms while reducing the total time consumption. And we hope to make the parameter settings more reasonable

by incorporating parameter adaptive control. In addition, we believe it would also be interesting to implement EHHOCBO to solve more practical optimization problems such as feature selection, path planning, predictive modeling, image segmentation, etc. [61]. We will enhance our work in this area.

Acknowledgement: The authors are grateful to the editor and reviewers for their constructive comments and suggestions, which have improved the presentation.

Funding Statement: This work is financially supported by the National Natural Science Foundation of China under Grant 52075090, Key Research and Development Program Projects of Heilongjiang Province under Grant GA21A403, the Fundamental Research Funds for the Central Universities under Grant 2572021BF01, and Natural Science Foundation of Heilongjiang Province under Grant YQ2021E002.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Xiao, Y., Sun, X., Guo, Y., Li, S., Zhang, Y. et al. (2022). An improved gorilla troops optimizer based on lens opposition-based learning and adaptive beta-Hill climbing for global optimization. *Computer Modeling in Engineering & Sciences, 131(2),* 815–850. https://doi.org/10.32604/cmes.2022.019198
2. Xiao, Y., Sun, X., Guo, Y., Cui, H., Wang, Y. et al. (2022). An enhanced honey badger algorithm based on Levy flight and refraction opposition-based learning for engineering design problems. *Journal of Intelligent & Fuzzy Systems, 43(4),* 4517–4540. https://doi.org/10.3233/JIFS-213206
3. Zhang, X. C., Zhao, K., Niu, Y. (2020). Improved Harris hawks optimization based on adaptive cooperative foraging and dispersed foraging strategies. *IEEE Access, 8,* 160297–160314. https://doi.org/10.1109/ACCESS.2020.3013332
4. Boussaid, I., Lepagnot, J., Siarry, P. (2013). A survey on optimization metaheuristics. *Information Sciences, 237,* 82–117. https://doi.org/10.1016/j.ins.2013.02.041
5. Jia, H. M., Li, Y., Sun, K. J., Cao, N., Zhou, H. M. (2021). Hybrid sooty tern optimization and differential evolution for feature selection. *Computer Systems Science and Engineering, 39(3),* 321–335. https://doi.org/10.32604/csse.2021.017536
6. Fan, Q. S., Huang, H. S., Yang, K., Zhang, S. S., Yao, L. G. et al. (2021). A modified equilibrium optimizer using opposition-based learning and novel update rules. *Expert Systems with Applications, 170(9),* 114575. https://doi.org/10.1016/j.eswa.2021.114575
7. Hussain, K., Mohd Salleh, M. N., Cheng, S., Shi, Y. (2019). Metaheuristic research: A comprehensive survey. *Artificial Intelligence Review, 52(4),* 2191–2233. https://doi.org/10.1007/s10462-017-9605-z
8. Zheng, R., Jia, H. M., Wang, S., Liu, Q. X. (2022). Enhanced slime mould algorithm with multiple mutation strategy and restart mechanism for global optimization. *Journal of Intelligent & Fuzzy Systems, 42(6),* 5069–5083. https://doi.org/10.3233/JIFS-211408
9. Xiao, Y., Guo, Y., Cui, H., Wang, Y., Li, J. et al. (2022). IHAOAVOA: An improved hybrid aquila optimizer and African vultures optimization algorithm for global optimization problems. *Mathematical Biosciences and Engineering, 19(11),* 10963–11017. https://doi.org/10.3934/mbe.2022512
10. Zhong, K. Y., Zhou, G., Deng, W., Zhou, Y. Q., Luo, Q. F. (2021). MOMPA: Multi-objective marine predator algorithm. *Computer Methods in Applied Mechanics and Engineering, 385(1),* 114029. https://doi.org/10.1016/j.cma.2021.114029

11. Yu, G., Wang, H., Zhou, H. Z., Zhao, S. S., Wang, Y. (2021). An efficient firefly algorithm based on modified search strategy and neighborhood attraction. *International Journal of Intelligent Systems, 36(8),* 4346–4363. https://doi.org/10.1002/int.22462
12. Xiao, Y., Sun, X., Zhang, Y., Guo, Y., Wang, Y. et al. (2021). An improved slime mould algorithm based on tent chaotic mapping and nonlinear inertia weight. *International Journal of Innovative Computing, Information and Control, 17(6),* 2151–2176. https://doi.org/10.24507/ijicic.17.06.2151
13. Nguyen, T. T., Wang, H. J., Dao, T. K., Pan, J. S., Liu, J. H. et al. (2020). An improved slime mold algorithm and its application for optimal operation of cascade hydropower stations. *IEEE Access, 8,* 226754–226772. https://doi.org/10.1109/ACCESS.2020.3045975
14. Dehghani, M., Montazeri, Z., Givi, H., Guerrero, J., Dhiman, G. (2020). Darts game optimizer: A new optimization technique based on darts game. *International Journal of Intelligent Engineering and Systems, 13(5),* 286–294. https://doi.org/10.22266/ijies2020.1031.26
15. Hamed, A. Y., Alkinani, M. H., Hassan, M. R. (2020). A genetic algorithm optimization for multi-objective multicast routing. *Intelligent Automation and Soft Computing, 26(6),* 1201–1216. https://doi.org/10.32604/iasc.2020.012663
16. Storn., R., Price., K. (1997). Differential evolution–A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization, 11(4),* 341–359. https://doi.org/10.1023/A:1008202821328
17. Beyer, H. G., Schwefel, H. P. (2002). Evolution strategies. A comprehensive introduction. *Natural Computing, 1(1),* 3–52. https://doi.org/10.1023/A:1015059928466
18. Kirkpatrick, S., Gelatt, C. D., Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220(4598),* 671–680. https://doi.org/10.1126/science.220.4598.671
19. Mirjalili, S., Mirjalili, S. M., Hatamlou, A. (2016). Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Computing & Applications, 27(2),* 495–513. https://doi.org/10.1007/s00521-015-1870-7
20. Zhao, W. G., Wang, L. Y., Zhang, Z. X. (2019). Atom search optimization and its application to solve a hydrogeologic parameter estimation problem. *Knowledge-Based Systems, 163(4598),* 283–304. https://doi.org/10.1016/j.knosys.2018.08.030
21. Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., Gandomi, A. H. (2021). The arithmetic optimization algorithm. *Computer Methods in Applied Mechanics and Engineering, 376(2),* 113609. https://doi.org/10.1016/j.cma.2020.113609
22. Kennedy, J., Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95—International Conference on Neural Networks*, pp. 1942–1948. Perth, WA, Australia. https://doi.org/10.1109/ICNN.1995.488968
23. Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems, 96(63),* 120–133. https://doi.org/10.1016/j.knosys.2015.12.022
24. Dhiman, G., Kaur, A. (2019). STOA: A bio-inspired based optimization algorithm for industrial engineering problems. *Engineering Applications of Artificial Intelligence, 82(2),* 148–174. https://doi.org/10.1016/j.engappai.2019.03.021
25. Khishe, M., Mosavi, M. R. (2020). Chimp optimization algorithm. *Expert Systems with Applications, 149(1),* 113338. https://doi.org/10.1016/j.eswa.2020.113338
26. Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A. A., Al-qaness, M. A. A. et al. (2021). Aquila optimizer: A novel meta-heuristic optimization algorithm. *Computers & Industrial Engineering, 157(11),* 107250. https://doi.org/10.1016/j.cie.2021.107250
27. Mirjalili, S. (2016). Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Computing & Applications, 27(4),* 1053–1073. https://doi.org/10.1007/s00521-015-1920-1

28. Arora, S., Singh, S. (2019). Butterfly optimization algorithm: A novel approach for global optimization. *Soft Computing, 23(3),* 715–734. https://doi.org/10.1007/s00500-018-3102-4
29. Li, S. M., Chen, H. L., Wang, M. J., Heidari, A. A., Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems, 111(Supplement C),* 300–323. https://doi.org/10.1016/j.future.2020.03.055
30. Mirjalili, S., Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software, 95(12),* 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
31. Mirjalili, S., Mirjalili, S. M., Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69,* 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
32. Dhiman, G., Kumar, V. (2019). Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. *Knowledge-Based Systems, 165(25),* 169–196. https://doi.org/10.1016/j.knosys.2018.11.024
33. Abdollahzadeh, B., Gharehchopogh, F. S., Mirjalili, S. (2021). Artificial gorilla troops optimizer: A new nature-inspired metaheuristic algorithm for global optimization problems. *International Journal of Intelligent Systems, 36(10),* 5887–5958. https://doi.org/10.1002/int.22535
34. Goncalves, M. S., Lopez, R. H., Miguel, L. F. F. (2015). Search group algorithm: A new meta-heuristic method for the optimization of truss structures. *Computers & Structures, 153(12),* 165–184. https://doi.org/10.1016/j.compstruc.2015.03.003
35. Wolpert, D. H., Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1(1),* 67–82. https://doi.org/10.1109/4235.585893
36. Jia, H. M., Sun, K. J., Zhang, W. Y., Leng, X. (2022). An enhanced chimp optimization algorithm for continuous optimization domains. *Complex & Intelligent Systems, 8(1),* 65–82. https://doi.org/10.1007/s40747-021-00346-5
37. Debnath, S., Baishya, S., Sen, D., Arif, W. (2021). A hybrid memory-based dragonfly algorithm with differential evolution for engineering application. *Engineering with Computers, 37(4),* 2775–2802. https://doi.org/10.1007/s00366-020-00958-4
38. Ziyu, T., Dingxue, Z. (2009). A modified particle swarm optimization with an adaptive acceleration coefficients. *2009 Asia-Pacific Conference on Information Processing*, pp. 330–332. Shenzhen, China. https://doi.org/10.1109/APCIP.2009.217
39. Li, S. L., Li, X. B., Chen, H., Zhao, Y. X., Dong, J. W. (2021). A novel hybrid hunger games search algorithm with differential evolution for improving the behaviors of non-cooperative animals. *IEEE Access, 9,* 164188–164205. https://doi.org/10.1109/ACCESS.2021.3132617
40. Jia, H., Jiang, Z., Li, Y. (2022). Simultaneous feature selection optimization based on improved bald eagle search algorithm. *Control and Decision, 37(2),* 445–454. https://doi.org/10.13195/j.kzyjc.2020.1025
41. Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M. et al. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems, 97,* 849–872. https://doi.org/10.1016/j.future.2019.02.028
42. Fan, Q., Chen, Z. J., Xia, Z. H. (2020). A novel quasi-reflected Harris hawks optimization algorithm for global optimization problems. *Soft Computing, 24(19),* 14825–14843. https://doi.org/10.1007/s00500-020-04834-7
43. Dokeroglu, T., Sevinc, E. (2022). An island parallel Harris hawks optimization algorithm. *Neural Computing and Applications, 34(21),* 18341–18368. https://doi.org/10.1007/s00521-022-07367-2
44. Wang, S. A., Jia, H. M., Abualigah, L., Liu, Q. X., Zheng, R. (2021). An improved hybrid aquila optimizer and Harris hawks algorithm for solving industrial engineering optimization problems. *Processes, 9(9),* 1551. https://doi.org/10.3390/pr9091551
45. Naruei, I., Keynia, F. (2021). A new optimization method based on COOT bird natural life model. *Expert Systems with Applications, 183(2),* 115352. https://doi.org/10.1016/j.eswa.2021.115352

46. Huang, Y. H., Zhang, J., Wei, W., Qin, T., Fan, Y. C. et al. (2022). Research on coverage optimization in a WSN based on an improved COOT bird algorithm. *Sensors, 22(9),* 3383. https://doi.org/10.3390/s22093383
47. Wang, Y., Cai, Z. X., Zhang, Q. F. (2011). Differential evolution with composite trial vector generation strategies and control parameters. *IEEE Transactions on Evolutionary Computation, 15(1),* 55–66. https://doi.org/10.1109/TEVC.2010.2087271
48. Long, W., Wu, T. B., Jiao, J. J., Tang, M. Z., Xu, M. (2020). Refraction-learning-based whale optimization algorithm for high-dimensional problems and parameter estimation of PV model. *Engineering Applications of Artificial Intelligence, 89(1),* 103457. https://doi.org/10.1016/j.engappai.2019.103457
49. Tizhoosh, H. R. (2005). Opposition-based learning: A new scheme for machine intelligence. *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce*, pp. 695–701. Vienna, Austria. https://doi.org/10.1109/CIMCA.2005.1631345
50. Sihwail, R., Omar, K., Ariffin, K. A. Z., Tubishat, M. (2020). Improved harris hawks optimization using elite opposition-based learning and novel search mechanism for feature selection. *IEEE Access, 8,* 121127–121145. https://doi.org/10.1109/ACCESS.2020.3006473
51. Digalakis, J. G., Margaritis, K. G. (2001). On benchmarking functions for genetic algorithms. *International Journal of Computer Mathematics, 77(4),* 481–506. https://doi.org/10.1080/00207160108805080
52. Cheng, R., Li, M., Tian, Y., Zhang, X., Jin, Y. et al. (2017). Benchmark functions for the CEC'2017 competition on evolutionary many-objective optimization. http://hdl.handle.net/2086/13857
53. García, S., Fernández, A., Luengo, J., Herrera, F. (2010). Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences, 180(10),* 2044–2064. https://doi.org/10.1016/j.ins.2009.12.010
54. Jena, B., Naik, M. K., Wunnava, A., Panda, R. (2021). A differential squirrel search algorithm. In: Das, S., Mohanty, M. N. (Eds.), *Advances in intelligent computing and communication*, pp. 143–152. Singapore: Springer.
55. Naik, M. K., Panda, R., Abraham, A. (2021). An entropy minimization based multilevel colour thresholding technique for analysis of breast thermograms using equilibrium slime mould algorithm. *Applied Soft Computing, 113,* 107955. https://doi.org/10.1016/j.asoc.2021.107955
56. Ma, C., Huang, H., Fan, Q., Wei, J., Du, Y. et al. (2022). Grey wolf optimizer based on aquila exploration method. *Expert Systems with Applications, 205,* 117629. https://doi.org/10.1016/j.eswa.2022.117629
57. Wu, D., Wang, S., Liu, Q., Abualigah, L., Jia, H. (2022). An improved teaching-learning-based optimization algorithm with reinforcement learning strategy for solving optimization problems. *Computational Intelligence and Neuroscience, 2022,* 1535957. https://doi.org/10.1155/2022/1535957
58. Naik, M. K., Panda, R., Wunnava, A., Jena, B., Abraham, A. (2021). A leader Harris hawks optimization for 2-D Masi entropy-based multilevel image thresholding. *Multimedia Tools and Applications, 80(28),* 35543–35583. https://doi.org/10.1007/s11042-020-10467-7
59. Hussain, K., Neggaz, N., Zhu, W., Houssein, E. H. (2021). An efficient hybrid sine-cosine Harris hawks optimization for low and high-dimensional feature selection. *Expert Systems with Applications, 176,* 114778. https://doi.org/10.1016/j.eswa.2021.114778
60. Coello Coello, C. A. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering, 191(11),* 1245–1287. https://doi.org/10.1016/S0045-7825(01)00323-1
61. Shehab, M., Mashal, I., Momani, Z., Shambour, M. K. Y., Al-Badareen, A. et al. (2022). Harris hawks optimization algorithm: Variants and applications. *Archives of Computational Methods in Engineering, 29(7),* 5579–5603. https://doi.org/10.1007/s11831-022-09780-1