IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 12 May 2024, accepted 29 May 2024, date of publication 3 June 2024, date of current version 23 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3408644

RESEARCH ARTICLE

Multi-Strategy Fusion Improved Dung Beetle Optimization Algorithm and Engineering Design Application

DAMING ZHANG[1,2], ZIJIAN WANG[1], YANQING ZHAO[1], AND FANGJIN SUN[3]

[1]College of Computer Science and Engineering, Guilin University of Technology, Guilin 541004, China
[2]Guangxi Key Laboratory of Embedded Technology and Intelligent System, Guilin University of Technology, Guilin 541004, China
[3]College of Civil Engineering and Architecture, Guilin University of Technology, Guilin 541004, China

Corresponding author: Fangjin Sun (e_dm@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 52178468 and Grant 52268023 and in part by the Natural Science Foundation of Guangxi under Grant 2023GXNSFAA026418.

ABSTRACT In this paper, a Multi-Strategy fusion Improved Dung Beetle Optimization (MSIDBO) algorithm is proposed to address the problems that the dung beetle optimization algorithm is prone to fall into local optimum, suffers from weak global exploration, and shows slow convergence to a solution. In MSIDBO, a good point set strategy is adopted to generate a more diverse initial population; Golden sine and t-distribution perturbation strategies are used to improve the global search capability of the algorithm; By introducing the adaptive Gaussian-Cauchy mutation strategy, the probability of the algorithm falling into local optimal is reduced. To verify the effectiveness of the MSIDBO algorithm, it was tested against 23 benchmark functions and 29 CEC-2017 test functions, comparing its performance with that of other well-known metaheuristic algorithms. The results show that the MSIDBO algorithm excelled in 17 out of 23 benchmark test functions, achieving higher solution accuracy and faster convergence compared to the original DBO algorithm, while the remaining 6 functions yielded comparable results. Similarly, among the 29 CEC-2017 test functions, the MSIDBO algorithm surpassed the original DBO algorithm in 25, and the remaining 4 functions yielded similar results. Additionally, to verify the practical application potential of the MSIDBO algorithm, this paper applies it to optimize three engineering design problems, and the experimental results show that the MSIDBO algorithm has a higher application potential compared with other algorithms.

INDEX TERMS Dung beetle optimization algorithm (DBO), MSIDBO, good point set, golden sine strategy, adaptive Gaussian-Cauchy mutation, engineering design problems.

I. INTRODUCTION

Inspired by natural physical laws and biological habits, scholars have developed a range of swarm intelligence optimization algorithms, including Particle Swarm Optimization Algorithm (PSO) [1], Grey Wolf Optimization Algorithm (GWO) [2], Harris Hawk Optimization Algorithm (HHO) [3], Whale Optimization Algorithm (WOA) [4], Sparrow Search Algorithm (SSA) [5], Slime Mould Algorithm (SMA) [6], Hunger Games Search Algorithm (HGS) [7], Differential Evolutionary Algorithm (DE) [8], and Sine Cosine Algorithm (SCA) [9]. These algorithms are widely used in various real-world fields, including fault detection [10], job shop scheduling [11], and engineering optimization fields [12], because of their stability and ease of use.

Dung beetle optimizer (DBO) is a novel meta-heuristic algorithm first proposed by Xue and Shen [13] in 2022, which is inspired by the social behaviors of dung beetle populations such as ball-rolling, dancing, foraging, breeding, and stealing. The algorithm divides the dung beetle population into four different species of dung beetles based on the different divisions of labour of individual dung beetles, namely the

The associate editor coordinating the review of this manuscript and approving it for publication was P. K. Gupta.

ball-rolling dung beetle, the spawning ball, the little dung beetle, and the stealing dung beetle. In order to validate the performance of the DBO algorithm, [13] uses a number of well-known test functions to evaluate DBO algorithm, including 23 benchmark functions and 29 CEC-2017 test functions [14]. In addition, the Wilcoxon signed-rank test [15] was used to evaluate the experimental results of the algorithm, which proved that the dung beetle optimization algorithm is very competitive with the current well-known optimization algorithms in terms of convergence speed, solution accuracy and stability. To further illustrate the practical application potential of the DBO algorithm, the DBO algorithm is successfully applied to three engineering design problems. The experimental results show that the proposed DBO algorithm can effectively handle practical application problems. Since the release of DBO, it has been applied to PV array fault diagnosis [16], lung cancer detection and classification [17], distribution network restructuring [18], wood thermal modification prediction [19], air quality prediction [20], and drone path planning [21], amongst others, due to its good performance and stability.

Although DBO has been successfully used in many different fields due to its superior performance, it still has the common drawbacks of optimization algorithms, i.e., it is prone to falling into local optima, suffers from weak global exploration, and shows slow convergence to a solution. Although the DBO algorithm was proposed recently, many scholars have improved it for these problems. For example, [22] combines the improved sine algorithm (MSA) [23], tent chaotic mapping and mutation operator to improve the DBO algorithm and apply it to solve engineering design problems; [16] improved DBO algorithm by introducing a variety of swarming mechanisms, and they propose an IDBO-LSTM model applied to the diagnosis of photovoltaic array faults; [18] proposes an improved multi-objective dung beetle optimization algorithm, which uses a variable spiral search strategy to enhance the search range and convergence accuracy of the DBO algorithm; [21] proposes a multi-strategy enhanced DBO algorithm incorporating Beta distribution and crossover operators and uses it for 3D UAV navigation; [19] uses the segmented linear chaos mapping (PWLCM), adaptive linear decreasing producer and dimension learning enhanced foraging (DLF) search strategies to improve the DBO algorithm.

This paper is dedicated to improving the traditional DBO algorithm for its shortcomings and enhancing the overall performance of the algorithm. Therefore, a Multi-Strategy fusion Improved Dung Beetle Optimization (MSIDBO) algorithm is proposed. In this paper, the original DBO algorithm is improved from three perspectives respectively. Firstly, the Good Point Set [24] method is introduced to generate an initial population to make the distribution of the population more even and to increase the population diversity; secondly, the position update formula is optimized in the exploration phase by combining the golden-sine and the t-distribution perturbation strategy, which enhances the algorithm's searching ability and convergence accuracy and expands the searching range; Finally, the adaptive Gaussian-Cauchy mutation operator is introduced to enhance the ability of the algorithm to jump out of the local optimal.

II. THE DUNG BEETLE OPTIMIZATION ALGORITHM (DBO)

The dung beetle optimization algorithm was designed to simulate various behaviours of dung beetles, including ball rolling, dancing, foraging, breeding and stealing behaviours. Based on these behaviours, five position update rules were designed respectively, and the dung beetle population was divided into four types of agent dung beetles according to their different behaviours, specifically, ball-rolling dung beetles, spawning balls, small dung beetles, and stealing dung beetles, each constituting a certain proportion of the dung beetle population.

A. THE BALL-ROLLING DUNG BEETLES

The dung beetle needs to be navigated through the sun as it rolls to ensure that the ball of dung rolls in a straight line. The intensity of sunlight also has an effect on the movement path of the dung beetle. During the rolling process, the position update formula of the ball-rolling dung beetle is denoted as:

$$x_i(t+1) = x_i(t) + \alpha \times k \times x_i(t-1) + b \times \Delta_x,$$
$$\Delta_x = |x_i(t) - X^w| \tag{1}$$

where t denotes the current number of iterations, $x_i(t)$ denotes the t^{th} iteration of the i^{th} dung beetle, $k \in [0, 0.2]$ denotes a deflection coefficient that primarily controls the degree of deviation of the dung beetle during its rolling motion, $b \in [0, 1]$ is a constant that represents the influence of changes in sunlight intensity on the dung beetle's movement. In original paper on DBO the k and b were set to 0.1 and 0.3. α is a natural coefficient, which is used to simulate that dung beetles deviate from the original route due to natural factors. When $\alpha = 1$, it indicates that there is no deviation. When $\alpha = -1$, it indicates deviation and the value of α is 1 or -1 with equal probability, X^w denotes the global worst solution. Δ_x is used for simulating the intensity of sunlight, a higher value of Δ_x indicates a weaker light source.

When the dung beetle encounters an obstacle, it will change direction by dancing to obtain a new path. The dancing action is simulated using a tangent function, which takes into account only the interval $[0, \pi]$ as its domain. Once the dung beetle has successfully determined a new direction, it should continue to roll the ball forward. Consequently, the position of the rolling dung beetle is updated and defined as:

$$x_i(t+1) = x_i(t) + \tan(\theta)|x_i(t) - x_i(t-1)| \tag{2}$$

The θ in Eq. (2) belongs to the angle of inclination of $[0, \pi]$, and $|x_i(t) - x_i(t-1)|$ is the difference between the position of i^{th} dung beetle at the t^{th} iteration and the position at the $t-1^{th}$ iteration. Thus, the position update of the ball-rolling dung beetle is closely related to the current and historical information.

B. THE SPAWNING BALL

The dung beetle rolls the dung ball to a safe location and conceals it, proceeding to lay its eggs thereafter. Locating the right spawning area is crucial for the reproduction of the dung beetle population. In the original paper on DBO, a boundary selection strategy was proposed to simulate the dung beetle's behavior in searching for a spawning area. This strategy takes into account not only the inherent boundaries of the optimization problem but also the dynamic changes in the current local optimal solution, thereby more realistically simulating the dung beetle's natural process of selecting the best spawning ground.

As the population iterates and evolves, the boundaries of the optimal spawning area undergo corresponding changes. This is because as the search process deepens, the algorithm gradually approaches the global optimal solution, and the boundaries of the spawning area gradually narrow down, focusing on more promising solution spaces. This dynamic boundary adjustment not only improves the search efficiency of the algorithm but also enhances its adaptability, making it more suitable for the demands of real-world problems. The specific definition of the boundary selection strategy is as follows:

$$\begin{aligned} Lb^* &= \max(X^* \times (1-R), Lb), \\ Ub^* &= \min(X^* \times (1+R), Ub) \\ R &= 1 - t/T_{\max} \end{aligned} \tag{3}$$

The X^* in Eq. (3) denotes the local optimal solution, Lb^* and Ub^* represent the upper and lower bounds of the spawning region, $T_{\max}$ represents the maximum number of iterations, R controls the dynamic change of the spawning area as the population iterates, and Lb and Ub denote the lower and upper bounds of the optimization problem, respectively; The spawning region of the dung beetle population changes dynamically with population iterations.

Once the population of dung beetles has found an optimal spawning area, female dung beetles lay their eggs on hatching balls in this area, laying only one egg per iteration. Because the optimal spawning area for dung beetles changes dynamically with population iterations, the location of the hatching balls also changes dynamically. This dynamic change can be defined as:

$$\begin{aligned} B_i(t+1) = X^* &+ b_1 \times (B_i(t) - Lb^*) \\ &+ b_2 \times (B_i(t) - Ub^*) \end{aligned} \tag{4}$$

The $B_i(t)$ in Eq. (4) denotes the positional information of i^{th} spawning ball at t^{th} iteration, and b_1 and b_2 denote two independent stochastic vectors of size $1 \times D$, where D denotes the dimension of the optimization problem.

C. THE LITTLE DUNG BEETLE

The little dung beetles that hatch out of the spawning ball will burrow out of the ground as adults in search of food, and the little dung beetles need to find food in the optimal foraging area, which is defined as:

$$\begin{aligned} Lb^b &= \max(X^b \times (1-R), Lb), \\ Ub^b &= \min(X^b \times (1+R), Ub) \\ R &= 1 - t/T_{\max} \end{aligned} \tag{5}$$

where X^b denotes the global optimal solution, Lb^b and Ub^b refer to the lower and upper bounds of the optimal foraging area. The $T_{\max}$ denotes the maximum number of iterations, and Lb and Ub refer to the lower and upper bounds of the optimization problem, respectively. The foraging area of the dung beetle population, like the spawning area, changes dynamically as the population iterates. The position update formula for the small dung beetle is as follows:

$$\begin{aligned} x_i(t+1) = x_i(t) &+ C_1 \times (x_i(t) - Lb^b) + \\ &C_2 \times (x_i(t) - Ub^b) \end{aligned} \tag{6}$$

where $x_i(t)$ denotes the position of i^{th} small dung beetle at the t^{th} iteration, C_1 denotes a random number obeying a normal distribution, and C_2 denotes a random vector with each element between 0 and 1.

D. THE STEALING DUNG BEETLE

In a dung beetle population, some individual dung beetles steal dung balls from other dung beetles, and these dung beetles are called stealing dung beetles. From Eq. (5), X^b is the best foraging area, so the area around X^b can be set up as the best area to compete for food. The position update formula for the stealing dung beetle can be defined as:

$$x_i(t+1) = X^b + S \times g \times (|x_i(t) - X^*| + |x_i(t) - X^b|) \tag{7}$$

where $x_i(t)$ denotes the position information of i^{th} stealing dung beetle at t^{th} iteration, g is a random vector of size $1 \times D$ obeying a normal distribution, and S is a constant.

III. THE MULTI-STRATEGY FUSION IMPROVED DUNG BEETLE OPTIMIZATION ALGORITHM(MSIDBO)

A. THE MOTIVATION FOR IMPROVEMENT

Although the DBO algorithm has been proven to be very competitive with well-known optimization algorithms in terms of convergence speed, solution accuracy and stability after several test experiments, it still suffers from the defects of low convergence accuracy, weak global exploration ability and easy-to-fall into local optimization. Therefore, in order to improve these defects of the DBO algorithm, this paper proposes the MSIDBO algorithm, which improves the original DBO algorithm by means of the Good Point Set [24] strategy, the Golden Sine [25] strategy, the t-distribution perturbation strategy, and the adaptive Gaussian-Cauchy mutation.

B. THE GOOD POINT SET

The merit of the initial population has an important impact on the search performance of the population. The original DBO algorithm initializes its population randomly, which

may result in an insufficiently uniform initial distribution and reduce the diversity of the population. This can make the algorithm prone to falling into local optima and may affect both the convergence speed and the global search capability of the algorithm. Therefore, choosing a population initialization strategy different from random distribution can effectively improve the diversity of the population and promote a more uniform distribution of the population within the search range.

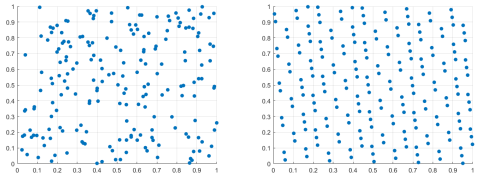
The Good Point Set was proposed by Hua and Wang [24] in 1978, and has been used by many scholars as a strategy for population initialization. Reference [26] demonstrates that the population generated by adopting the good point set strategy has a more uniform distribution, which can effectively improve the performance of the algorithm. Therefore, the good point set strategy is used to initialize the DBO population in this paper. The basic definition and construction of the good point set are as follows: Let G_s be the unit cube in s-dimensional Euclidean space. If $r \in G_s$, which can be expressed as:

$$p_n(k) = \{(r_1^n, r_2^n, \ldots, r_m^n) \times k\}, k = 1, 2, \ldots, n \quad (8)$$

$$\Phi(n) = C(r, \epsilon)n^{(-1+\epsilon)} \quad (9)$$

where $\Phi(n)$ represents the deviation of Eq. (8), $C(r, \epsilon)$ is a constant that depends solely on r and ϵ, ϵ is an arbitrary positive number, n is the number of points in the population, P_n^k is the set of good points, and r stands for a good point.

In this paper, we use the random method and the good point set method to generate a 2D initial population for comparison, as shown in Fig. 2, with the same population size, the initial population generated by the good point set method is more uniformly distributed than that generated by the random method and has higher stability. In contrast, the distribution of the initial population generated by the random method is disordered and even overlapping. Therefore, initializing the population using the good point set method greatly enhances the quality of the initial population and lays a solid foundation for the subsequent iterative search process.



(a) Random population (b) Goodpoint set of population

FIGURE 1. Comparison chart of initialized populations.

C. THE GOLDEN SINE STRATEGY

Golden sine algorithm (Golden-SA) is a meta-inspired algorithm proposed by Tanyildizi and Demir [25] in 2017. It employs the sine function to traverse the entire unit circle based on the angular relationship between the sine function and the unit circle. The Golden Ratio coefficient is utilized to reduce the solution space, thereby obtaining a superior search area and enhancing both local exploitation ability and solving accuracy. The position update formula of the golden sine algorithm is as follows:

$$\begin{aligned} V_i^{t+1} &= V_i^t |\sin(r_1)| - r_2 sin(r_1)|y_1 D_i^t - y_2 V_i^t| \\ y_1 &= a \times (1-\tau) + b \times \tau \\ y_2 &= a \times \tau + b * (1-\tau) \end{aligned} \quad (10)$$

where r_1 is a random number within $[0, 2\pi]$; r_2 is a random number within $[0, \pi]$; V_i^t represents the position of the i^{th} individual at the t^{th} iteration, τ is the golden number defined as $\frac{1-\sqrt{5}}{2}$, y_1 and y_2 are the Golden Ratio coefficients that aim to balance the search and development processes. The initial values of a and b are $-\pi$ and π, respectively.

In the DBO algorithm, dung beetles adjust their direction by performing a dance-like behavior when they encounter obstacles during ball rolling. Once they have determined a new rolling direction for the ball, they continue to move forward. The dance-like behavior of dung beetles is simulated using a tangential function. In this paper, the golden sine strategy is employed to reduce the solution space and replaces the dance-like behavior of the rolling dung beetle when encountering an obstacle. When the rolling dung beetle encounters an obstacle, it modifies its current position by implementing the golden sine strategy, locating its position closer to the optimal solution. Subsequently, the rolling dung beetle continues its forward movement, effectively enhancing the local mining ability of the DBO algorithm. The specific position update formula of the golden sine strategy is as follows:

$$\begin{aligned} x_i(t+1) = x_i(t)|\sin(r_1)| - r_2 sin(r_1)|y_1 x_{best}(t) \\ - y_2 x_i(t)| \end{aligned} \quad (11)$$

where $x_i(t)$ is the position of i th dung beetle at t th iteration, $x_{best}(t)$ is the optimal solution at t th iteration, and the rest of the variables are defined by the Eq. (10).

D. THE T-DISTRIBUTION STRATEGY

The t-distribution [27], also known as the student distribution, contains a parameter degree of freedom n, and its curve shape is related to the size of the degree of freedom n. In this paper, we perturb the positions of egg-laying dung beetles, small dung beetles, and stealing dung beetles with the current number of iterations as the parameter of the degrees of freedom.

The improved position update formula for the spawning ball is defined as:

$$\begin{aligned} B_i(t+1) = X^* + (b_1 \times (B_i(t) - Lb^*) \\ + b_2 \times (B_i(t) - Ub^*)) \times t(iter) \end{aligned} \quad (12)$$

The improved position update formula for the little dung beetle is defined as:

$$\begin{aligned} x_i(t+1) = x_i(t) + (C_1 \times (x_i(t) - Lb^b) \\ + C_2 \times (x_i(t) - Ub^b)) \times t(iter) \end{aligned} \quad (13)$$

The improved position update formula for the stealing dung beetle is defined as:

$$x_i(t+1) = X^b + (S \times g \times (|x_i(t) - X^*| + |x_i(t) - X^b|)) \times t(iter) \quad (14)$$

where *iter* is the current number of iterations, as the iteration count *iter* increases, the t-distribution gradually converges to the Gaussian distribution. This ensures that the algorithm exhibits good global exploration capabilities in early iterations and strong local exploitation abilities in later iterations, thereby enhancing the convergence speed of the algorithm.

E. THE ADAPTIVE GAUSSIAN-CAUCHY MUTATION STRATEGY

In the later iterations of the algorithm, the population tends to converge around the position of the optimal individual. If this optimal position is not the theoretical global optimum, the algorithm may become stuck in a local optimum. Therefore, to reduce the possibility of the population falling into a local optimum, this paper introduces the Gaussian-Cauchy mutation strategy to randomly perturb the optimal dung beetle individual. Furthermore, research in the reference [22] has demonstrated that Cauchy mutation helps enhance the algorithm's global search ability, while Gaussian mutation aids in improving its local search capability. Inspired by the work in reference [28], this paper introduces adaptive weights into the Gaussian-Cauchy mutation strategy. The specific definition of the adaptive Gaussian-Cauchy mutation strategy is as follows:

$$xs = x_{best}(t) + x_{best}(t) \times (w_1 \times C + w_2 \times G) \quad (15)$$

where xs is the result after the mutation of the optimal solution at the t^{th} iteration, $x_{best}(t)$ is the optimal solution at the t^{th} iteration, T_{max} is the maximum number of iterations. C is a random variable that satisfies the Cauchy distribution, representing the Cauchy mutation operator. G is a random variable that satisfies the Gaussian distribution, representing the Gaussian mutation operator. w_1 is the weight for the Cauchy mutation operator, defined as $1 - \frac{t}{T_{max}}$. w_2 is the weight for the Gaussian mutation operator, defined as $\frac{t}{T_{max}}$.

In the early stages of the algorithm iteration, when individuals in the population are more dispersed, assigning a higher weight to the Cauchy mutation operator is beneficial for generating large step sizes to jump out of the current position, thereby improving the global search ability of the population. In the later stages of the algorithm iteration, assigning a higher weight to the Gaussian mutation operator allows for small-step perturbations around the optimal individual position, which in turn helps enhance the local search ability of the population.

The adaptive Gaussian-Cauchy mutation strategy does not ensure that the location of each mutation is better than before. Therefore, this paper adopts the greedy strategy and only retains solutions with better fitness values after position change, which is defined as follows:

$$x_{best}(t) = \begin{cases} xs & xs \le x_{best}(t) \\ x_{best}(t) & xs > x_{best}(t) \end{cases} \quad (16)$$

F. THE STEPS OF MSIDBO ALGORITHM

The steps of the Multi-Strategy fusion Improved Dung Beetle Optimization (MSIDBO) algorithm are as follows.

Step 1: Set the basic parameters of MSIDBO;

Step 2: Initialize the population using the good point set method, Eq. (8);

Step 3: Calculate the fitness value of each individual in the initial population and select the global optimal position based on the fitness value;

Step 4: Calculate the position of the ball-rolling dung beetle based on Eq. (1). If it deviates from the path, execute the golden sine strategy according to Eq. (11);

Step 5: Update the local optimal solution;

Step 6: Update the positions of spawning balls, little dung beetles and stealing dung beetles according to the Eq. (12)-(14), respectively;

Step 7: Calculate and record the fitness of each individual based on the updated population location information to update the global optimal solution;

Step 8: Execute the adaptive Gaussian-Cauchy mutation strategy to the optimal individual's position based on Eq. (15), and select the retained solution according to Eq. (16);

Step 9: If the current iteration number reaches the maximum iteration number, then execute Step 10, otherwise continue to jump to Step 4;

Step 10: End the MSIDBO algorithm and output the global optimal solution x_{best} and its fitness $fitness_b$.

The flow of the MSIDBO algorithm can be seen in Fig. 2.

G. TIME COMPLEXITY AND COMPUTING TIME ANALYSIS

The mean value and standard deviation of each test function of the algorithm mentioned above show that the algorithm has better optimization accuracy and stability, respectively. The time complexity and computing time are also important evaluation indexes of algorithm performance.

The time complexity of the algorithm is expressed by O notation. Let the size of dung beetle population be N, the maximum number of iterations be M, and the dimension of the optimization problem be D. The time complexity of initialization operation for each dung beetle individual is $O(N \times D)$, each iteration will update the position of each individual in the population, and the time complexity is $O(M \times N \times D)$. After each iteration, fitness value will be used to evaluate whether to replace the best dung beetle individual. Time complexity is $O(M \times N)$. The overall complexity of the DBO algorithm is $O(N \times D)+O(M \times N \times D)+O(M \times N)$, which can be simplified to $O(M \times N \times D)$. The improvement strategy of MSIDBO algorithm does not increase the number of cycles, so the overall time complexity is still $O(M \times N \times D)$.

The calculation time of MSIDBO and DBO algorithm was tested by F1 in 23 benchmark test functions, the population

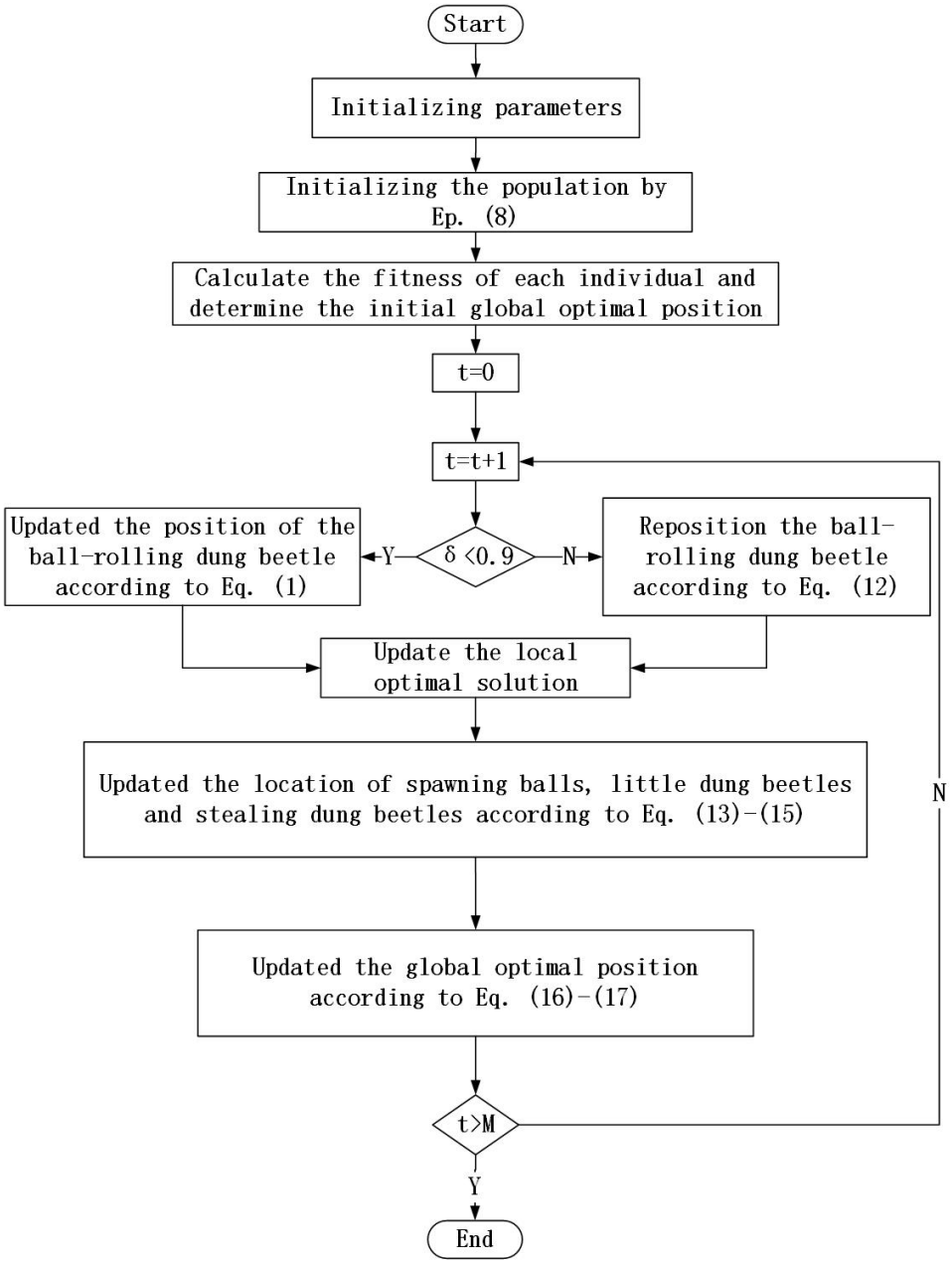


FIGURE 2. MSIDBO algorithm flow chart.

size of the two algorithms was set to 30, and the maximum number of iterations was 500. The two algorithms were run independently for 10 times, and the average value was taken as the evaluation standard of the algorithm calculation time. The calculation time test results of the two algorithms, with the data units in seconds, can be found in Tab. 1. The average calculation time of MSIDBO is 0.1627 seconds, and that of DBO algorithm is 0.1417 seconds. The average computing time of the two algorithms is close. This shows that MSIDBO can improve the optimization accuracy and stability without causing too much time overhead, and further proves that the overall performance of MSIDBO algorithm is higher than that of the original DBO algorithm.

IV. SIMULATION EXPERIMENT AND RESULT ANALYSIS

In [13], 23 benchmark functions and CEC-2017 test functions were used to test the optimization ability of the DBO algorithm. Therefore, these two sets of test functions are still used in this paper to verify the performance of the MSIDBO algorithm. For the 23 benchmark functions, the maximum number of iterations is set to 500, and the dimension settings are depicted in Tab. 3. For the 29 CEC-2017 test functions, the maximum number of iterations is set to 10000 and the dimensions are set to 10. The parameter settings of HGS algorithm are obtained from [7], and the parameters of other algorithms come from [13]. The specific parameter settings are detailed in Tab. 2. To ensure the fairness of the experiment,

TABLE 1. Computing time of MSIDBO and DBO.

Run times	MSIDBO (seconds)	DBO (seconds)
1	0.1683	0.1659
2	0.1646	0.1329
3	0.1584	0.1471
4	0.1584	0.1343
5	0.1582	0.1287
6	0.1654	0.1524
7	0.1635	0.1428
8	0.1606	0.1372
9	0.1618	0.1383
10	0.1675	0.1369
Mean	0.1627	0.1417

all algorithms will be run on the same hardware and software environment. The experimental environment was an AMD Ryzen 5 2500U CPU @ 2.00 GHz, with 8.00 GB of RAM, running Windows 10, and Matlab R2020a. Each algorithm will be executed independently 30 times, and the average value and standard deviation will be used as the evaluation criteria for the algorithm's solution accuracy and stability, respectively.

The 23 benchmark functions are commonly used in the field of intelligent optimization algorithms to evaluate their performance, as shown in Tab. 3. Among them, F1-F7 are high-dimensional single-peak functions, F8-F13 are high-dimensional multi-peak functions, and F14-F23 are fixed-dimensional multi-peak functions. In the experiments conducted with these 23 benchmark test functions, the population size of all algorithms is set to 30, and the maximum number of iterations is set to 500.

The 29 CEC-2017 test functions include F1 and F3 as unimodal functions, F4-F10 as simple multimodal functions, F11-F20 as hybrid functions, and F21-F30 as composition functions. Function F2 has been removed from the CEC-2017 test functions set. In the CEC-2017 test experiments, the population size of all algorithms is set to 30, the dimension is set to 10, and the maximum iteration number is set to 10000.

For a more rigorous comparison, the Wilcoxon signed-rank test is used to determine whether there is a significant difference in performance between the MSIDBO algorithm and the other optimization methods. The p-value is calculated to assess the statistical significance of the difference between the algorithms. If $p < 0.05$, then it indicates that there is a significant difference between the two algorithms.

A. COMPARISON WITH CLASSICAL OPTIMIZATION ALGORITHMS

1) COMPARISON ON 23 BENCHMARK FUNCTIONS

In this section, five classical intelligent optimization algorithms are chosen to compare with the MSIDBO algorithm proposed in this paper. They are Dung Beetle Optimization Algorithm (DBO) [13], Grey Wolf Optimization Algorithm (GWO) [2], Particle Swarm Optimization Algorithm (PSO) [1], Whale Optimization Algorithm (WOA) [4], and Harris Hawk Optimization Algorithm (HHO) [3]; The algorithm parameters are set according to Tab. 2.

TABLE 2. Parameter settings for various algorithms.

Algorithm	Parameter	Value
MSIDBO	k	0.1
	b	0.3
	S	0.5
DBO	k	0.1
	b	0.3
	S	0.5
GWO	amin and amax	0 and 2
PSO	vmax	6
	c1 and c2	2 and 2
	w	1
WOA	a1	[2,0]
	a2	[-2,-1]
	b	1
HHO	Interval of E0	[-1,1]
HGS	1 and LH	0.08 and 100
SSA	Leader position update probability	0.5
	v0	0
DE	scalingfactor	0.5
	crossoverprobability	0.5
SCA	a	2

From Tab. 4, it can be seen that the search of the MSIDBO algorithm is better than the DBO algorithm in 17 of the test functions, and equal to the DBO algorithm in the rest of the test functions. First of all, there is only one extreme point in the high-dimensional single-peak functions F1-F7, which mainly tests the local convergence ability of the algorithm, and comparing the mean and standard deviation, we can see that the accuracy of the MSIDBO algorithm is ranked in the first place except F5, which finds the optimal solution in the function of F1-F4, and the accuracy of the function is weaker than the first algorithm in the function of F5 in the second place, and it is superior to the DBO algorithm in the function of F1-F7. This fully reflects the advantage of using the good point set method to generate the initial population, which generates a more evenly distributed and more diverse initial population, so the local fast convergence ability of MSIDBO is better than that of DBO and also in the leading level compared with other classical algorithms; there are multiple local optima in the high-dimensional multi-peak functions F8-F13, which is mainly to test the global optimality search and the ability of the algorithm to jump out of the local optima. MSIDBO is ranked first in F9-F12, which reflects the advantages of introducing the golden sine guidance strategy as well as the t-distribution perturbation strategy and the mutation operator strategy, which makes the ability of the global searching and jumping out of the local optimum of the MSIDBO algorithm improve tremendously; in the fixed-dimensional multi-peak function F14-F23, except for F14, which is ranked second, the rest of the algorithms are all ranked first. In summary, the MSIDBO algorithm improves greatly in high-dimensional single-peak and high-dimensional multi-peak functions and still has greater competitiveness in fixed-dimensional multi-peak functions.

In order to demonstrate more intuitively the effectiveness of MSIDBO and the remaining five traditional intelligent

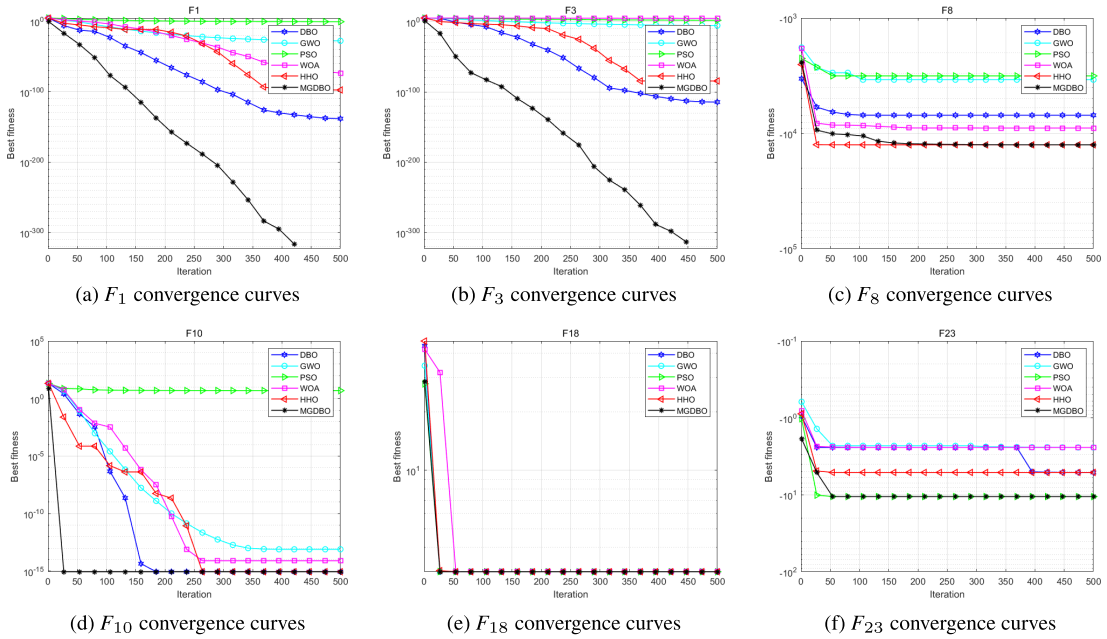
TABLE 3. 23 Benchmark test function.

ID	Function Equation	Dim	Range	fmin
F1	$\sum_{i=1}^{n} x_i^2$	30	[-100,100]	0
F2	$\sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[-10,10]	0
F3	$\sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	30	[-100,100]	0
F4	$\max\{\lvert x_i \rvert, 1 \le i \le n\}$	30	[-100,100]	0
F5	$\sum_{i=1}^{n-1} [100(x_{i-1} - x_i^2)^2 + (x_i - 1)^2]$	30	[-30,30]	0
F6	$\sum_{i=1}^{n} (x_i + 0.5)^2$	30	[-100,100]	0
F7	$\sum_{i=1}^{n} i x_i^4 + \text{random}[0, 1]$	30	[-1.28,1.28]	0
F8	$\sum_{i=1}^{n} -x_i sin(\sqrt{\lvert x_i \rvert})$	30	[-500,500]	-12569.5
F9	$\sum_{i=1}^{n} \lvert x_i^2 - 10\cos(2\pi x_i) + 10 \rvert$	30	[-5.12,5.12]	0
F10	$-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos(x\pi x_i)) + 20 + e$	30	[-32,32]	0
F11	$\frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$	30	[-600,600]	0
F12	$\frac{\pi}{n}\{10sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10sin^2(\pi y_{i+1}) + (y_n - 1)^2]\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$	30	[-50,50]	0
F13	$0.1\{sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + sin^2(3\pi x_i)] + (x_n - 1)^2[1 + sin^2(2\pi x_n)]\}$ $+ \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	[-50,50]	0
F14	$(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{\sum_{i=1}^{n}(x_i - a_j)^6})^{-1}$	2	[-65,65]	1
F15	$\sum_{i=1}^{11}[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}]^2$	4	[-5,5]	0.0003
F16	$4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[-5,5]	-1.0316
F17	$(x_2 - \frac{5.1}{4\pi^2} + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos x_1 + 10$	2	[-5,5]	0.398
F18	$[30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$ $[1 + (x_1 + x_1 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]$	2	[-2,2]	3
F19	$-\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{n} a_{ij}(x_j - p_{ij})^2)$	3	[0,1]	-3.8628
F20	$-\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{n} a_{ij}(x_j - p_{ij})^2)$	6	[0,1]	-3.32
F21	$-\sum_{i=1}^{5}[(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	[0,10]	-10.1532
F22	$-\sum_{i=1}^{7}[(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	[0,10]	-10.4028
F23	$-\sum_{i=1}^{10}[(x - a_i)(x - a_i)^T + c_i]^{-1}$	4	[0,10]	-10.5364

TABLE 4. Results of MSIDBO, DBO, GWO, PSO, WOA, and HHO on 23 benchmark test function.

ID	Fmin	MSIDBO		DBO		GWO		PSO		WOA		HHO	
		AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
F1	0	0	0	6.53E-111	3.58E-110	1.81E-27	2.93E-27	0.3971	0.2324	8.62E-75	2.76E-74	1.55E-95	8.16E-95
F2	0	0	0	4.62E-52	2.53E-51	9.63E-17	5.45E-17	6.8200	3.0401	2.39E-52	8.75E-52	9.20E-52	3.79E-51
F3	0	0	0	2.25E-85	1.23E-84	3.84E-05	1.62E-04	49.4307	20.5661	43565.2189	14254.2623	4.41E-78	1.69E-77
F4	0	0	0	2.95E-53	1.61E-52	5.08E-07	4.27E-07	4.2793	1.5365	55.3487	28.2725	2.27E-45	1.24E-44
F5	0	20.6488	0.7045	25.7695	0.2248	26.7586	0.7388	289.0633	275.2683	27.9948	0.4630	0.0078	0.0121
F6	0	1.68E-19	6.37E-19	0.0010	0.0018	0.7776	0.4233	0.4127	0.2099	0.4608	0.2370	1.18E-04	1.34E-04
F7	0	1.17E-05	1.43E-05	0.0016	0.0010	0.0020	0.0013	0.4232	1.9492	0.0024	0.0027	1.26E-04	1.40E-04
F8	-12569.5	-11307.5	1105.7	-8142.4	1758.8	-5752.7	938.7	-2999.1	474.4	-10143.7	1839.1	-12568.7	1.3
F9	0	0	0	0	0	2.7396	3.2092	74.9908	19.1257	0	0	0	0
F10	0	8.88E-16	0	1.01E-15	6.49E-16	1.04E-13	1.40E-14	4.7013	1.0808	4.91E-15	2.42E-15	8.88E-16	0
F11	0	0	0	0	0	0.0019	0.0050	0.5993	0.1736	0	0	0	0
F12	0	9.59E-09	5.25E-08	2.38E-04	0.0011	0.0403	0.0176	2.9416	1.4696	0.0237	0.0171	1.35E-05	1.87E-05
F13	0	0.0251	0.0394	0.6462	0.3908	0.5509	0.2252	9.3461	13.1642	0.5160	0.3050	1.58E-04	2.72E-04
F14	1	1.3288	0.7056	1.1303	0.8476	5.5604	4.6659	2.0219	1.2829	2.0791	2.4796	1.3612	0.9520
F15	0.0003	0.0003	2.54E-06	8.11E-04	3.71E-04	0.0017	0.0051	0.0019	0.0051	8.61E-04	0.0011	3.42E-04	2.95E-05
F16	-1.0316	-1.0316	6.65E-16	-1.0316	6.05E-16	-1.0316	1.70E-08	-1.0316	5.30E-16	-1.0316	3.73E-09	-1.0316	2.08E-09
F17	0.398	0.398	0	0.398	0	0.398	7.80E-07	0.398	0	0.398	1.52E-05	0.398	7.85E-06
F18	3	3	1.85E-15	3	3.64E-15	3	4.33E-05	3	3.30E-15	3.9004	4.9307	3	1.47E-06
F19	-3.8628	-3.8623	0.0020	-3.8617	0.0027	-3.8617	0.0021	-3.8609	0.0034	-3.8503	0.0228	-3.8597	0.0055
F20	-3.32	-3.28	0.04	-3.23	0.11	-3.23	0.10	-3.18	0.16	-3.26	0.08	-3.10	0.11
F21	-10.1532	-10.1531	2.96E-04	-7.2289	2.7127	-9.2289	2.1362	-5.9724	3.3731	-7.7894	2.6486	-5.2063	0.8488
F22	-10.4028	-10.4028	2.96E-04	-8.1093	2.9069	-10.1466	1.3940	-8.6993	3.1672	-7.3136	2.9799	-5.4176	1.2677
F23	-10.5364	-10.5364	3.53E-05	-8.5819	2.8883	-10.5346	9.33E-04	-8.2952	3.5414	-6.9449	3.2302	-5.1233	0.0061

optimization algorithms in finding the optimal result, this paper selects two functions from each of the 23 benchmark test functions to demonstrate. The population size is set to 30, and the number of iterations is set to 500. From the convergence curves shown in Fig. 3, it can be seen that MSIDBO makes up for the defects of the DBO algorithm in that it easily falls into the local optimum and slow convergence speed, and has a faster convergence speed and



(a) F_1 convergence curves (b) F_3 convergence curves (c) F_8 convergence curves

(d) F_{10} convergence curves (e) F_{18} convergence curves (f) F_{23} convergence curves

FIGURE 3. Comparison chart of convergence of 6 classical optimization algorithm.

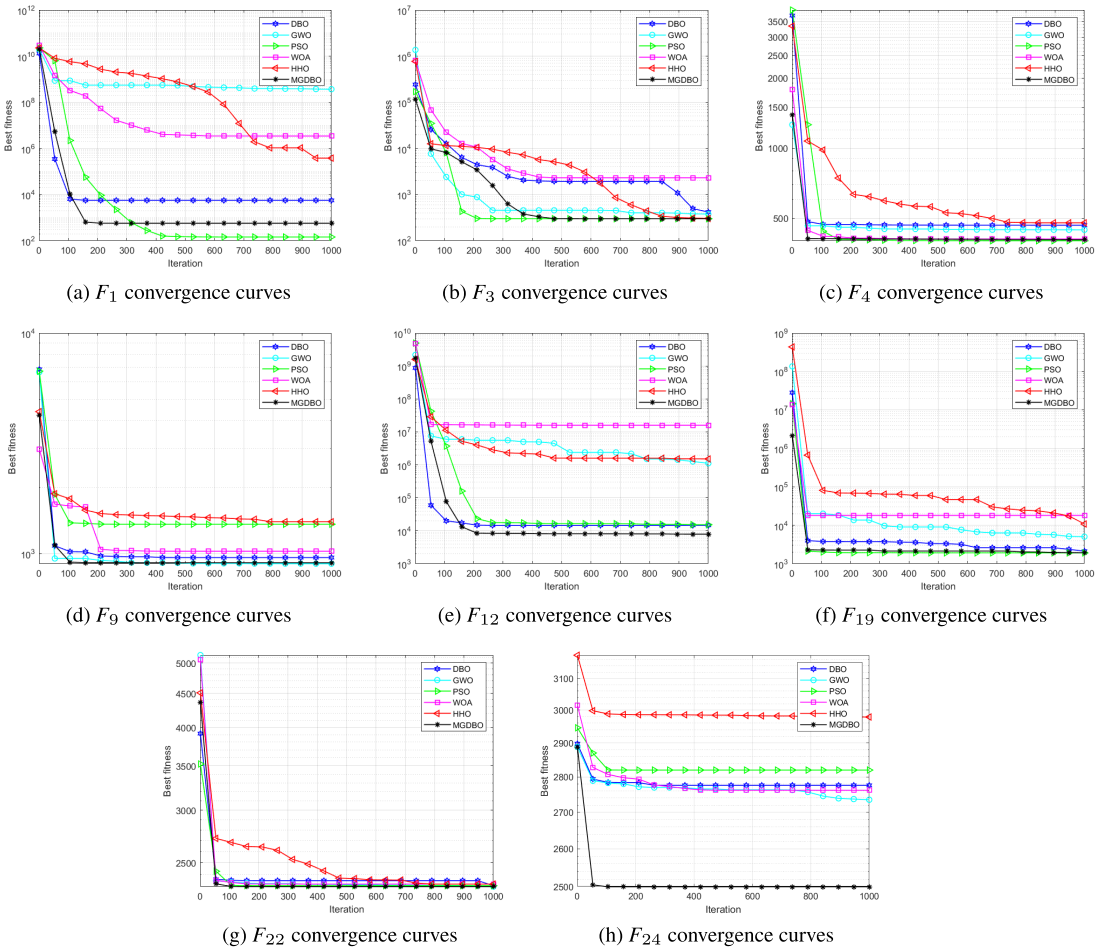
TABLE 5. Results of MSIDBO, DBO, GWO, PSO, WOA, and HHO on CEC-2017 test function.

ID	Fmin	MSIDBO		DBO		GWO		PSO		WOA		HHO	
		AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
F1	100	1635.6111	1969.4843	6511.4883	4533.0051	40825311.8064	122905565.5087	230.2326	374.2233	5978.8602	7145.5445	133577.2604	44296.5120
F3	300	300	0	300	0	1514.9132	1735.6006	300	0	353.9785	56.6499	300.4292	0.1543
F4	400	400	0	408.8351	12.6778	412.5718	15.3554	402.0384	3.7940	419.0653	32.2647	409.4953	17.0739
F5	500	521.9762	6.6717	529.0671	11.5025	514.2453	8.3826	548.2221	17.5893	543.0082	19.8805	540.9892	15.4708
F6	600	604.1206	3.7864	605.0396	4.3992	600.7567	0.9587	629.1999	9.0584	624.9646	11.4711	617.5445	11.6529
F7	700	750.2899	15.5095	743.1755	16.8891	725.8294	7.8464	753.6723	18.0149	777.5618	18.2441	759.0553	16.8274
F8	800	823.4797	7.0902	823.2117	9.6747	813.9806	6.0607	824.5423	12.0542	837.7896	14.3170	827.9779	7.4699
F9	900	913.7022	47.5154	922.0497	37.5812	911.6634	21.9531	1158.0330	194.8548	1224.7682	266.7086	1223.3772	244.5281
F10	1000	1608.3103	184.1716	1753.0015	291.5177	1538.5880	291.7981	2135.2683	307.5211	1976.2573	353.2707	1849.0553	266.5152
F11	1100	1146.8861	53.7463	1153.8141	56.1606	1124.3598	27.0445	1130.3726	15.4809	1192.3256	88.5318	1152.2710	54.8691
F12	1200	12274.0006	6380.1605	1017113.2130	2527967.7633	417228.5515	596786.2851	10095.5312	14615.2258	2295784.4114	3460397.6773	431727.8257	576682.9502
F13	1300	1725.1796	276.5568	11592.3529	12324.2612	8788.0735	4672.8731	1725.3125	463.0602	19098.1617	11301.6100	15003.1451	11327.9028
F14	1400	1481.5294	34.8552	1497.9804	44.5202	1832.4466	1123.3758	1453.2522	25.6492	1500.2601	30.3641	1492.4358	17.5831
F15	1500	1545.2092	44.7423	1677.9869	155.7860	2664.4608	1539.4300	1556.8848	38.1652	2198.2288	733.8283	1597.7453	57.8825
F16	1600	1707.3260	91.8406	1708.4252	100.9134	1691.3620	93.3570	1931.6774	106.5095	1775.1443	106.8068	1877.7312	112.3947
F17	1700	1743.6069	14.2149	1749.7150	19.3479	1741.4411	16.0232	1767.9658	57.7608	1774.8031	39.3827	1763.0495	34.5706
F18	1800	1978.8559	244.9041	15383.9400	14994.7780	26972.5515	12184.5776	2976.9659	5983.8746	16571.2705	11271.7644	14259.0233	9794.6645
F19	1900	1937.1496	39.4487	2014.3187	186.9641	3481.6018	3915.4157	1925.9143	23.6586	12747.1611	11180.4787	5672.4224	5819.1269
F20	2000	2035.1214	18.8071	2064.1010	40.6264	2069.0190	51.0274	2122.4368	54.3895	2113.9240	56.7011	2127.7811	77.1074
F21	2100	2201.9464	1.4317	2203.2419	1.3820	2309.1250	20.9289	2312.3529	64.3768	2285.6255	72.3959	2310.3913	69.6053
F22	2200	2302.2530	12.2229	2307.8631	6.4898	2322.3763	86.3422	2364.8715	236.6866	2362.6484	290.9609	2386.2618	224.7818
F23	2300	2618.9296	53.1318	2630.4188	11.6658	2612.6091	7.2758	2732.4703	59.0952	2642.7834	16.3989	2666.2478	22.0435
F24	2400	2504.0280	22.0624	2599.8996	126.2447	2742.5739	9.2229	2808.8873	127.1187	2763.8339	53.9170	2757.2876	120.6285
F25	2500	2924.6524	23.4910	2914.3396	64.1312	2931.6438	16.5741	2919.6063	63.9885	2937.9858	24.9978	2933.5826	36.4639
F26	2600	2974.6434	65.0461	3044.4027	105.0789	3053.6066	321.7734	3467.3214	486.7282	3272.6919	487.6469	3306.4199	548.2832
F27	2700	3094.6660	2.8653	3098.9625	8.7777	3099.3574	15.9978	3238.1575	61.5016	3118.3995	34.5653	3135.8759	35.4339
F28	2800	3295.4424	140.2688	3241.1026	118.0135	3387.9049	78.1473	3288.7017	140.6839	3397.3013	202.6768	3352.8162	140.2755
F29	2900	3175.5286	24.3219	3196.2487	40.0086	3181.9714	43.8868	3299.6887	91.0403	3328.6253	88.0956	3260.7446	72.9357
F30	3000	405204.7130	512886.8929	889870.5814	1194957.3641	504094.1025	812962.4750	97385.2837	291050.7265	281687.3993	494049.6108	373659.0554	662873.4422

higher convergence accuracy, which proves the effectiveness of the improved strategy.

2) COMPARISON ON CEC-2017 TEST FUNCTIONS

To further demonstrate the performance of the MSIDBO algorithm, this section compares five well-known meta-heuristic algorithms, including the classical DBO algorithm, with the MSIDBO algorithm among the 29 test functions in the CEC-2017 test functions. The experimental results are shown in Tab. 5. In 25 of the test functions, the accuracy of the MSIDBO algorithm is superior to that of the original DBO algorithm. For the remaining 4 functions, all of them yield similar results to the original DBO. Among the single-peak functions, MSIDBO performs best in F1 and second best in F2. Among the multi-peak functions, MSIDBO outperforms the original DBO algorithm in most cases, except for F7, and ranks among the top three in all tested functions. F11-F30 are hybrid and combination functions, which are typically used to evaluate the balance between algorithm exploitation and exploration. From Tab. 5, it can be seen that MSIDBO ranks first in F13, F15, F17, F18, F20-F22, F24, F26, F27, and F29, with the remaining func tions ranking in the top three. This



(a) F_1 convergence curves (b) F_3 convergence curves (c) F_4 convergence curves

(d) F_9 convergence curves (e) F_{12} convergence curves (f) F_{19} convergence curves

(g) F_{22} convergence curves (h) F_{24} convergence curves

FIGURE 4. Comparison chart of convergence of classical optimization algorithm on CEC-2017 test functions.

analysis of the standard deviation indicates that the MSIDBO algorithm also exhibits good stability. In conclusion, this demonstrates that the MSIDBO algorithm possesses a strong ability to handle complex problems and achieves a good balance between exploitation and exploration.

In order to demonstrate more intuitively the effectiveness of MSIDBO and the remaining five traditional intelligent optimization algorithms in finding the optimal results, two of each type from CEC-2017 test functions are selected for demonstration in this paper. The population size is set to 30, and the number of iterations is set to 1000. As can be seen in Fig. 4, the MSIDBO algorithm converges the fastest in most of the functions, and even though it converges a little slower in some of the functions, the search accuracy catches up to the rest of the algorithms as the number of iterations increases. The superior performance of MSIDBO in hybrid and combinatorial functions also shows that the MSIDBO algorithm has a good balance between development and exploration.

B. COMPARISON WITH IMPROVED ALGORITHMS

In this section, the MSIDBO algorithm and five improved algorithms—including the dung beetle optimization algorithm guided by improved sine algorithm (MSADBO) [22], the improved dung beetle optimization algorithm (IDBO) [16], the improved multi-objective dung beetle optimizer (IMODBO) [18], the elite opposition-based learning and t-Distribution hunger games search algorithm (EtHGS) [29], and the Multi-strategy improved adaptive dynamic whale optimization algorithm (MSIWOA) [30]—are used for comparative optimization search experiments. The experimental data for MSADBO was obtained by replicating from [22]. Similarly, the data for IDBO was obtained by replicating from [16], the data for IMODBO was obtained by replicating from [18], the data for EtHGS was obtained by replicating from [29], and the data for MSIWOA was obtained by replicating from [30].

1) COMPARISON ON 23 BENCHMARK FUNCTIONS

The experimental results of the MSIDBO algorithm and the five improved algorithms on 23 benchmark test functions are shown in Tab. 6.

From Tab. 6, the MSIDBO algorithm demonstrates excellent accuracy and stability in high-dimensional

TABLE 6. Results of MSIDBO, MSADBO, IDBO, IMODBO, EtHGS and MSIWOA on 23 benchmark function.

ID	Fmin	MSIDBO		MSADBO		IDBO		IMODBO		EtHGS		MSIWOA	
		AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
F1	0	0	0	0	0	0	0	3.21E-223	0	0	0	0	0
F2	0	0	0	0	0	0	0	3.28E-114	1.26E-113	7.56E-177	0	0	0
F3	0	0	0	0	0	0	0	3.23E-213	0	0	0	0	0
F4	0	0	0	0	0	0	0	1.71E-110	6.61E-110	1.46E-169	0	0	0
F5	0	20.6488	0.7045	25.1227	0.2678	0.0213	0.0582	27.1875	0.3234	25.1269	0.5519	0.0101	0.0204
F6	0	1.68E-19	6.37E-19	2.32E-05	3.02E-05	1.64E-07	2.67E-07	1.1457	0.5999	0.6159	0.3212	0.0001	0.0003
F7	0	1.17E-05	1.43E-05	7.84E-05	0.0001	8.50E-05	7.32E-05	0.0004	0.0004	0.0002	0.0001	8.16E-05	7.92E-05
F8	-12569.5	-11307.5	1105.7	-8697.5	1633.4	-9834.3	1296.2	-11699.9	1266.7	-12125.8	378.4	-12522.3	72.6
F9	0	0	0	0	0	0	0	0	0	0	0	0	0
F10	0	8.88E-16	0	8.88E-16	0	8.88E-16	0	8.88E-16	0	8.88E-16	0	8.88E-16	0
F11	0	0	0	0	0	0	0	0	0	0	0	0	0
F12	0	9.59E-09	5.25E-08	1.54E-06	4.10E-06	4.17E-08	8.37E-08	0.0526	0.0498	0.0131	0.0195	2.81E-06	6.44E-06
F13	0	0.0251	0.0394	0.7799	0.3946	0.0080	0.0142	0.2037	0.1221	0.4184	0.3528	4.10E-05	0.0001
F14	1	1.3288	0.7056	1.5551	1.8462	1.5855	2.1791	1.2964	0.6283	2.0407	2.9792	1.8914	0.8778
F15	0.0003	0.0003	2.54E-06	0.0007	0.0004	0.0005	0.0003	0.0004	6.48E-05	0.0005	0.0002	0.0006	0.0004
F16	-1.0316	-1.0316	6.65E-16	-1.0316	6.12E-16	-1.0316	5.90E-16	-1.0316	4.23E-05	-1.0316	6.71E-16	-1.0316	1.52E-10
F17	0.398	0.398	0	0.398	0	0.398	0	0.398	0	0.398	0	0.411	0.040
F18	3	3	1.85E-15	3.9	4.9295	3	3.23E-15	3.0021	0.0022	3	4.05E-15	3	4.80E-10
F19	-3.8628	-3.8628	0.0020	-3.8617	0.0027	-3.8628	2.55E-15	-3.8561	0.0044	-3.8628	2.65E-15	-3.6867	0.2215
F20	-3.32	-3.28	0.04	-3.28	0.09	-3.25	0.06	-3.28	0.04	-3.28	0.06	-2.86	0.33
F21	-10.1532	-10.1532	0.0003	-10.0839	0.1955	-10.1512	0.0040	-9.6130	0.5774	-10.1532	0.0006	-10.1094	0.1319
F22	-10.4028	-10.4028	0.0003	-10.3910	0.0323	-10.4010	0.0031	-9.8428	0.7290	-10.4028	0.0009	-10.3774	0.0552
F23	-10.5364	-10.5364	3.53E-05	-10.4679	0.2877	-10.2220	1.4342	-9.8857	0.8148	-10.5364	4.06E-05	-10.4658	0.1524

TABLE 7. Results of MSIDBO, MSADBO, IDBO, IMODBO, EtHGS and MSIWOA on CEC-2017 test functions.

ID	Fmin	MSIDBO		MSADBO		IDBO		IMODBO		EtHGS		MSIWOA	
		AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
F1	100	1635.6111	1969.4843	2770.1449	2553.2606	1969.4252	1916.8529	22785573.2270	8956710.8619	2470.8320	1678.8973	306767063.3866	235601380.3960
F3	300	300	0	300	0	300	0	460.6426	90.9145	306.1629	31.4278	2092.8951	1229.8055
F4	400	400	0	405.7233	21.9660	400.0272	0.0492	415.2446	20.3894	400.1549	0.0257	470.3139	41.8510
F5	500	521.9762	6.6717	525.2652	8.5090	527.6209	12.7599	532.8486	10.4455	523.0008	4.9502	572.2093	19.4478
F6	600	604.1206	3.7864	604.4000	4.2640	605.1307	3.9579	604.1671	1.0931	604.2514	0.0000	641.9202	11.2421
F7	700	750.2899	15.5095	742.2342	12.1884	753.1124	18.6555	749.4303	7.7739	726.3938	8.0667	800.3557	19.6792
F8	800	823.4797	7.0902	822.6557	6.2985	827.8302	7.6932	825.9319	5.1142	811.4420	4.1210	836.5835	8.5363
F9	900	913.7022	47.5154	958.1437	85.3873	968.4219	91.3041	910.3964	16.1328	906.0331	32.6163	1536.0852	254.7793
F10	1000	1608.3103	184.1716	1774.8514	321.8956	1807.5611	300.8365	1879.3306	214.0432	1632.2432	193.9470	2127.2888	240.8917
F11	1100	1146.8861	53.7463	1171.1517	44.2970	1151.4871	42.7751	1137.9883	24.1240	1109.1381	5.8452	1228.0103	85.2627
F12	1200	12274.0006	6380.1605	18160.0814	18817.0024	10341.1522	12211.8534	1929473.3121	1924607.0278	16167.7769	16014.5593	6210936.2115	4666672.9839
F13	1300	1725.1796	276.5568	6072.4410	5457.6103	7298.2729	6230.0158	11743.2394	6947.4432	6247.6246	6412.1374	17444.1642	9173.7452
F14	1400	1481.5294	34.8552	1492.0148	46.9135	1477.3764	34.5178	1525.5296	22.2382	1483.6049	28.1295	1746.1559	202.2488
F15	1500	1545.2092	44.7423	1602.8613	77.9869	1581.5027	69.1623	1821.1328	215.0746	1696.5236	451.1661	8813.7587	2719.4398
F16	1600	1707.3260	91.8406	1747.3175	130.7422	1679.4228	75.0503	1681.8408	52.7125	1691.4615	110.8161	2002.9525	133.8732
F17	1700	1743.6069	14.2149	1746.5725	16.4014	1743.8860	21.1570	1756.3383	9.7958	1751.9149	16.8110	1778.2969	25.0934
F18	1800	1978.8559	244.9041	5430.5167	8327.1750	3166.5236	3398.6876	19354.4984	17669.4600	10270.0742	8579.6409	2766.2804	2489.7435
F19	1900	1937.1496	39.4487	1973.0269	100.0095	1955.3104	55.3213	2241.2115	251.0121	2318.7159	1896.6527	31846.5298	65957.3592
F20	2000	2035.1214	18.8071	2054.6766	44.8937	2065.0372	40.7706	2061.2875	20.4163	2004.8082	7.5775	2239.0458	67.7207
F21	2100	2201.9464	1.4317	2201.6353	1.4045	2201.8526	2.0489	2204.3400	1.7127	2264.5734	61.5386	2227.6744	34.6221
F22	2200	2302.2530	12.2229	2303.9266	2.9532	2299.8834	21.4024	2309.0168	16.6217	2296.0396	20.1662	2598.8030	529.3550
F23	2300	2618.9296	53.1318	2633.1393	11.4000	2631.9054	14.5071	2627.1180	4.5751	2620.0141	7.3503	2667.4272	26.0021
F24	2400	2504.0280	22.0624	2537.5186	85.9575	2543.8587	99.4120	2568.0934	91.4116	2736.3098	81.4144	2813.4216	35.7772
F25	2500	2924.6524	23.4910	2925.5515	24.7746	2916.2494	64.1873	2924.6259	22.7239	2923.9600	23.9968	2982.3352	30.8142
F26	2600	2974.6434	65.0461	3009.8564	141.7861	3040.4106	147.4256	2905.1894	84.9644	2917.2137	85.0857	3829.9478	539.8920
F27	2700	3094.6660	2.8653	3097.8738	6.7829	3098.5809	5.9208	3098.4171	7.9590	3105.5114	13.2870	3208.8258	70.9205
F28	2800	3295.4424	140.2688	3211.6280	127.3166	3256.4256	134.4814	3335.9496	117.8120	3225.8126	143.4856	3523.4240	96.1169
F29	2900	3175.5286	24.3219	3207.2918	42.9489	3196.4875	38.7764	3199.7180	35.8416	3188.5565	36.6978	3447.5801	78.6925
F30	3000	405204.7130	512886.8929	341907.6320	504972.1196	486682.7756	710978.1173	345618.5520	300013.0989	182187.1328	383820.9766	681298.3780	1171332.5944

single-peaked functions F1-F7. Specifically, it ranks first on F1-F4, F6-F7, F9-F12, and F15-F23. Furthermore, the global optimal solution has been found on F1-F4, F9, F11, F16-F19, and F21-F23. Additionally, MSIDBO performs competitively with several other improved algorithms on F5, F8, F13, and F14, often outperforming them. Overall, MSIDBO holds a significant advantage over the other five improved algorithms.

2) COMPARISON ON CEC-2017 TEST FUNCTIONS

The comparison data of the six improved algorithms on the CEC-2017 test functions are shown in Tab. 7, in which MSIDBO ranks first on 15 test functions, and the algorithms that are slightly weaker than the first algorithm rank in the top three on 9 test functions. This again proves that MSIDBO is still competitive compared to the improved algorithm. Analyzing the standard deviation, it becomes evident that

TABLE 8. Results of Wilcoxon signed-rank test on 23 benchmark functions.

ID	DBO	GWO	PSO	WOA	SSA	DE	SCA	HHO	Total
F1	1.21E-12	1.21E-12	1.21E-12	1.21E-12	5.77E-11	1.21E-12	1.21E-12	1.21E-12	8
F2	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	8
F3	1.21E-12	1.21E-12	1.21E-12	1.21E-12	4.57E-12	1.21E-12	1.21E-12	1.21E-12	8
F4	1.21E-12	1.21E-12	1.21E-12	1.21E-12	4.57E-12	1.21E-12	1.21E-12	1.21E-12	8
F5	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	8
F6	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	8
F7	3.02E-11	3.02E-11	3.02E-11	4.50E-11	1.78E-10	3.02E-11	3.02E-11	2.67E-09	8
F8	2.39E-08	3.02E-11	3.02E-11	2.07E-02	4.74E-06	1.47E-07	3.02E-11	3.02E-11	8
F9	NaN	1.20E-12	1.21E-12	NaN	NaN	1.21E-12	1.21E-12	NaN	4
F10	3.34E-01	1.14E-12	1.21E-12	3.57E-10	NaN	1.21E-12	1.21E-12	NaN	5
F12	3.34E-11	3.02E-11	3.02E-11	3.02E-11	5.57E-10	3.02E-11	3.02E-11	6.07E-11	8
F13	9.92E-11	3.02E-11	3.02E-11	3.02E-11	2.06E-01	1.00E+00	3.02E-11	9.94E-01	5
F14	7.60E-02	9.93E-10	8.79E-06	1.35E-06	6.69E-12	5.58E-03	3.29E-07	1.21E-05	7
F15	4.07E-11	9.91E-11	2.49E-06	4.50E-11	1.20E-10	3.02E-11	3.02E-11	6.69E-11	8
F16	9.55E-05	2.36E-12	5.91E-07	2.36E-12	7.75E-10	5.70E-01	2.36E-12	2.36E-12	7
F17	NaN	1.21E-12	NaN	1.21E-12	2.79E-03	NaN	1.21E-12	1.21E-12	5
F18	6.67E-04	1.27E-11	1.75E-07	1.27E-11	1.04E-09	7.99E-03	1.27E-11	1.48E-11	8
F19	6.54E-03	1.54E-09	4.83E-06	2.15E-10	1.05E-09	8.15E-02	6.09E-12	1.05E-09	7
F20	6.73E-03	1.02E-01	6.98E-01	3.72E-03	1.01E-02	1.76E-11	7.51E-11	2.51E-04	6
F21	5.82E-06	7.38E-11	3.62E-02	3.02E-11	4.51E-02	9.94E-01	3.02E-11	3.02E-11	7
F22	6.09E-02	1.46E-10	2.79E-02	3.33E-11	9.62E-02	2.39E-01	3.02E-11	3.02E-11	5
F23	2.44E-01	2.89E-11	4.02E-01	2.89E-11	6.44E-05	2.04E-01	2.89E-11	2.89E-11	5

MSIDBO maintains the first place on 12 functions and is among the top three on another 9 functions. This underscores the significant advantage of MSIDBO over the other improved algorithms, both in terms of accuracy and stability.

C. WILCOXON SIGNED-RANK TEST

In our previous experiments, we analyzed MSIDBO along with the classical algorithm and the improved version of DBO on 23 benchmark functions and the CEC-2017 test functions. MSIDBO demonstrated a significantly superior performance in both test sets, whether comparing accuracy or stability with other algorithms. This fully reflects the effectiveness of the strategy employed by the MSIDBO algorithm. To further compare the differences between MSIDBO and other algorithms, we conducted Wilcoxon rank and non-parametric statistical tests in this paper, based on the experimental data of MSIDBO on both the 23 benchmark test functions and the CEC-2017 test functions. When the p-value is less than 0.05 for an algorithm, it indicates that MSIDBO is significantly different from that algorithm. In this paper, eight well-known intelligent optimization algorithms were selected: DBO [13], GWO [2], PSO [1], SSA [5], DE [8], SCA [9], and HHO [3]. The p-values of the statistical results are shown in Tab. 8 and Tab. 9. The last column displays the total number of data points with significant differences.

From Tab. 8, it can be seen that the data of MSIDBO in 70% of the functions are significantly different from those of the traditional DBO. This indicates that the similarity between the search results of the MSIDBO and DBO algorithms based on the 23 benchmark functions is low. When comparing MSIDBO with other algorithms, it is evident that the search results of the MSIDBO algorithm differ significantly from those of most other algorithms on the majority of functions. Analysis of Tab. 9 further reveals that the search results of MSIDBO are significantly different from those of the DBO algorithm on 58.6% of the functions, and they also stand out as distinct from the results of other optimization algorithms on most functions. Taken together, these analyses demonstrate that, among the various metaheuristic algorithms, the MSIDBO algorithm exhibits the most outstanding comprehensive performance.

V. ENGINEERING DESIGN PROBLEMS

In order to test the potential of MSIDBO algorithm in practical engineering application, this paper uses the proposed MSIDBO algorithm to optimize three engineering design problems. The reference [31] puts forward a benchmark suite containing 57 constraint optimization problems to

TABLE 9. Results of Wilcoxon signed-rank test on CEC-2017 test functions.

ID	DBO	GWO	PSO	WOA	SSA	DE	SCA	HHO	Total
F1	2.10E-05	8.20E-07	4.69E-08	4.12E-06	3.02E-11	1.21E-12	3.02E-11	3.02E-11	8
F3	3.84E-01	2.24E-11	2.24E-11	2.24E-11	2.24E-11	1.09E-07	2.24E-11	2.24E-11	7
F4	3.02E-11	3.02E-11	4.22E-04	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	8
F5	5.20E-03	3.18E-04	1.43E-08	2.68E-06	5.07E-10	3.02E-11	1.07E-09	6.52E-07	8
F6	4.73E-01	1.11E-06	4.98E-11	1.17E-09	3.02E-11	1.21E-12	5.07E-10	2.57E-07	7
F7	1.26E-01	6.53E-08	6.10E-01	3.01E-07	3.02E-11	3.02E-11	3.50E-03	3.64E-02	6
F8	5.35E-01	2.32E-06	8.65E-01	2.28E-05	8.89E-10	3.02E-11	2.77E-05	1.84E-02	6
F9	1.71E-01	4.23E-03	1.29E-09	9.26E-09	3.02E-11	4.57E-12	8.12E-04	3.35E-08	7
F10	3.51E-02	2.06E-01	1.56E-08	5.97E-05	1.55E-09	3.34E-11	3.82E-09	2.13E-04	7
F11	4.92E-01	8.77E-02	8.88E-01	6.10E-03	2.39E-08	4.50E-11	7.62E-03	2.97E-01	4
F12	3.18E-04	1.85E-08	1.95E-03	1.31E-08	1.78E-10	1.33E-10	3.02E-11	5.49E-11	8
F13	2.87E-10	4.98E-11	4.38E-01	4.50E-11	6.70E-11	5.57E-10	3.02E-11	3.82E-10	7
F14	1.54E-01	3.26E-01	4.22E-04	3.03E-02	2.37E-10	1.27E-11	1.86E-03	6.79E-02	5
F15	1.61E-06	6.05E-07	9.63E-02	4.98E-11	3.02E-11	3.02E-11	1.29E-09	1.32E-04	7
F16	7.17E-01	4.04E-01	2.60E-08	9.07E-03	9.92E-11	3.02E-11	7.01E-02	2.15E-06	5
F17	4.38E-01	2.40E-01	4.36E-02	2.32E-06	3.02E-11	2.57E-11	6.74E-06	1.17E-02	6
F18	5.27E-05	3.02E-11	1.54E-01	4.50E-11	3.69E-11	6.07E-11	3.02E-11	6.07E-11	7
F19	5.83E-03	6.00E-01	3.40E-01	3.69E-11	3.02E-11	1.21E-12	2.44E-09	1.29E-09	6
F20	1.78E-04	2.25E-04	6.72E-10	1.70E-08	3.69E-11	5.14E-12	4.31E-08	9.06E-08	8
F21	3.24E-03	2.13E-10	1.10E-06	7.29E-11	2.98E-11	1.08E-10	4.91E-11	3.55E-06	8
F22	4.06E-02	2.84E-01	3.34E-03	1.61E-06	3.02E-11	5.57E-10	6.74E-06	9.76E-10	7
F23	4.38E-01	1.11E-06	5.49E-11	2.13E-04	4.50E-11	5.57E-10	1.25E-07	2.23E-09	7
F24	1.25E-06	2.04E-11	3.06E-11	2.26E-11	2.04E-11	3.06E-11	3.06E-11	3.39E-11	8
F25	9.65E-01	6.63E-01	2.71E-01	1.77E-03	3.33E-11	1.99E-05	8.23E-02	8.88E-01	3
F26	4.27E-04	7.50E-01	7.56E-05	3.52E-04	5.30E-11	5.77E-06	6.48E-04	1.62E-01	6
F27	1.86E-03	8.53E-01	3.02E-11	1.25E-07	3.02E-11	1.69E-09	2.61E-10	9.91E-11	7
F28	8.99E-01	5.54E-05	9.05E-01	1.67E-04	3.60E-09	7.00E-01	6.05E-02	8.95E-02	3
F29	2.81E-02	7.96E-01	3.47E-10	1.29E-09	3.34E-11	2.15E-10	7.30E-04	1.60E-07	7
F30	9.66E-03	4.21E-02	9.05E-02	1.81E-01	1.28E-09	9.23E-01	1.15E-01	9.33E-02	4

test the practical application potential of the algorithm. In this paper, MSIDBO algorithm is used to search and test three well-known optimization problems. Respectively, the pressure vessel design problem (PVD) [32], welded beam design problem (WBD) [33] and tension/compression spring design problem (TSCD) [34], and the optimization results are compared with the results of other classical optimization algorithms. The test method is to convert these three engineering problems into a mathematical model, use penalty function to deal with inequality constraints, and then use each algorithm to find the optimal solution. Set the fill size to 30 and the maximum number of iterations to 500.

A. PRESSURE VESSEL DESIGN PROBLEMS (PVD)

The main object of the pressure vessel design problem is to optimize the welding cost, material, and forming of a vessel. This problem refers to finding pressure vessel design parameters that satisfy the constraints, and using these design parameters to calculate the manufacturing cost of the pressure vessel. The pressure vessel design parameters are shell thickness, head thickness, inner radius, and length of the vessel without including the head. In this section, the optimization results of the MSIDBO algorithm are used to compare with the results of DBO [13], DE [8], GWO [2], HHO [3], PSO [1], SCA [9], SSA [5] and WOA [4].

The specific formulation of the pressure vessel design problem (PVD) is as follows:

$$\begin{aligned} \min f(s) &= 0.6224 s_1 s_3 s_4 + 1.7781 s_2 s_3^3 \\ &\quad + 3.1661 s_1^2 s_4 + 19.84 s_1^2 s_3 \\ g_1 &= -s_1 + 0.0193 s_3 \le 0, \end{aligned}$$

$$
\begin{aligned}
&g_2 = -s_2 + 0.00954 s_3 \le 0,\\
&g_3 = -\pi s_3^2 s_4 - \frac{4}{3}\pi s_3^3 + 1296000 \le 0,\\
&g_4 = s_4 - 240 \le 0\\
&s_i \in [0, 99](i = 1, 2), s_i \in [0, 200](i = 3, 4)
\end{aligned} \tag{17}
$$

where s_1 is the shell thickness, s_2 is the head thickness, s_3 is the inner radius, and s_4 is the length of the vessel without including the head, $g_i(i = 1, 2, 3, 4)$ is the four constraints. $f(s)$ is the fitness function and denote the manufacturing cost of the pressure vessel.

The optimization results of all the algorithms are shown in Tab. 10, among the optimal costs of all the algorithms, the MSIDBO algorithm has the lowest cost and the optimal parameter obtained is s = (0.778169, 0.384649, 40.31962, 200) and the optimal cost is f(s)=5885.332954.

TABLE 10. Result of PVD.

algorithm	s_1	s_2	s_3	s_4	Optimal cost	Rank
MSIDBO	0.7782	0.3846	40.3196	200.0000	5885.3330	1
DBO	0.7868	0.3874	40.3697	199.3149	5949.5803	3
DE	0.9634	0.4762	49.9179	98.9991	6283.5212	6
GWO	0.7792	0.3907	40.3738	199.4158	5906.9946	2
HHO	0.9056	0.4471	46.8555	125.4289	6145.8319	4
PSO	0.9634	0.4762	49.9176	99.0000	6283.3282	5
SCA	0.8217	0.5491	42.0625	180.7961	6566.8536	8
SSA	0.7803	0.6257	40.3196	200.0000	6597.7467	9
WOA	0.9288	0.5138	47.9584	115.4158	6436.8613	7

B. WELDED BEAM DESIGN PROBLEMS (WBD)

The main objective of this problem is to optimize the weight of the welded beam. The welded beam design problem is to find four design parameters that satisfy the constraints of shear stress, bending stress, bending load of the beam and terminal deviation. These design parameters are the length, height, thickness of the beam, and thickness of the welded joint, respectively. Finally, these design parameters are used to calculate the manufacturing cost of the welded beam. In this section, the results of the MSIDBO algorithm's optimization search are compared with those of DBO [13], GWO [2], HGS [7], HHO [3], SCA [9], SMA [6], SSA [5], and WOA [4]. The welded beam design problem (WBD) is formulated as follows:

$$
\begin{aligned}
&\min f(s) = 1.10471 s_1^2 s_2\\
&\qquad + 0.04811 s_3 s_4 (14.0 + s_2)\\
&g_1(s) = \tau(s) - 13600 \le 0,\\
&g_2(s) = \sigma(s) - 30000 \le 0,\\
&g_3(s) = \gamma(s) - 0.25 \le 0,\\
&g_4 = s_1 - s_4 \le 0, g_5 = p - p_c \le 0,\\
&g_6 = 0.125 - s_1 \le 0,\\
&g_7 = 1.10471 s_1^2 +\\
&\quad 0.04811 s_3 s_4 (14.0 + s_2) - 5.0 \le 0\\
&s_i \in [0.1, 2.0](i = 1, 4), s_i \in [0.1, 10.0](i = 2, 3),\\
&\tau = \sqrt{\tau_1^2 + 2\tau_1\tau_2(\frac{s_2}{2r}) + \tau_2^2}, \tau_1 = \frac{p}{s_1 s_2 \sqrt{2}}\\
&m = p(l + \frac{s_2}{2}), j = 2\{\sqrt{2} s_1 s_2 [\frac{s_2^2}{12} + (\frac{s_1 + s_2}{2})^2]\}\\
&r = \sqrt{\frac{s_2^2}{4} + (\frac{s_1 + s_3}{2})^2}, \sigma = \frac{6pl}{s_4 s_3^2}, \gamma = \frac{6pl^3}{E s_3^2 s_4},\\
&p_c = \frac{4.013E\sqrt{\frac{s_3^2 s_4^6}{36}}}{l^2}(1 - \frac{s_3}{2l}\sqrt{\frac{E}{4G}}),\\
&G = 12 \times 10^6 psi, E = 30 \times 10^6 psi\\
&p = 6000 lb, l = 14, \tau_2 = \frac{mr}{j}
\end{aligned} \tag{19}
$$

where s_1 is the thickness of the welded joint, s_2 is the length of the beam, s_3 is the height of the beam, s_4 is the thickness of the beam, g_i(i=1,2,...,7) is the seven constraints. τ is the shear stress, σ denotes the bending stress, p_c is the bending load of the beam, and γ is the terminal deviation. $f(s)$ is the fitness function and denotes the manufacturing cost of the welded beam.

The optimization results of all algorithms for the welded beam design problem (WBD) are shown in Tab. 11, the MSIDBO algorithm obtains the minimum cost in solving the welded beam design problem, the optimum parameter obtained is s=(0.205734,3.253036,9.036624,0.20573) and the minimum cost is 1.69524492.

TABLE 11. Results of WBD.

Algorithm	s1	s2	s3	s4	Optimal cost	Rank
MSIDBO	0.2057	3.2530	9.0366	0.2057	1.6952	1
DBO	0.1989	3.3826	9.0383	0.2058	1.7033	4
GWO	0.2013	3.3341	9.0376	0.2058	1.7000	3
HGS	0.2280	3.0132	8.5845	0.2280	1.7749	8
HHO	0.1928	3.4816	9.0993	0.2088	1.7411	6
SCA	0.2134	3.1893	8.8412	0.2175	1.7510	7
SMA	0.2030	3.3035	9.0356	0.2058	1.6981	2
SSA	0.1250	10.0000	10.0000	0.2014	2.4978	9
WOA	0.2058	3.4018	9.0320	0.2059	1.7164	5

C. TENSION/COMPRESSION SPRING DESIGN PROBLEMS (TSCD)

The main objective of this problem is to optimize the weight of a tension/compression spring. This problem involves finding the design parameters of the tension/compression spring that satisfy the constraints and calculating the weight of the spring through these parameters. The constraints include minimum deflection, vibration frequency, shear stress, and outer diameter limitations. The design parameters include the average diameter of the spring coil, the diameter of the spring wire, and the number of effective coils of the spring. In this section, the optimization results of the MSIDBO algorithm are compared with the results of DBO [13], DE [8], GWO [2], HGS [7], HHO [3], PSO [1], SCA [9], SMA [6], SSA [5], and WOA [4].

The specific formulation of the tension/compression spring design problem (TSCD) is as follows:

$$\begin{aligned}
&\min f(s) = (s_3 + 2)s_2 s_1^2 \\
&g_1(s) = 1 - \frac{s_3 s_2^3}{71785 s_1^4} \leq 0, \\
&g_2(s) = \frac{4s_2^2 - s_1 s_2}{12566(s_2 s_1^3 - s_1^4)} + \frac{1}{5108 s_1^2} - 1 \leq 0, \\
&g_3(s) = 1 - \frac{140.45 s_1}{s_2^2 s_3} \leq 0, \\
&g_4(s) = \frac{s_1 + s_2}{1.5} - 1 \leq 0 \\
&s_1 \in [0.05, 2], s_2 \in [0.25, 1.3], s_3 \in [2, 15] \qquad (20)
\end{aligned}$$

where s_1 is the average diameter of the spring coil, s_2 is the diameter of the spring wire, s_3 is the number of effective coils of the spring, $g_i(i = 1, 2, 3, 4)$ is the constraint. $f(s)$ is the fitness function and denotes the weight of the tension/compression spring.

The optimization results of all the algorithms for the Tension Compression Spring Design Problem (TSCD) are shown in Tab. 12. The MSIDBO algorithm is ranked first among all the algorithms in terms of finding the optimal parameter s=(0.05107,0.341998,12.20717) and the minimum cost of 0.012672312 is obtained.

TABLE 12. Result of TSCD.

Algorithm	s1	s2	s3	Optimal cost	Rank
MSIDBO	0.0511	0.3420	12.2072	0.0127	1
DBO	0.0500	0.3174	14.0278	0.0127	3
DE	0.0690	0.9334	2.0000	0.0178	10
GWO	0.0505	0.3292	13.1187	0.0127	2
HGS	0.0836	1.2068	2.3140	0.0364	11
HHO	0.0571	0.5008	6.0670	0.0132	5
PSO	0.0690	0.9334	2.0000	0.0178	9
SCA	0.0500	0.3107	15.0000	0.0132	7
SMA	0.0629	0.5950	5.3603	0.0173	8
SSA	0.0500	0.3104	15.0000	0.0132	6
WOA	0.0549	0.4376	7.7515	0.0128	4

VI. CONCLUSION AND FUTURE PERSPECTIVES

Aiming at the problems that the dung beetle optimization algorithm (DBO) cannot guarantee the diversity of the population, has poor ability to jump out of the local optimum, has poor convergence accuracy, and has weak global exploration ability, a multi-strategy fusion of the improved golden sine dung beetle optimization algorithm (MSIDBO) is proposed. The algorithm adopts a good point set strategy to make the population more evenly distributed within the search range, enhance the population diversity, and provide a good basis for the algorithm to find the optimal. The golden sine strategy is introduced, and the golden sine coefficient is used to complete the dancing action, which effectively strengthens the global exploration ability of the algorithm. The t-distribution strategy is introduced to perturb the egg-laying dung beetles, small dung beetles and stealing dung beetles, so as to expand the search range of the population. Finally, in order to prevent the population from falling into the local optimal, the adaptive Gauss-Cauchy mutation strategy is introduced to mutate the current optimal solution under certain conditions, thus improving the ability of the algorithm to jump out of the local optimal. These improvement strategies effectively improve the algorithm's convergence accuracy, global search capability, and ability to jump out of local optima.

In order to verify the effectiveness of the improved strategy, this paper presents simulation experiments with 23 benchmark test functions as well as CEC-2017 test functions and compares them with five well-known traditional optimization algorithms as well as five improved algorithms. The experimental results show that the MSIDBO algorithm is much improved in many aspects. This paper also verifies the significant difference between MSIDBO and traditional DBO as well as other traditional algorithms using the Wilcoxon signed-rank test based on the experimental data of 23 benchmark test functions and CEC-2017 test functions, and the experimental results show that MSIDBO is significantly different from the rest of the algorithms. In order to test the practical application potential of the MSIDBO algorithm, this paper uses the MSIDBO algorithm to perform the optimization test on three well-known engineering design problems and compares it with other well-known algorithms, and the results of the comparison show that the MSIDBO algorithm has a greater competitiveness in dealing with such problems. In the future, the MSIDBO algorithm will be applied to more practical and engineering problems, such as image segmentation, UAV path planning, data mining and so on.

REFERENCES

[1] R. E. J. Kennedy, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw.*, Perth, WA, Australia, Nov. 1995, pp. 1942–1948.

[2] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.

[3] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," *Future Gener. Comput. Syst.*, vol. 97, pp. 849–872, Aug. 2019.

[4] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016.

[5] J. Xue and B. Shen, "A novel swarm intelligence optimization approach: Sparrow search algorithm," *Syst. Sci. Control Eng.*, vol. 8, no. 1, pp. 22–34, Jan. 2020.

[6] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, "Slime mould algorithm: A new method for stochastic optimization," *Future Gener. Comput. Syst.*, vol. 111, pp. 300–323, Oct. 2020.

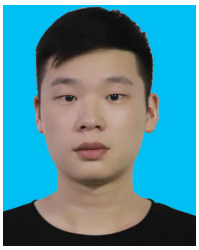
[7] Y. Yang, H. Chen, A. A. Heidari, and A. H. Gandomi, "Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts," *Expert Syst. Appl.*, vol. 177, Sep. 2021, Art. no. 114864.

[8] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[9] S. Mirjalili, "SCA: A sine cosine algorithm for solving optimization problems," *Knowl.-Based Syst.*, vol. 96, pp. 120–133, Mar. 2016.

[10] D.-H. Lee, J.-H. Ahn, and B.-H. Koh, "Fault detection of bearing systems through EEMD and optimization algorithm," *Sensors*, vol. 17, no. 11, p. 2477, Oct. 2017.
[11] C. Chong, M. H. Low, A. Sivakumar, and K. Gay, "A bee colony optimization algorithm to job shop scheduling," in *Proc. Winter Simulation Conf.*, Dec. 2006, pp. 1954–1961.
[12] A. Shabani, B. Asgarian, M. Salido, and S. Asil Gharebaghi, "Search and rescue optimization algorithm: A new optimization method for solving constrained engineering optimization problems," *Expert Syst. Appl.*, vol. 161, Dec. 2020, Art. no. 113698.
[13] J. Xue and B. Shen, "Dung beetle optimizer: A new meta-heuristic algorithm for global optimization," *J. Supercomput.*, vol. 79, no. 7, pp. 7305–7336, May 2023.
[14] G. Wu, R. Mallipeddi, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2017 competition on constrained real-parameter optimization," Nat. Univ. Defense Technol., Changsha, Hunan, PR China Kyungpook Nat. Univ., Daegu, South Korea Nanyang Technol. Univ., Singapore, Tech. Rep., 2017.
[15] R. F. Woolson, "Wilcoxon signed-rank test," in *Wiley Encyclopedia of Clinical Trials*. Hoboken, NJ, USA: Wiley, 2007, pp. 1–3.
[16] B. Li, P. Gao, and Z. Guo, "Improved dung beetle algorithm to optimize LSTM photovoltaic array fault diagnosis," *J. Power Syst. Autom.*, 2023.
[17] M. Alamgeer, N. Alruwais, H. M. Alshahrani, A. Mohamed, and M. Assiri, "Dung beetle optimization with deep feature fusion model for lung cancer detection and classification," *Cancers*, vol. 15, no. 15, p. 3982, Aug. 2023.
[18] N. Tu and Z. Fan, "IMODBO for optimal dynamic reconfiguration in active distribution networks," *Processes*, vol. 11, no. 6, p. 1827, Jun. 2023.
[19] R. Zhang and Y. Zhu, "Predicting the mechanical properties of heat-treated woods using optimization-algorithm-based BPNN," *Forests*, vol. 14, no. 5, p. 935, May 2023.
[20] J. Duan, Y. Gong, J. Luo, and Z. Zhao, "Air-quality prediction based on the ARIMA-CNN-LSTM combination model optimized by dung beetle optimizer," *Sci. Rep.*, vol. 13, no. 1, Jul. 2023.
[21] Q. Shen, D. Zhang, M. Xie, and Q. He, "Multi-strategy enhanced dung beetle optimizer and its application in three-dimensional UAV path planning," *Symmetry*, vol. 15, no. 7, p. 1432, Jul. 2023.
[22] J. Pan, S. Li, P. Zhou, g. Yang, and D. Lv, "Dung beettle optimization algorithm guided by improved sine algorithm," *Comput. Eng. Appl.*, vol. 59, pp. 92–110, Apr. 2023.
[23] S. Gupta and K. Deep, "Improved sine cosine algorithm with crossover scheme for global optimization," *Knowledge-Based Syst.*, vol. 165, pp. 374–406, Feb. 2019.
[24] L.-G. Hua and Y. Wang, *The Applications of Number Theory to Approximate Analysis*, vol. 10. Beijing, China: Science Press, 1978, pp. 83–87.
[25] E. Tanyildizi and G. Demir, "Golden sine algorithm: A novel math-inspired algorithm," *Adv. Electr. Comput. Eng.*, vol. 17, no. 2, pp. 71–78, 2017.
[26] D. Zhang, Y. Zhao, J. Ding, Z. Wang, and J. Xu, "Multi-strategy fusion improved adaptive hunger games search," *IEEE Access*, vol. 11, pp. 67400–67410, 2023.
[27] Y. Hu and H. Zhang, "Moth weighted centroid location algorithm based on T distribution variation," *Comput. Eng. Sci.*, vol. 34, no. 2, p. 5, 2012.
[28] P. J. Van Laarhoven, E. H. Aarts, P. J. van Laarhoven, and E. H. Aarts, *Simulated Annealing*. Cham, Switzerland: Springer, 1987.
[29] Y. Xu, S. Liu, W. Zhang, and Y. Liu, "Elite opposition-based learning and t-distribution hunger games search algorithm," *Comput. Simul.*, vol. 40, nos. 425–434, 2023.
[30] Z. Wang, Z. Dou, J. Dong, S. Si, and C. Wang, "Multi-strategy improved adaptive dynamic whale optimization algorithm," *Comput. Eng. Des.*, vol. 43, nos. 2638–2645, 2022.
[31] A. Kumar, G. Wu, M. Z. Ali, R. Mallipeddi, P. N. Suganthan, and S. Das, "A test-suite of non-convex constrained optimization problems from the real-world and some baseline results," *Swarm Evol. Comput.*, vol. 56, Aug. 2020, Art. no. 100693.
[32] E. Sandgren, "Nonlinear integer and discrete programming in mechanical design," in *Proc. 14th Design Autom. Conf.*, Sep. 1988, pp. 95–105.
[33] K. Ragsdell and D. Phillips, "Optimal design of a class of welded structures using geometric programming," Tech. Rep., 1976.
[34] A. D. Belegundu and J. S. Arora, "A study of mathematical programming methods for structural optimization. Part I: Theory," *Int. J. Numer. Methods Eng.*, vol. 21, no. 9, pp. 1583–1599, Sep. 1985.

DAMING ZHANG received the Ph.D. degree in resource information and decision-making from Northeastern University, China, in 2017. He is currently an Associate Professor and a Masters Supervisor with the School of Information Science and Engineering, Guilin University of Technology. His main research interests include resource information, decision making, and software engineering.



ZIJIAN WANG is currently pursuing the master's degree with the School of Information Science and Engineering, Guilin University of Technology, Guilin, China.

YANQING ZHAO is currently pursuing the master's degree with the School of Information Science and Engineering, Guilin University of Technology, Guilin, China.

FANGJIN SUN received the Ph.D. degree in engineering mechanics from Liaoning Technical University, in 2009. Currently, she works as a Professor and a Ph.D. Supervisor with the School of Civil Engineering and Architecture, Guilin University of Technology. Her main research interests include disaster prevention and reduction as well as intelligent optimization algorithms.