Mathematical Biosciences and Engineering

MBE, 20(7): 12263–12297.
DOI: 10.3934/mbe.2023546
Received: 09 December 2022
Revised: 15 March 2023
Accepted: 31 March 2023
Published: 19 May 2023

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

Research article

Hybrid multi-strategy chaos somersault foraging chimp optimization algorithm research

Xiaorui Yang[2,3], **Yumei Zhang**[1,2,3,*], **Xiaojiao Lv**[2,3], **Honghong Yang**[1,3], **Zengguo Sun**[1,2,3] **and Xiaojun Wu**[1,2,3]

[1] Key Laboratory of Modern Teaching Technology, Ministry of Education, Shaanxi Normal University, Xi'an, China
[2] School of Computer Science, Shaanxi Normal University, Xi'an, China
[3] Key Laboratory of Intelligent Computing and Service Technology for Folk Song, Xi'an, China

* **Correspondence:** Email: zym0910@snnu.edu.cn.

Abstract: To address the problems of slow convergence speed and low accuracy of the chimp optimization algorithm (ChOA), and to prevent falling into the local optimum, a chaos somersault foraging ChOA (CSFChOA) is proposed. First, the cat chaotic sequence is introduced to generate the initial solutions, and then opposition-based learning is used to select better solutions to form the initial population, which can ensure the diversity of the algorithm at the beginning and improve the convergence speed and optimum searching accuracy. Considering that the algorithm is likely to fall into local optimum in the final stage, by taking the optimal solution as the pivot, chimps with better adaptation at the mirror image position replace chimps from the original population using the somersault foraging strategy, which can increase the population diversity and expand the search scope. The optimization search tests were performed on 23 standard test functions and CEC2019 test functions, and the Wilcoxon rank sum test was used for statistical analysis. The CSFChOA was compared with the ChOA and other improved intelligent optimization algorithms. The experimental results show that the CSFChOA outperforms most of the other algorithms in terms of mean and standard deviation, which indicates that the CSFChOA performs well in terms of the convergence accuracy, convergence speed and robustness of global optimization in both low-dimensional and high-dimensional experiments. Finally, through the test and analysis comparison of two complex engineering design problems, the CSFChOA was shown to outperform other algorithms in terms of optimal cost. For the design of the speed reducer, the performance of the CSFChOA is 100% better than other algorithms in terms of optimal cost; and, for the design of a three-bar truss, the performance of the CSFChOA is 6.77% better than other algorithms in terms of optimal cost, which verifies the feasibility, applicability and superiority of the CSFChOA in practical engineering problems.

Keywords: chimp optimization algorithm; cat chaotic sequence; opposition-based learning; somersault foraging; convergence; local optimum

1. Introduction

Optimization problems are prevalent in a wide range of disciplines and social life tasks, such as resource scheduling, task allocation, path planning, etc. The core idea is to choose the best possible solution from feasible solutions which satisfy the constraints to make the objective sought be as optimal as possible. Most of the engineering research fields need to solve optimization problems with multi-dimensional, complex and large computational loads. Optimization problems have been explored in the pursuit of optimal goals in the long term. To a certain extent, traditional optimization algorithms provide reliable solutions to optimization problems in various fields based on the precise mathematical characteristics of the optimization problem, and they are dedicated to using definite structures and parameters to give optimal solutions. However, for complex optimization problems with non-differential objective and constraint functions and multiple peaks, traditional optimization methods are often complicated to calculate or difficult to yield satisfactory solutions. Swarm intelligence (SI) is an intelligent computing technique, which has been very popular in recent years. The term “swarm intelligence” was introduced by Beni and Wang [1] in the context of cellular robotic systems; Brezočnik et al. [2] consider the advantages of SI to include 1) autonomy, where each agent of the swarm controls its own behavior autonomously, and 2) self-organization, where intelligence appears in the swarm itself, so SI is extremely flexible in solving problems. In actual situations, optimization problems are increasingly characterized by dynamic changes, nonlinearity, uncertainty and high latitude. To effectively solve complex optimization problems in reality, researchers have proposed SI-based optimization algorithms inspired by the behavior of animal groups, such as particle swarm optimization [3], the artificial bee colony algorithm [4], cuckoo search [5], ant colony optimization [6], etc.

The chimp optimization algorithm (ChOA) was proposed by Khishe and Mosavi [7] in 2020; it is a heuristic intelligent optimization algorithm based on the social behavior of chimp populations, and it originates from the simulation of individual chimp intelligence, sexual motivation and predatory behavior in nature; it constructs an effective optimization solution through the processes of driving, chasing, blocking and attacking. For various intelligent optimization algorithms, many improvement strategies have been proposed by numerous scholars. Feng and Song [8] improved the performance of the whale optimization algorithm in the global exploration and local search phases by introducing nonlinear time-variant adaptive weights, introduced a differential mutation perturbation factor in the encircling prey phase of the whale optimization algorithm to avoid premature convergence and improved the logarithmic spiral curve search path for individual whales to improve the algorithm’s ability to traverse solutions. Shen et al. [9] adjusted the population diversity through an adaptive multi-scale control factor and designed an assisted swarm in implementing a local search to balance faster convergence and improved solution quality. Tang et al. [10] enhanced the population diversity through an elite ranking strategy, used a nonlinear energy factor adjustment strategy to balance the exploration and exploitation of the algorithm, and used a Gaussian random walk strategy to apply perturbations to the optimal individuals to effectively jump out of the local optimum. Zhang et al. [11] let the gulls choose spiral flight paths in different directions so that the gull flight paths are no longer single, which increased the algorithm diversity and avoided premature convergence of the algorithm by conducting a small search around the target according to the convergence of the algorithm. Deng et al. [12] used the Faure sequence to generate quasi-random numbers to initialize the population to improve the ChOA, approximated the uniform distribution with higher quality, improved the population search speed and solved the Kullback-Leibler problem of stochastic neighbor embedding with better results than the commonly used gradient descent method and stochastic gradient descent method. Xiao et al. [13] used the sine map to initialize the population and modify chaotic vectors to overcome the early convergence problem and applied it to test systems of four cascade reservoirs and three thermal plants with the valve point effect to demonstrate the improvements. Khishe et al. [14] developed weight-based location weighting equations to speed up convergence and improve exploration and exploitation. Mansoor et al. [15] used sine and cosine functions to search for the global optimal solution around the population generated by the ChOA, which was used to train the Fuzzy Neural Network (FNN) model to predict wind power generation for smart grid applications in the short term. Jia et al. [16] used highly destructive polynomial variants to further explore the regions and boundaries of the initial space, used Spearman's rank correlation coefficient to refine the candidate solutions in need of improvement and used the beetle antennae operator to avoid getting trapped in a local optimum to improve the ChOA, and they applied it to train a multilayer perceptron. Dhiman [17] used the sine and cosine functions and the attack strategy from the spotted hyena optimizer [18] to improve the ChOA and increase convergence efficiency. Chen et al. [19] introduced the bubble-net attacking method and the random search mechanism in the whale optimization algorithm [20] into the position updating process of the ChOA in order to improve the local search capability of the ChOA, and they demonstrated the effectiveness of this algorithm for structural damage detection through the validation of a two-story rigid frame model and a simply supported beam model. Kaur et al. [21] used the mathematical equations by a random walk in the cuckoo search algorithm [5] to update the position equations of the ChOA for the optimal design of digital infinite pulse filters. Zhang et al. [22] used Sobol sequences to initialize populations to improve search efficiency, and they used it to enhance the take-off performance of a cruise aircraft.

The motivation of this research is, compared with other intelligent optimization algorithms, to demonstrate that the ChOA has the advantages of no gradient mechanism, few control parameters, being a simple model, easy implementation and high stability, and that it has been successfully applied to various fields and achieved good effects. However, the ChOA is similar to other swarm intelligence optimization algorithms in that it suffers from low solution accuracy, slow convergence speed and the tendency to fall into the local optimum, which needs further improvement in order to achieve better results when applied to practical life. Although the above-mentioned scholars have achieved certain results in the improvement of the algorithm, there are still shortcomings in terms of knowing how to balance the convergence accuracy and convergence speed. Therefore, in order to improve the ChOA's performance and applicability in the optimum search, we propose a chaos somersault foraging ChOA (CSFChOA) on the basis of the ChOA. Simulation experiments were conducted on 23 standard test functions, as well as CEC2019 test functions, to find the best; and, statistical analysis with the Wilcoxon rank sum test was used to verify the superiority of the algorithm. Finally, the feasibility, applicability and superiority of the CSFChOA in practical engineering problems are demonstrated by comparing the test analysis on two complex engineering design problems. The following is a summary of this research's significant contributions.

a) To address the problems of low solution accuracy and slow convergence of the ChOA, the cat chaotic sequence is introduced to generate initial solutions, and opposition-based learning is used to select the better solutions to form the initial population, which can ensure diversity at the initial phase of the algorithm and improve the convergence speed and the accuracy of the algorithm to find the best solution.

b) In order to solve the problem of the ChOA easily falling into a local optimum at the later phase, by taking the optimal solution as the pivot, chimps with better adaptation at the mirror image position replace chimps from the original population using the somersault foraging strategy, which increases the population diversity and expands the search range.

c) The performance of the CSFChOA has been tested by using 23 standard test functions in low dimensions, 13 standard test functions in high dimensions and CEC2019 test functions to find the optimal solution; also, it is statistically analyzed using the Wilcoxon rank sum test and the optimization results are compared with the ChOA and other improved intelligent optimization algorithms.

d) Application of the CSFChOA to two complex engineering designs: the speed reducer design and the three-bar truss design have been tested and analyzed, and the feasibility, applicability and superiority of CSFChOA in practical engineering are verified.

The rest of the paper is organized as follows. After the introduction in Section 1, the CSFChOA is described in detail in Section 2, and a time complexity analysis is performed. Simulations and analysis are given in Section 3. The analysis of the engineering example applications of the CSFChOA is reported in Section 4. Finally, the full work is summarized and discussed in Section 5, which explains the direction of future work.

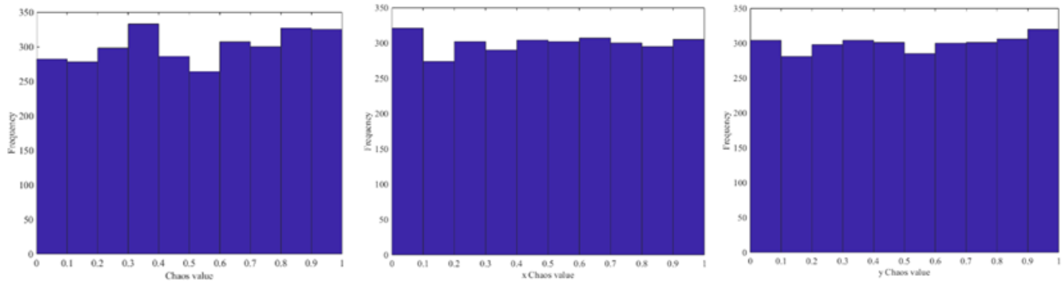
2. Chaos somersault foraging chimp optimization algorithm

2.1. Description of CSFChOA

First, the CSFChOA uses cat mapping to generate uniformly distributed populations. Cat mapping features a simple structure, good traversal and is less likely to fall into small loops and unstable periodical points. It traverses all states in the search scope without repeating; a uniformly distributed population can be produced, which can guarantee the quality of the initial solutions and prevent some individuals from being far from the optimal solution due to the uneven distribution of the population; uneven distribution of the population may reduce the speed of the population search, the accuracy of the solution and the diversity of the algorithm. The cat mapping formula is defined as follows.

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} \bmod 1 \tag{1}$$

The rand function and cat mapping were used to generate 3000 numbers between [0, 1], as shown in Figure 1; Figure 1(a) shows the frequency histogram generated by the rand function. The cat mapping is a two-dimensional reversible chaotic mapping, and the two histograms in Figure 1(b),(c) are the frequency histograms of the *x* and *y* sequences. It can be seen that the cat mapping produces a uniform distribution of chaotic values between [0, 1], and the results are excellent, with the frequency of each unit interval being around 300 for both the *x* and *y* sequences. In contrast, the frequency of the rand function per unit interval is uneven, particularly in [0.3, 0.4], where the chaos values is nearly 90 more than in [0.5, 0.6], which is quite different. This shows that the cat mapping is better than the rand function in generating a uniformly distributed population.



(a) (b) (c)

Figure 1. Frequency histogram generated by using the rand function and cat chaos mapping.

In order to increase the speed of convergence of the population and to find the optimal solution faster, opposition-based learning is introduced. If we consider the opposite solution of each individual at the same time with a uniform distribution of the population, the probability of the current individual and the opposite individual being closer to the optimal solution is 50%. Selecting the closer individual as the initial population, i.e., the individuals with higher fitness, means that each individual in the population is one step closer to the optimal individual, which greatly accelerates the algorithm search efficiency and convergence speed.

The cat mapping and opposition-based learning strategy is viewed as a chaotic opposition-based learning strategy. Steps to initialize the population based on this strategy are as follows: first, generate *N* initial solutions $\mathbf{x}_i$ using the cat chaotic sequence in the search space [*ub*, *lb*]; then, generate the corresponding opposite solutions for each initial solution in the following way.

$$\mathbf{x}_j = k\left(\mathbf{x}_{\min}^d + \mathbf{x}_{\max}^d\right) - \mathbf{x}_i \tag{2}$$

where *k* is a random number between [0, 1]; $\mathbf{x}_j$ is the opposite solution corresponding to each initial solution $\mathbf{x}_i$; x_{min}^d and x_{max}^d denote the minimum and maximum values of the d^{th} dimension vector in all initial solutions, respectively. Finally, the initial solutions $\mathbf{x}_i$ and the opposite solutions $\mathbf{x}_j$ are combined and sorted in ascending order according to the fitness value, and the top *N* solutions with the best fitness value are selected as the initial population.

Next, the chimp hunting phase is divided into the exploration phase and the exploitation phase. The exploration phase includes driving, blocking and chasing the prey, and the exploitation phase includes attacking the prey. The population divides chimps into four different roles to perform the division of duties during the hunt, namely, attacker, barrier, driver and chaser, which are the optimal, sub-optimal, third-optimal and fourth-optimal solutions of the chimp group, respectively, with the latter three types of individuals assisting the attacker in the search for prey and other chimps' positions being guided by these four types of chimps for updates. The fitness value of each individual in the population is recalculated at each iteration and compared with the four best fitness values from the previous iteration; the four better individuals are selected as the leaders of the chimp population for this iteration. Each type of chimp has its own independent thinking capability and uses its own search strategy to explore and predict the location of the prey; meanwhile, they have their own individual tasks and will engage in chaotic individual hunting behavior in the final stages of the hunt because of the social motivation from the obtainable behavior and benefits. During the exploration phase, any chimp will change its position depending on the location of its prey, and the mathematical model for driving and chasing the prey is shown in Eqs (3)–(7).

$$\mathbf{d} = \left|\mathbf{c}\cdot\mathbf{x}_{prey}(t) - \mathbf{m}\cdot\mathbf{x}_{chimp}(t)\right| \tag{3}$$

$$\mathbf{x}_{chimp}(t+1) = \mathbf{x}_{prey}(t) - \mathbf{a}\cdot\mathbf{d} \tag{4}$$

$$\mathbf{a} = 2\cdot\mathbf{f}\cdot\mathbf{r}_1 - \mathbf{f} \tag{5}$$

$$\mathbf{c}=2\cdot\mathbf{r}_2 \tag{6}$$

$$\mathbf{m} = \mathbf{Chaotic\_value} \tag{7}$$

where **d** is the distance between the chimp and the prey and *t* is the current number of iterations; $\mathbf{x}_{prey}$ is the prey position vector; $\mathbf{x}_{chimp}$ is the current chimp position vector; $\mathbf{r}_1$ and $\mathbf{r}_2$ are random vectors between [0, 1]; **f** is the convergence factor whose value decreases nonlinearly from 2.5 to 0 with increasing number of iterations (including the exploration phase and the exploitation phase); **a** is a random vector that determines the distance between the chimp and the prey, and its value is a random number between [-2**f**, 2**f**]; when $|\mathbf{a}| < 1$, the position of the chimp tends to approach that of the prey $\mathbf{x}_{prey}$; when $|\mathbf{a}| > 1$, it means that the chimp is forced to be away from the prey position and search for the prey in a wider scope; **c** is a coefficient that controls the chimp driving and chasing the prey, and it is a random vector in the range of [0, 2]; when $\mathbf{c} < 1$, the impact of the prey position on the chimp position diminishes; otherwise, it increases. **m** is a chaotic vector calculated from various chaotic mappings, representing the impact of individual motivation on the chimp position during the hunting process.

The mathematical model of chimps attacking the prey during the exploitation phase is shown in Eqs (8)–(16).

$$\mathbf{d}_{Attacker} = \left|\mathbf{c}_1\cdot\mathbf{x}_{Attacker} - \mathbf{m}_1\cdot\mathbf{x}\right| \tag{8}$$

$$\mathbf{d}_{Barrier} = \left|\mathbf{c}_2\cdot\mathbf{x}_{Barrier} - \mathbf{m}_2\cdot\mathbf{x}\right| \tag{9}$$

$$\mathbf{d}_{Chaser} = \left|\mathbf{c}_3\cdot\mathbf{x}_{Chaser} - \mathbf{m}_3\cdot\mathbf{x}\right| \tag{10}$$

$$\mathbf{d}_{Driver} = \left|\mathbf{c}_4\cdot\mathbf{x}_{Driver} - \mathbf{m}_4\cdot\mathbf{x}\right| \tag{11}$$

$$\mathbf{x}_1 = \mathbf{x}_{Attacker} - \mathbf{a}_1\cdot\mathbf{d}_{Attacker} \tag{12}$$

$$\mathbf{x}_2 = \mathbf{x}_{Barrier} - \mathbf{a}_2\cdot\mathbf{d}_{Barrier} \tag{13}$$

$$\mathbf{x}_3 = \mathbf{x}_{Chaser} - \mathbf{a}_3 \cdot \mathbf{d}_{Chaser} \tag{14}$$

$$\mathbf{x}_4 = \mathbf{x}_{Driver} - \mathbf{a}_4 \cdot \mathbf{d}_{Driver} \tag{15}$$

$$\mathbf{x}(t+1) = \frac{\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 + \mathbf{x}_4}{4} \tag{16}$$

where $\mathbf{x}$ denotes the position vector of the current chimp, $\mathbf{x}_{Attacker}$ denotes the position vector of the attacker, $\mathbf{x}_{Barrier}$ denotes the position vector of the barrier, $\mathbf{x}_{Chaser}$ denotes the position vector of the chaser, $\mathbf{x}_{Driver}$ denotes the position vector of the driver, $\mathbf{x}(t + 1)$ denotes the updated position vector of the current chimp and $\mathbf{c}_1$, $\mathbf{c}_2$, $\mathbf{c}_3$ and $\mathbf{c}_4$ are all random vectors in the range of [0, 1]. It is assumed that the attacker, the driver, the barrier and the chaser have a better knowledge of the location of the potential prey. Thus, the four best solutions obtained so far are stored and other chimps update their position according to the best chimps' positions.

When individuals in the final phase are satisfied with food, the subsequent social motivation will make chimps unleash their natural instincts; individual chimps will attempt to get food in the forced chaotic situation. The chaotic behavior of chimps in the final phase helps to further alleviate the twin problems of local optimum traps and slow convergence when solving high-dimensional problems. The ChOA uses six chaotic mappings [7] of deterministic processes with random behavior. In order to simulate this social behavior, assuming a 50% probability of choosing either the normal update position mechanism or the chaotic model to update the chimps' position, the social stimulus behavior is mathematically modeled as shown in Eq (17).

$$\mathbf{x}_{chimp}(t+1) = \begin{cases} \mathbf{x}_{prey}(t) - \mathbf{a} \cdot \mathbf{d}, & if \;\; \mu < 0.5 \\ \mathbf{Chaotic\_value}, & if \;\; \mu \geq 0.5 \end{cases} \tag{17}$$

where μ is a random number taking the value [0, 1] and **Chaotic_value** is a chaotic mapping used to update the position of the solution.

To avoid falling into the local optimum, inspired by the manta ray foraging optimization algorithm [23], which suddenly flips over to catch plankton, a somersault foraging strategy has been introduced to improve the ability of the ChOA to jump out of the local optimum. In this behavior, the position of the current optimal individual is considered as a pivot. Chimps other than the optimal individual tend to turn around the pivot and then somersault to the mirror image position of their current position. Thus, they always update their position around the best position they have found. The mathematical model is as follows.

$$\mathbf{x}_{chimp}(t+1) = \mathbf{x}_{chimp}(t) + S \cdot (r_1 \cdot \mathbf{x}_{Attacker}(t) - r_2 \cdot \mathbf{x}_{chimp}(t)) \tag{18}$$

where S represents the somersault factor, which determines the somersault range, taken as $S = 2$; $\mathbf{x}_{Attacker}(t)$ is the position of the optimal individual; r_1, r_2 are two random numbers in the range of [0, 1].

Within the defined somersault range, each chimp has the possibility to move to any position in the new search domain located between the current position and the symmetric position around $\mathbf{x}_{Attacker}(t)$. As the distance between a single location and the best location found so far decreases, all individuals gradually approach the optimal solution in the search space. Thus, as the number of iterations increases, the somersault search range adaptively decreases. In each iteration, the current chimp $\mathbf{x}_{chimp}(t)$ is compared for fitness with the chimp in the mirror image position, and it is replaced by the chimp in the mirror image position if it has fallen into local optimum. At the same time, it also increases the population diversity for the next iteration and expands the search range. When the iteration is completed, the population has the optimal solution $\mathbf{x}_{Attacker}$ in $\mathbf{x}_{chimp}$, which is the final result.

2.2. Implementation of CSFChOA

First, initialize the population through the chaotic opposition-based learning strategy, and then divide the chimps into four different roles to perform the division of duties during the hunt, namely, attacker, barrier, driver and chaser; the positions of other chimps are updated by these four chimp species for guidance. Individuals in the final stages of the hunt are satisfied with food, and subsequent social motivation leads chimps to unleash their natural instincts. Next, the ability of the ChOA to jump out of a local optimum is improved by the somersault foraging strategy. The steps of the CSFChOA implementation proposed in this paper are shown in Figure 2, with *rand* being a random number of [0, 1].

2.3. Time complexity analysis of CSFChOA

Time complexity is a key indicator to test the operational efficiency and convergence performance of the algorithm. To the ChOA, assuming that the population size is N, the dimension of the search space is d, the time required to initialize the parameters is t_1 and the time to generate random numbers is t_2, the time complexity of the population initialization phase is

$$O(t_1 + N(dt_2)) = O(d + f(d)) \tag{19}$$

In the iterative phase of the algorithm, the time to calculate the individual fitness value of the population is designed to be $f(d)$, the time to compare the individual fitness values of the population and select the four optimal individual positions is t_3, the time to update the convergence factor is t_4 and the time to update the position of other chimps in the population to follow the four optimal individual positions is t_5; then, the time complexity of this phase is

$$O(N(f(d) + t_3 + t_4 + t_5)) = O(d + f(d)) \tag{20}$$

So, the total time complexity of the ChOA when finding the best solution is

$$T(d) = O(d + f(d)) + O(d + f(d)) = O(d + f(d)) \tag{21}$$

The CSFChOA is derived from the ChOA, so the time to initialize parameters remains the same as the time required by the ChOA. Let the population initialization time to execute the chaotic opposition-based learning strategy be t_6 and the time to execute the somersault foraging strategy be t_7; then, the time complexity of the CSFChOA is

$$T'(d) = O(t_1 + N(dt_6 + dt_7 + f(d))) = O(d + f(d)) \tag{22}$$

In summary, the time complexity of the CSFChOA is the same as that of the ChOA, and the improvement strategies proposed in this paper to address the shortcomings of the ChOA do not increase the time complexity.

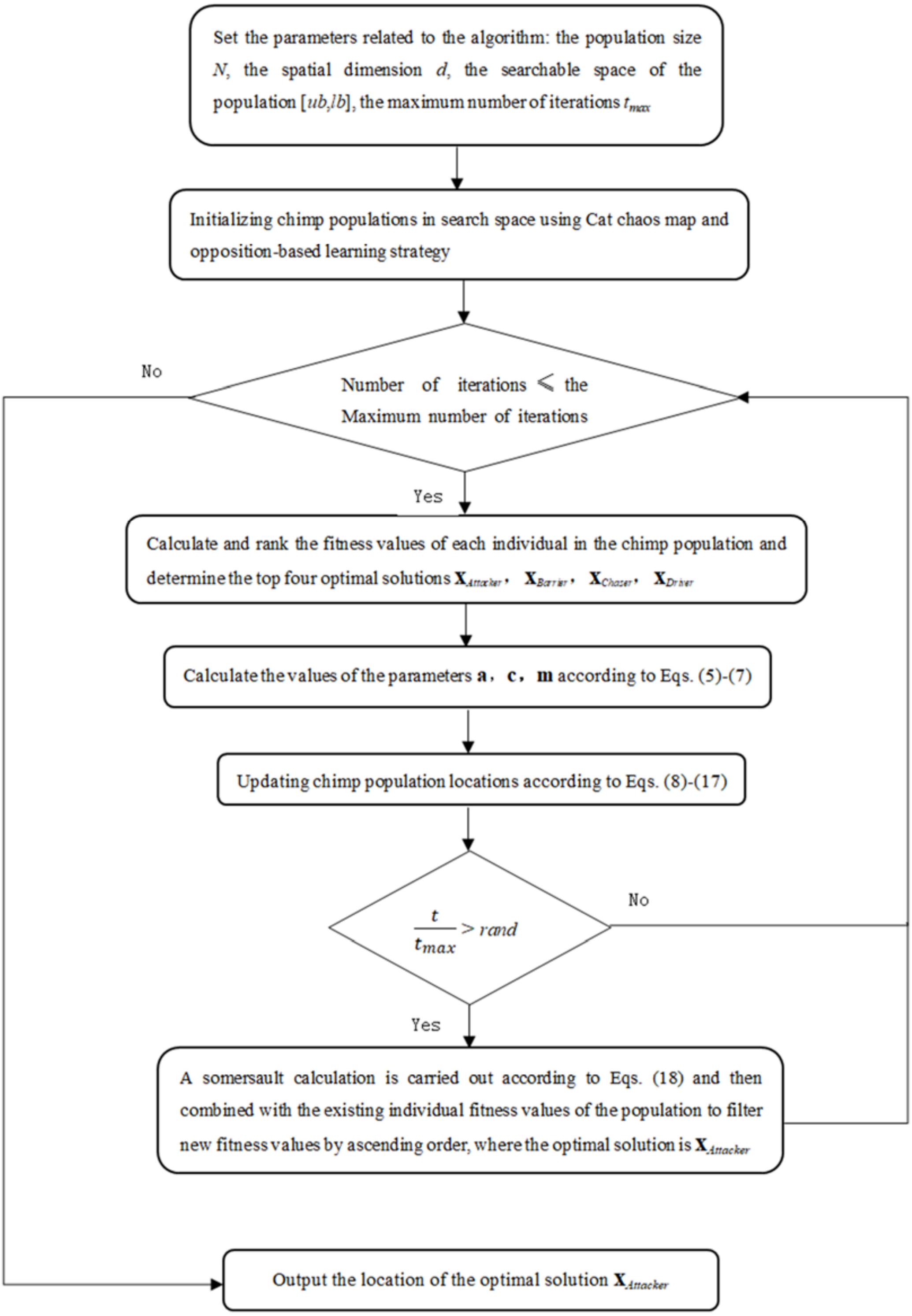

Set the parameters related to the algorithm: the population size N, the spatial dimension d, the searchable space of the population $[ub,lb]$, the maximum number of iterations t_{max}
Initializing chimp populations in search space using Cat chaos map and opposition-based learning strategy
Number of iterations ⩽ the Maximum number of iterations
No
Yes
Calculate and rank the fitness values of each individual in the chimp population and determine the top four optimal solutions $\mathbf{X}_{Attacker}$, $\mathbf{X}_{Barrier}$, $\mathbf{X}_{Chaser}$, $\mathbf{X}_{Driver}$
Calculate the values of the parameters **a**, **c**, **m** according to Eqs. (5)-(7)
Updating chimp population locations according to Eqs. (8)-(17)
$\frac{t}{t_{max}} > rand$
No
Yes
A somersault calculation is carried out according to Eqs. (18) and then combined with the existing individual fitness values of the population to filter new fitness values by ascending order, where the optimal solution is $\mathbf{X}_{Attacker}$
Output the location of the optimal solution $\mathbf{X}_{Attacker}$

Figure 2. Flowchart of CSFChOA implementation steps.

3. Simulation experiments and analysis results

3.1. Experimental design and test functions

The simulation environment was set to a 64-bit Windows 10 operating system; the CPU was an Intel$^{(R)}$ Core$^{(TM)}$ i7-10700 with a basic frequency of 2.90 GHz and memory of 16.0 GB; the programming software was MATLAB R2021b. To test the CSFChOA's search performance, 23 reference test functions were taken from the literature [7] to carry out simulation experiments. The expressions, search dimensions, variable boundaries and theoretical optimal values of the reference test functions are given in Table 1. *Function* denotes the function expression, *D* denotes the dimension of the problem, f_{min} is the theoretical optimal value of the function and *Range* denotes the boundary of the problem search space. F_1–F_7 are unimodal functions with relatively flat images and only a single extreme value point, mainly used to test the search accuracy and convergence speed of the algorithm. F_5 is a Rosenbrock function, which is a single-module function when the dimension is 2 or 3. It has many local extreme value points in the high-dimensional case and belongs to the canyon-type function, which has a narrow and gently changing valley, and its global minimum value appears at the bottom of the valley. This test function is used to test the efficiency of the function search. F_8–F_{13} are multimodal functions; F_9 has a number of widely existing minima, and it is regularly distributed and has high requirements for the function to jump out of a local optimum. F_{14}–F_{23} are fixed-dimension multimodal functions, which are used to test the algorithm's global search ability, local optimum avoidance ability and convergence speed.

Table 1. Introduction to benchmark functions.

Function	D	Range	f_{min}
$F_1(x)=\sum_{i=1}^{n} x_i^2$	3×10^1, 1×10^3	[-100, 100]	0
$F_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	3×10^1, 1×10^3	[-10, 10]	0
$F_3(x)=\sum_{i=1}^{n}\left(\sum_{j-1}^{i} x_j\right)^2$	3×10^1, 1×10^3	[-100, 100]	0
$F_4(x)=\max_i\left\{\lvert x_i\rvert, 1\le i\le n\right\}$	3×10^1, 1×10^3	[-100, 100]	0
$F_5(x)=\sum_{i=1}^{n-1}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(x_i-1\right)^2\right]$	3×10^1, 1×10^3	[-30, 30]	0
$F_6(x)=\sum_{i=1}^{n}\left([x_i+0.5]\right)^2$	3×10^1, 1×10^3	[-100, 100]	0
$F_7(x)=\sum_{i=1}^{n} i x_i^4+random[0,1)$	3×10^1, 1×10^3	[-1.28, 1.28]	0
$F_8(x)=\sum_{i=1}^{n}-x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	3×10^1, 1×10^3	[-500, 500]	-418.9829 × Dim
$F_9(x)=\sum_{i=1}^{n}\left[x_i^2-10\cos\left(2\pi x_i\right)+10\right]$	3×10^1, 1×10^3	[-5.12, 5.12]	0
$F_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i\right)+20+e$	3×10^1, 1×10^3	[-32, 32]	0

Continued on next page

Function	D	Range	f_{min}
$F_{11}(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	3×10^1, 1×10^3	[-600, 600]	0
$F_{12}(x)=\frac{\pi}{n}\{10\sin^2(\pi y_i)+\sum_{i=1}^{n-1}(y_i-1)^2\left[1+10\sin^2(\pi y_i+1)\right]$ $+(y_n-1)^2\}+\sum_{i=1}^{n}u(x_i,10,100,4)$, $y_i=1+\frac{1}{4}(x_i+1)$ $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, & x_i>a,\\ 0, & -a\le x_i\le a,\\ k(-x_i-a)^m, x_i<-a\end{cases}$	3×10^1, 1×10^3	[-50, 50]	0
$F_{13}(x)=0.1\{\sin^2(3\pi x_1)+\sum_{i=1}^{n-1}(x_i-1)^2\left[1+\sin^2(3\pi x_i+1)\right]$ $+(x_n-1)^2\left[1+\sin^2(2\pi x_n)\right]\}+\sum_{i=1}^{n}u(x_i,5,100,4)$	3×10^1, 1×10^3	[-50, 50]	0
$F_{14}(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$	2×10^0	[-65, 65]	1
$F_{15}(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}\right]^2$	4×10^0	[-5, 5]	0.00030
$F_{16}(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	2×10^0	[-5, 5]	-1.0316
$F_{17}(x)=(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	2×10^0	[-5, 5]	0.398
$F_{18}(x)=\left[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)\right]$ $\times\left[30+(2x_1-3x_2)^2\times(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)\right]$	2×10^0	[-2, 2]	3
$F_{19}(x)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{3}a_{ij}(x_j-p_{ij})^2\right)$	3×10^0	[1, 3]	-3.86
$F_{20}(x)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{6}a_{ij}(x_j-p_{ij})^2\right)$	6×10^0	[0, 1]	-3.32
$F_{21}(x)=-\sum_{i=1}^{5}\left[(x-a_i)(x-a_i)^T+c_i\right]^{-1}$	4×10^0	[0, 10]	-10.1532
$F_{22}(x)=-\sum_{i=1}^{7}\left[(x-a_i)(x-a_i)^T+c_i\right]^{-1}$	4×10^0	[0, 10]	-10.4028
$F_{23}(x)=-\sum_{i=1}^{10}\left[(x-a_i)(x-a_i)^T+c_i\right]^{-1}$	4×10^0	[0, 10]	-10.5363

3.2. Analysis of the impact of different improved strategies on algorithm performance in low dimensions

To fully validate the effectiveness and superiority of the CSFChOA, the ChOA [7], the ChOA using only the chaotic opposition-based learning strategy for population initialization (CChOA), the ChOA applying only the somersault foraging strategy (SFChOA), the enhanced whale optimization algorithm (EWOA) [8], the multi-scale sine cosine algorithm (MSCA) [9], the multi-directional exploring seagull optimization algorithm based on a chaotic map (MESOA) [11], the multi-strategy ChOA (EOSMICOA) [24] and the CSFChOA were compared through simulation experiments on 23 reference test functions with different optimum finding characteristics. In this research, the parameter values of these algorithms were selected according to the corresponding literature. The general conditions of algorithms were set to be the same: the population size $N = 30$, the spatial dimension $d = 30$, the maximum times of iterations $t_{max} = 1000$, and each algorithm was run 30 times independently. The performance of each algorithm was evaluated in terms of four performance metrics: the best value, the worst value, the mean value and the standard deviation. The simulation results of the eight algorithms for the 23 reference test functions are shown in Table 2.

Table 2. Comparison of the search results for different improved algorithms at $d = 30$.

Function	Algorithm	Worst	Best	Mean	Std
F_1	ChOA	6.02×10^{-16}	1.27×10^{-24}	2.86×10^{-17}	1.10×10^{-16}
	CChOA	5.18×10^{-17}	1.23×10^{-22}	3.92×10^{-18}	1.15×10^{-17}
	SFChOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EWOA	8.98×10^{-298}	7.91×10^{-323}	3.01×10^{-299}	0.00×10^{0}
	MSCA	2.14×10^{-6}	9.85×10^{-27}	7.29×10^{-8}	3.91×10^{-7}
	MESOA	1.57×10^{-118}	9.85×10^{-159}	5.24×10^{-120}	2.86×10^{-119}
	EOSMICOA	3.20×10^{-24}	3.69×10^{-29}	3.44×10^{-25}	7.01×10^{-25}
	CSFChOA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**
F_2	ChOA	2.92×10^{11}	1.04×10^{-16}	9.72×10^{9}	5.32×10^{10}
	CChOA	1.31×10^{-11}	5.50×10^{-22}	1.54×10^{-12}	2.79×10^{-12}
	SFChOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EWOA	6.24×10^{-162}	5.25×10^{-183}	2.08×10^{-163}	0.00×10^{0}
	MSCA	2.35×10^{-10}	1.55×10^{-18}	9.85×10^{-12}	4.39×10^{-11}
	MESOA	2.43×10^{-89}	5.56×10^{-117}	1.54×10^{-90}	5.81×10^{-90}
	EOSMICOA	3.77×10^{-16}	2.88×10^{-18}	6.29×10^{-17}	7.89×10^{-17}
	CSFChOA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**
F_3	ChOA	2.58×10^{0}	4.01×10^{-7}	2.26×10^{-1}	5.85×10^{-1}
	CChOA	1.02×10^{0}	7.38×10^{-8}	7.14×10^{-2}	2.11×10^{-1}
	SFChOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EWOA	7.08×10^{-15}	6.32×10^{-95}	2.36×10^{-16}	1.29×10^{-15}
	MSCA	2.95×10^{3}	3.23×10^{-3}	5.78×10^{2}	8.16×10^{2}
	MESOA	1.34×10^{-60}	9.00×10^{-108}	4.46×10^{-62}	2.44×10^{-61}
	EOSMICOA	1.83×10^{-5}	3.44×10^{-11}	7.66×10^{-7}	3.34×10^{-6}
	CSFChOA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**
F_4	ChOA	9.63×10^{-2}	2.97×10^{-5}	6.70×10^{-3}	2.23×10^{-2}
	CChOA	1.49×10^{-2}	5.93×10^{-7}	7.25×10^{-4}	2.71×10^{-3}
	SFChOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EWOA	3.90×10^{-100}	2.76×10^{-123}	1.31×10^{-101}	7.12×10^{-101}
	MSCA	3.04×10^{1}	4.31×10^{-2}	1.13×10^{1}	8.12×10^{0}
	MESOA	9.65×10^{-48}	2.68×10^{-67}	3.22×10^{-49}	1.76×10^{-48}
	EOSMICOA	8.88×10^{-2}	8.19×10^{-8}	2.98×10^{-3}	1.62×10^{-2}
	CSFChOA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**
F_5	ChOA	2.90×10^{1}	2.81×10^{1}	2.89×10^{1}	1.68×10^{-1}
	CChOA	2.90×10^{1}	2.73×10^{1}	2.88×10^{1}	3.61×10^{-1}
	SFChOA	2.88×10^{1}	2.58×10^{1}	2.75×10^{1}	8.94×10^{-1}
	EWOA	2.87×10^{1}	2.61×10^{1}	2.73×10^{1}	7.74×10^{-1}
	MSCA	2.91×10^{1}	2.67×10^{1}	2.75×10^{1}	6.65×10^{-1}
	MESOA	2.90×10^{1}	2.72×10^{1}	2.80×10^{1}	7.81×10^{-1}
	EOSMICOA	2.88×10^{1}	2.63×10^{1}	2.77×10^{1}	7.58×10^{-1}
	CSFChOA	**2.88×10^{1}**	**2.61×10^{1}**	**2.74×10^{1}**	**7.29×10^{-1}**

Continued on next page

Function	Algorithm	Worst	Best	Mean	Std
F_6	ChOA	4.15×10^{0}	2.22×10^{0}	3.39×10^{0}	4.27×10^{-1}
	CChOA	3.93×10^{0}	2.53×10^{0}	3.28×10^{0}	3.51×10^{-1}
	SFChOA	1.92×10^{0}	2.59×10^{-1}	8.74×10^{-1}	4.10×10^{-1}
	EWOA	1.03×10^{0}	2.77×10^{-3}	4.25×10^{-1}	3.35×10^{-1}
	MSCA	9.77×10^{-1}	2.98×10^{-3}	2.66×10^{-1}	2.03×10^{-1}
	MESOA	4.75×10^{0}	3.71×10^{0}	4.29×10^{0}	2.24×10^{-1}
	EOSMICOA	2.93×10^{0}	1.37×10^{0}	2.18×10^{0}	3.50×10^{-1}
	CSFChOA	**1.75×10^{0}**	**1.90×10^{-1}**	**8.44×10^{-1}**	**4.06×10^{-1}**
F_7	ChOA	6.41×10^{-3}	9.59×10^{-5}	1.22×10^{-3}	1.40×10^{-3}
	CChOA	2.75×10^{-3}	1.36×10^{-4}	6.50×10^{-4}	6.11×10^{-4}
	SFChOA	5.69×10^{-4}	2.79×10^{-5}	1.67×10^{-4}	1.33×10^{-4}
	EWOA	6.32×10^{-3}	1.97×10^{-5}	1.49×10^{-3}	1.56×10^{-3}
	MSCA	4.96×10^{-2}	6.12×10^{-4}	1.51×10^{-2}	1.20×10^{-2}
	MESOA	5.05×10^{-4}	1.46×10^{-5}	1.71×10^{-4}	1.42×10^{-4}
	EOSMICOA	2.55×10^{-3}	3.92×10^{-5}	7.97×10^{-4}	6.40×10^{-4}
	CSFChOA	**4.82×10^{-4}**	**5.22×10^{-6}**	**1.37×10^{-4}**	**1.22×10^{-4}**
F_8	ChOA	-5.64×10^{3}	-5.99×10^{3}	-5.75×10^{3}	7.36×10^{1}
	CChOA	-5.66×10^{3}	-5.92×10^{3}	-5.76×10^{3}	6.01×10^{1}
	SFChOA	-6.34×10^{3}	-8.69×10^{3}	-7.46×10^{3}	6.19×10^{2}
	EWOA	-5.17×10^{3}	-9.85×10^{3}	-8.03×10^{3}	9.96×10^{2}
	MSCA	-5.63×10^{3}	-7.03×10^{3}	-6.28×10^{3}	4.03×10^{2}
	MESOA	-3.85×10^{3}	-4.95×10^{3}	-4.33×10^{3}	2.68×10^{2}
	EOSMICOA	-5.65×10^{3}	-5.92×10^{3}	-5.76×10^{3}	6.05×10^{1}
	CSFChOA	**-6.74×10^{3}**	**-8.64×10^{3}**	**-7.66×10^{3}**	**4.50×10^{2}**
F_9	ChOA	4.90×10^{1}	0.00×10^{0}	3.32×10^{0}	9.31×10^{0}
	CChOA	7.77×10^{0}	0.00×10^{0}	9.45×10^{-1}	1.89×10^{0}
	SFChOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EWOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	MSCA	4.32×10^{1}	0.00×10^{0}	6.47×10^{0}	1.31×10^{1}
	MESOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EOSMICOA	2.27×10^{-13}	0.00×10^{0}	9.47×10^{-15}	4.24×10^{-14}
	CSFChOA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**
F_{10}	ChOA	2.00×10^{1}	2.00×10^{1}	2.00×10^{1}	1.21×10^{-3}
	CChOA	2.00×10^{1}	3.30×10^{-11}	1.86×10^{1}	5.06×10^{0}
	SFChOA	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	0.00×10^{0}
	EWOA	4.44×10^{-15}	8.88×10^{-16}	2.90×10^{-15}	1.79×10^{-15}
	MSCA	2.00×10^{1}	2.70×10^{0}	1.94×10^{1}	3.15×10^{0}
	MESOA	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	0.00×10^{0}
	EOSMICOA	2.00×10^{1}	2.00×10^{1}	2.00×10^{1}	1.60×10^{-3}
	CSFChOA	**8.88×10^{-16}**	**8.88×10^{-16}**	**8.88×10^{-16}**	**0.00×10^{0}**
F_{11}	ChOA	8.15×10^{-2}	0.00×10^{0}	1.45×10^{-2}	2.43×10^{-2}
	CChOA	6.61×10^{-2}	0.00×10^{0}	1.37×10^{-2}	2.02×10^{-2}
	SFChOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EWOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	MSCA	1.09×10^{-1}	0.00×10^{0}	1.92×10^{-2}	3.35×10^{-2}
	MESOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EOSMICOA	5.50×10^{-2}	0.00×10^{0}	1.83×10^{-3}	1.00×10^{-2}
	CSFChOA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**

Continued on next page

Function	Algorithm	Worst	Best	Mean	Std
F_{12}	ChOA	8.69×10^{-1}	2.29×10^{-1}	4.12×10^{-1}	1.89×10^{-1}
	CChOA	9.49×10^{-1}	2.00×10^{-1}	3.83×10^{-1}	1.86×10^{-1}
	SFChOA	5.69×10^{-1}	2.20×10^{-2}	7.13×10^{-2}	9.67×10^{-2}
	EWOA	1.19×10^{-1}	6.11×10^{-4}	2.82×10^{-2}	3.01×10^{-2}
	MSCA	1.65×10^{0}	2.30×10^{-3}	2.59×10^{-1}	4.60×10^{-1}
	MESOA	1.12×10^{0}	3.25×10^{-1}	6.34×10^{-1}	2.25×10^{-1}
	EOSMICOA	7.10×10^{-1}	1.09×10^{-1}	2.08×10^{-1}	1.32×10^{-1}
	CSFChOA	**1.51×10^{-1}**	**2.47×10^{-2}**	**5.96×10^{-2}**	**2.92×10^{-2}**
F_{13}	ChOA	2.99×10^{0}	2.57×10^{0}	2.85×10^{0}	1.20×10^{-1}
	CChOA	2.99×10^{0}	2.48×10^{0}	2.80×10^{0}	1.36×10^{-1}
	SFChOA	1.31×10^{0}	3.20×10^{-1}	8.04×10^{-1}	2.52×10^{-1}
	EWOA	1.63×10^{0}	1.94×10^{-1}	8.89×10^{-1}	3.74×10^{-1}
	MSCA	3.16×10^{0}	3.63×10^{-1}	1.21×10^{0}	6.05×10^{-1}
	MESOA	2.87×10^{0}	1.99×10^{0}	2.37×10^{0}	2.28×10^{-1}
	EOSMICOA	2.10×10^{0}	1.38×10^{0}	1.75×10^{0}	1.78×10^{-1}
	CSFChOA	**1.20×10^{0}**	**2.87×10^{-1}**	**6.96×10^{-1}**	**3.08×10^{-1}**
F_{14}	ChOA	1.08×10^{1}	9.98×10^{-1}	1.32×10^{0}	1.78×10^{0}
	CChOA	2.04×10^{0}	9.98×10^{-1}	1.03×10^{0}	1.90×10^{-1}
	SFChOA	1.08×10^{1}	9.98×10^{-1}	2.70×10^{0}	3.31×10^{0}
	EWOA	1.27×10^{1}	9.98×10^{-1}	5.20×10^{0}	4.79×10^{0}
	MSCA	2.98×10^{0}	9.98×10^{-1}	1.20×10^{0}	5.47×10^{-1}
	MESOA	2.98×10^{0}	9.98×10^{-1}	1.34×10^{0}	5.20×10^{-1}
	EOSMICOA	1.08×10^{1}	9.98×10^{-1}	1.46×10^{0}	1.83×10^{0}
	CSFChOA	**1.08×10^{1}**	**9.98×10^{-1}**	**2.24×10^{0}**	**2.97×10^{0}**
F_{15}	ChOA	2.09×10^{-2}	1.23×10^{-3}	1.93×10^{-3}	3.59×10^{-3}
	CChOA	1.43×10^{-3}	1.23×10^{-3}	1.28×10^{-3}	4.18×10^{-5}
	SFChOA	1.25×10^{-3}	3.08×10^{-4}	4.78×10^{-4}	2.56×10^{-4}
	EWOA	1.23×10^{-3}	3.10×10^{-4}	6.38×10^{-4}	2.76×10^{-4}
	MSCA	1.22×10^{-3}	4.06×10^{-4}	8.18×10^{-4}	2.76×10^{-4}
	MESOA	7.68×10^{-4}	3.10×10^{-4}	4.25×10^{-4}	9.37×10^{-5}
	EOSMICOA	1.31×10^{-3}	1.23×10^{-3}	1.27×10^{-3}	2.01×10^{-5}
	CSFChOA	**7.65×10^{-4}**	**3.11×10^{-4}**	**4.55×10^{-4}**	**1.49×10^{-4}**
F_{16}	ChOA	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	7.79×10^{-6}
	CChOA	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	7.34×10^{-6}
	SFChOA	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	6.94×10^{-6}
	EWOA	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	1.12×10^{-11}
	MSCA	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	1.44×10^{-9}
	MESOA	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	6.82×10^{-7}
	EOSMICOA	-1.03×10^{0}	-1.03×10^{0}	-1.03×10^{0}	3.44×10^{-5}
	CSFChOA	**-1.03×10^{0}**	**-1.03×10^{0}**	**-1.03×10^{0}**	**1.10×10^{-11}**
F_{17}	ChOA	3.994×10^{-1}	3.979×10^{-1}	3.982×10^{-1}	3.24×10^{-4}
	CChOA	3.993×10^{-1}	3.979×10^{-1}	3.982×10^{-1}	3.06×10^{-5}
	SFChOA	3.984×10^{-1}	3.979×10^{-1}	3.980×10^{-1}	1.35×10^{-7}
	EWOA	3.979×10^{-1}	3.979×10^{-1}	3.979×10^{-1}	9.86×10^{-9}
	MSCA	3.979×10^{-1}	3.979×10^{-1}	3.979×10^{-1}	2.19×10^{-9}
	MESOA	3.979×10^{-1}	3.979×10^{-1}	3.979×10^{-1}	2.79×10^{-6}
	EOSMICOA	3.982×10^{-1}	3.979×10^{-1}	3.979×10^{-1}	6.23×10^{-5}
	CSFChOA	**3.980×10^{-1}**	**3.980×10^{-1}**	**3.980×10^{-1}**	**1.41×10^{-9}**

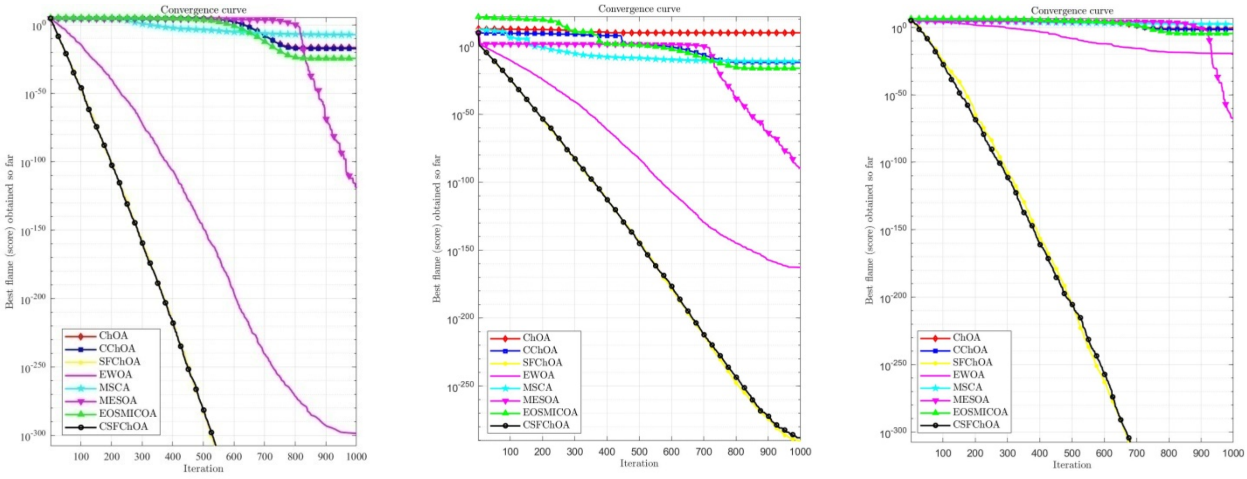
Continued on next page

Function	Algorithm	Worst	Best	Mean	Std
F_{18}	ChOA	3.00×10^{0}	3.00×10^{0}	3.00×10^{0}	8.51×10^{-5}
	CChOA	3.00×10^{0}	3.00×10^{0}	3.00×10^{0}	5.08×10^{-5}
	SFChOA	3.00×10^{0}	3.00×10^{0}	3.00×10^{0}	1.77×10^{-5}
	EWOA	3.01×10^{0}	3.00×10^{0}	3.00×10^{0}	2.47×10^{-3}
	MSCA	3.00×10^{0}	3.00×10^{0}	3.00×10^{0}	7.43×10^{-6}
	MESOA	3.00×10^{0}	3.00×10^{0}	3.00×10^{0}	2.56×10^{-7}
	EOSMICOA	3.00×10^{0}	3.00×10^{0}	3.00×10^{0}	9.26×10^{-6}
	CSFChOA	**3.00×10^{0}**	**3.00×10^{0}**	**3.00×10^{0}**	**1.55×10^{-8}**
F_{19}	ChOA	-3.85×10^{0}	-3.86×10^{0}	-3.85×10^{0}	1.49×10^{-3}
	CChOA	-3.85×10^{0}	-3.85×10^{0}	-3.85×10^{0}	3.98×10^{-4}
	SFChOA	-3.85×10^{0}	-3.86×10^{0}	-3.86×10^{0}	3.49×10^{-3}
	EWOA	-3.86×10^{0}	-3.86×10^{0}	-3.86×10^{0}	1.66×10^{-3}
	MSCA	-3.85×10^{0}	-3.85×10^{0}	-3.86×10^{0}	3.64×10^{-3}
	MESOA	-3.30×10^{0}	-3.86×10^{0}	-3.81×10^{0}	1.20×10^{-1}
	EOSMICOA	-3.85×10^{0}	-3.86×10^{0}	-3.86×10^{0}	1.95×10^{-3}
	CSFChOA	**-3.86×10^{0}**	**-3.86×10^{0}**	**-3.86×10^{0}**	**3.08×10^{-4}**
F_{20}	ChOA	-1.06×10^{0}	-3.04×10^{0}	-2.48×10^{0}	6.07×10^{-1}
	CChOA	-1.70×10^{0}	-3.30×10^{0}	-2.71×10^{0}	4.51×10^{-1}
	SFChOA	-2.27×10^{0}	-3.32×10^{0}	-3.10×10^{0}	1.80×10^{-1}
	EWOA	-2.84×10^{0}	-3.32×10^{0}	-3.23×10^{0}	1.35×10^{-1}
	MSCA	-3.02×10^{0}	-3.32×10^{0}	-3.19×10^{0}	7.70×10^{-2}
	MESOA	-1.53×10^{0}	-3.03×10^{0}	-2.17×10^{0}	3.93×10^{-1}
	EOSMICOA	-1.46×10^{0}	-3.08×10^{0}	-2.65×10^{0}	5.17×10^{-1}
	CSFChOA	**-2.86×10^{0}**	**-3.32×10^{0}**	**-3.22×10^{0}**	**1.32×10^{-1}**
F_{21}	ChOA	-4.97×10^{-1}	-5.04×10^{0}	-3.03×10^{0}	2.10×10^{0}
	CChOA	-3.51×10^{-1}	-5.04×10^{0}	-1.25×10^{0}	1.51×10^{0}
	SFChOA	-5.03×10^{0}	-9.81×10^{0}	-5.36×10^{0}	1.19×10^{0}
	EWOA	-8.82×10^{-1}	-1.02×10^{1}	-6.20×10^{0}	2.81×10^{0}
	MSCA	-4.98×10^{-1}	-1.02×10^{1}	-7.85×10^{0}	3.29×10^{0}
	MESOA	-1.62×10^{0}	-9.27×10^{0}	-3.93×10^{0}	2.06×10^{0}
	EOSMICOA	-8.81×10^{-1}	-5.05×10^{0}	-3.51×10^{0}	2.03×10^{0}
	CSFChOA	**-8.82×10^{-1}**	**-5.05×10^{0}**	**-4.76×10^{0}**	**1.05×10^{0}**
F_{22}	ChOA	-5.21×10^{-1}	-5.08×10^{0}	-4.04×10^{0}	1.79×10^{0}
	CChOA	-5.21×10^{-1}	-4.99×10^{0}	-8.53×10^{-1}	8.04×10^{-1}
	SFChOA	-5.05×10^{0}	-5.09×10^{0}	-5.07×10^{0}	1.09×10^{-2}
	EWOA	-2.77×10^{0}	-1.04×10^{1}	-6.63×10^{0}	3.01×10^{0}
	MSCA	-5.09×10^{0}	-1.04×10^{1}	-8.81×10^{0}	2.48×10^{0}
	MESOA	-1.97×10^{0}	-6.02×10^{0}	-3.80×10^{0}	1.05×10^{0}
	EOSMICOA	-5.24×10^{-1}	-5.08×10^{0}	-3.80×10^{0}	1.95×10^{0}
	CSFChOA	**-9.12×10^{-1}**	**-5.09×10^{0}**	**-4.93×10^{0}**	**7.60×10^{-1}**
F_{23}	ChOA	-5.57×10^{-1}	-9.03×10^{0}	-4.21×10^{0}	2.00×10^{0}
	CChOA	-5.54×10^{-1}	-9.49×10^{-1}	-8.29×10^{-1}	1.81×10^{-1}
	SFChOA	-5.09×10^{0}	-5.13×10^{0}	-5.11×10^{0}	1.08×10^{-2}
	EWOA	-2.42×10^{0}	-1.05×10^{1}	-6.21×10^{0}	3.26×10^{0}
	MSCA	-5.13×10^{0}	-1.05×10^{1}	-1.00×10^{1}	1.65×10^{0}
	MESOA	-2.15×10^{0}	-5.17×10^{0}	-3.48×10^{0}	9.40×10^{-1}
	EOSMICOA	-5.54×10^{-1}	-5.13×10^{0}	-3.29×10^{0}	2.11×10^{0}
	CSFChOA	**-5.10×10^{0}**	**-5.13×10^{0}**	**-5.11×10^{0}**	**7.00×10^{-3}**

As shown in Table 2, the CSFChOA can obtain theoretical optimum convergence when solving the unimodal functions F_1–F_4, multimodal functions F_9, F_{11} and fixed-dimensional multimodal functions F_{16}, F_{17}, F_{18} with the standard deviation of 0 on F_1–F_4, F_9, F_{10}, F_{11}. Regarding the mean value, for F_1–F_{13}, F_{15}, F_{17}, F_{18}, F_{19} and F_{20}, the CChOA and SFChOA yielded values smaller than that for the ChOA, which shows the effectiveness of using the chaotic opposition-based learning strategy and the somersault foraging strategy, respectively, both of which can improve the convergence accuracy of the ChOA to some extent. For F_{16}, although the ChOA, CChOA and SFChOA can all converge to the theoretical optimum, the standard deviations of both the CChOA and SFChOA were smaller than the ChOA, which has better robustness. For F_1–F_4, F_7, F_9–F_{11}, F_{13} and F_{19}, the CSFChOA had the highest convergence accuracy compared with the other seven algorithms. For F_{16} and F_{18}, although other algorithms could find the theoretical optimal solution, the CSFChOA had the smallest standard deviation and demonstrated better robustness. For F_5, F_8, F_{12} and F_{20}, the convergence accuracy of the CSFChOA was stronger than that of the other six algorithms, except the EWOA. Although it is weaker than the EWOA, it is more stable. For F_{15}, the CSFChOA was stronger than the other six algorithms, except MESOA. In terms of standard deviation, for F_1–F_4, F_7, F_9–F_{12}, F_{16}, F_{19}, F_{21}, F_{23}, the CSFChOA had the smallest standard deviation and the highest stability compared with the other seven algorithms.

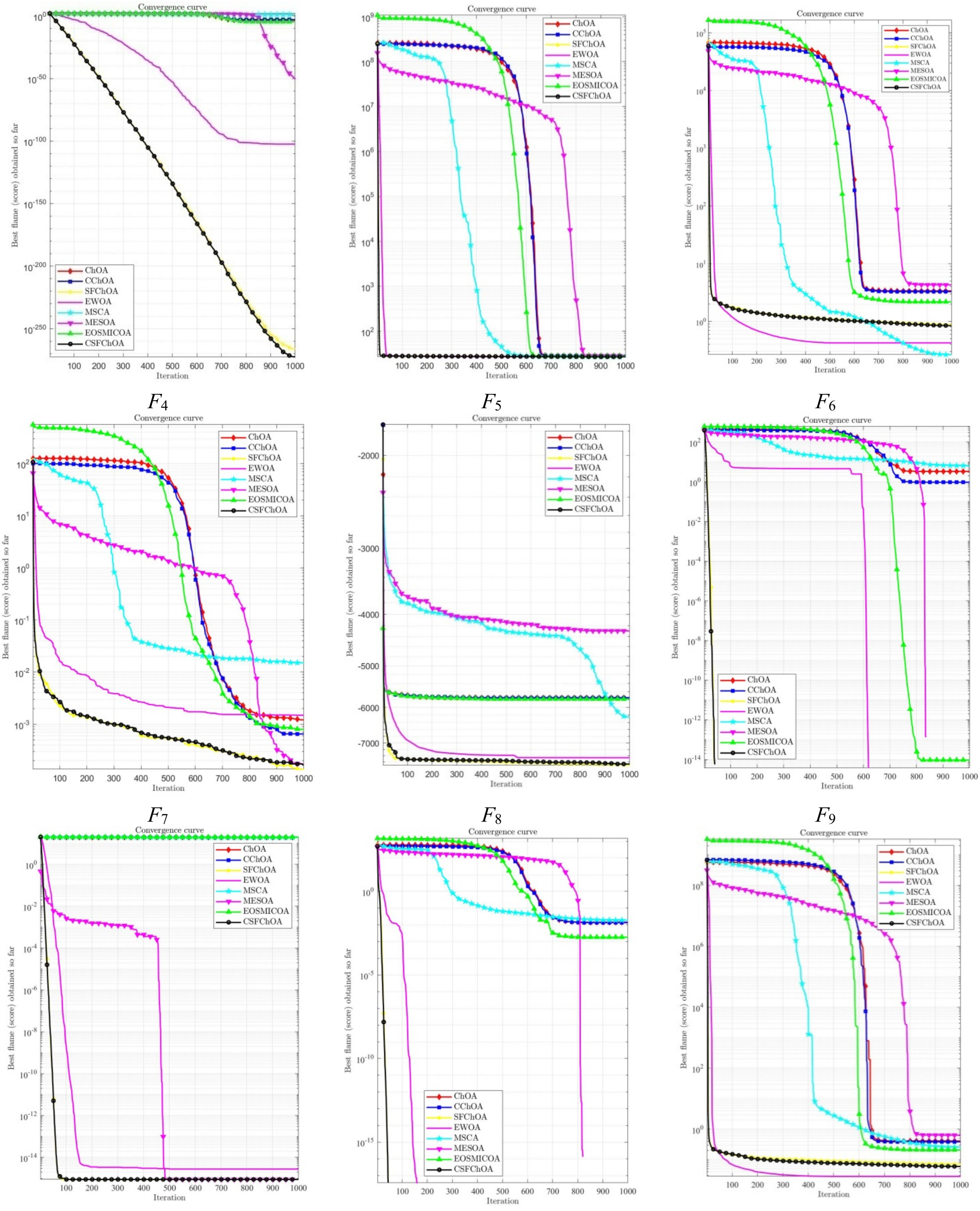
3.3. Convergence analysis

To more intuitively reflect the convergence performance of the CSFChOA, the average convergence curve graphs are used to describe the convergence characteristics of the eight algorithms used in Section 3.2 under the conditions of the population size N = 30, the spatial dimension d = 30, the maximum number of iterations t_{max} = 1000 and 30 times of independent runs of each algorithm. The average convergence curve comparison diagrams of solving the 23 reference test functions F_1–F_{23} of all algorithms are given in Figure 3. In addition, the data for each algorithm run 30 times on each function is plotted as a box plot, which gives a direct indication of how well the different algorithms perform on each function in terms of search performance, as shown in Figure 4.



F_1 F_2 F_3

Continued on next page

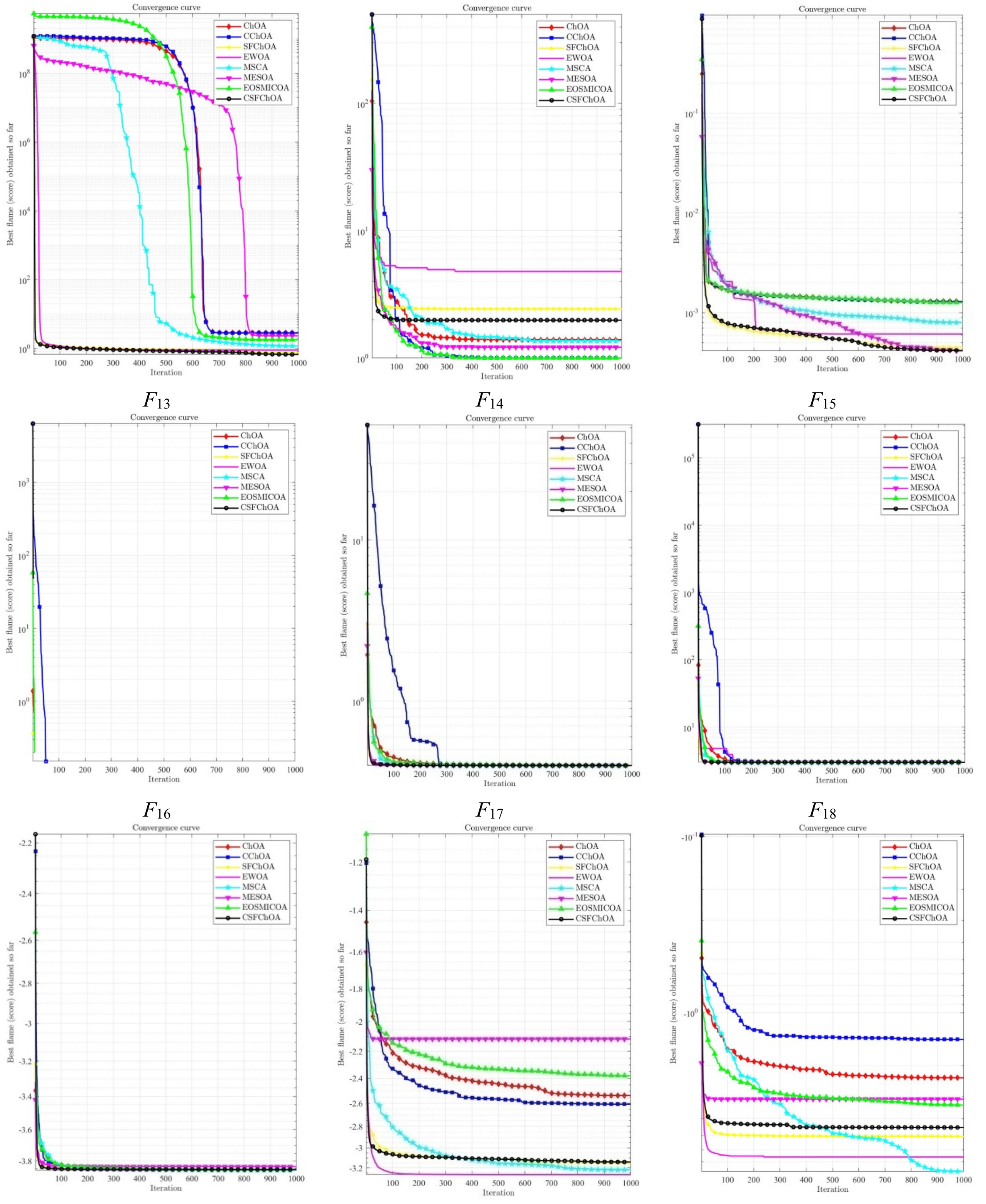


F_4 F_5 F_6

F_7 F_8 F_9

F_{10} F_{11} F_{12}

Continued on next page

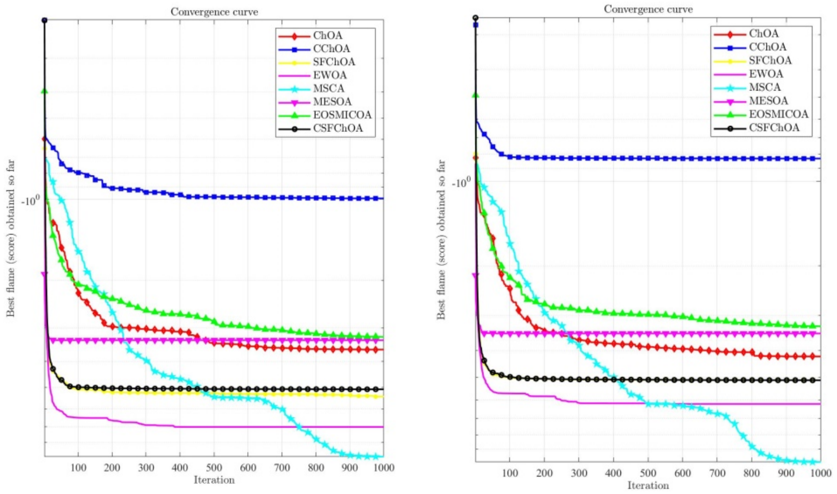


F_{13} F_{14} F_{15}

F_{16} F_{17} F_{18}

F_{19} F_{20} F_{21}

Continued on next page

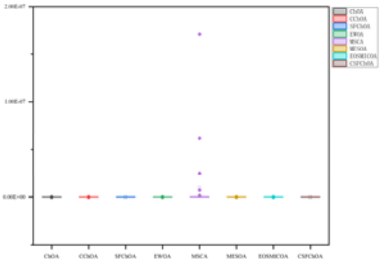


F_{22} F_{23}

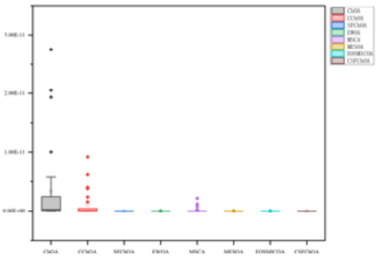
Figure 3. Average convergence curves of different strategy algorithms on 23 benchmark functions.

As can be seen in Figure 3, for F_1–F_5, F_7–F_{11}, F_{13}, F_{15} and F_{19}, the average convergence curves of the CSFChOA are all located below the ChOA, CChOA, EWOA, MSCA, MESOA and EOSMICOA; it indicates that the CSFChOA has obvious superiority in terms of exploration performance in the early iteration and exploitation performance in the late iteration, both of which are better than the other algorithms, with higher solution accuracy and faster convergence speed for the same times of iterations; it verifies that the CSFChOA can fully guarantee the exploration capability and exploration capability without losing the population diversity and the stability of the search for superiority. In particular, on F_1–F_4, F_9, F_{11}, F_{16}–F_{18}, the CSFChOA can effectively jump out of the local optimum and converge to the theoretical optimum quickly. For functions F_5, F_8, F_{10}, F_{13}, F_{18} and F_{19}, although the CSFChOA falls into the local optimum like other algorithms, the optimization-seeking accuracy and the convergence speed of the CSFChOA are much better than those of other algorithms.

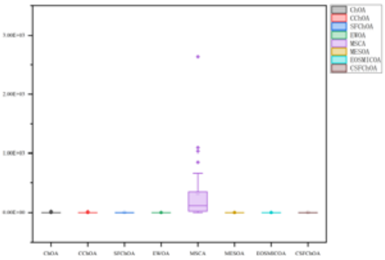
The CChOA is below the ChOA for the same times of iterations, which means that the CChOA converges faster than the ChOA. The initialization of the population with the chaotic opposition-based learning strategy can improve the initial population diversity and helps to improve the convergence speed and accuracy of the algorithm. Although this advantage is not obvious, it is more helpful for the introduction of other strategies to find the optimum later. The SFChOA converges significantly faster than the ChOA; it also has a significant improvement in convergence accuracy, and does not stall significantly throughout the iterations. This indicates that the somersault foraging strategy can effectively optimize the globally optimal chimp position, help the population to jump out of the local optimum and make the algorithm less likely to fall into the local optimum so that the theoretical optimal solution can be explored effectively. It can be seen that different improvement strategies have different degrees of improvement on the algorithm's optimality-seeking performance, and that the CSFChOA, which combines the two strategies, shows certain advantages.



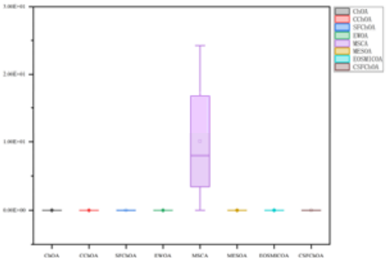
F_1



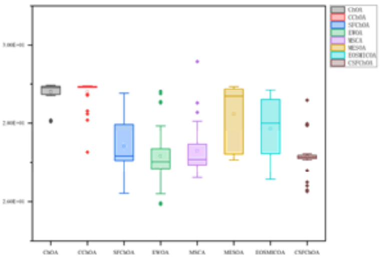
F_2



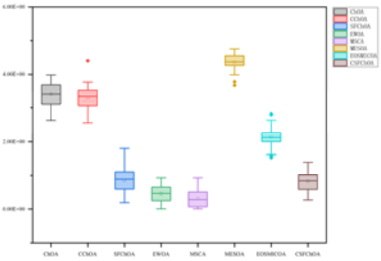
F_3



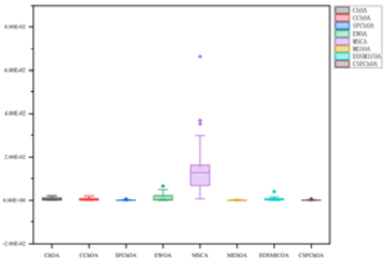
F_4



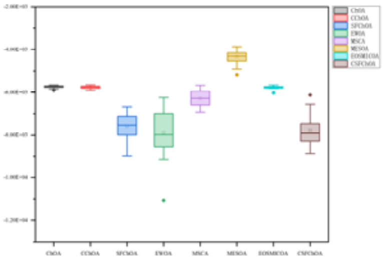
F_5



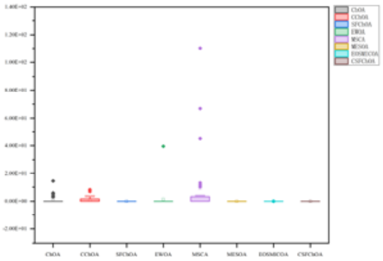
F_6



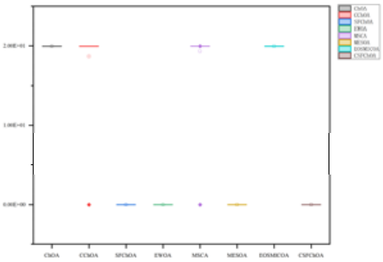
F_7



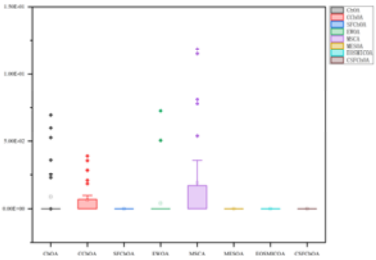
F_8



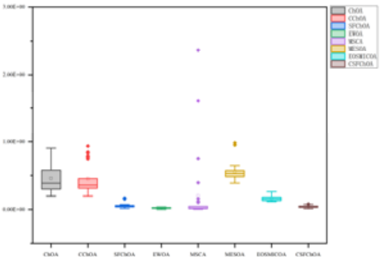
F_9



F_{10}

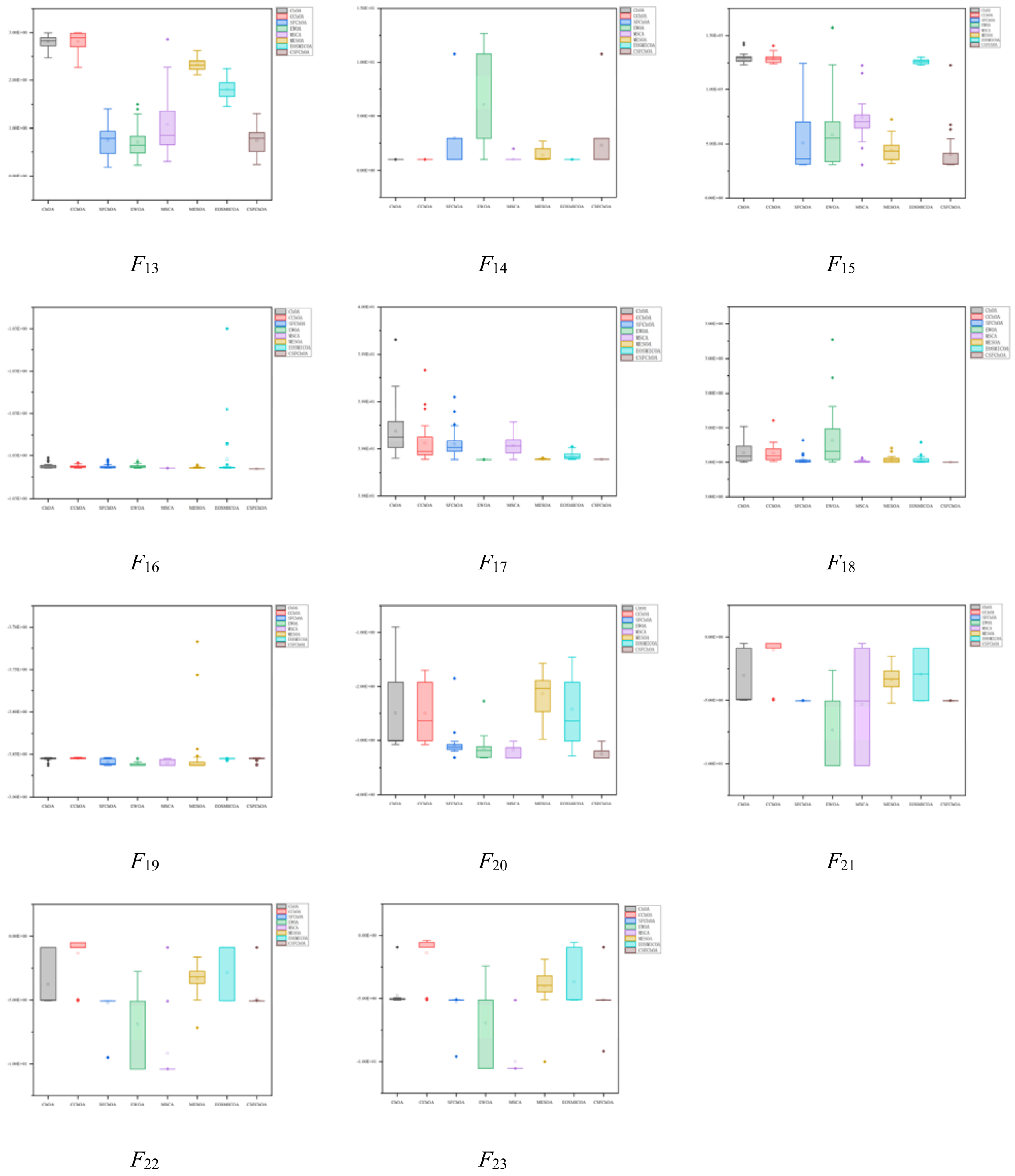


F_{11}



F_{12}

Continued on next page



F_{13} F_{14} F_{15}

F_{16} F_{17} F_{18}

F_{19} F_{20} F_{21}

F_{22} F_{23}

Figure 4. Box plot graphs of different strategy algorithms on 23 benchmark functions.

As can be seen in Figure 4, for F_1–F_4, F_7, F_9–F_{11}, the solution results of the CSFChOA were all around the theoretical optimum, which is both outlier-free and stable compared with the other seven algorithms. For F_5, F_{12}, F_{16}, F_{18}, F_{19}, F_{21} and F_{23}, the boxes of the CSFChOA are close to a line, which indicates high stability. It is very obvious in the figure that, for F_5–F_{13}, F_{15}–F_{20}, the convergence accuracies of the CChOA and SFChOA are higher than that of the ChOA, which further proves that using a chaotic opposition-based learning strategy and the somersault foraging strategy both have different degrees of improvement on the convergence ability of the ChOA. The mean value represented by the hollow square inside the box in the figure shows that, compared with the other seven algorithms, the convergence accuracy of the CSFChOA is higher for F_5, and that, for F_8, the convergence ability of the CSFChOA is only lower than that of the EWOA, but its box is flatter and more stable. For F_{12}, F_{15}, the CSFChOA outperforms the other seven algorithms in terms of box width, outliers and mean value.

The experimental results in Table 2, the average convergence curves in Figure 3 and box plot graphs in Figure 4 verify the effectiveness of the algorithm proposed in this research; its convergence speed is much faster than that of other algorithms, although the convergence accuracy gap is not obvious for some functions. The CSFChOA's solving accuracy is higher and more stable than other algorithms.

3.4. Analysis of the impact of different improvement strategies on algorithm performance in high dimensions

For analysis in this section, the population size $N = 30$, the spatial dimension $d = 1000$, the maximum number of iterations $t_{max} = 1000$ and each algorithm was run 30 times independently. The performance of each algorithm described in Section 3.2 were evaluated in terms of the best value, the worst value, the mean value and the standard deviation of the four performance metrics. The results of the eight algorithms for solving the F_1–F_{13} functions from the 23 reference test functions are shown in Table 3.

Table 3. Comparison of the results for different improved algorithms at $d = 1000$.

Function	Algorithm	Worst	Best	Mean	Std
F_1	ChOA	1.72×10^{2}	3.48×10^{1}	8.41×10^{1}	4.29×10^{1}
	CChOA	1.39×10^{2}	1.70×10^{1}	6.98×10^{1}	3.26×10^{1}
	SFChOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EWOA	7.64×10^{-282}	5.97×10^{-318}	2.55×10^{-283}	0.00×10^{0}
	MSCA	1.00×10^{5}	3.37×10^{-6}	2.66×10^{4}	3.45×10^{4}
	MESOA	2.43×10^{-61}	7.68×10^{-90}	1.01×10^{-62}	4.50×10^{-62}
	EOSMICOA	1.02×10^{4}	1.68×10^{3}	3.95×10^{3}	2.04×10^{3}
	CSFChOA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**
F_2	ChOA	1.78×10^{263}	3.23×10^{2}	1.78×10^{262}	Inf
	CChOA	2.75×10^{2}	2.45×10^{0}	5.36×10^{1}	1.07×10^{2}
	SFChOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EWOA	9.74×10^{-162}	2.09×10^{-176}	1.17×10^{-162}	3.14×10^{-162}
	MSCA	Inf	Inf	Inf	NaN
	MESOA	6.97×10^{-37}	7.09×10^{-50}	7.36×10^{-38}	2.19×10^{-37}
	EOSMICOA	1.29×10^{3}	2.73×10^{1}	2.69×10^{2}	4.67×10^{2}
	CSFChOA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**
F_3	ChOA	6.70×10^{7}	5.11×10^{6}	2.06×10^{7}	1.34×10^{7}
	CChOA	2.26×10^{7}	5.55×10^{6}	1.15×10^{7}	3.71×10^{6}
	SFChOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EWOA	6.02×10^{6}	2.88×10^{1}	7.43×10^{5}	1.50×10^{6}
	MSCA	1.20×10^{7}	2.99×10^{6}	7.56×10^{6}	2.55×10^{6}
	MESOA	1.05×10^{2}	1.78×10^{-22}	9.22×10^{0}	2.52×10^{1}
	EOSMICOA	8.02×10^{6}	3.08×10^{6}	5.60×10^{6}	1.21×10^{6}
	CSFChOA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**

Continued on next page

Function	Algorithm	Worst	Best	Mean	Std
F_4	ChOA	9.94×10^{1}	9.57×10^{1}	9.80×10^{1}	1.05×10^{0}
	CChOA	9.10×10^{1}	5.12×10^{1}	7.40×10^{1}	1.11×10^{1}
	SFChOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EWOA	1.11×10^{-79}	5.13×10^{-110}	3.69×10^{-81}	2.02×10^{-80}
	MSCA	9.94×10^{1}	9.81×10^{1}	9.89×10^{1}	3.27×10^{-1}
	MESOA	2.76×10^{-35}	7.16×10^{-230}	1.09×10^{-36}	5.08×10^{-36}
	EOSMICOA	8.98×10^{1}	6.33×10^{1}	7.45×10^{1}	6.74×10^{0}
	CSFChOA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**
F_5	ChOA	5.22×10^{8}	1.27×10^{5}	4.90×10^{7}	1.27×10^{8}
	CChOA	2.25×10^{5}	6.35×10^{3}	4.51×10^{4}	6.44×10^{4}
	SFChOA	9.96×10^{2}	9.94×10^{2}	9.95×10^{2}	4.52×10^{-1}
	EWOA	9.97×10^{2}	9.93×10^{2}	9.95×10^{2}	1.18×10^{0}
	MSCA	5.76×10^{9}	2.89×10^{8}	1.88×10^{9}	1.78×10^{9}
	MESOA	9.99×10^{2}	9.99×10^{2}	9.99×10^{2}	3.95×10^{-2}
	EOSMICOA	2.35×10^{7}	7.31×10^{5}	5.97×10^{6}	5.42×10^{6}
	CSFChOA	**9.96×10^{2}**	**9.94×10^{2}**	**9.95×10^{2}**	**4.62×10^{-1}**
F_6	ChOA	5.66×10^{2}	2.89×10^{2}	3.79×10^{2}	8.94×10^{1}
	CChOA	4.87×10^{2}	2.95×10^{2}	3.73×10^{2}	7.36×10^{1}
	SFChOA	1.65×10^{2}	8.58×10^{1}	1.41×10^{2}	2.14×10^{1}
	EWOA	1.61×10^{2}	6.11×10^{1}	1.22×10^{2}	3.37×10^{1}
	MSCA	9.08×10^{4}	2.28×10^{2}	3.17×10^{4}	3.67×10^{4}
	MESOA	2.45×10^{2}	2.44×10^{2}	2.45×10^{2}	3.27×10^{-1}
	EOSMICOA	6.99×10^{3}	1.68×10^{3}	3.66×10^{3}	1.59×10^{3}
	CSFChOA	**1.51×10^{2}**	**1.04×10^{2}**	**1.32×10^{2}**	**1.52×10^{1}**
F_7	ChOA	1.59×10^{2}	1.16×10^{0}	4.48×10^{1}	5.33×10^{1}
	CChOA	8.77×10^{-1}	5.42×10^{-1}	7.20×10^{-1}	1.16×10^{-1}
	SFChOA	7.52×10^{-4}	2.02×10^{-5}	2.80×10^{-4}	2.11×10^{-4}
	EWOA	1.05×10^{-2}	2.20×10^{-5}	2.19×10^{-3}	3.14×10^{-3}
	MSCA	7.66×10^{4}	8.05×10^{0}	2.26×10^{4}	2.63×10^{4}
	MESOA	1.01×10^{-3}	4.90×10^{-5}	3.83×10^{-4}	3.16×10^{-4}
	EOSMICOA	2.84×10^{2}	1.85×10^{1}	9.02×10^{1}	9.56×10^{1}
	CSFChOA	**1.03×10^{-3}**	**1.05×10^{-5}**	**2.38×10^{-4}**	**2.89×10^{-4}**
F_8	ChOA	-1.66×10^{5}	-1.69×10^{5}	-1.68×10^{5}	8.65×10^{2}
	CChOA	-1.66×10^{5}	-1.69×10^{5}	-1.68×10^{5}	8.53×10^{2}
	SFChOA	-1.87×10^{5}	-2.97×10^{5}	-2.43×10^{5}	3.47×10^{4}
	EWOA	-1.32×10^{5}	-3.00×10^{5}	-2.65×10^{5}	3.21×10^{4}
	MSCA	-3.74×10^{4}	-5.36×10^{4}	-4.28×10^{4}	4.10×10^{3}
	MESOA	-2.18×10^{4}	-2.94×10^{4}	-2.50×10^{4}	1.69×10^{3}
	EOSMICOA	-1.73×10^{5}	-1.77×10^{5}	-1.75×10^{5}	8.52×10^{2}
	CSFChOA	**-1.92×10^{5}**	**-2.91×10^{5}**	**-2.44×10^{5}**	**2.90×10^{4}**
F_9	ChOA	1.71×10^{2}	1.57×10^{1}	5.98×10^{1}	3.25×10^{1}
	CChOA	1.37×10^{2}	1.23×10^{1}	6.71×10^{1}	3.38×10^{1}
	SFChOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EWOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	MSCA	3.06×10^{3}	1.09×10^{-8}	6.28×10^{2}	8.49×10^{2}
	MESOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EOSMICOA	9.64×10^{2}	2.85×10^{2}	6.24×10^{2}	1.56×10^{2}
	CSFChOA	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**	**0.00×10^{0}**

Continued on next page

Function	Algorithm	Worst	Best	Mean	Std
F_{10}	ChOA	2.01×10^{1}	2.01×10^{1}	2.01×10^{1}	9.15×10^{-3}
	CChOA	1.12×10^{0}	2.28×10^{-1}	4.93×10^{-1}	1.97×10^{-1}
	SFChOA	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	0.00×10^{0}
	EWOA	4.44×10^{-15}	8.88×10^{-16}	3.02×10^{-15}	1.77×10^{-15}
	MSCA	2.00×10^{1}	2.00×10^{1}	2.00×10^{1}	1.67×10^{-5}
	MESOA	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	0.00×10^{0}
	EOSMICOA	2.00×10^{1}	2.00×10^{1}	2.00×10^{1}	7.20×10^{-3}
	CSFChOA	$\mathbf{8.88 \times 10^{-16}}$	$\mathbf{8.88 \times 10^{-16}}$	$\mathbf{8.88 \times 10^{-16}}$	$\mathbf{0.00 \times 10^{0}}$
F_{11}	ChOA	2.40×10^{0}	1.17×10^{0}	1.66×10^{0}	2.96×10^{-1}
	CChOA	4.25×10^{0}	1.21×10^{0}	1.80×10^{0}	6.01×10^{-1}
	SFChOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EWOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	MSCA	1.02×10^{3}	2.94×10^{-6}	3.69×10^{2}	3.71×10^{2}
	MESOA	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}	0.00×10^{0}
	EOSMICOA	6.28×10^{1}	1.63×10^{1}	3.46×10^{1}	1.22×10^{1}
	CSFChOA	$\mathbf{0.00 \times 10^{0}}$	$\mathbf{0.00 \times 10^{0}}$	$\mathbf{0.00 \times 10^{0}}$	$\mathbf{0.00 \times 10^{0}}$
F_{12}	ChOA	9.72×10^{9}	4.24×10^{6}	1.06×10^{9}	2.07×10^{9}
	CChOA	9.06×10^{4}	1.55×10^{0}	3.20×10^{3}	1.65×10^{4}
	SFChOA	5.37×10^{-1}	2.18×10^{-1}	3.59×10^{-1}	7.98×10^{-2}
	EWOA	4.29×10^{-1}	6.76×10^{-2}	2.25×10^{-1}	7.82×10^{-2}
	MSCA	1.90×10^{10}	4.42×10^{8}	8.89×10^{9}	5.86×10^{9}
	MESOA	1.17×10^{0}	1.15×10^{0}	1.16×10^{0}	6.72×10^{-3}
	EOSMICOA	1.54×10^{8}	3.58×10^{5}	2.12×10^{7}	3.30×10^{7}
	CSFChOA	$\mathbf{4.61 \times 10^{-1}}$	$\mathbf{2.28 \times 10^{-1}}$	$\mathbf{3.51 \times 10^{-1}}$	$\mathbf{6.34 \times 10^{-2}}$
F_{13}	ChOA	1.93×10^{9}	4.28×10^{5}	3.08×10^{8}	4.80×10^{8}
	CChOA	4.96×10^{4}	1.04×10^{2}	5.70×10^{3}	1.22×10^{4}
	SFChOA	8.72×10^{1}	5.29×10^{1}	6.95×10^{1}	9.27×10^{0}
	EWOA	8.92×10^{1}	2.96×10^{1}	6.04×10^{1}	1.55×10^{1}
	MSCA	3.73×10^{10}	1.09×10^{9}	1.68×10^{10}	1.27×10^{10}
	MESOA	1.00×10^{2}	9.93×10^{1}	9.98×10^{1}	2.30×10^{-1}
	EOSMICOA	1.98×10^{8}	2.58×10^{6}	2.29×10^{7}	3.64×10^{7}
	CSFChOA	$\mathbf{8.56 \times 10^{1}}$	$\mathbf{4.91 \times 10^{1}}$	$\mathbf{6.79 \times 10^{1}}$	$\mathbf{9.13 \times 10^{0}}$

Table 3 shows that the CSFChOA can find convergence to the theoretical optimum with the standard deviation of 0 for F_1–F_4 and multi-peak functions F_9 and F_{11}. Regarding the mean value, for F_1–F_8, F_{10}, F_{12} and F_{13}, the CChOA and SFChOA values were smaller than that from the ChOA, which proves that the improvement of the ChOA using chaotic opposition-based learning strategy and the somersault foraging strategy is still effective in high dimensions. For F_1–F_4, F_7, F_9 and F_{11}, the CSFChOA had the highest convergence accuracy compared with the other seven algorithms. For the remaining functions, F_5, F_6, F_8, F_{12} and F_{13}, the CSFChOA had higher convergence accuracy than the other six algorithms, except EWOA. Although weaker than the EWOA, the standard deviation of the CSFChOA was smaller than the EWOA for all five functions, which is more stable than the EWOA. In terms of standard deviation, for F_1–F_4, F_9–F_{11}, the CSFChOA had the smallest standard deviation compared with the other seven algorithms. For F_6, F_{12} and F_{13}, the stability of the CSFChOA was only lower than the MESOA, but its convergence accuracy was higher than the MESOA. For F_5, the stability of the CSFChOA was only weaker than the SFChOA and MESOA, but its convergence accuracy was higher than the SFChOA and MESOA. For F_7, the stability of the CSFChOA was only weaker than the SFChOA, but its convergence accuracy was higher than that of the SFChOA.

In summary, the improved strategy proposed in this paper has strong performance and robustness in finding the best solution for low-dimensional and high-dimensional problems.

3.5. Wilcoxon rank-sum test

Table 4. Wilcoxon rank-sum test p-value results.

Function	ChOA	AOA	CEHHO	EWOA	MSCA	MESOA	EOSMICOA
F_1	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}
F_2	3.02×10^{-11}	1.21×10^{-12}	1.21×10^{-12}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F_3	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}
F_4	3.02×10^{-11}	3.02×10^{-11}	1.21×10^{-12}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}
F_5	3.96×10^{-8}	7.70×10^{-4}	3.02×10^{-11}	2.92×10^{-2}	1.45×10^{-1}	5.27×10^{-5}	3.15×10^{-2}
F_6	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	1.25×10^{-7}	9.83×10^{-8}	3.02×10^{-11}	3.02×10^{-11}
F_7	1.39×10^{-6}	1.85×10^{-8}	5.09×10^{-6}	2.28×10^{-5}	3.02×10^{-11}	3.95×10^{-2}	1.39×10^{-6}
F_8	3.02×10^{-11}	4.50×10^{-11}	3.02×10^{-11}	1.33×10^{-2}	3.16×10^{-10}	3.02×10^{-11}	3.02×10^{-11}
F_9	4.55×10^{-12}	NaN	NaN	NaN	1.70×10^{-8}	NaN	8.14×10^{-2}
F_{10}	1.21×10^{-12}	NaN	NaN	1.79×10^{-7}	1.21×10^{-12}	NaN	1.21×10^{-12}
F_{11}	4.77×10^{-8}	1.21×10^{-12}	1.21×10^{-12}	NaN	4.79×10^{-8}	NaN	3.34×10^{-1}
F_{12}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	3.32×10^{-6}	3.33×10^{-1}	3.02×10^{-11}	3.02×10^{-11}
F_{13}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	4.36×10^{-2}	1.03×10^{-2}	3.02×10^{-11}	6.07×10^{-11}
F_{14}	3.79×10^{-1}	5.19×10^{-7}	1.69×10^{-9}	6.55×10^{-4}	1.43×10^{-8}	5.57×10^{-3}	1.87×10^{-7}
F_{15}	3.02×10^{-11}	2.00×10^{-5}	1.75×10^{-5}	4.08×10^{-5}	1.33×10^{-10}	7.24×10^{-2}	3.02×10^{-11}
F_{16}	3.11×10^{-1}	2.37×10^{-10}	3.02×10^{-11}	3.02×10^{-11}	3.02×10^{-11}	7.74×10^{-6}	1.30×10^{-3}
F_{17}	3.02×10^{-11}	4.98×10^{-11}	1.33×10^{-10}	3.02×10^{-11}	3.02×10^{-11}	8.10×10^{-10}	1.53×10^{-5}
F_{18}	6.36×10^{-5}	6.77×10^{-5}	2.49×10^{-6}	1.24×10^{-3}	3.39×10^{-2}	5.49×10^{-11}	3.63×10^{-1}
F_{19}	3.34×10^{-3}	4.71×10^{-4}	6.20×10^{-4}	7.77×10^{-9}	1.70×10^{-8}	4.71×10^{-4}	1.45×10^{-1}
F_{20}	3.16×10^{-10}	2.50×10^{-2}	2.64×10^{-1}	1.78×10^{-4}	3.16×10^{-5}	3.69×10^{-11}	2.38×10^{-7}
F_{21}	2.68×10^{-6}	2.15×10^{-2}	3.02×10^{-11}	2.44×10^{-9}	5.60×10^{-7}	2.24×10^{-2}	1.25×10^{-4}
F_{22}	6.52×10^{-9}	4.23×10^{-3}	3.02×10^{-11}	7.60×10^{-7}	5.57×10^{-10}	6.28×10^{-6}	6.74×10^{-6}
F_{22}	7.70×10^{-8}	2.84×10^{-4}	3.02×10^{-11}	8.35×10^{-8}	8.48×10^{-9}	1.85×10^{-8}	1.86×10^{-6}
F_{23}	1.55×10^{-9}	2.84×10^{-4}	3.34×10^{-11}	4.64×10^{-5}	5.57×10^{-10}	7.77×10^{-9}	1.25×10^{-7}
+/=/-	22/0/1	21/2/0	21/2/0	21/2/0	21/0/2	19/3/1	19/0/4

The Wilcoxon rank-sum test is a non-parametric statistical test to fully validate the effectiveness of the improved algorithm; it tested whether the CSFChOA is statistically different from the ChOA [7], arithmetic optimization algorithm (AOA) [25], the chaotic elite Harris hawks optimization algorithm (CEHHO) [10], the EWOA [8], the MSCA [9], the MESOA [11] and the EOSMICOA [24] based on 23 reference test functions at a significance level of p = 5%, 30 dimensions and 30 times of independent runs for each algorithm. In this research, the parameter values of these algorithms were selected according to the corresponding literature. When the p-value is less than 5%, if the H_0 hypothesis is rejected, it indicates that there is a significant difference between the two compared algorithms; if the H_0 hypothesis is accepted, it indicates that the two algorithms are about identical in terms of their performance in terms of finding the optimum. The p-values of the rank-sum test between the CSFChOA and the seven algorithms on the 23 reference test functions are shown in Table 4. A value marked "NaN" means that it is not applicable, i.e., no significance judgment can be made. The symbols "+", "-" and "=" indicate that the CSFChOA is better, worse or equal compared with the other algorithms, respectively. Most of the p-values were much less than 5%, which indicates that there is a statistical significant difference between the CSFChOA and the other seven algorithms in terms of the effectiveness of the search for the optimum; it verifies that the CSFChOA is significantly superior.

3.6. Experimental analysis of CEC2019 test functions

In order to verify the effectiveness and robustness of the CSFChOA, CEC2019 test functions were used to further verify the performance of the improved algorithm. There are 10 single-objective test functions in CEC2019, each with a theoretical optimum value of 1. These functions are challenging and can test the performance of the algorithm effectively. The introduction of CEC2019 test functions is shown in Table 5. The ChOA [7], AOA [25], CEHHO [10], EWOA [8], MSCA [9], MESOA [11] and EOSMICOA [24] have been compared. In this research, the parameter values of these algorithms were selected according to the corresponding literature. The experimental parameters were taken as the population size N = 30, the maximum times of iterations t_{max} = 1000, and each algorithm was run 30 times independently to take the worst value, the best value, mean value and standard deviation, as shown in Table 6.

Table 5. CEC2019 test functions.

No.	Functions	$F_i^* = F_i(x^*)$	D	Search Range
1	Storn's Chebyshev Polynomial Fitting Problem	1×10^0	9×10^0	[-8192,8192]
2	Inverse Hilbert Matrix Problem	1×10^0	16×10^0	[-16384,16384]
3	Lennard-Jones Minimum Energy Cluster	1×10^0	18×10^0	[-4,4]
4	Rastrigin's Function	1×10^0	10×10^0	[-100,100]
5	Griewangk10's Function	1×10^0	10×10^0	[-100,100]
6	Weierstrass Function	1×10^0	10×10^0	[-100,100]
7	Modified Schwefel's Function	1×10^0	10×10^0	[-100,100]
8	Expanded Schaffer's F6 Function	1×10^0	10×10^0	[-100,100]
9	Happy Cat Function	1×10^0	10×10^0	[-100,100]
10	Ackley Function	1×10^0	10×10^0	[-100,100]

Table 6. Comparison of CEC2019 test function optimization results.

Function	Algorithm	Worst	Best	Mean	Std
F_1	ChOA	3.80×10^6	1.00×10^0	3.26×10^5	8.00×10^5
	AOA	1.06×10^6	1.00×10^0	3.86×10^4	1.94×10^5
	CEHHO	1.00×10^0	1.00×10^0	1.00×10^0	2.30×10^{-9}
	EWOA	1.30×10^3	1.00×10^0	1.11×10^2	2.70×10^2
	MSCA	5.24×10^6	1.00×10^0	6.37×10^5	1.06×10^6
	MESOA	1.14×10^5	1.00×10^0	4.40×10^3	2.10×10^4
	EOSMICOA	1.09×10^7	1.00×10^0	1.25×10^6	2.73×10^6
	CSFChOA	$\mathbf{1.00 \times 10^0}$	$\mathbf{1.00 \times 10^0}$	$\mathbf{1.00 \times 10^0}$	$\mathbf{1.50 \times 10^{-9}}$
F_2	ChOA	1.48×10^3	3.90×10^0	3.39×10^2	3.86×10^2
	AOA	3.47×10^3	3.90×10^0	1.63×10^2	6.30×10^2
	CEHHO	3.66×10^0	3.14×10^0	3.35×10^0	1.28×10^{-1}

Continued on next page

Function	Algorithm	Worst	Best	Mean	Std
F_2	EWOA	3.01×10^{2}	3.17×10^{0}	1.37×10^{1}	5.42×10^{1}
	MSCA	7.08×10^{2}	1.58×10^{1}	3.81×10^{2}	2.07×10^{2}
	MESOA	1.29×10^{1}	3.06×10^{0}	4.01×10^{0}	1.76×10^{0}
	EOSMICOA	1.30×10^{3}	3.26×10^{0}	2.94×10^{2}	3.08×10^{2}
	CSFChOA	$\mathbf{4.00 \times 10^{0}}$	$\mathbf{3.15 \times 10^{0}}$	$\mathbf{3.26 \times 10^{0}}$	$\mathbf{2.04 \times 10^{-1}}$
F_3	ChOA	7.34×10^{0}	3.54×10^{0}	5.59×10^{0}	1.01×10^{0}
	AOA	1.16×10^{1}	8.29×10^{0}	9.88×10^{0}	8.46×10^{-1}
	CEHHO	6.77×10^{0}	1.71×10^{0}	4.07×10^{0}	9.39×10^{-1}
	EWOA	9.71×10^{0}	1.61×10^{0}	5.05×10^{0}	2.28×10^{0}
	MSCA	1.04×10^{1}	5.31×10^{0}	7.68×10^{0}	1.21×10^{0}
	MESOA	6.96×10^{0}	1.00×10^{0}	4.52×10^{0}	1.92×10^{0}
	EOSMICOA	6.75×10^{0}	1.85×10^{0}	4.11×10^{0}	1.24×10^{0}
	CSFChOA	$\mathbf{5.79 \times 10^{0}}$	$\mathbf{2.05 \times 10^{0}}$	$\mathbf{3.72 \times 10^{0}}$	$\mathbf{9.38 \times 10^{-1}}$
F_4	ChOA	9.73×10^{1}	4.26×10^{1}	6.23×10^{1}	1.43×10^{1}
	AOA	8.27×10^{1}	2.01×10^{1}	4.98×10^{1}	1.53×10^{1}
	CEHHO	9.06×10^{1}	2.06×10^{1}	4.98×10^{1}	1.73×10^{1}
	EWOA	8.96×10^{1}	1.16×10^{1}	5.37×10^{1}	2.00×10^{1}
	MSCA	8.68×10^{1}	2.15×10^{1}	4.17×10^{1}	1.48×10^{1}
	MESOA	6.22×10^{1}	3.98×10^{1}	5.20×10^{1}	4.77×10^{0}
	EOSMICOA	6.38×10^{1}	2.32×10^{1}	4.12×10^{1}	1.03×10^{1}
	CSFChOA	$\mathbf{1.75 \times 10^{1}}$	$\mathbf{7.25 \times 10^{0}}$	$\mathbf{1.22 \times 10^{1}}$	$\mathbf{2.96 \times 10^{0}}$
F_5	ChOA	1.52×10^{2}	1.21×10^{1}	5.36×10^{1}	3.47×10^{1}
	AOA	1.11×10^{2}	1.71×10^{1}	5.82×10^{1}	2.26×10^{1}
	CEHHO	6.85×10^{0}	1.94×10^{0}	2.79×10^{0}	1.04×10^{0}
	EWOA	2.67×10^{1}	1.55×10^{0}	3.45×10^{0}	4.52×10^{0}
	MSCA	1.56×10^{0}	1.14×10^{0}	1.39×10^{0}	8.83×10^{-2}
	MESOA	1.44×10^{1}	3.28×10^{0}	5.42×10^{0}	2.88×10^{0}
	EOSMICOA	3.41×10^{1}	2.41×10^{0}	6.28×10^{0}	6.29×10^{0}
	CSFChOA	$\mathbf{5.17 \times 10^{1}}$	$\mathbf{2.61 \times 10^{0}}$	$\mathbf{1.27 \times 10^{1}}$	$\mathbf{1.25 \times 10^{1}}$
F_6	ChOA	1.16×10^{1}	5.04×10^{0}	7.41×10^{0}	1.49×10^{0}
	AOA	1.29×10^{1}	7.85×10^{0}	1.07×10^{1}	1.19×10^{0}
	CEHHO	1.15×10^{1}	6.22×10^{0}	9.27×10^{0}	1.28×10^{0}
	EWOA	1.28×10^{1}	4.68×10^{0}	8.50×10^{0}	1.81×10^{0}
	MSCA	6.46×10^{0}	2.28×10^{0}	4.10×10^{0}	9.99×10^{-1}
	MESOA	9.00×10^{0}	4.82×10^{0}	6.39×10^{0}	9.33×10^{-1}
	EOSMICOA	8.94×10^{0}	4.35×10^{0}	6.90×10^{0}	1.14×10^{0}
	CSFChOA	$\mathbf{7.83 \times 10^{0}}$	$\mathbf{4.98 \times 10^{0}}$	$\mathbf{6.02 \times 10^{0}}$	$\mathbf{6.79 \times 10^{-1}}$
F_7	ChOA	2.10×10^{3}	1.33×10^{3}	1.75×10^{3}	2.40×10^{2}
	AOA	1.79×10^{3}	8.15×10^{2}	1.27×10^{3}	2.40×10^{2}
	CEHHO	1.93×10^{3}	7.75×10^{2}	1.31×10^{3}	2.76×10^{2}
	EWOA	1.99×10^{3}	8.23×10^{2}	1.38×10^{3}	3.31×10^{2}
	MSCA	1.70×10^{3}	8.73×10^{2}	1.25×10^{3}	2.03×10^{2}
	MESOA	1.72×10^{3}	9.99×10^{2}	1.36×10^{3}	1.71×10^{2}
	EOSMICOA	2.12×10^{3}	1.04×10^{3}	1.66×10^{3}	2.77×10^{2}
	CSFChOA	$\mathbf{1.09 \times 10^{3}}$	$\mathbf{2.50 \times 10^{2}}$	$\mathbf{6.77 \times 10^{2}}$	$\mathbf{1.88 \times 10^{2}}$
F_8	ChOA	5.06×10^{0}	4.58×10^{0}	4.81×10^{0}	1.66×10^{-1}
	AOA	5.41×10^{0}	3.77×10^{0}	4.72×10^{0}	3.30×10^{-1}
	CEHHO	5.09×10^{0}	3.89×10^{0}	4.76×10^{0}	3.30×10^{-1}
	EWOA	5.42×10^{0}	3.77×10^{0}	4.67×10^{0}	3.74×10^{-1}

Continued on next page

Function	Algorithm	Worst	Best	Mean	Std
F_8	MSCA	4.50×10^{0}	3.03×10^{0}	4.03×10^{0}	2.98×10^{-1}
	MESOA	5.59×10^{0}	4.86×10^{0}	5.35×10^{0}	1.43×10^{-1}
	EOSMICOA	5.04×10^{0}	4.13×10^{0}	4.71×10^{0}	2.44×10^{-1}
	CSFChOA	**4.66×10^{0}**	**3.56×10^{0}**	**4.14×10^{0}**	**2.86×10^{-1}**
F_9	ChOA	1.60×10^{0}	1.32×10^{0}	1.50×10^{0}	7.33×10^{-2}
	AOA	3.44×10^{0}	1.29×10^{0}	2.52×10^{0}	6.95×10^{-1}
	CEHHO	1.60×10^{0}	1.15×10^{0}	1.37×10^{0}	1.26×10^{-1}
	EWOA	1.93×10^{0}	1.14×10^{0}	1.43×10^{0}	1.85×10^{-1}
	MSCA	1.27×10^{0}	1.09×10^{0}	1.17×10^{0}	4.17×10^{-2}
	MESOA	1.52×10^{0}	1.19×10^{0}	1.40×10^{0}	8.33×10^{-2}
	EOSMICOA	1.52×10^{0}	1.20×10^{0}	1.32×10^{0}	6.84×10^{-2}
	CSFChOA	**1.60×10^{0}**	**1.20×10^{0}**	**1.41×10^{0}**	**1.07×10^{-1}**
F_{10}	ChOA	2.16×10^{1}	2.12×10^{1}	2.15×10^{1}	1.11×10^{-1}
	AOA	2.12×10^{1}	2.10×10^{1}	2.11×10^{1}	2.86×10^{-2}
	CEHHO	2.14×10^{1}	2.10×10^{1}	2.12×10^{1}	1.23×10^{-1}
	EWOA	2.16×10^{1}	2.10×10^{1}	2.11×10^{1}	1.20×10^{-1}
	MSCA	2.14×10^{1}	2.10×10^{1}	2.12×10^{1}	8.18×10^{-2}
	MESOA	2.20×10^{1}	2.15×10^{1}	2.18×10^{1}	1.25×10^{-1}
	EOSMICOA	2.16×10^{1}	2.13×10^{1}	2.15×10^{1}	8.10×10^{-2}
	CSFChOA	**2.16×10^{1}**	**7.74×10^{0}**	**2.07×10^{1}**	**2.83×10^{0}**

From Table 6, we can see that, for F_1, the CSFChOA is the algorithm which can find the theoretical optimum and has the best stability among all of the algorithms. For F_2–F_{10}, although the CSFChOA could not find the theoretical optimum like the other algorithms, the convergence accuracy is better than the standard ChOA. For F_2–F_4, F_7, F_{10}, F_{16}–F_{18}, the optimum value obtained by the CSFChOA is much closer to the theoretical optimum than the other seven algorithms. For F_6, F_8, the convergence ability of the CSFChOA was only worse than the MSCA, but more stable than the MSCA. For F_4, F_6, the standard deviation of the CSFChOA was lower than those of the other seven algorithms. For F_1, F_2, F_3, F_7, the stability of the CSFChOA was in the second place among all of the algorithms. In summary, the CSFChOA has good convergence accuracy and robustness when applied to the challenging CEC2019 test functions.

4. Application analysis of CSFChOA on engineering examples

To further explore the superiority of the CSFChOA when handling practical engineering application cases, two types of nonlinear constrained optimization problems in engineering were selected: the speed reducer design and the three-bar truss design. The ChOA [7], AOA [25], CEHHO [10], EWOA [8], MSCA [9], MESOA [11] and EOSMICOA [24] were used to solve these two problems respectively. The parameter values of these algorithms were selected according to the corresponding literature. The experimental parameters were taken as the population size $N = 30$, the maximum times of iterations $t_{max} = 1000$, and each algorithm was run 30 times independently.

4.1. Speed reducer design

To a mechanical system, the speed reducer is one of the important components of the gear case, and it can be used for a variety of applications. Design of the speed reducer is optimized to minimize the weight of the speed reducer, which is associated with seven variables, i.e., the tooth width *b*, modulus of gear *m*, number of teeth on the pinion *z*, length of the first shaft between bearings l_1, length of the second shaft between bearings l_2, diameter of the first shaft d_1 and diameter of the second shaft d_2. The structure of the speed reducer is schematically shown in Figure 5. The specific mathematical model of the speed reducer is as follows.

Objective function:

$$X = [x_1\ x_2\ x_3\ x_4\ x_5\ x_6\ x_7] = [b\ \ m\ \ z\ \ l_1\ l_2\ d_1\ d_2]$$

$$\begin{gathered} f(X) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) \\ -1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2) \end{gathered} \tag{23}$$

Constraint conditions:

$$g_1(X) = \frac{27}{x_1x_2^2x_3} - 1 \le 0 \tag{24}$$

$$g_2(X) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \le 0 \tag{25}$$

$$g_3(X) = \frac{1.93x_4^3}{x_2x_6^4x_3} - 1 \le 0 \tag{26}$$

$$g_4(X) = \frac{1.93x_5^3}{x_2x_7^4x_3} - 1 \le 0 \tag{27}$$

$$g_5(X) = \frac{\sqrt{(745x_4/x_2x_3)^2 + 16.9\times10^6}}{110x_6^3} - 1 \le 0 \tag{28}$$

$$g_6(X) = \frac{\sqrt{(745x_5/x_2x_3)^2 + 157.5\times10^6}}{85x_7^3} - 1 \le 0 \tag{29}$$

$$g_7(X) = \frac{x_2x_3}{40} - 1 \le 0 \tag{30}$$

$$g_8(X) = \frac{5x_2}{x_1} - 1 \le 0 \tag{31}$$

$$g_9(X) = \frac{x_1}{12x_2} - 1 \le 0 \tag{32}$$

$$g_{10}(X) = \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0 \tag{33}$$

$$g_{11}(X) = \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0 \tag{34}$$

Variable range:

$$2.6 \le x_1 \le 3.6 \tag{35}$$

$$0.7 \le x_2 \le 0.8 \tag{36}$$

$$x_3 \in \{17, 18, 19, ..., 28\} \tag{37}$$

$$7.3 \le x_4 \tag{38}$$

$$x_5 \le 8.3 \tag{39}$$

$$2.9 \le x_6 \le 3.9 \tag{40}$$

$$5 \le x_7 \le 5.5 \tag{41}$$

The solution results of the eight algorithms are shown in Table 7.

The comparison results in Table 7 show that the CSFChOA performs the best in terms of solving the complex optimization problem of the speed reducer design and can achieve the lowest weight; and, the overall optimization effect was much better than the other seven algorithms. The optimal cost was 64.03 lower than the ChOA, 93.31 lower than the AOA, 2636.45 lower than the CEHHO, 107.67 lower than the EWOA, 20.18 lower than the MSCA, 99,999,394.17 lower than the MESOA and 80.28 lower than the EOSMICOA. The performance of the CSFChOA was 100% better than that of the other algorithms. The comparison results verify the superior performance of the CSFChOA in terms of solving the speed reducer design.

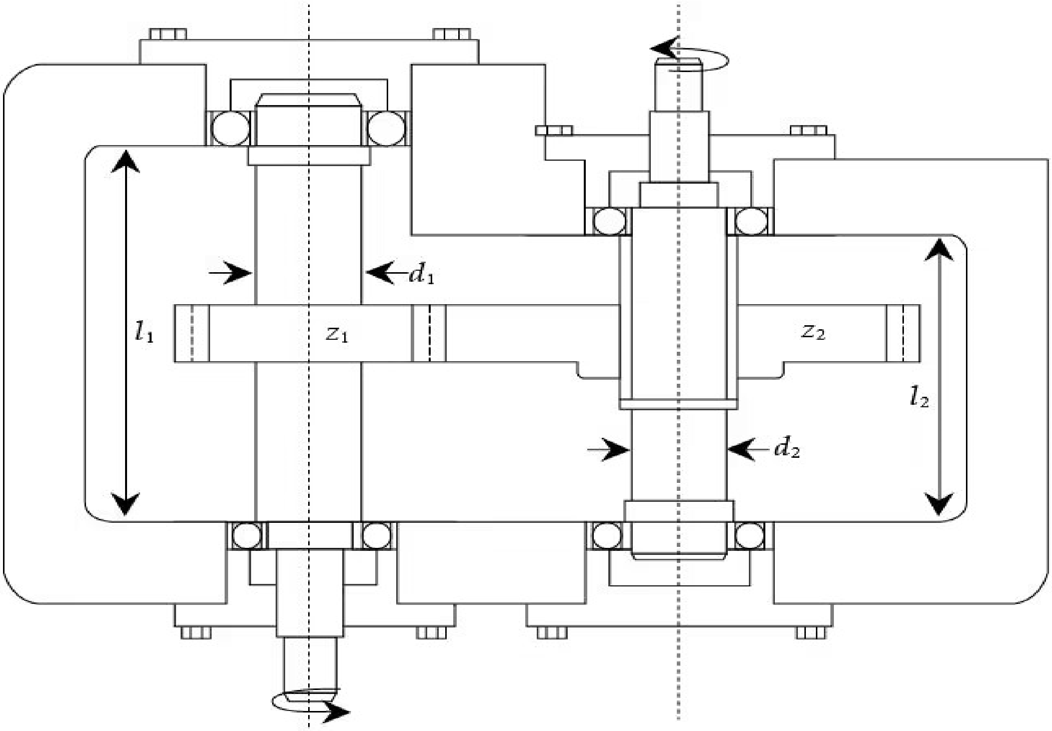


Figure 5. Principle diagram of the speed reducer.

Table 7. Comparison results of eight algorithms in solving the speed reducer design problem.

Algorithm	Optimum variables							Optimal cost
	b	m	p	l_1	l_2	d_1	d_2	
ChOA	3.6000	0.7000	17.0000	7.3000	7.8000	3.4020	5.2942	3081.3530
AOA	3.6000	0.7000	17.0000	7.3000	7.8000	3.3672	5.2883	3110.6266
CEHHO	3.5002	0.7000	17.0000	7.5651	7.8000	3.7735	5.2867	5653.7677
EWOA	3.5015	0.7000	17.0000	7.3000	7.8000	3.3504	5.2867	3124.9913
MSCA	3.5356	0.7000	17.0000	7.3000	7.8000	3.3925	5.2982	3037.5032
MESOA	2.6154	0.7001	17.0132	7.8088	8.0721	2.9132	5.0074	100,002,411.4893
EOSMICOA	3.6000	0.7000	17.0000	7.3000	7.8000	3.3930	5.2943	3097.5938
CSFChOA	**3.5040**	**0.7000**	**17.0000**	**7.3303**	**7.8617**	**3.3652**	**5.2869**	**3017.3183**

4.2. Three-bar truss design

The three-bar truss design is a structural problem in the field of civil engineering; the objective is to minimize the volume of the truss under hydrostatic pressure while satisfying the stress (σ) constraint on each truss member. A schematic model of a three-bar truss is shown in Figure 6, where H is the height of the truss and P is the concentrated force. The variables for this problem are the optimum cross-sectional areas A_1, A_2, which is mathematically modeled as follows.

Objective function:

$$X = [A_1\ A_2] = [x_1\ x_2]$$

$$f(X) = (2\sqrt{2}x_1 + x_2) \times l \tag{42}$$

Constraint conditions:

$$g_1(X) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \le 0 \tag{43}$$

$$g_2(X) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \le 0 \tag{44}$$

$$g_3(X) = \frac{1}{\sqrt{2}x_2 + x_1}P - \sigma \le 0 \tag{45}$$

$$l = 100cm \tag{46}$$

$$P = 2kN / cm^3 \tag{47}$$

$$\sigma = 2kN / cm^3 \tag{48}$$

Variable range:

$$0 \le x_1 \tag{49}$$

$$x_2 \le 1 \tag{50}$$

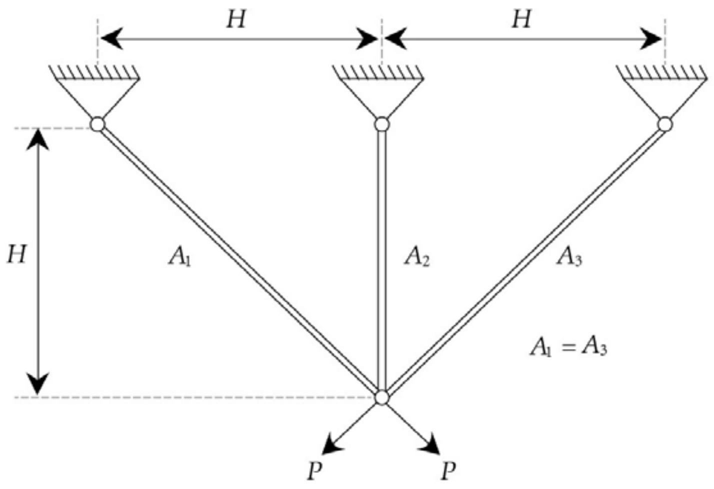


Figure 6. Schematic model of a three-bar truss.

The solution results of the eight algorithms are shown in Table 8.

Table 8. Comparison of the results of eight algorithms for solving the three-bar truss design problem.

Algorithm	Optimum variables		Optimal cost
	A_1	A_2	
ChOA	0.7857	0.4081	263.4646
AOA	0.7890	0.3995	263.9554
CEHHO	0.7861	0.4068	263.4635
EWOA	0.7863	0.4063	263.4639
MSCA	0.7858	0.4077	263.4638
MESOA	0.7862	0.4067	263.4648
EOSMICOA	0.7865	0.4057	282.5938
CSFChOA	**0.7861**	**0.4069**	**263.4634**

The comparison results in Table 8 show that the CSFChOA performs the best in terms of solving the complex optimization problem of the three-bar truss design, and that it can achieve the minimum truss volume and the overall optimization effect much better than the other seven algorithms. The optimal cost was 0.0012 lower than the ChOA, 0.4920 lower than the AOA, 0.0001 lower than the CEHHO, 0.0005 lower than the EWOA, 0.0005 lower than the MSCA, 0.0004 lower than the MESOA and 19.1304 lower than the EOSMICOA. The performance of the CSFChOA was 6.77% better than that of the other algorithms. The comparison results verify the superior performance of the CSFChOA in terms of solving the three-bar truss design problem.

In summary, the testing of the above two well-known engineering constraint problems with different levels of complexity fully demonstrates the strong potential and superiority of the CSFChOA on handling different types of engineering optimization problems, and it demonstrates the robustness of the CSFChOA.

5. Conclusions

To address the problems of slow convergence speed, low accuracy and susceptibility to local optimum traps of the ChOA, a hybrid improvement strategy, i.e., the CSFChOA has been proposed. By combining a chaos initialization method and opposition-based learning initialization strategy, the initial population is generated, which can ensure the diversity of the algorithm at the beginning, as well as improve the convergence speed and optimum searching accuracy. The somersault foraging strategy is used to maintain the population diversity and prevent homogeneity, which can improve ability of the algorithm to jump out of the local optimum. Regarding simulation, the optimization search tests were performed on 23 standard test functions and CEC2019 test functions, and the Wilcoxon rank-sum test was used for statistical analysis. The CSFChOA has been compared with the ChOA and other improved intelligent optimization algorithms. The experimental results show that the CSFChOA was demonstrated to be better than most of the other algorithms in terms of optimum search accuracy, convergence speed and robustness of global optimization in both low-dimensional and high-dimensional experiments. Good results were obtained from the application of the CSFChOA to two complex engineering problems, that is, the design of a three-bar truss and speed reducer; this verifies the feasibility, applicability and superiority of the CSFChOA in practical engineering applications, as well as provides a new idea for solving complex practical engineering problems. However, the CSFChOA has its limitations, like other optimization algorithms, and needs further improvement. It was found during experimental runs that the CSFChOA runs for a long time in high dimensions; we think that this can be solved by introducing a parallel strategy. The research on the human brain has become a hot spot in recent years, and numerous research results have been widely used in clinical diagnosis, artificial intelligence, brain computer interface, etc. After the collection of Electroencephalogram (EEG) signals, frequency components should be extracted, unnecessary components should be removed or noise should be eliminated; therefore, the filtering of EEG signals is very important. The design of the filters can be treated as an optimization problem; we need to find the optimal filter with the most proper specifications, and we are planning to apply the CSFChOA for the optimization of EEG filtering.

Acknowledgments

This work was partially supported by the Shaanxi Key Science and Technology Innovation Team Project (No. 2022TD-26), the Science and Technology Plan of Xi’an city (No. 22GXFW0020), Shaanxi Science and Technology Plan Project (No. 2023YBGY158) and the Fundamental Research Fund for the Central Universities (No. GK202101004).

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. G. Beni, J. Wang, Swarm intelligence in cellular robotic system, in *Robots and Biological Systems: Towards a New Bionics*, (1993), 703–712. https://doi.org/10.1007/978-3-642-58069-7_38
2. L. Brezočnik, I. Fister Jr., V. Podgorelec, Swarm intelligence algorithms for feature selection: a review, *Appl. Sci.*, **8** (2018), 1521. https://doi.org/doi:10.3390/app8091521

3. R. Eeberhart, J. Kennedy, A new optimizer using particle swarm theory, in *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, (1995), 39–43. https://doi.org/10.1109/MHS.1995.494215
4. D. Karaboga, An idea based on honey bee swarm for numerical optimization, *Technical report-tr06, Erciyes University, Engineering Faculty, Computer Engineering Departm*ent, 2005.
5. X. S. Yang, S. Deb, Cuckoosearch via lévy flight, in *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*, (2009), 210–214. https://doi.org/10.1109/NABIC.2009.5393690
6. A. Colorni, M. Dorigo, V. Maniezzo, D. di Elettronica, P. di Milano, P. L. da Vinci, et al., Distributed optimization by ant colonies, in *Proceedings of the First European Conference on Artificial Life*, (1991), 134–142.
7. M. Khishe, M. R. Mosavi, Chimp optimization algorithm, *Expert Syst. Appl.*, **149** (2020), 113338. https://doi.org/10.1016/j.eswa.2020.113338
8. W. Feng, K. Song, An enhanced whale optimization algorithm, *Comput. Simul.*, **37** (2020), 275–279+357.
9. Y. Shen, X. Zhang, X. Fang, X. Wang, A multi-scale sine cosine algorithm for optimization problems, *Control Decis.*, **37** (2022), 2860–2868. https://doi.org/doi:10.13195/j.kzyjc.2021.0513
10. A. Tang, T. Han, D. Xu, L. Xie, Chaotic elite harris hawks optimization algorithm, *J. Comput. Appl.*, **41** (2021), 2265–2272.
11. B. Zhang, Q. He, S. Dai, N. Du, Multi-directional exploring seagull optimization algorithm based on chaotic map, *J. Chin. Mini-Micro Comput. Syst.*, **2021** (2021), 1–10.
12. J. Deng, J. Cao, S. Zhao, Z. Yang, W. Nai, D. Li, Stochastic neighbor embedding based on faure sequence initialized chimp optimization algorithm, in *2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, (2022), 2493–2497. https://doi.org/10.1109/ITAIC54216.2022.9836748
13. Y. Xiao, G. Chen, S. Wang, A modified chimp optimization algorithm for short-term hydrothermal scheduling, in *2021 China Automation Congress (CAC)*, (2021), 647–651. https://doi.org/10.1109/CAC53003.2021.9728072
14. M. Khishe, M. Nezhadshahbodaghi, M. R. Mosavi, D. Martín, A weighted chimp optimization algorithm, *IEEE Access*, **9** (2021), 158508–158539. https://doi.org/10.1109/ACCESS.2021.3130933
15. M. Mansoor, Q. Ling, M. H. Zafar, Short term wind power prediction using Feedforward Neural Network (FNN) trained by a novel Sine-Cosine fused Chimp Optimization Algorithm (SChoA), in *2022 5th International Conference on Energy Conservation and Efficiency (ICECE)*, (2022), 1–6. https://doi.org/10.1109/ICECE54634.2022.9758965
16. H. Jia, K. Sun, W. Zhang, X. Leng, An enhanced chimp optimization algorithm for continuous optimization domains, *Complex Intell. Syst.*, **2021** (2021), 1–18. https://doi.org/10.1007/s40747-021-00346-5
17. G. Dhiman, SSC: A hybrid nature-inspired meta-heuristic optimization algorithm for engineering applications, *Knowledge-Based Syst.*, **222** (2021), 106926. https://doi.org/10.1016/j.knosys.2021.106926
18. G. Dhiman, A. Kaur, Spotted hyena optimizer for solving engineering design problem, in *2017 International Conference on Machine Learning and Data Science (MLDS)*, (2017), 114–119. https://doi.org/10.1109/MLDS.2017.5

19. Z. Chen, K. Zhang, T. H. Chan, X. K. Li, S. B. Zhao, A novel hybrid whale-chimp optimization algorithm for structural damage detection, *Appl. Sci.*, **12** (2022), 9036. https://doi.org/10.3390/app12189036
20. S. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Software*, **95** (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
21. M. Kaur, R. Kaur, N. Singh, A novel hybrid of chimp with cuckoo search algorithm for the optimal designing of digital infinite impulse response filter using high-level synthesis, *Soft Comput.*, **26** (2022), 13843–13867. https://doi.org/10.1007/s00500-022-07410-3
22. X. Zhang, J. Yan, S. Liu, B. Yan, Enhancing the take-off performance of hypersonic vehicles using the improved chimp optimisation algorithm, *Aeronaut. J.*, **2022** (2022), 1–17. https://doi.org/10.1017/aer.2022.70
23. W. Zhao, Z. Zhang, L. Wang, Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications, *Eng. Appl. Artif. Intell.*, **87** (2020), 103300. https://doi.org/10.1016/j.engappai.2019.103300
24. Q. Huang, S. Liu, M. Li, Y. Guo, Multi-strategy chimp optimization algorithm and its application of engineering proble, *Comput. Eng. Appl.*, **58** (2022), 174–183. https://doi.org/10.3778/j.issn.1002-8331.2101-0520
25. L. Abualigah, A. Diabat, S. Mirjalili, M. A. Elaziz, A. H. Gandomi, The arithmetic optimization algorithm, *Comput. Methods Appl. Mech. Eng.*, **376** (2021), 113609. https://doi.org/10.1016/j.cma.2020.113609