Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2021.017310

ARTICLE

A Chaos Sparrow Search Algorithm with Logarithmic Spiral and Adaptive Step for Engineering Problems

Andi Tang, Huan Zhou*, Tong Han and Lei Xie

Aeronautics Engineering College, Air Force Engineering University, Xi'an, 710038, China

*Corresponding Author: Huan Zhou. Email: kgy_zhouh@163.com

Received: 30 April 2021 Accepted: 15 July 2021

ABSTRACT

The sparrow search algorithm (SSA) is a newly proposed meta-heuristic optimization algorithm based on the sparrow foraging principle. Similar to other meta-heuristic algorithms, SSA has problems such as slow convergence speed and difficulty in jumping out of the local optimum. In order to overcome these shortcomings, a chaotic sparrow search algorithm based on logarithmic spiral strategy and adaptive step strategy (CLSSA) is proposed in this paper. Firstly, in order to balance the exploration and exploitation ability of the algorithm, chaotic mapping is introduced to adjust the main parameters of SSA. Secondly, in order to improve the diversity of the population and enhance the search of the surrounding space, the logarithmic spiral strategy is introduced to improve the sparrow search mechanism. Finally, the adaptive step strategy is introduced to better control the process of algorithm exploitation and exploration. The best chaotic map is determined by different test functions, and the CLSSA with the best chaotic map is applied to solve 23 benchmark functions and 3 classical engineering problems. The simulation results show that the iterative map is the best chaotic map, and CLSSA is efficient and useful for engineering problems, which is better than all comparison algorithms.

KEYWORDS

Sparrow search algorithm; global optimization; adaptive step; benchmark function; chaos map

1 Introduction

The optimization problem is a common real-world problem that requires seeking the maximum or minimum value of a given objective function and they can be classified as single-objective optimization problems and multi-objective optimization problems [1,2]. There are two types of methods commonly used for optimization problems. One type is the traditional gradient-based approach. One is the metaheuristic algorithm [3,4]. Generally speaking, the traditional gradient-based methods often encounter difficulties in solving complex engineering problems [5]. The existing research shows that the traditional mathematical or numerical programming methods are difficult to deal with many non-differentiable and discontinuous problems efficiently [6]. In order to overcome these shortcomings, a kind of metaheuristic optimization algorithm is proposed and used to solve global optimization problems. Metaheuristic algorithms are usually divided into three categories: evolutionary algorithms, physics-based algorithms, and swarm-based algorithms.

Evolutionary algorithm is a kind of algorithm inspired by the mechanism of natural evolution. Genetic Algorithm (GA) [7] based on Darwin's theory of survival of the fittest is one of the most famous evolutionary algorithms. There are also some other evolutionary algorithms such as Evolution Strategy (ES) [8], Evolutionary Programming (EP) [9], Differential Evolution (DE) [10] and Biogeography Based Optimization (BBO) [11]. Physical-based algorithms are based on physical concepts to establish optimization models, such as Simulated Annealing (SA) [12], Gravity Search Algorithm (GSA) [13], Nuclear Reaction Optimization (NRO) [14], and Black Hole Algorithm (BHA) [15]. Swarm-based algorithms based on the characteristics of group behavior are the focus of research in recent years. These algorithms establish optimization models by imitating the behavior of gregarious animals [16]. Particle Swarm Optimization (PSO) [17] is the most well-known swarm intelligence optimization algorithm among these algorithms and has been applied to many fields. Other swarm intelligence optimization algorithms include Ant Colony Optimization (ACO) [18], Monarch Butterfly Optimization (MBO) [19], Moth Search Algorithm (MSA) [20], and Harris Hawk Optimization (HHO) [21]. In addition to the algorithms mentioned above, there are more algorithms proposed, such as Earthworm Optimisation Algorithm (EOA) [22], Elephant Herding Optimization (EHO) [23] and Slime Mould Algorithm (SMA) [24]. Besides proposing new algorithms to solve the optimization problems, more researchers also solve them by modifying existing algorithms. Gao et al. [25] propose a new selection mechanism to improve the DE performance and apply it to solve the job-shop scheduling problem. To enhance the population diversity of the equilibrium optimizer, Tang et al. [26] suggested the utilization of distribution estimation strategies and selection pools and perform well in solving the UAV path planning problem. Chen et al. [27] enhanced the performance of neighborhood search algorithm by introducing ad hoc destroy/repair heuristics and a periodic perturbation procedure, with successful solution of the dynamic vehicle routing problem Wang et al. [28] proposed a new newsvendor model and apply a histogram-based distribution estimation algorithm to solve it. However, the no free lunch theory states that no single algorithm can solve all problems well [29]. This motivates us to continuously propose and improve algorithms to be applicable to more problems. SSA is a new swarm-based optimization algorithm based on sparrow foraging principle proposed by XUE in 2020 [30], which has the advantages of simple structure and few control parameters. In SSA, each sparrow finds the best position by looking for food and anti-predation behavior.

However, similar to other metaheuristic algorithms, there are also problems such as reduction of population diversity and early convergence in the late iterations when solving complex optimization problems.

Based on the discussion above, a chaos sparrow search algorithm based on logarithmic spiral search strategy and adaptive step size strategy (CLSSA) is proposed in this paper, which employs three strategies to enhance the global search ability of SSA. In CLSSA, different chaotic maps are used to change the random values of the parameters in the SSA. Logarithmic spiral search strategy is used to expand the search space and enhance population diversity. Two adaptive step size strategies are applied to adjust the development and exploration ability of the algorithm. To verify the performance of CLSSA, 23 benchmark functions and three engineering problems were used for the tests. Simulation results show that the CLSSA proposed in this paper is superior to the existing methods in terms of accuracy, convergence speed and stability.

The rest of this article is organized as follows: Section 2 introduces the principle and structure of SSA. Section 3 introduces the improvement strategy of CLSSA. Section 4 introduces the experimental results and analysis based on benchmark functions and engineering problems. In Section 5, the full text is summarized, and the direction of further research is pointed out.

2 The Basic Sparrow Search Algorithm

SSA is a novel swarm-based optimization algorithm that mainly simulates the process of sparrow foraging. The sparrow foraging process is a kind of discoverer-follower model, and the detection and early warning mechanism is also superimposed. Individuals with good fitness in sparrows are the producers, and other individuals are the followers. At the same time, a certain proportion of individuals in the population are selected for detection and early warning. If a danger is found, these individuals fly away to find new position.

There are producers, followers, and guards in SSA. The location update is per-formed according to their respective rules. The update rules are as follows:

$$\mathbf{X}_{i,j}^{t+1} = \begin{cases} \mathbf{X}_{i,j}^{t} \times exp\left(\dfrac{-i}{\alpha \times \text{iter}_{\max}}\right) & if \;\; R_2 < \text{ST} \\ \mathbf{X}_{i,j}^{t} + Q \times \mathbf{L} & if \;\; R_2 \geq \text{ST} \end{cases} \tag{1}$$

where t indicates the current iteration, $\mathbf{X}_{i,j}^{t}$ represents the value of the j^{th} dimension of the i^{th} sparrow at iteration t. $\text{iter}_{\max}$ is a constant with the largest number of iterations. $\alpha \in (0,1]$ is a random number. $R_2(R_2 \in (0,1))$ and $\text{ST}(\text{ST} \in [0.5,1))$ represent the alarm value and the safety threshold respectively, where R_2 is randomly generated and ST is usually set to 0.8. Q is a random number which obeys normal distribution. $\mathbf{L}$ shows a matrix of $1 \times \text{D}$ for which each element inside is 1.

$$\mathbf{X}_{i,j}^{t+1} = \begin{cases} Q \times exp\left(\dfrac{\mathbf{X}_{worst}^{t} - \mathbf{X}_{i,j}^{t}}{i^2}\right) & if \;\; i > n/2 \\ \mathbf{X}_{p}^{t+1} + |\mathbf{X}_{i,j}^{t+} - \mathbf{X}_{p}^{t+1}| \times \mathbf{A}^{+} \times \mathbf{L} & if \;\; others \end{cases} \tag{2}$$

where $\mathbf{X}_p$ indicates the best position occupied by the discoverer, $\mathbf{X}_{worst}$ indicates the current worst position, and $\mathbf{A}$ is a matrix with a row of multi-dimensional elements of 1 or -1.

$$\mathbf{X}_{i,j}^{t+1} = \begin{cases} \mathbf{X}_{best}^{t} + \beta \times |\mathbf{X}_{i,j}^{t} - \mathbf{X}_{best}^{t}| & if \;\; f_i > f_g \\ \mathbf{X}_{i,j}^{t} + K \times \left(\dfrac{|\mathbf{X}_{i,j}^{t} - \mathbf{X}_{worst}^{t}|}{(f_i - f_w) + \varepsilon}\right) & if \;\; f_i = f_g \end{cases} \tag{3}$$

where $\mathbf{X}_{best}$ is the current global best position, β is a step size control parameter that obeys Gaussian distribution, $K \in (-1,1)$ is a random number, f_i is the fitness of the current sparrow, f_g and f_w is the best fitness and the worst fitness at present, and $\varepsilon$is a constant to avoid zero denominator. The pseudo code of SSA is shown in Algorithm 1:

Algorithm 1: The framework of the SSA.

Input:
$\text{iter}_{\max}$: the maximum iteration
PD: the number of producers
SD: the number of sparrows who perceive the danger
R_2: the alarm value
N: the number of sparrows
Initialize a population of N sparrows and define its relevant parameters.

(Continued)

Algorithm 1: (Continued)

Output: $\mathbf{X}_{best}$, f_g.
1: while ($t < \text{iter}_{\max}$)
2: Rank the Fitness Values and Find the Current Best individual and the current worst individual.
3: $R_2 =$ rand
4: for $i =$ 1: PD
5: Using Eq. (1) update the sparrow's location;
6: end for
7: for $i =$ (PD + 1): n
8: Using Eq. (2) update the sparrow's location;
9: end for
10: for $l =$ 1: SD
11: Using Eq. (3) update the sparrow's location;
12: end for
13: Get the current new location;
14: If the new location is better than before, update it;
15: $t = t + 1$
16: end while
17: return $\mathbf{X}_{best}$, f_g.

3 The Improved Sparrow Search Algorithm

In Section 3, we introduce a new SSA variant called CLSSA, which can improve the performance of the basic SSA. We introduce three strategies to improve the SSA algorithm. Firstly, we use chaotic map sequence to replace the random parameter R_2 of the algorithm. Secondly, we use the combination of logarithmic helix strategy and original search strategy to balance the discoverer's development and exploration ability. finally, we use two adaptive step size strategies to update the alert position and adjust the algorithm exploitation and exploration ability.

3.1 Chaotic Maps

Chaos is a random phenomenon in nonlinear dynamic systems, which is regular and random, and is sensitive to initial conditions and ergodicity. According to these characteristics, chaotic graphs represented by different equations are constructed to update the random variables in the optimization algorithm. Table 1 and Fig. 1 show ten chaotic maps which are used in the experiments. These ten chaotic maps have different effects in generating numerical values. More details about the 10 chaotic maps can be found in the literature [31,32]. Many researchers have demonstrated the effectiveness of chaotic maps in their studies, investigating the contribution of chaotic operators in the HHO [33], Krill Herd Algorithm (KHA) [34] and WOA [35].

3.2 Logarithmic Spiral Strategy

Through experiments, it is found that the original SSA is easy to fall into the local optimum, which leads to premature convergence. As shown in Fig. 2b, each iteration update of its discoverer approaches the individual optimal solution straight line, which has a strong exploitation ability, but loses the exploration of the nearby search space in the process of approaching the optimal individual, the population diversity is reduced, and it is easy to fall into the local optimum. Therefore, we introduce a logarithmic spiral search model [21] to solve this problem. The mathematical

model is described as follows:

$$\mathbf{X}_{pbest}^{t} = |\mathbf{X}_{i,j}^{t} - \mathbf{X}_{pbest}^{t}| \cdot e^{al} \cdot \cos(2\pi\theta) + \mathbf{X}_{pbest}^{t} \tag{4}$$

$$l = 2(1 - t/\text{iter}_{\max}) - 1 \tag{5}$$

where a is constant that determines the shape of the spiral, whose value is 1, l is a parameter that linearly decreases from 1 to -1, and $\mathbf{X}_{pbest}$ is the optimal position of the current iteration individual.

Table 1: Description of the ten chaotic maps used

ID	Mapping type	Function
1	Chebyshev map	$x_{i+1} = \cos(i\cos^{-1}(x_i))$
2	Circle map	$x_{i+1} = \text{mod}(x_i + b - (a/2\pi x_k)), 1, a = 0.5 \ and \ b = 0.2$
3	Gauss map	$x_{i+1} = \begin{cases} 1 & \text{x}_\text{i} = 0 \\ 1/\ \text{mod}\ (x_i, 1) & otherwaise \end{cases}$
4	Iterative map	$x_{i+1} = \sin(a\pi/x_i), \ a = 0.7$
5	Logistic map	$x_{i+1} = ax_i(1 - x_i), \ a = 4$
6	Precewise map	$x_{i+1} = \begin{cases} x_i/p & 0 \le \text{x}_\text{i} < p \\ (x_i - p)/(0.5 - p) & \text{p} \le \text{x}_\text{i} < 0.5 \\ (1 - x_i - p)/(0.5 - p) & 0.5 \le \text{x}_\text{i} < 1 - p \\ (1 - x_i)/(p) & 1 - p \le \text{x}_\text{i} < 1 \end{cases}$
7	Sine map	$x_{i+1} = a/4 \cdot \sin(\pi x_i), \ a = 4$
8	Singer map	$x_{i+1} = \mu(7.86x_i - 23.32{x_i}^2 + 28.75{x_i}^3 - 13.301875{x_i}^4), \mu = 1.07$
9	Sinusoidal map	$x_{i+1} = a{x_i}^2 \sin(\pi x_i), \ a = 2.3$
10	Tent map	$x_{i+1} = \begin{cases} xi/0.7 & \text{xi} < 0.7 \\ 10/3 \times (1 - x_i) & \text{xi} \ge 0.7 \end{cases}$

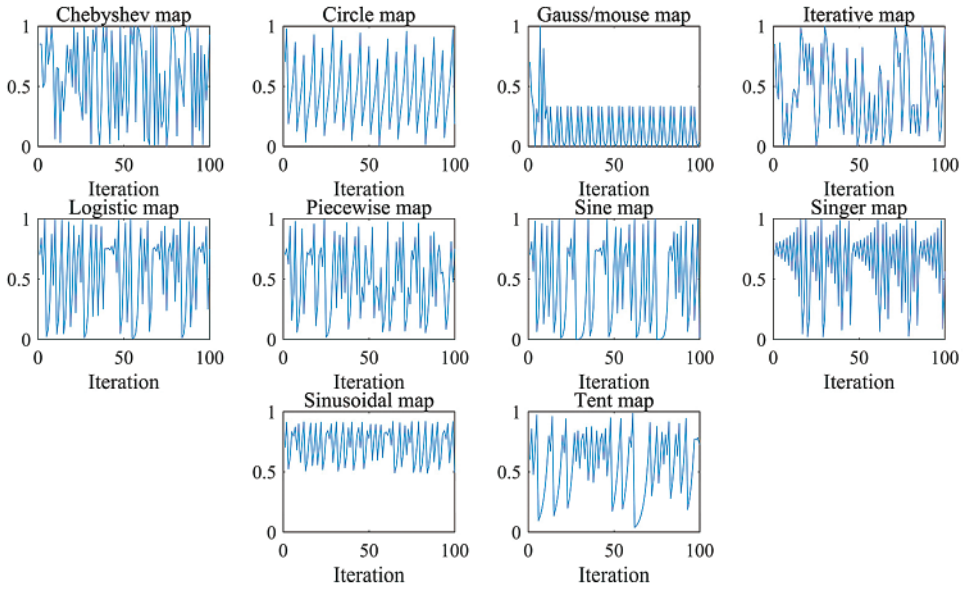

Figure 1: Chaotic maps visualization

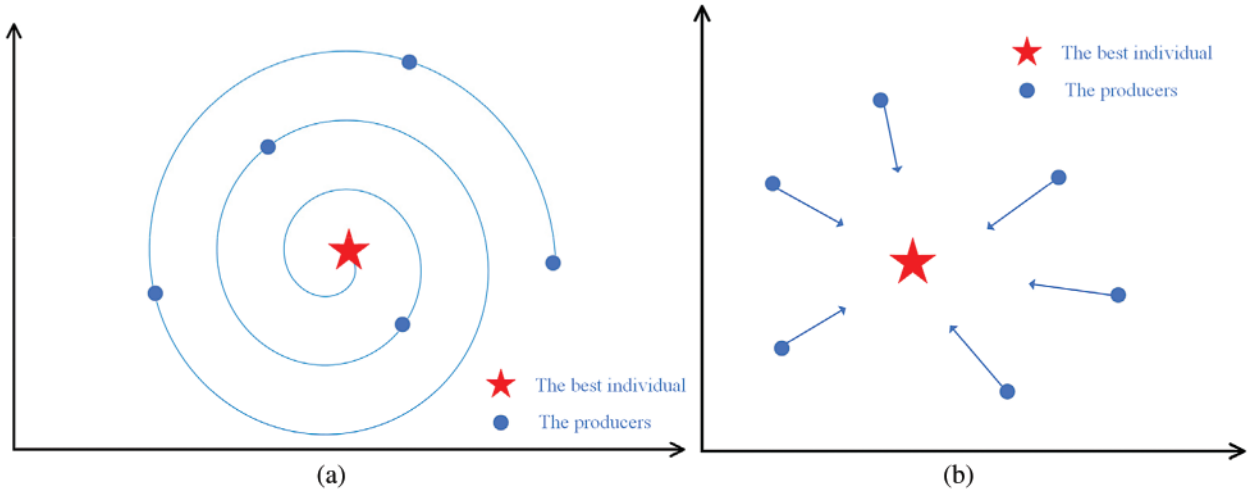

Figure 2: The illustration of two search model (a) the logarithmic spiral search model (b) the original search model

It can be seen from the Fig. 2a that when individuals of each generation update their positions, they gradually approach in a spiral shape, increasing the search for the surrounding space, maintaining the diversity of the population, and enhancing the exploration ability of the algorithm. Based on this analysis, the position update formula is adjusted as follows:

$$\mathbf{X}_{pbest}^{t} = \begin{cases} \mathbf{X}_{pbest}^{t} \times exp\left(\dfrac{-i}{\alpha \times \text{iter}_{\max}}\right) & if\ \ R_2 < \text{ST}\ \ \&\ \ if\ \ R_3 < \text{p} \\ |\mathbf{X}_{i,j}^{t} - \mathbf{X}_{pbest}^{t}| \times e^{al} \times \cos(2\pi\theta) + \mathbf{X}_{pbest}^{t} & if\ \ R_2 < \text{ST}\ \ \&\ \ if\ \ R_3 \geq \text{p} \\ \mathbf{X}_{i,j}^{t} + Q \times \mathbf{L} & if\ \ R_2 \geq \text{ST} \end{cases} \tag{6}$$

where R_3 is a uniformly distributed random number from 0 to 1, p is a constant and the value is 0.5.

3.3 Adaptive Step Strategy

In the SSA, two strategies are used for the location update of the guards. The Gaussian distribution is used to generate the step size for individuals with poor fitness. It can be seen from the Fig. 3 that the probability of the Gaussian distribution producing a smaller step size is higher. Conducive to the global search of the algorithm. The random step strategy is used for individuals with better fitness, and there is still a greater probability of large step in the later iterations, which is not conducive to algorithm convergence. Based on the above analysis, in order to balance the exploitation and exploration capabilities of the algorithm and enhance the convergence speed of the algorithm, an adaptive step size update formula is proposed for two strategies:

$$\beta = \begin{cases} Cauchyrnd() & if\ \ fit_{mean}^{SEL}(t) \geq fit_{mean}^{SEL}(t+1) \\ Gaussianrnd() & if\ \ fit_{mean}^{SEL}(t) < fit_{mean}^{SEL}(t+1) \end{cases} \tag{7}$$

$$K = (2rand - 1) \times (1 - t/\text{iter}_{\max})^{1/2} \tag{8}$$

$$\mathbf{X}_{i,j}^{t+1} = \begin{cases} \mathbf{X}_{best}^{t} + \beta \times |\mathbf{X}_{i,j}^{t} - \mathbf{X}_{best}^{t}| & if\ f_i > f_g \\ \mathbf{X}_{i,j}^{t} + K \times \left(\dfrac{|\mathbf{X}_{i,j}^{t} - B_{worst}^{t}|}{(f_i - f_w) + \varepsilon} \right) & if\ f_i = f_g \end{cases} \tag{9}$$

where fit_{mean}^{SEL} is the average fitness of the dominant population and SEL is the ratio of the dominant population which is 0.35.

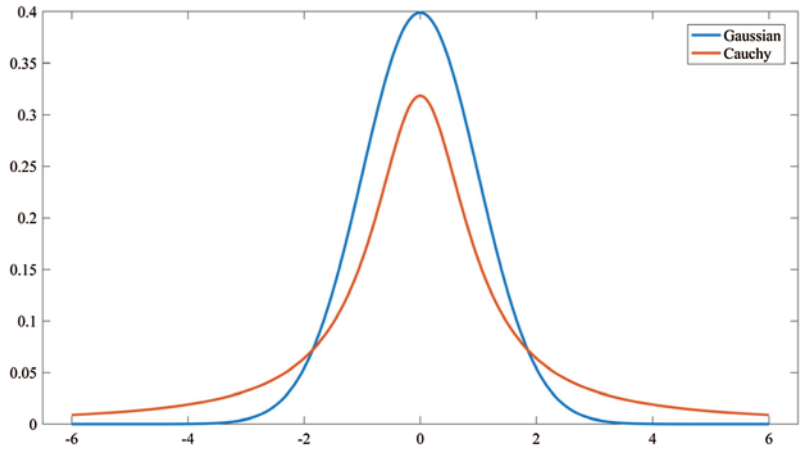

Figure 3: Gauss-cauchy distribution density function

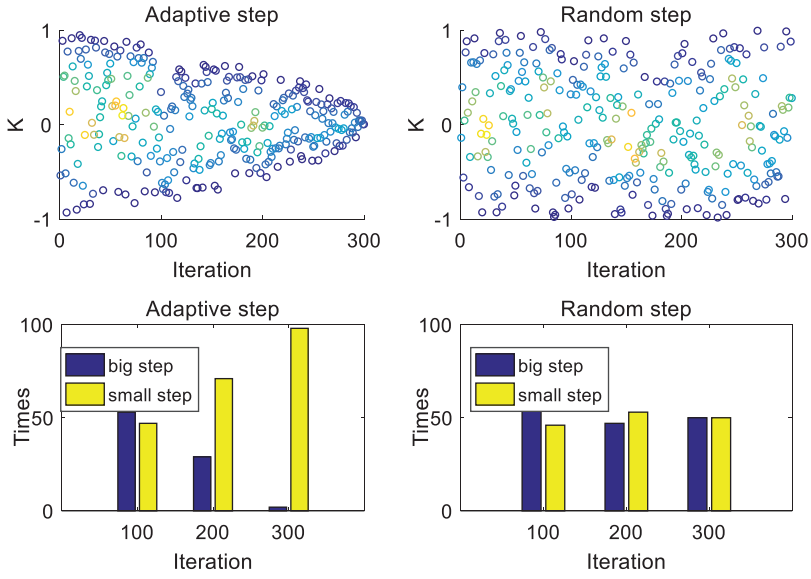

Figure 4: Comparison of new and old step strategies

For the individuals with poor fitness, when the dominant population of the updated sparrow is better than the dominant population of the previous generation, the larger step size of the

Cauchy distribution is used to make the poor individual approach to the dominant population quickly; while when the dominant population of the updated sparrow is weaker than the dominant population of the previous generation, indicating that the renewal effect of this generation is not good, the smaller step size of Gaussian distribution is used to strengthen the search of the space near the individual. For individuals with better fitness, the adaptive step strategy is used. As can be seen from the Fig. 4, the large step size produced by the large probability in the early stage is beneficial for the individual to jump out of the local optimization, maintain the population diversity, increase the probability of small step size in the later stage, and impose only a small disturbance on the dominant individual, which is conducive to the convergence of the algorithm.

The pseudo code and flow chart of CLSSA is shown in Algorithm 2 and Fig. 5.

Algorithm 2: The framework of the CLSSA.

Input:
$iter_{\max}$: the maximum iterations
PD: the number of producers
SD: the number of sparrows who perceive the danger
R_2: the alarm value
N: the number of sparrows
Initialize a population of N sparrows and define its relevant parameters.
Output: $\mathbf{X}_{best}$, fg.
1: while ($t < \text{iter}_{\max}$)
2: Rank the fitness values and find the current best
individual and the current worst individual.
3: Using iterative map update the R_2
4: for $i = 1$: PD
5: Using Eq. (6) update the sparrow's location;
6: end for
7: for $i = (\text{PD} + 1)$: n
8: Using Eq. (2) update the sparrow's location;
9: end for
10: for $l = 1$: SD
11: Using Eq. (9) update the sparrow's location;
12: end for
13: Get the current new location;
14: If the new location is better than before, update it;
15: $t = t + 1$
16: end while
17: return $\mathbf{X}_{best}$, f_g.

4 Experimental Results and Discussion

In Section 4, the benchmark function will be used to evaluate various chaotic map combination algorithms, and then determine which chaotic map sequence to replace the original SSA parameters. Secondly, we need to explore the impact of different improvement strategies in CLSSA on the optimization performance of the algorithm. Finally, we evaluate the performance of the CLSSA and compare the results with other latest algorithms.

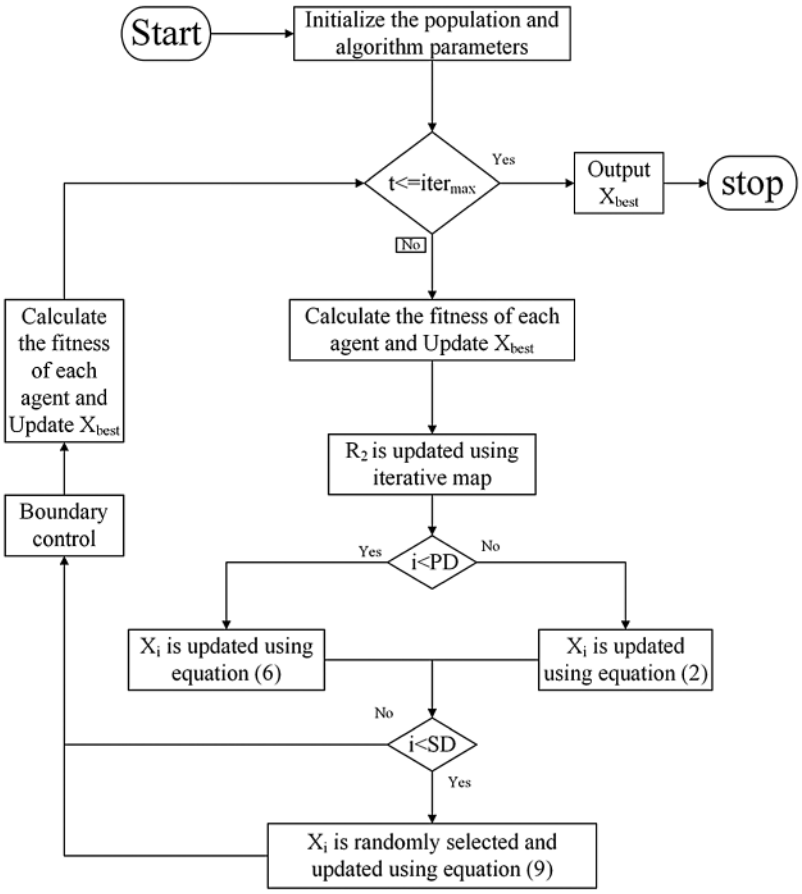

Figure 5: Flow chart of CLSSA

4.1 Introduction of Benchmark Function

In this paper, 23 classical test functions are employed, including 7 unimodal functions, 6 multimodal functions and 10 fixed dimensional functions. The above test functions are all single-objective functions. The unimodal function F1–F7 has only one global optimal value, which is mainly used to test the development ability of the algorithm; the multimodal function has multiple local minima, which can be used to test the exploration ability of the algorithm. The benchmark function is shown in Table 2. The 3D view of each test function is shown in Figs. 6a–6d.

Table 2: Benchmark functions (M: Multimodal, U: Unimodal, S: Separable, N: Non-separable, D: Dimension, Range: Limits of search space, Optimum: Global optimal value)

Test function	Name	Type	*Dim*	Range	Optimum
$f_{01}(x) = \sum_{i=1}^{D} x_i^2$	Sphere	US	30	[−100, 100]	0
$f_{02}(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i-1}^{D} \lvert x_i \rvert$	Schwefel 2.22	UN	30	[−10, 10]	0
$f_{03}(x) = \sum_{i=1}^{D} \left(\sum_{j-1}^{D} x_i \right)^2$	Schwefel 1.2	UN	30	[−100, 100]	0

(continued)

Table 2 (continued)

Test function	Name	Type	Dim	Range	Optimum
$f_{04}(x)=\max_i\{\lvert x_i\rvert, 1\le i\le D\}$	Schwefel 2.21	US	30	$[-100, 100]$	0
$f_{05}(x)=\sum_{i=1}^{D}100(x_{i+1}^2-x_i^2)^2+(x_i-1)^2$	Rosenbrock	UN	30	$[-30, 30]$	0
$f_{06}(x)=\sum_{i=1}^{D}(\lfloor x_i+0.5\rfloor)^2$	Step	US	30	$[-100, 100]$	0
$f_{07}(x)=\sum_{i=1}^{D}ix_i^4+random[0,1)$	Quartic	US	30	$[-1.28, 1.28]$	0
$f_{08}(x)=\sum_{i=1}^{D}-x_i\sin(\sqrt{\lvert x_i\rvert})$	Schwefel 2.26	MS	30	$[-500, 500]$	-418.9829^*D
$f_{09}(x)=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$	Rastrigin	MS	30	$[-5.12, 5.12]$	0
$f_{10}(x)=20+e-20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2})-\exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i))$	Ackley	MS	30	$[-32, 32]$	8.8818e−16
$f_{11}(x)=\frac{1}{4000}\sum_{i=1}^{D}(x_i^2)-\left(\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})\right)+1$	Griewank	MN	30	$[-600, 600]$	0
$f_{12}(x)=\frac{\pi}{D}\{10\sin^2(\pi y_i)+\sum_{i-1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_D-1)\}+\sum_{i-1}^{D}u(x_i,10,100,4)\,y_i=1+\frac{x_i+1}{4}$ $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m\ x_i>a\\0-a<x_i<a\\k(-x_i-a)^m x_i<a\end{cases}$	Penalized	MN	30	$[-50, 50]$	0
$f_{13}(x)=0.1\{\sin^2(3\pi x_i)+\sum_{i=1}^{D}(x_i-1)^2[1+\sin^2(3\pi x_i)]+(x_D-1)^2[1+\sin^2(2\pi x_D)]\}+\sum_{i-1}^{D}u(x_i,5,100,4)$	Penalized2	MN	30	$[-50, 50]$	0
$f_{14}(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$	Foxholes	MS	2	$[-65.53, 65.53]$	0.998004
$f_{15}(x)=\sum_{i=1}^{11}\left(a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}\right)^{-1}$	Kowalik	MS	4	$[-5, 5]$	0.0003075
$f_{16}(x)=4x_1^2-2.1x_1^4+1/3x_1^6+x_1x_2-4x_2^2+x_2^4$	Six Hump Camel Back	MN	2	$[-5, 5]$	−1.03163
$f_{17}(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	Branin	MS	2	$[-5, 10]\times[0, 15]$	0.398
$f_{18}(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]\times[30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	Goldstein Price	MN	2	$[-5, 5]$	3
$f_{19}(x)=-\sum_{i=1}^{4}(c_i\exp(-\sum_{j-1}^{3}a_{ij}(x_j-p_{ij})^2)$	Hartman 3	MN	3	$[0, 1]$	−3.8628
$f_{20}(x)=-\sum_{i=1}^{4}(c_i\exp(-\sum_{j-1}^{6}a_{ij}(x_j-p_{ij})^2)$	Hartman 6	MN	6	$[0, 1]$	−3.32
$f_{21}(x)=-\sum_{i=1}^{5}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	Langermann 5	MN	4	$[0, 10]$	−10.1532
$f_{22}(x)=-\sum_{i=1}^{7}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	Langermann 7	MN	4	$[0, 10]$	−10.4029
$f_{23}(x)=-\sum_{i=1}^{10}[(X-a_i)(X-a_i)^T+c_i]^{-1}$	Langermann 10	MN	4	$[0, 10]$	−10.5364

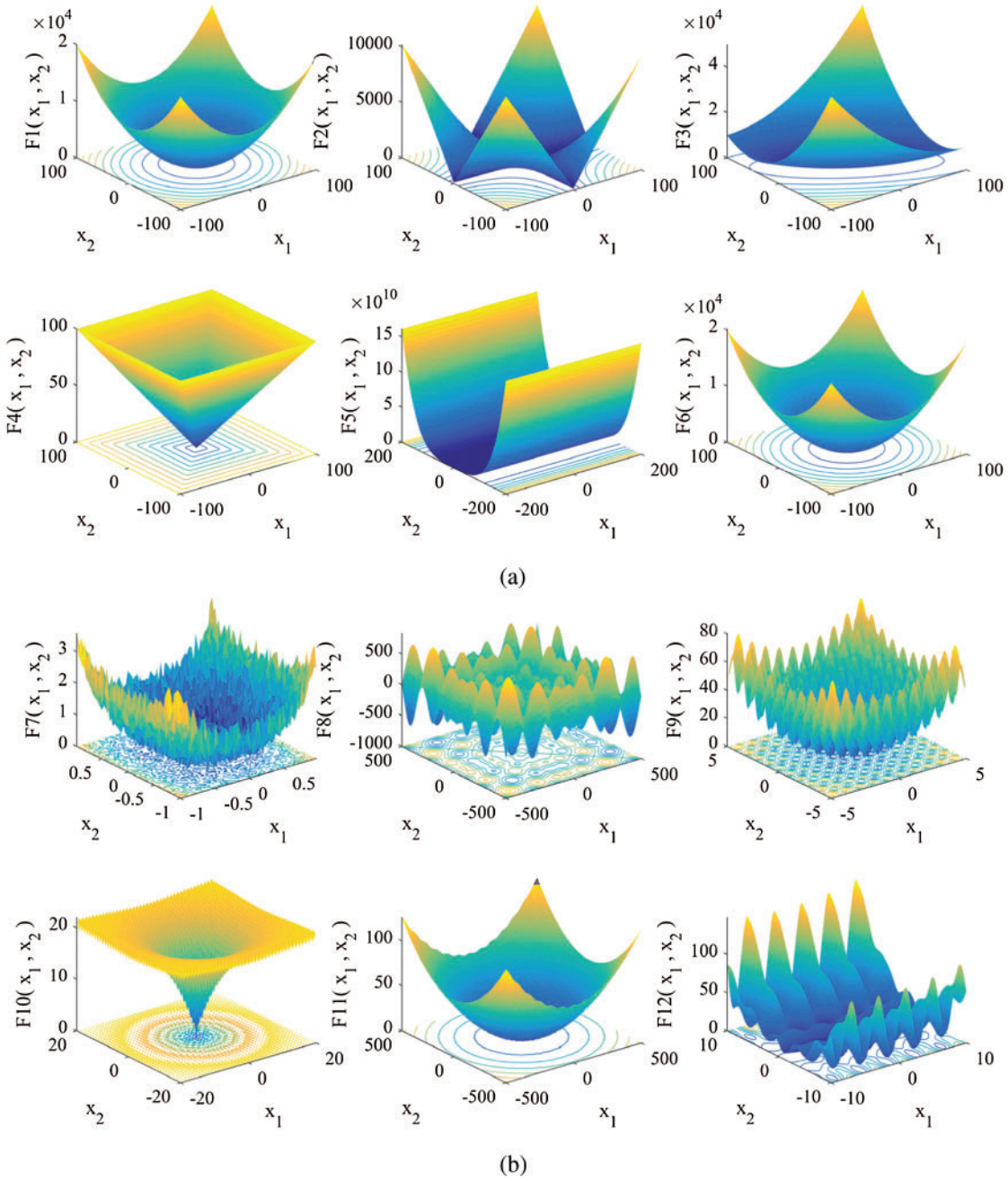

Figure 6: (continued)

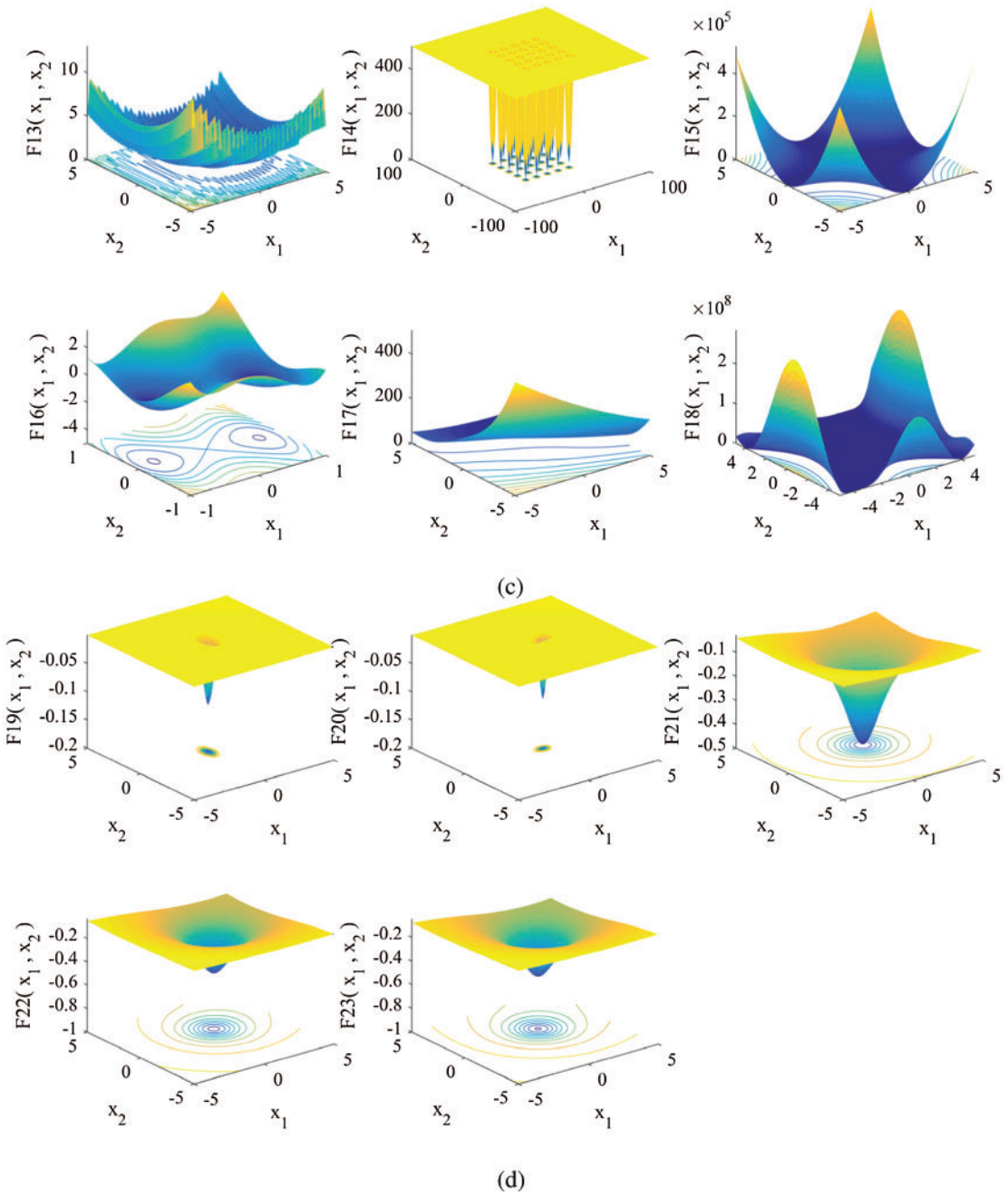

Figure 6: 3D view of benchmark functions (a) 3D view of benchmark F1–F6 (b) 3D view of benchmark F7–F12 (c) 3D view of benchmark F13–F18 (d) 3D view of benchmark F19–F23

4.2 Chaos Map Test

Ten kinds of chaotic maps are combined with SSA algorithm to form new algorithms, the first chaotic map combined algorithm is named SSA-1, the second chaotic map combined algorithm is named SSA-2, and so on. The ten combined algorithms are compared with SSA in the benchmark

function. In order to make a fair comparison, on the same experimental platform, the number of populations is set to 50, and the maximum number of iterations is 300. Except for using chaotic sequences to replace parameter R_2, the other parameters are consistent with the original literature, and the initial values of chaotic mapping sequences are set to 0.7. All the algorithms are implemented in MATLAB R2016a and the test environment is set up on a computer with AMD R7 4700U CPU@1.80 GHz 16GB RAM, running on Windows 10. The average value is used to measure the accuracy of the algorithm, and the standard deviation is used to measure the robustness of the algorithm, so the average value and standard deviation are used to measure the performance of the algorithm. The results are recorded in Table 3. The last row in this table presents the count of the better than, equal to or worse than SSA obtained by each chaotic map over all functions.

It can be observed from Table 3 that SSA-8 (Singer map) outperforms or equals SSA in 14 test functions. SSA-3 (Gauss map), SSA-9 (Sinusoidal map) and SSA-10 (Tent map) perform better than or equal to SSA in 15 test functions. SSA-5 (Logistic map), SSA-6 (Precewise map) and SSA-7 (Sine map) perform better than or equal to SSA in 16 test functions. SSA-1 (Chebyshev map), SSA-2 (Circle map) and SSA-4 (Iterative map) are superior or equal to SSA in 17 test functions.

In order to further analyze the optimization ability of the eleven algorithms, the results of these algorithms in each test function are compared and sorted according to the mean value of Table 3. The results are shown in Table 4, and the average sorting results of each algorithm in the last behavior of the table. SSA-4 ranks first, indicating that iterative mapping is the best alternative to the original parameter R_2. In addition to SSA-4, SSA-2, SSA-3, SSA-5, and SSA-7 also rank higher than SSA. The above analysis shows that using chaotic map sequence to replace the random parameter R_2 in the algorithm can better improve the algorithm's optimization performance, and each chaotic map sequence has different improvement effects on the algorithm. In order to visually show the performance of each algorithm in different test functions, a block diagram is used to plot the ranking results in Table 4, as shown in Fig. 7. The larger the area of the circle in the Fig. 7, the darker the color, indicating that the algorithm has a stronger performance in this test function, and the numbers in the figure indicate the ranking of each algorithm in each test function. It can be seen from Fig. 7 that SSA-4 performs better in the test function, and only performs poorly on F12 and F14.

In order to further prove the effectiveness of using chaotic sequences to replace SSA algorithm parameter, Figs. 8 and 9 list the convergence curves and box plots of eleven algorithms. It can be seen from Fig. 8 that SSA performs generally in each test function, and all chaotic mapping combination algorithms are better than SSA in convergence speed and convergence accuracy. The box diagram is used to show the distribution of the solutions of each algorithm. It can be seen from Fig. 9 that the optimal, median and worst values of the improved algorithm are better than those of SSA in most functions.

Combined with the above analysis, the chaotic mapping sequence can promote the improvement of SSA performance, and iterative mapping has the best effect on improving the performance of the SSA. Therefore, in the next part of the CLSSA performance test, the iterative mapping sequence is used to replace the random value parameter R_2 in the SSA.

Table 3: Results of 10 chaotic maps on all benchmark functions on SSA

ID		SSA	SSA-1	SSA-2	SSA-3	SSA-4	SSA-5	SSA-6	SSA-7	SSA-8	SSA-9	SSA-10
F1	Mean	1.72E-129	3.58E-114	7.16E-147	5.43E-156	5.83E-128	9.76E-116	6.93E-112	6.58E-124	2.74E-89	7.39E-94	1.39E-120
	Std	9.40E-129	1.96E-113	2.73E-146	2.45E-155	3.19E-111	5.35E-115	3.76E-127	3.60E-123	1.50E-88	4.05E-93	7.64E-120
F2	Mean	1.78E-53	4.33E-54	2.22E-69	3.61E-75	1.77E-66	4.26E-56	1.21E-71	1.43E-62	5.11E-38	8.90E-52	1.02E-60
	Std	9.73E-66	1.49E-53	1.22E-68	1.98E-74	9.71E-53	2.30E-55	6.45E-71	7.68E-62	2.80E-37	3.42E-51	5.60E-60
F3	Mean	1.04E-88	1.82E-71	9.32E-90	1.84E-116	4.05E-82	1.12E-83	7.00E-91	3.29E-80	4.80E-59	1.63E-72	1.32E-96
	Std	5.70E-88	9.96E-82	5.10E-89	1.00E-115	2.22E-70	6.15E-83	3.83E-90	1.80E-79	2.63E-58	8.93E-72	5.56E-96
F4	Mean	9.18E-79	2.12E-60	2.77E-73	7.99E-97	5.34E-66	4.10E-54	3.08E-61	2.41E-61	5.70E-46	2.36E-47	8.31E-64
	Std	5.03E-78	1.16E-59	1.52E-72	4.02E-96	2.92E-65	2.08E-53	1.68E-60	1.32E-60	3.12E-45	1.30E-46	3.27E-63
F5	Mean	1.65E-04	1.50E-04	3.15E-04	1.79E-04	1.24E-04	1.16E-04	1.20E-04	1.21E-04	1.09E-04	8.45E-05	1.11E-04
	Std	3.36E-04	5.73E-04	7.26E-04	4.46E-04	2.16E-04	1.95E-04	1.97E-04	2.51E-04	1.90E-04	1.96E-04	2.00E-04
F6	Mean	5.81E-08	8.00E-08	3.81E-08	3.36E-08	4.50E-08	5.23E-08	6.19E-08	1.85E-07	2.27E-07	6.58E-08	5.57E-08
	Std	9.15E-08	1.52E-07	5.87E-08	6.62E-08	9.31E-08	8.24E-08	1.55E-07	4.43E-07	4.66E-07	1.03E-07	1.37E-07
F7	Mean	3.49E-04	4.32E-04	4.46E-04	4.14E-04	3.36E-04	5.13E-04	4.15E-04	5.22E-04	4.37E-04	5.32E-04	6.03E-04
	Std	2.81E-04	4.05E-04	3.12E-04	3.93E-04	2.85E-04	4.20E-04	4.29E-04	4.80E-04	3.49E-04	3.61E-04	4.67E-04
F8	Mean	-8.19E+03	-7.93E+03	-8.15E+03	-8.16E+03	-8.21E+03	-8.21E+03	-8.12E+03	-8.03E+03	-8.09E+03	-8.27E+03	-8.08E+03
	Std	6.62E+02	7.21E+02	6.60E+02	6.95E+02	7.76E+02	5.45E+02	5.86E+02	4.94E+02	7.59E+02	6.80E+02	6.19E+02
F9	Mean	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F10	Mean	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F11	Mean	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F12	Mean	3.16E-09	3.05E-09	4.30E-09	1.68E-09	4.77E-09	1.78E-09	7.68E-10	4.42E-09	2.99E-09	5.57E-09	6.23E-09
	Std	5.79E-09	5.65E-09	1.41E-08	3.14E-09	9.89E-09	2.65E-09	1.53E-09	8.63E-09	5.75E-09	1.60E-08	1.29E-08
F13	Mean	5.12E-08	9.81E-08	2.00E-08	6.78E-08	3.39E-08	1.36E-07	1.31E-08	3.03E-08	3.99E-08	3.50E-08	2.29E-08
	Std	1.50E-07	3.33E-07	4.66E-08	1.60E-07	5.96E-08	3.90E-07	2.01E-08	5.49E-08	6.15E-08	5.93E-08	5.03E-08
F14	Mean	4.51E+00	5.55E+00	6.19E+00	4.95E+00	5.87E+00	5.93E+00	5.80E+00	3.15E+00	5.54E+00	5.32E+00	7.10E+00
	Std	5.07E+00	5.44E+00	5.79E+00	5.56E+00	5.57E+00	5.52E+00	5.61E+00	4.37E+00	5.46E+00	5.35E+00	5.47E+00
F15	Mean	3.08E-04	3.08E-04	3.08E-04	3.18E-04	3.08E-04	3.08E-04	3.08E-04	3.08E-04	3.08E-04	3.08E-04	3.08E-04
	Std	4.44E-08	9.86E-07	4.64E-07	5.56E-05	1.27E-06	5.17E-07	2.58E-06	4.41E-07	8.24E-07	3.34E-06	2.78E-07
F16	Mean	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00
	Std	5.53E-16	6.12E-16	5.90E-16	5.76E-16	5.53E-16	5.98E-16	5.68E-16	5.61E-16	5.98E-16	6.12E-16	6.12E-16
F17	Mean	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F18	Mean	3.90E+00	3.00E+00	3.00E+00	3.90E+00	3.00E+00	3.00E+00	5.70E+00	3.90E+00	6.60E+00	3.00E+00	4.80E+00
	Std	4.93E+00	2.87E-15	2.11E-15	4.93E+00	2.11E-15	1.84E-15	8.24E+00	4.93E+00	9.34E+00	2.51E-15	6.85E+00
F19	Mean	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00	-3.86E+00
	Std	2.42E-15	2.39E-15	2.39E-15	2.34E-15	2.39E-15	2.37E-15	2.36E-15	2.39E-15	2.40E-15	2.34E-15	2.48E-15
F20	Mean	-3.24E+00	-3.27E+00	-3.27E+00	-3.25E+00	-3.26E+00	-3.25E+00	-3.25E+00	-3.29E+00	-3.29E+00	-3.26E+00	-3.28E+00
	Std	5.70E-02	6.03E-02	5.99E-02	5.83E-02	6.05E-02	5.92E-02	5.99E-02	5.54E-02	5.54E-02	6.05E-02	5.83E-02
F21	Mean	-8.79E+00	-9.81E+00	-8.62E+00	-7.94E+00	-9.64E+00	-9.81E+00	-9.13E+00	-8.79E+00	-9.81E+00	-9.47E+00	-9.13E+00
	Std	2.29E+00	1.29E+00	2.38E+00	2.57E+00	1.56E+00	1.29E+00	2.07E+00	2.29E+00	1.29E+00	1.76E+00	2.07E+00
F22	Mean	-8.81E+00	-1.02E+01	-8.45E+00	-8.28E+00	-9.87E+00	-1.00E+01	-8.99E+00	-1.02E+01	-1.00E+01	-9.87E+00	-9.69E+00
	Std	2.48E+00	9.70E-01	2.61E+00	2.65E+00	1.62E+00	1.35E+00	2.39E+00	9.70E-01	1.35E+00	1.62E+00	1.84E+00
F23	Mean	-9.82E+00	-1.05E+01	-9.82E+00	-9.09E+00	-1.04E+01	-1.02E+01	-9.82E+00	-9.82E+00	-1.05E+01	-9.82E+00	-9.82E+00
	Std	1.87E+00	9.27E-04	1.87E+00	2.43E+00	9.87E-01	1.37E+00	1.87E+00	1.87E+00	7.86E-07	1.86E+00	1.87E+00
'+/=/–			8/9/6	9/8/6	8/7/8	**11/6/6**	10/6/7	10/6/7	7/9/9	8/6/9	9/6/8	9/6/8

Table 4: Ranking of 11 algorithms in the benchmark function

ID	SSA	SSA-1	SSA-2	SSA-3	SSA-4	SSA-5	SSA-6	SSA-7	SSA-8	SSA-9	SSA-10
F1	3	8	2	1	4	7	9	5	11	10	6
F2	9	8	3	1	4	7	2	5	11	10	6
F3	5	10	4	1	7	6	3	8	11	9	2
F4	2	8	3	1	4	9	7	6	11	10	5
F5	9	8	11	10	7	4	5	6	2	1	3
F6	6	9	2	1	3	4	7	10	11	8	5
F7	2	5	7	3	1	8	4	9	6	10	11
F8	4	11	6	5	3	2	7	10	8	1	9
F9	1	1	1	1	1	1	1	1	1	1	1
F10	1	1	1	1	1	1	1	1	1	1	1
F11	1	1	1	1	1	1	1	1	1	1	1
F12	6	5	7	2	9	3	1	8	4	10	11
F13	8	10	2	9	5	11	1	4	7	6	3
F14	2	6	10	3	8	9	7	1	5	4	11
F15	1	8	5	11	7	3	9	4	6	10	2
F16	1	1	1	1	1	1	1	1	1	1	1
F17	1	1	1	1	1	1	1	1	1	1	1
F18	7	5	1	7	3	3	10	6	11	2	9
F19	1	1	1	1	1	1	1	1	1	1	1
F20	11	5	4	10	6	9	8	1	2	7	3
F21	8	2	10	11	4	1	7	9	3	5	6
F22	9	1	10	11	6	3	8	2	4	5	7
F23	10	2	9	11	3	4	8	7	1	5	6
Mean ranks	4.69	5.08	4.43	4.52	3.91	4.30	4.73	4.65	5.21	5.17	4.82

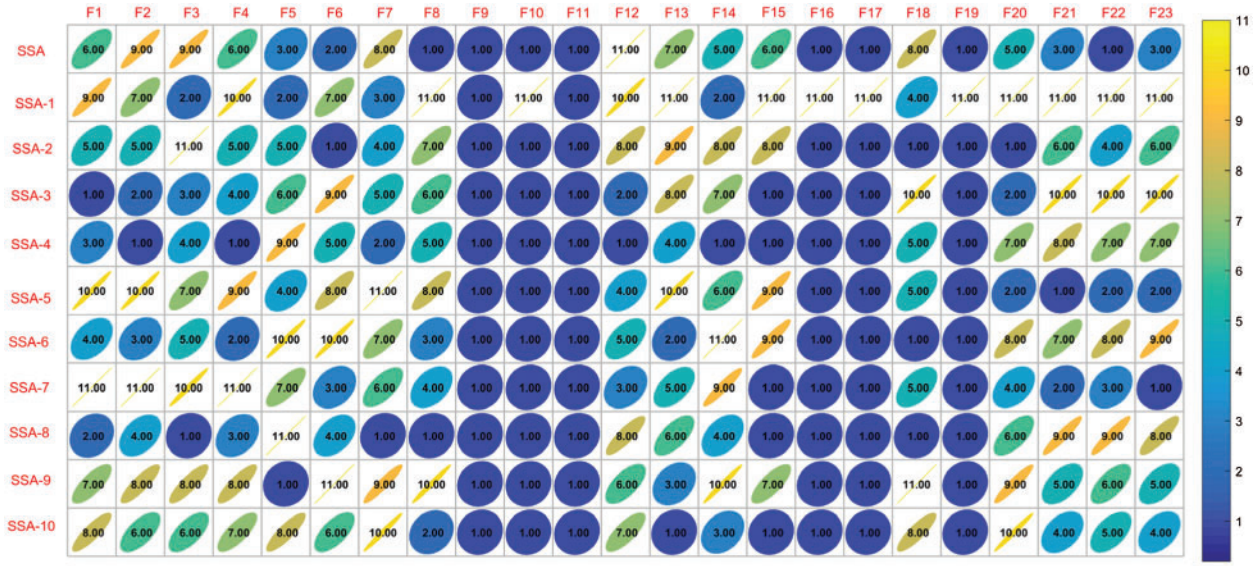

Figure 7: Block diagram of algorithm ranking

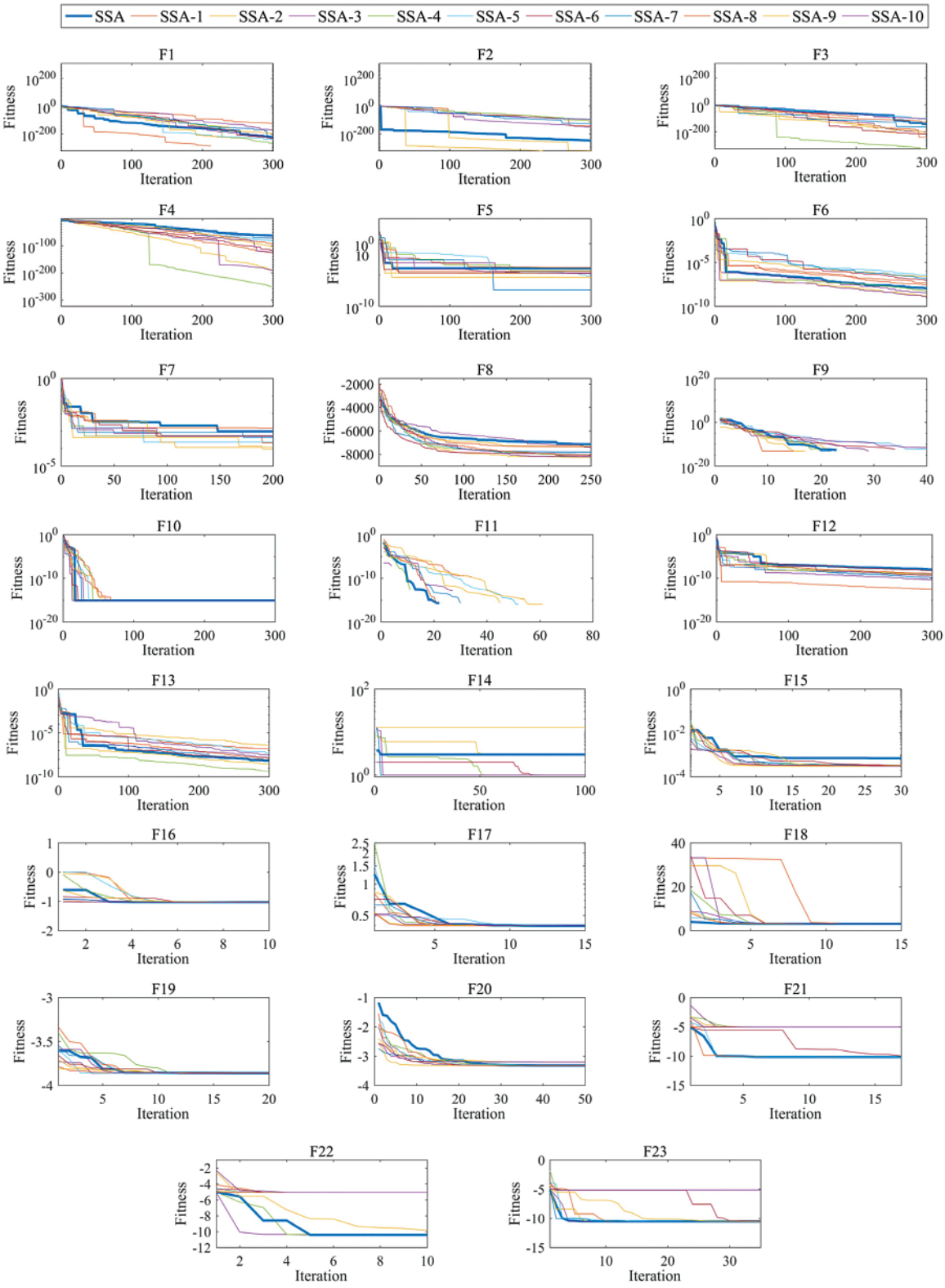

Figure 8: Convergence graphs of 11 algorithms on 23 representative functions

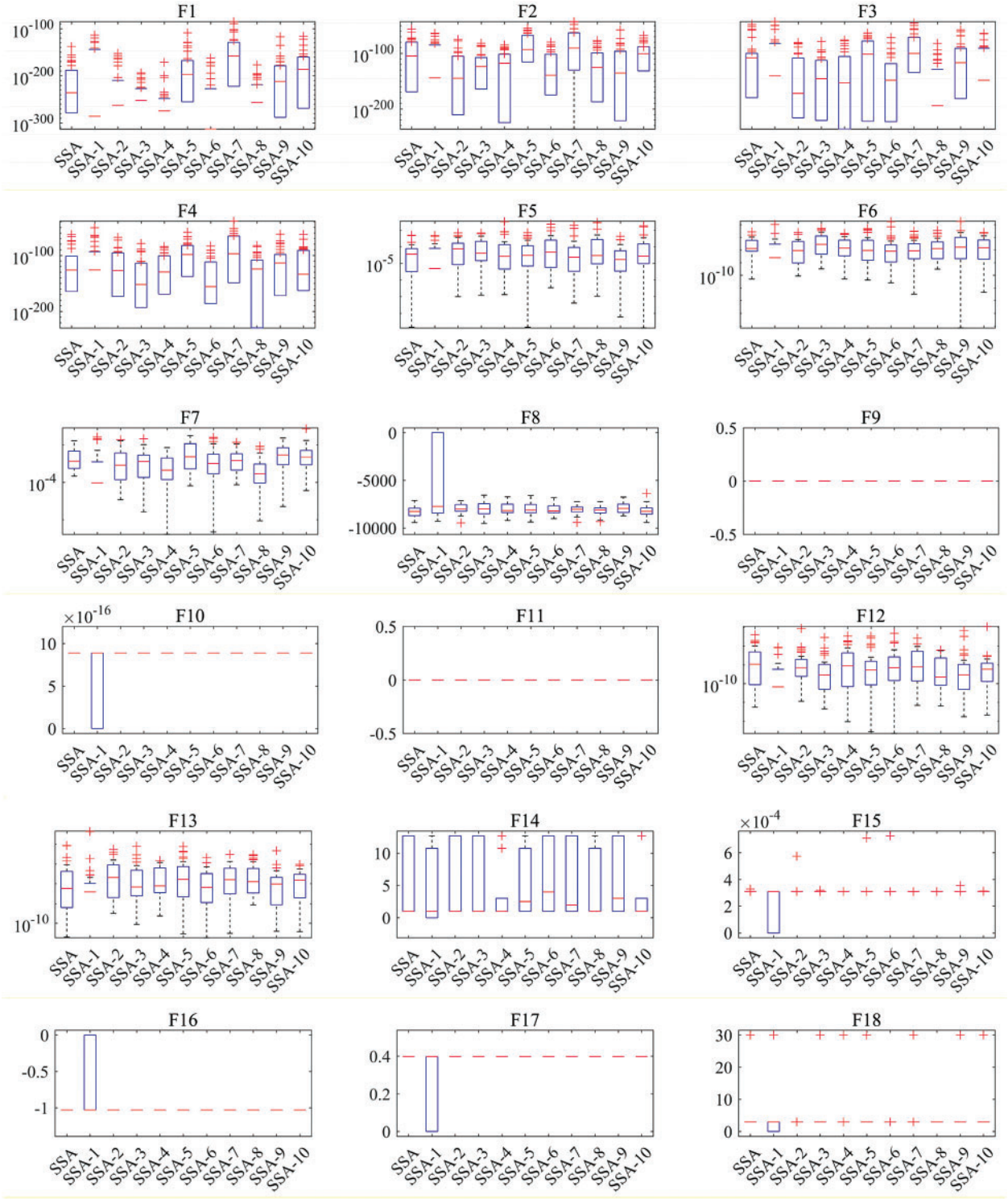

Figure 9: (continued)

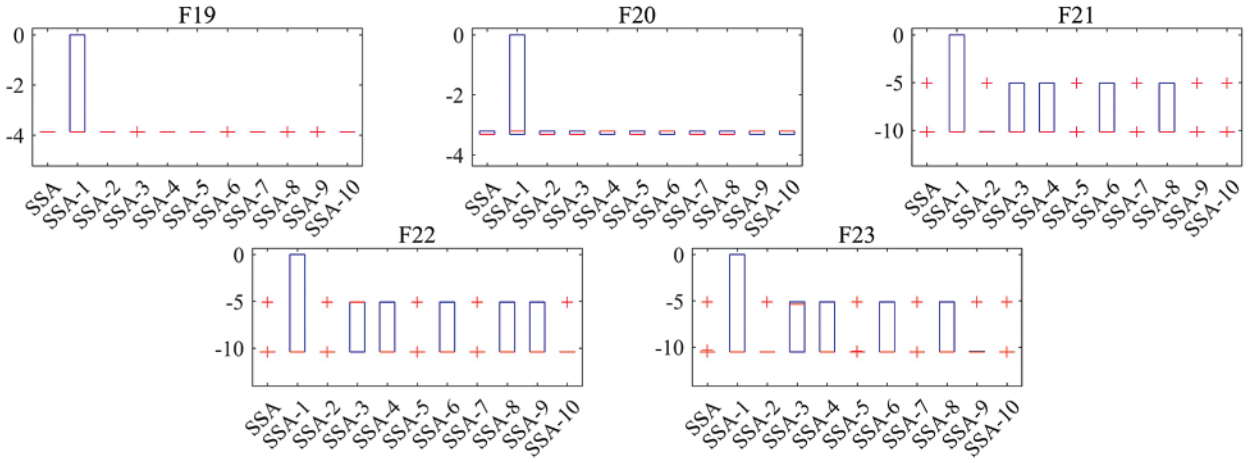

Figure 9: Box diagrams of solutions obtained by 11 algorithms on 23 benchmark functions with 30 independent runs

4.3 Comparison of Different Improvement Strategies

As mentioned above, this paper mainly uses three strategies to improve SSA, so three different derivative algorithms are designed to evaluate the impact of these three strategies on the algorithm. These three derivation algorithms are obtained by removing the corresponding improvement strategy from CLSSA. CLSSA-1 removes both logarithmic spiral strategy and adaptive step strategy; CLSSA-2 removes chaotic map and adaptive step strategy at the same time; CLSSA-3 removes chaotic map strategy and logarithmic spiral strategy at the same time. 23 benchmark functions are used to compare the performance of the three derived algorithms with SSA and CLSSA. Each algorithm runs 30 times independently on each test function, and the statistical average results are shown in Tables 5 and 6 show the ranking of each algorithm in the test function. Obviously, CLSSA which includes all the improvement strategies, performed best, ranking first on average. The performance of the derived algorithm with one improved strategy is better than that of SSA. From the specific optimization results and sorting table provided by the Table 6, the chaotic mapping strategy mainly improves the development ability, while the logarithmic spiral strategy enhances the exploration ability of the algorithm, while the adaptive step strategy enhances the exploitation ability and exploration ability of the algorithm to some extent. The above analysis proves the effectiveness of each improvement strategy.

4.4 Performance Test of CLSSA

In order to verify the performance of CLSSA, the proposed CLSSA is compared with the SSA, WOA, BSO [36], PSO, GSA, HHO, GWO [37], SCA [38], MVO [39], MFO [40], BBO, FPA [41] Flower pollination algorithm for global optimization. The experimental environment is the same as the previous article, the number of populations is set to 50, the maximum number of iterations is 300, and the parameters of each algorithm are consistent with the original literature. Meanwhile, to reduce the influence of randomness on the experimental results, all algorithms need to run 30 times independently. Table 7 lists the best fitness, mean fitness and standard deviation, in which the best mean fitness is marked in bold.

Table 5: Results of 10 different derived algorithms on all benchmark functions

Function	SSA	CLSSA-1	CLSSA-2	CLSSA-3	CLSSA
F1	5.16E-109	0.00E+00	6.10E-157	5.56E-235	6.90E-201
F2	6.02E-67	1.26E-144	2.07E-70	1.56E-124	1.40E-110
F3	4.72E-95	2.73E-219	3.83E-109	1.15E-234	7.26E-159
F4	1.42E-69	9.39E-156	1.07E-79	5.59E-138	5.99E-100
F5	1.02E-04	3.44E-04	1.94E-06	3.06E-04	1.79E-05
F6	5.09E-08	6.23E-08	4.87E-10	9.87E-07	2.97E-09
F7	5.31E-04	2.08E-04	3.80E-04	2.79E-04	3.09E-04
F8	−8.13E+03	−7.75E+03	−7.78E+03	−8.65E+03	−8.48E+03
F9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F10	8.88E-16	8.88E-16	8.88E-16	8.88E-16	8.88E-16
F11	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F12	1.73E-09	3.65E-08	2.36E-11	4.41E-08	7.08E-10
F13	6.66E-08	7.19E-08	7.27E-09	5.72E-07	1.70E-09
F14	5.80E+00	3.01E+00	1.20E+00	2.37E+00	1.52E+00
F15	3.08E-04	3.08E-04	3.35E-04	3.08E-04	3.08E-04
F16	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00	−1.03E+00
F17	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01
F18	3.00E+00	3.90E+00	4.80E+00	6.60E+00	3.00E+00
F19	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00	−3.86E+00
F20	−3.27E+00	−3.26E+00	−3.27E+00	−3.28E+00	−3.27E+00
F21	−8.96E+00	−8.80E+00	−1.02E+01	−9.81E+00	−1.02E+01
F22	−9.34E+00	−8.28E+00	−1.04E+01	−1.00E+01	−1.04E+01
F23	−9.63E+00	−8.91E+00	−1.04E+01	−1.05E+01	−1.05E+01

As shown in Table 7, when solving the unimodal test functions F1–F7, CLSSA can stably converge to the optimal value in F1–F4 and F6, and the performance is better than the comparison algorithm. CLSSA could not obtain the optimal value of F5, but it was 19 orders of magnitude higher than SSA. HHO perform best on F7, with CLSSA in second position. In the unimodal test functions F1–F7, CLSSA is better than SSA, indicating that the proposed chaotic map sequence substitution strategy can effectively improve the local search ability of the algorithm.

When solving the multimodal test functions F8–F13, GSA, PSO, BBO and MFO outperform CLSSA in solving F8. For F9–F11, CLSAA, SSA, HHO can all stably converge to the optimal value. WOA can obtain the optimal value, but it is not stable. The CLSSA has the highest accuracy for F12–F13, with optimal values improved by 17 and 11 orders of magnitude compared to SSA. When solving the fixed-dimensional multimodal functions F14–F23, the CLSSA performs poorly for F14, outperforming only SSA, WOA, GSA, GWO and BBO. For the F15, optimal values can be obtained for CLSSA and SSA, but CLSSA is more stable than SSA. All algorithms have similar performance at F16, and all can obtain optimal values. GSA is the most stable and CLSAA is the second most stable. The CLSSA outperforms WOA, HHO, GSA, CSA, MVO, BBO and FPA for F17, with performance comparable to other algorithms. As for F18, the stability of CLSSA is only weaker than BSO, PSO, and GSA. The CLSSA outperforms all comparison algorithms for F19, F21 and F23. The GSA performs best for F22, with the CLSSA second

best. In all multimodal test functions, CLSSA performs better than SSA, which shows that the logarithmic spiral strategy proposed in this paper can significantly improve the performance of algorithm exploration.

Table 6: Ranking of 5 algorithms in the benchmark function

Function	SSA	CLSSA-1	CLSSA-2	CLSSA-3	CLSSA
F1	5	1	4	2	3
F2	5	1	4	2	3
F3	5	2	4	1	3
F4	5	1	4	2	3
F5	3	5	1	4	2
F6	3	4	1	5	2
F7	5	1	4	2	3
F8	3	5	4	1	2
F9	1	1	1	1	1
F10	1	1	1	1	1
F11	1	1	1	1	1
F12	3	4	1	5	2
F13	3	4	2	5	1
F14	5	4	1	3	2
F15	2	4	5	1	3
F16	1	1	1	1	1
F17	1	1	1	1	1
F18	2	3	4	5	1
F19	1	1	1	1	1
F20	4	5	2	1	3
F21	4	5	1	3	1
F22	4	5	2	3	1
F23	4	5	3	2	1
Mean ranks	3.086957	2.826087	2.304348	2.304348	1.826087

Table 7: Results and comparison of different algorithms for 23 benchmark functions

ID	Index	WOA	BSO	PSO	SSA	GSA	HHO	GWO	SCA	MVO	MFO	BBO	PFA	CLASSA
F1	Best	2.0E-42	6.9E+02	2.1E-02	9.9E-120	3.9E+01	4.9E-63	1.1E-18	1.1E+02	2.1E+00	1.2E+02	2.8E+00	4.4E+03	2.8E-204
	Mean	9.7E-52	2.8E+02	4.4E-04	0.0E+00	4.5E-17	3.8E-79	3.9E-20	2.0E+00	1.2E+00	2.8E+01	1.4E+00	2.2E+03	0.0E+00
	Std	1.1E-41	4.5E+02	4.2E-02	5.4E-119	4.1E+01	2.6E-62	9.0E-19	1.4E+02	5.7E-01	7.7E+01	7.4E-01	1.5E+03	0.0E+00
F2	Best	1.0E-29	4.9E+00	2.3E-01	2.9E-67	2.7E-02	8.6E-33	1.9E-11	1.2E-01	1.9E+01	2.1E+01	5.3E-01	5.6E+01	1.2E-105
	Mean	3.0E-34	4.9E-01	1.1E-02	0.0E+00	2.8E-08	3.7E-39	4.2E-12	1.6E-02	5.8E-01	2.8E+00	3.4E-01	3.3E+01	0.0E+00
	Std	2.2E-29	4.1E+00	2.0E-01	1.6E-66	8.2E-02	4.4E-32	1.2E-11	1.3E-01	3.8E+01	1.7E+01	6.5E-02	1.3E+01	6.5E-105
F3	Best	4.4E+04	5.3E+05	6.3E+02	3.2E-94	8.9E+02	3.6E-53	1.3E-03	1.1E+04	3.5E+02	2.2E+04	1.1E+03	4.5E+03	1.4E-144
	Mean	2.6E+04	5.4E+04	1.3E+02	0.0E+00	3.6E+02	3.5E-64	4.6E-06	1.4E+03	9.1E+01	5.4E+03	4.9E+02	1.6E+03	0.0E+00
	Std	8.7E+03	6.4E+05	3.6E+02	1.7E-93	3.2E+02	1.9E-52	2.2E-03	7.3E+03	1.4E+02	1.1E+04	5.5E+02	1.5E+03	7.9E-144
F4	Best	5.5E+01	9.5E+00	5.3E+00	6.2E-65	7.2E+00	2.8E-33	9.8E-05	3.6E+01	2.5E+00	6.3E+01	1.6E+00	3.5E+01	5.2E-117
	Mean	6.1E-01	3.6E+00	2.6E+00	0.0E+00	3.9E+00	3.0E-39	1.9E-05	1.7E+01	1.0E+00	3.9E+01	1.1E+00	2.5E+01	0.0E+00
	Std	2.7E+01	3.3E+00	1.8E+00	3.4E-64	1.6E+00	1.2E-32	6.5E-05	1.0E+01	1.2E+00	8.8E+00	2.1E-01	3.7E+00	1.8E-116

(continued)

Table 7 (continued)

ID	Index	WOA	BSO	PSO	SSA	GSA	HHO	GWO	SCA	MVO	MFO	BBO	PFA	CLASSA
F5	Best	2.9E+01	2.4E+03	1.1E+02	1.0E-04	1.2E+02	1.1E-02	2.7E+01	2.2E+05	4.5E+02	5.4E+06	2.3E+02	1.6E+06	6.1E-07
	Mean	2.8E+01	2.0E+02	2.3E+01	3.1E-08	2.6E+01	1.2E-03	2.5E+01	1.1E+03	4.5E+01	5.0E+03	5.8E+01	4.4E+05	3.2E-27
	Std	3.0E-01	3.0E+03	8.0E+01	1.6E-04	7.2E+01	1.8E-02	8.2E-01	5.1E+05	6.2E+02	2.9E+07	2.1E+02	8.3E+05	1.6E-06
F6	Best	1.8E+00	6.6E+02	2.7E-02	4.4E-08	4.6E+01	8.3E-05	5.6E-01	8.7E+01	2.2E+00	1.5E+03	2.9E+00	4.1E+03	4.6E-10
	Mean	8.4E-01	2.4E+02	2.3E-04	2.2E-11	4.8E-17	7.6E-07	7.0E-05	9.6E+00	9.1E-01	3.7E+01	1.7E+00	2.4E+03	0.0E+00
	Std	6.4E-01	3.0E+02	3.3E-02	8.1E-08	4.3E+01	8.9E-05	3.4E-01	1.3E+02	6.2E-01	4.4E+03	7.0E-01	8.1E+02	1.7E-09
F7	Best	4.7E-03	7.2E-02	2.9E-02	6.7E-04	4.5E-02	1.5E-04	2.2E-03	2.1E-01	4.0E-02	5.3E-01	1.3E-02	1.1E+00	3.2E-04
	Mean	2.1E-05	1.0E-02	1.5E-02	3.1E-05	1.1E-02	1.0E-05	3.1E-04	1.6E-02	1.2E-02	1.2E-01	4.2E-03	3.7E-01	2.0E-05
	Std	6.1E-03	5.2E-02	1.1E-02	5.7E-04	2.5E-02	1.2E-04	9.5E-04	1.9E-01	1.4E-02	8.1E-01	4.4E-03	4.2E-01	2.7E-04
F8	Best	-1.1E+82	-4.9E+03	-9.7E+03	-8.1E+03	-2.7E+03	-1.2E+04	-6.3E+03	-3.8E+03	-7.7E+03	-9.1E+03	-8.6E+03	-6.6E+03	-8.0E+03
	Mean	-1.4E+81	-6.2E+03	-1.2E+04	-9.6E+03	-3.4E+03	-1.3E+04	-7.2E+03	-4.6E+03	-9.2E+03	-1.1E+04	-9.7E+03	-7.1E+03	-9.6E+03
	Std	3.9E+82	3.1E+02	1.6E+03	6.1E+02	4.0E+02	2.9E+02	9.1E+02	3.4E+02	6.8E+02	8.9E+02	6.8E+02	2.5E+02	8.6E+02
F9	Best	0.0E+00	2.5E+01	5.7E+01	0.0E+00	1.7E+01	0.0E+00	4.2E+00	6.3E+01	1.2E+02	1.6E+02	3.8E+01	1.9E+02	0.0E+00
	Mean	0.0E+00	2.3E+00	4.1E+01	0.0E+00	1.1E+01	0.0E+00	5.1E-13	6.8E+00	6.7E+01	6.8E+01	2.3E+01	1.6E+02	0.0E+00
	Std	0.0E+00	1.3E+01	1.5E+01	0.0E+00	4.7E+00	0.0E+00	4.3E+00	3.8E+01	3.1E+01	4.6E+01	1.1E+01	1.6E+01	0.0E+00
F10	Best	6.7E-15	4.8E+00	1.3E+00	8.9E-16	1.6E-03	8.9E-16	2.2E-10	1.4E+01	2.0E+00	1.4E+01	6.5E-01	1.3E+01	8.9E-16
	Mean	8.9E-16	2.1E+00	3.2E-02	8.9E-16	4.5E-09	8.9E-16	8.6E-11	2.2E-01	6.4E-01	2.7E+00	3.6E-01	7.2E+00	8.9E-16
	Std	4.2E-15	2.0E+00	9.5E-01	0.0E+00	6.2E-03	0.0E+00	1.0E-10	8.0E+00	4.5E-01	7.0E+00	8.8E-02	2.2E+00	0.0E+00
F11	Best	7.4E-18	2.2E+02	5.3E-02	0.0E+00	1.1E+02	0.0E+00	6.9E-03	1.7E+00	9.7E-01	7.9E+00	1.0E+00	3.9E+01	0.0E+00
	Mean	0.0E+00	1.5E+02	3.7E-03	0.0E+00	8.3E+01	0.0E+00	0.0E+00	6.6E-01	8.4E-01	1.5E+00	9.8E-01	2.4E+01	0.0E+00
	Std	2.8E-17	3.0E+01	7.4E-02	0.0E+00	1.6E+01	0.0E+00	8.6E-03	1.1E+00	4.4E-02	2.3E+01	2.7E-02	8.1E+00	0.0E+00
F12	Best	9.3E-02	1.8E+00	9.3E-01	2.8E-09	1.8E+00	6.3E-06	3.8E-02	3.5E+05	2.5E+00	8.5E+06	1.1E-02	6.7E+04	4.9E-11
	Mean	3.2E-02	3.5E-01	3.8E-03	1.2E-11	3.7E-01	5.6E-08	7.0E-03	1.2E+00	4.3E-01	6.6E+00	4.0E-03	2.5E+02	2.4E-28
	Std	9.0E-02	1.4E+00	8.1E-01	7.4E-09	1.0E+00	7.1E-06	2.1E-02	9.1E+05	1.6E+00	4.7E+07	1.9E-02	9.2E+04	1.5E-10
F13	Best	1.3E+00	2.5E+01	6.4E-01	2.4E-08	1.5E+01	5.7E-05	4.7E-01	1.6E+06	2.1E-01	3.9E+03	1.4E-01	1.7E+06	3.6E-09
	Mean	7.7E-01	5.4E+00	1.3E-03	1.9E-11	6.3E-02	7.1E-08	2.5E-04	1.1E+01	9.4E-02	3.5E+01	7.8E-02	2.5E+05	2.9E-22
	Std	4.4E-01	1.4E+01	8.6E-01	5.6E-08	7.1E+00	7.2E-05	2.3E-01	3.5E+06	9.1E-02	5.6E+03	3.7E-02	1.4E+06	1.7E-08
F14	Best	2.3E+00	1.0E+00	1.0E+00	5.3E+00	7.0E+00	1.3E+00	3.9E+00	1.5E+00	1.0E+00	1.6E+00	4.6E+00	1.0E+00	1.7E+00
	Mean	1.0E+00	1.0E+00	1.0E+00	1.0E+00	1.1E+00	1.0E+00	1.0E+00	1.0E+00	1.0E+00	1.0E+00	1.0E+00	1.0E+00	1.0E+00
	Std	2.6E+00	1.8E-16	5.8E-17	5.3E+00	4.2E+00	9.5E-01	3.8E+00	8.9E-01	6.7E-11	1.2E+00	3.9E+00	1.8E-03	2.5E+00
F15	Best	1.1E-03	1.4E-03	5.6E-04	3.1E-04	6.0E-03	4.1E-04	4.4E-03	1.1E-03	5.4E-03	1.0E-03	2.4E-03	7.9E-04	3.1E-04
	Mean	3.1E-04	3.1E-04	3.1E-04	3.1E-04	1.8E-03	3.1E-04	3.1E-04	4.8E-04	4.9E-04	4.9E-04	3.8E-04	5.4E-04	3.1E-04
	Std	6.3E-04	3.6E-03	3.8E-04	1.7E-06	4.1E-03	2.4E-04	8.1E-03	3.6E-04	8.4E-03	3.5E-04	4.9E-03	1.4E-04	2.4E-07
F16	Best	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00
	Mean	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00	-1.0E+00
	Std	1.8E-09	6.0E-16	6.3E-16	5.5E-16	5.0E-16	3.0E-10	3.4E-08	3.6E-05	7.3E-07	6.8E-16	4.1E-12	1.8E-07	5.5E-16
F17	Best	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01
	Mean	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01	4.0E-01
	Std	2.6E-05	0.0E+00	0.0E+00	0.0E+00	0.0E+00	2.5E-05	1.3E-06	1.4E-03	1.7E-07	0.0E+00	2.1E-11	1.9E-09	0.0E+00
F18	Best	3.9E+00	3.0E+00	3.0E+00	3.9E+00	3.0E+00	3.0E+00	3.0E+00	3.0E+00	3.0E+00	3.0E+00	3.9E+00	3.0E+00	3.0E+00
	Mean	3.0E+00	3.0E+00	3.0E+00	3.0E+00	3.0E+00	3.0E+00	3.0E+00	3.0E+00	3.0E+00	3.0E+00	3.0E+00	3.0E+00	3.0E+00
	Std	4.9E+00	1.5E-15	1.7E-15	4.9E+00	4.0E-15	3.4E-07	8.2E-05	4.4E-05	5.8E-06	1.4E-15	4.9E+00	8.5E-07	4.9E-15
F19	Best	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00
	Mean	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00	-3.9E+00
	Std	3.8E-03	3.7E-03	2.7E-15	2.3E-15	2.5E-03	3.7E-03	1.9E-03	1.6E-03	2.3E-06	2.7E-15	5.7E-14	1.2E-06	2.4E-15
F20	Best	-3.2E+00	-3.1E+00	-3.3E+00	-3.3E+00	-3.3E+00	-3.1E+00	-3.3E+00	-2.9E+00	-3.2E+00	-3.2E+00	-3.3E+00	-3.3E+00	-3.3E+00
	Mean	-3.3E+00	-3.3E+00	-3.3E+00	-3.3E+00	-3.3E+00	-3.3E+00	-3.3E+00	-3.3E+00	-3.3E+00	-3.3E+00	-3.3E+00	-3.3E+00	-3.3E+00
	Std	7.1E-02	3.6E-01	6.0E-02	6.0E-02	1.4E-15	1.1E-01	6.9E-02	2.5E-01	6.0E-02	6.2E-02	6.0E-02	1.5E-02	5.9E-02
F21	Best	-8.5E+00	-1.0E+01	-6.8E+00	-9.5E+00	-7.0E+00	-5.1E+00	-9.8E+00	-3.1E+00	-7.4E+00	-6.6E+00	-6.0E+00	-1.0E+01	-1.0E+01
	Mean	-1.0E+01	-1.0E+01	-1.0E+01	-1.0E+01	-1.0E+01	-1.0E+01	-1.0E+01	-5.1E+00	-1.0E+01	-1.0E+01	-1.0E+01	-1.0E+01	-1.0E+01
	Std	3.0E+00	5.8E-15	3.3E+00	1.8E+00	3.6E+00	4.1E-03	1.3E+00	2.1E+00	3.1E+00	3.7E+00	3.6E+00	1.3E-01	5.4E-15
F22	Best	-6.5E+00	-1.0E+01	-8.1E+00	-1.0E+01	-1.0E+01	-5.6E+00	-1.0E+01	-3.9E+00	-8.5E+00	-8.3E+00	-6.4E+00	-1.0E+01	-1.0E+01
	Mean	-1.0E+01	-1.0E+01	-1.0E+01	-1.0E+01	-1.0E+01	-1.0E+01	-1.0E+01	-6.0E+00	-1.0E+01	-1.0E+01	-1.0E+01	-1.0E+01	-1.0E+01
	Std	3.4E+00	1.3E+00	3.4E+00	1.3E+00	1.4E-15	1.6E+00	1.9E-03	1.7E+00	3.2E+00	3.3E+00	3.6E+00	3.3E-01	1.1E-07
F23	Best	-6.7E+00	-9.7E+00	-8.1E+00	-1.0E+01	-1.0E+01	-4.8E+00	-9.3E+00	-4.0E+00	-8.2E+00	-7.8E+00	-8.6E+00	-1.0E+01	-1.1E+01
	Mean	-1.1E+01	-1.1E+01	-1.1E+01	-1.1E+01	-1.1E+01	-8.5E+00	-1.1E+01	-7.6E+00	-1.1E+01	-1.1E+01	-1.1E+01	-1.1E+01	-1.1E+01
	Std	3.7E+00	2.1E+00	3.6E+00	1.7E+00	1.5E+00	8.9E-01	2.8E+00	1.5E+00	3.3E+00	3.7E+00	3.4E+00	3.4E-01	5.6E-09

Combined with the above analysis, the CLSSA proposed in this paper is better than all the comparison algorithms in 12 of the 23 benchmark functions, 11 comparison algorithms in 6 test functions, 9 comparison algorithms in 3 test functions, and CLSSA is better than SSA, in all test functions, which proves that our proposed CLSSA has obvious advantages in optimization accuracy.

In order to directly show the performance differences of each algorithm in solving the test function, the algorithms are sorted according to the mean fitness of Table 7, the results are shown in Table 8, and the last column is the average ranking of each algorithm.

Table 8: Ranking of 13 algorithms in the benchmark function

ID	WOA	BSO	PSO	SSA	GSA	HHO	GWO	SCA	MVO	MFO	BBO	FPA	CLSSA
F1	4	12	6	2	9	3	5	10	7	11	8	13	1
F2	4	10	3	2	6	3	5	7	11	12	9	13	1
F3	12	13	6	2	7	3	4	10	5	11	8	9	1
F4	12	9	7	2	8	3	4	11	6	13	5	10	1
F5	5	10	6	2	8	3	7	11	9	13	8	12	1
F6	6	11	4	2	9	3	5	10	7	12	8	13	1
F7	5	10	7	3	9	1	4	11	8	12	6	13	2
F8	8	11	2	6	13	1	10	12	7	3	4	9	5
F9	1	7	9	1	6	1	5	10	11	12	8	13	1
F10	4	10	8	1	6	1	5	13	9	12	7	11	1
F11	4	13	6	1	12	1	5	9	7	10	8	11	1
F12	6	8	7	2	9	3	5	10	10	13	4	11	1
F13	8	10	7	2	9	3	6	12	5	11	4	13	1
F14	9	1	1	12	13	5	10	6	1	7	11	4	8
F15	7	9	4	1	13	3	11	7	12	6	10	5	1
F16	9	4	5	3	1	8	10	13	12	6	7	11	2
F17	13	1	1	1	1	12	7	8	9	1	11	10	1
F18	13	2	3	12	4	6	10	9	8	1	11	7	5
F19	10	11	3	2	8	12	9	13	7	4	5	6	1
F20	8	11	6	5	1	12	4	13	9	10	7	2	3
F21	6	1	9	5	8	12	4	13	7	10	11	3	1
F22	10	4	9	5	1	12	3	13	7	8	11	6	2
F23	11	5	9	4	2	12	6	13	8	10	7	3	1
Mean ranks	7.60	7.95	5.56	3.39	7.08	5.34	6.26	10.6	7.91	9.04	7.73	9.04	1.86

Fig. 10 is drawn according to the ranks in Table 8. The smaller the area of the algorithm performance curve, the better the performance of the algorithm.

The black bold line is the sorting result curve of CLSSA, and it can be seen intuitively that the performance of CLSSA is in the middle level on F8 and F14, and performs better in other test functions, and its surrounding area is the smallest, indicating that CLSSA has the best optimization performance as a whole.

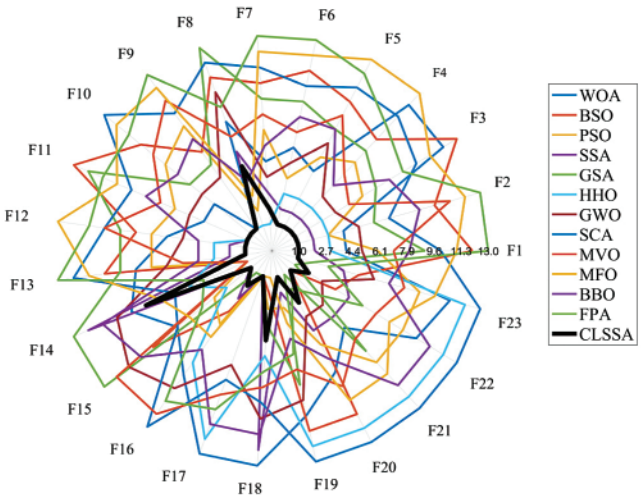

Figure 10: Ranks of average of 13 algorithms

To further illustrate the convergence performance of CLSSA, Fig. 11 lists the mean convergence curves of 13 algorithms to solve these functions. For the unimodal test function F1–F7, CLSSA has the best performance, the convergence speed is faster than all comparison algorithms, and the convergence accuracy is also higher than all comparison algorithms. For the multimodal functions F8–F23, CLSSA performs best in most of them. However, for functions F14, F17, and F18, CLSSA converges slowly in the early iterations. The CLSSA converges slower in the early iterations on F21 and F22 but can converge to better results afterwards. CLSSA converges faster than SSA on all test functions, which shows that the variable step strategy proposed in this paper can effectively improve the convergence speed.

To analyze the distribution characteristics of each algorithm in the test function, Fig. 12 lists box plots of 13 algorithms. Compared with other comparison algorithms, the CLSSA proposed in this paper performs well on most functions, and the obtained maximum, minimum, and median values are almost the same as the optimal solution, especially for F9, F10, F11, F16, F17, F19 and F20. In other test functions, although there are individual outliers, the overall distribution is still more concentrated than the comparison algorithm. Therefore, the CLSSA proposed in this paper has stronger stability.

The above analysis shows that CLSSA shows strong optimization ability on low-dimensional functions. However, the optimization algorithm is prone to fail in solving high-dimensional complex function problems. Real-world optimization problems are mostly large-scale complex optimization problems. Therefore, to verify the performance of CLSSA in high-dimensional problems, 13 algorithms were compared on the 100D test functions, and the experimental results are shown in Table 9. CLSSA is better than all comparison algorithms in F1–F6 and F12–F13. HHO performs best in F7 and F8, CLSSA ranks second and third respectively. When solving F9–F11, CLSSA and SSA can get the best value. It can also be seen from Figs. 13 and 14 that CLSSA performs well in other test functions except for the pool performance on F8 and can steadily and quickly converge to a better value.

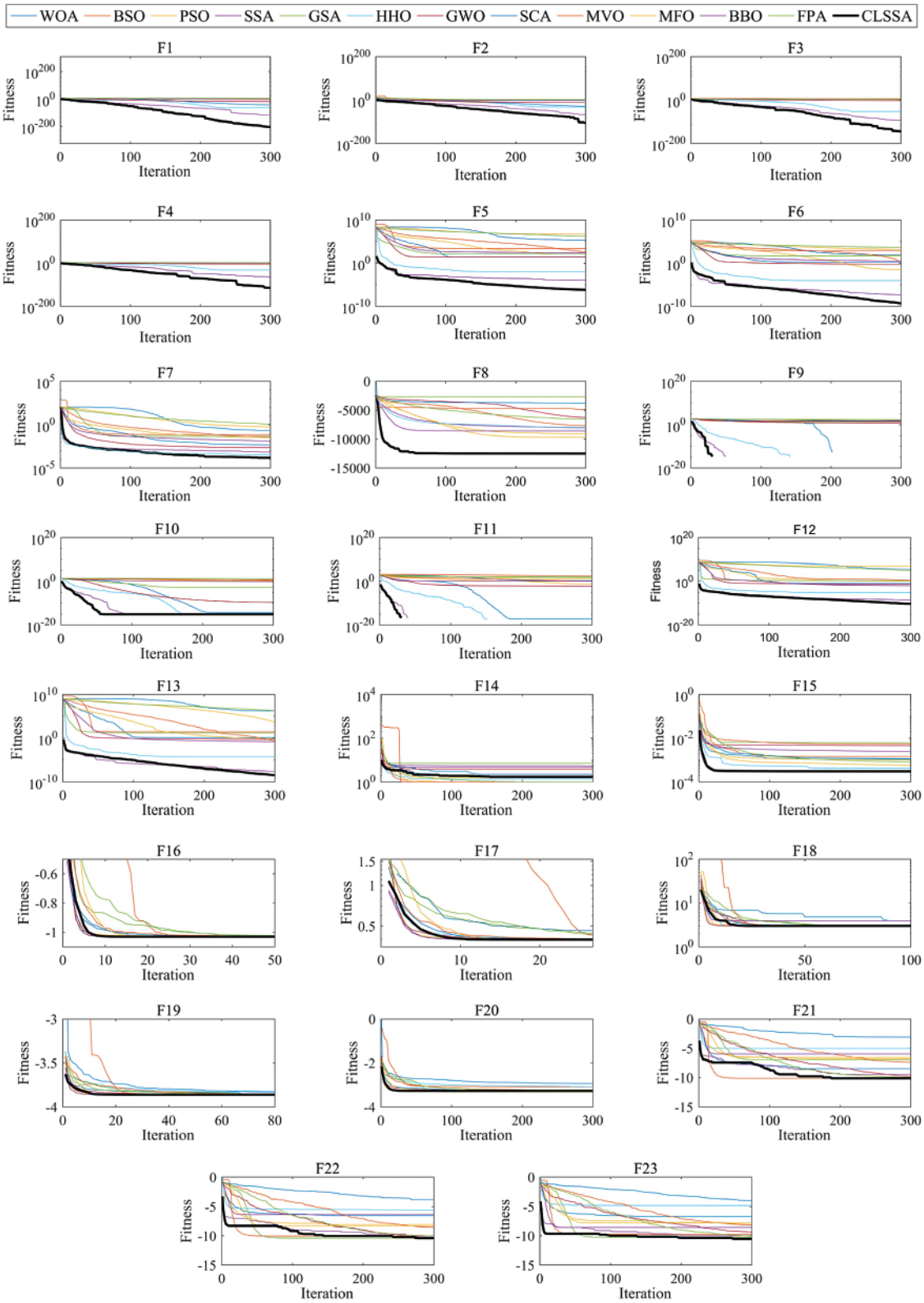

Figure 11: Convergence graphs of 13 algorithms on 23 representative functions

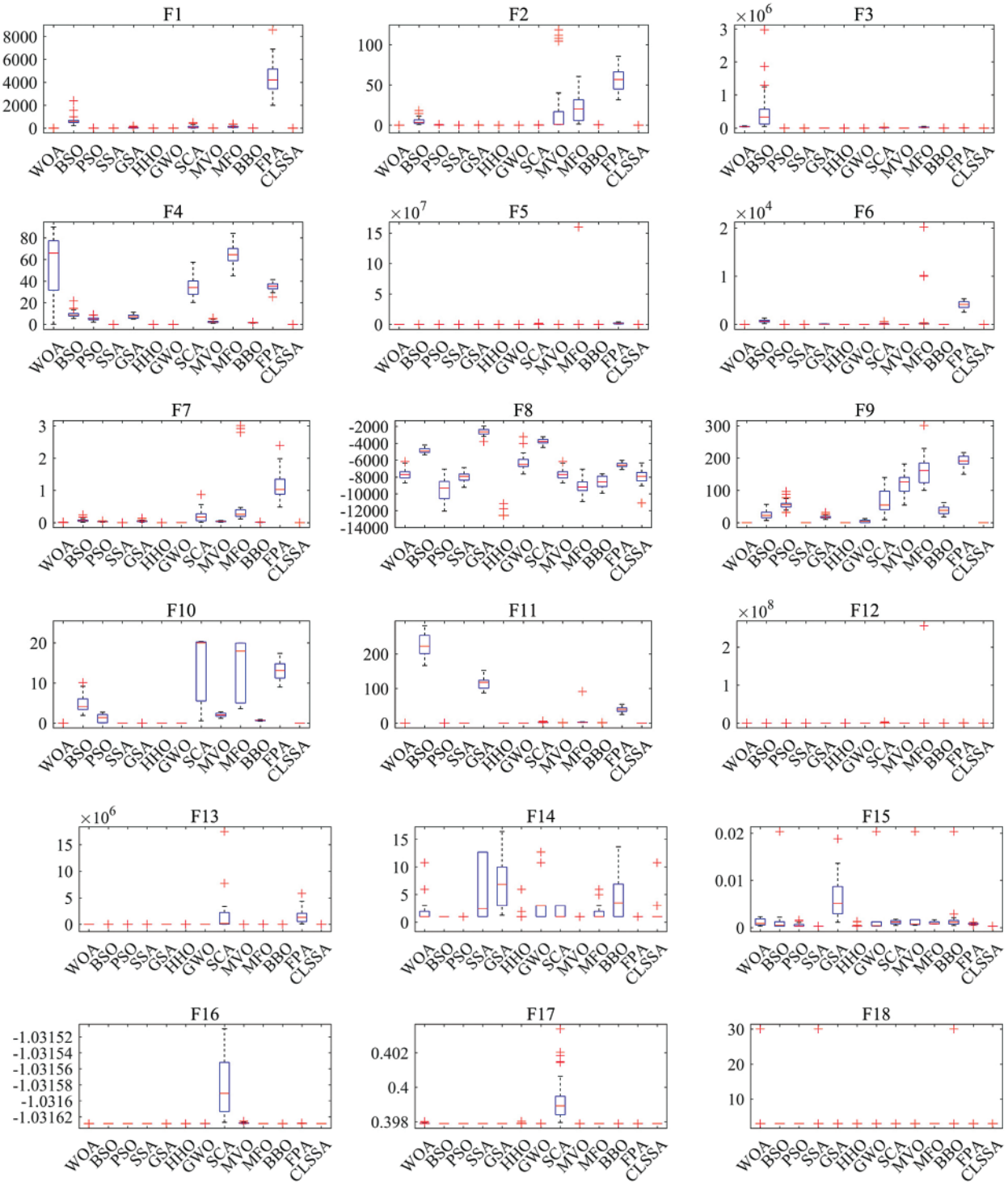

Figure 12: (continued)

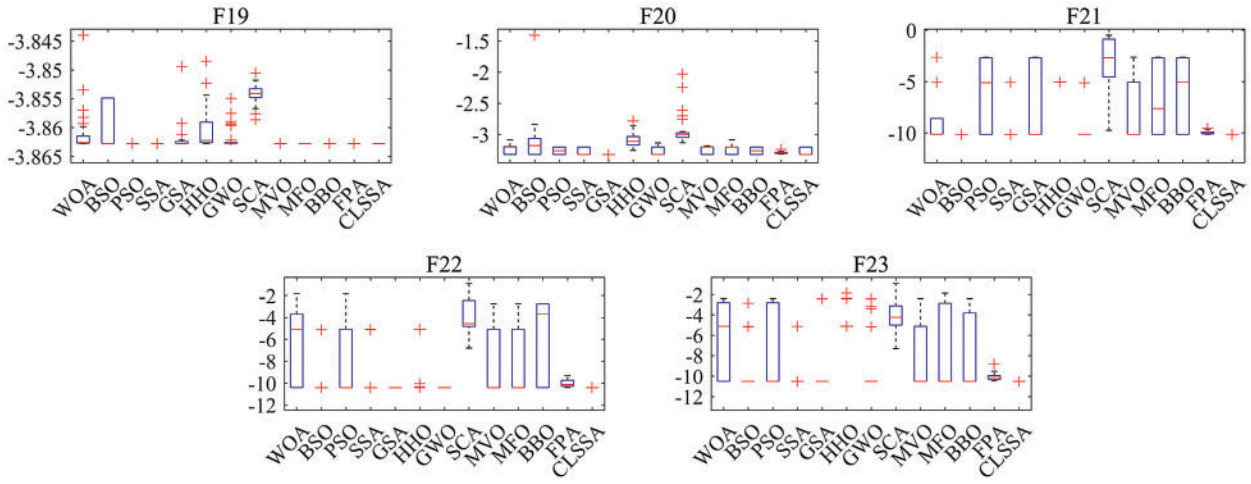

Figure 12: Box diagrams of solutions obtained by 13 algorithms on 23 benchmark functions with 30 independent runs

Table 9: Results and comparison of different algorithms for 13 benchmark functions with 100 D

ID	WOA	BSO	PSO	SSA	GSA	HHO	GWO	SCA	MVO	MFO	BBO	FPA	CLSSA
F1	7.29E-41	8.74E+03	1.94E+03	3.16E-113	4.31E+03	6.49E-62	2.34E-07	1.65E+04	2.62E+02	9.33E+04	2.17E+02	2.35E+04	2.04E-201
F2	4.08E-28	7.01E+01	5.46E+01	3.81E-69	1.42E+01	1.55E-33	4.77E-05	1.46E+01	2.96E+28	3.06E+02	9.41E+00	1.37E+06	4.62E-106
F3	8.91E+05	8.62E+07	6.17E+04	3.53E-88	1.62E+04	1.25E-40	1.93E+03	2.48E+05	7.26E+04	2.61E+05	6.26E+04	5.38E+04	8.17E-115
F4	8.26E+01	4.01E+01	4.11E+01	1.41E-71	1.54E+01	1.80E-32	2.84E+00	9.05E+01	5.62E+01	9.16E+01	1.95E+01	4.85E+01	6.51E-113
F5	9.84E+01	2.40E+05	4.86E+05	6.60E-04	9.81E+04	7.12E-02	9.81E+01	1.40E+08	1.76E+04	2.58E+08	5.12E+03	1.60E+07	2.77E-04
F6	1.16E+01	7.93E+03	2.03E+03	2.24E-06	4.26E+03	5.35E-04	9.91E+00	1.58E+04	2.64E+02	9.32E+04	2.11E+02	2.59E+04	1.29E-07
F7	3.74E-03	1.53E+00	2.32E+00	4.68E-04	1.62E+00	2.04E-04	1.04E-02	2.18E+02	6.50E-01	3.92E+02	1.15E-01	2.34E+01	2.84E-04
F8	-2.21E+04	-1.37E+04	-2.77E+04	-2.25E+04	-5.15E+03	-4.18E+04	-1.66E+04	-6.88E+03	-2.26E+04	-2.22E+04	-2.35E+04	-1.35E+04	-2.32E+04
F9	0.00E+00	1.32E+02	4.10E+02	0.00E+00	1.44E+02	0.00E+00	2.23E+01	3.42E+02	7.59E+02	9.12E+02	2.50E+02	9.30E+02	0.00E+00
F10	7.76E-15	9.03E+00	7.75E+00	8.88E-16	4.92E+00	8.88E-16	4.90E-05	1.87E+01	6.95E+00	1.99E+01	3.30E+00	1.26E+01	8.88E-16
F11	1.57E-02	1.00E+03	2.02E+01	0.00E+00	1.21E+03	0.00E+00	1.13E-02	1.54E+02	3.55E+00	7.75E+02	2.87E+00	2.38E+02	0.00E+00
F12	2.39E-01	7.44E+00	5.88E+02	1.85E-08	7.39E+00	2.79E-06	2.94E-01	4.34E+08	2.48E+01	4.14E+08	2.73E+00	4.66E+06	8.72E-09
F13	5.18E+00	4.84E+03	1.24E+05	1.70E-06	4.73E+02	2.45E-04	6.97E+00	7.62E+08	1.82E+02	9.06E+08	1.07E+01	3.34E+07	1.90E-07

In summary, compared with other algorithms, the CLSSA proposed in this paper is competitive, and the proposed improvement strategy can handle the relationship between exploitation and exploration well.

4.5 CLSSA for Engineering Problems

Engineering design problem is a nonlinear optimization problem with complex geometric shapes, various design variables and many practical engineering constraints. The performance of the proposed algorithm is evaluated by solving practical engineering problems. In the simulation, the population size is set to 50, and the maximum iterations is 500. The results of 30 independent runs of CLSSA are compared with those in other literatures.

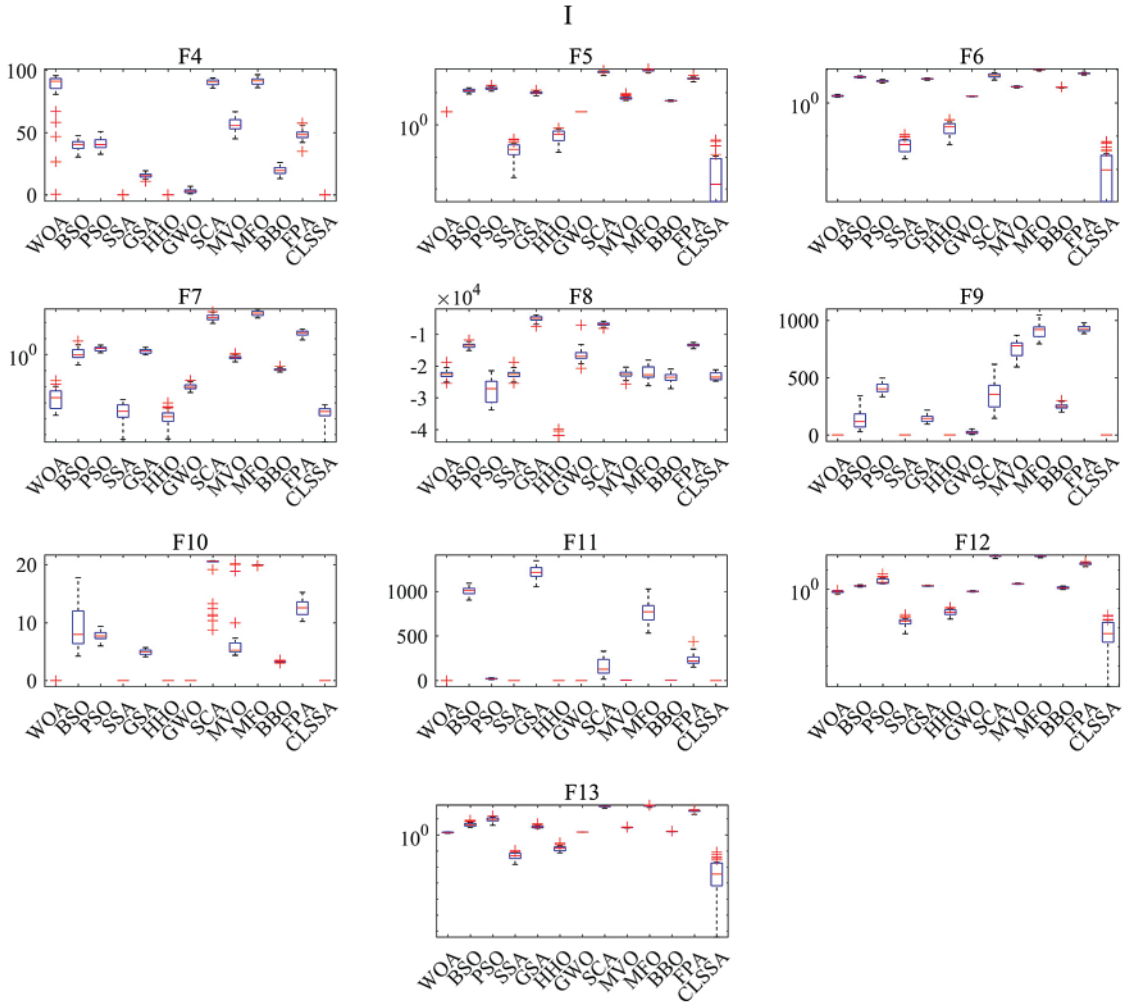

Figure 13: Box diagrams of solutions obtained by 13 algorithms on 13 benchmark functions with 30 independent runs

4.5.1 Pressure Vessel Design Problem

The pressure vessel design optimization problem shown in Fig. 15 is a typical hybrid optimization problem, whose goal is to reduce the total cost, including forming cost, material cost and welding cost. There are four different variables: container thickness Ts(x1), head thickness Th(x2), inner diameter R(x3) and container cylindrical section length L(x4). The comparison results are shown in Table 10. The problem can be described as Eq. (10).

$$\min f(x_1, x_2, x_3, x_4) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \tag{10}$$

subject to

$g_1(X) = -x_1 + 0.0193x_3 \leq 0$

$g_2(X) = -x_2 + 0.00954x_3 \leq 0$

$g_3(X) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^2 + 1,296,000 \leq 0$

$g_4(X) = x_4 - 240 \leq 0$

Variable ranges : $1 \times 0.0625 \leq x_1,\ x_2 \leq 99 \times 0.0625,\ 10 \leq x_3,\ x_4 \leq 200$

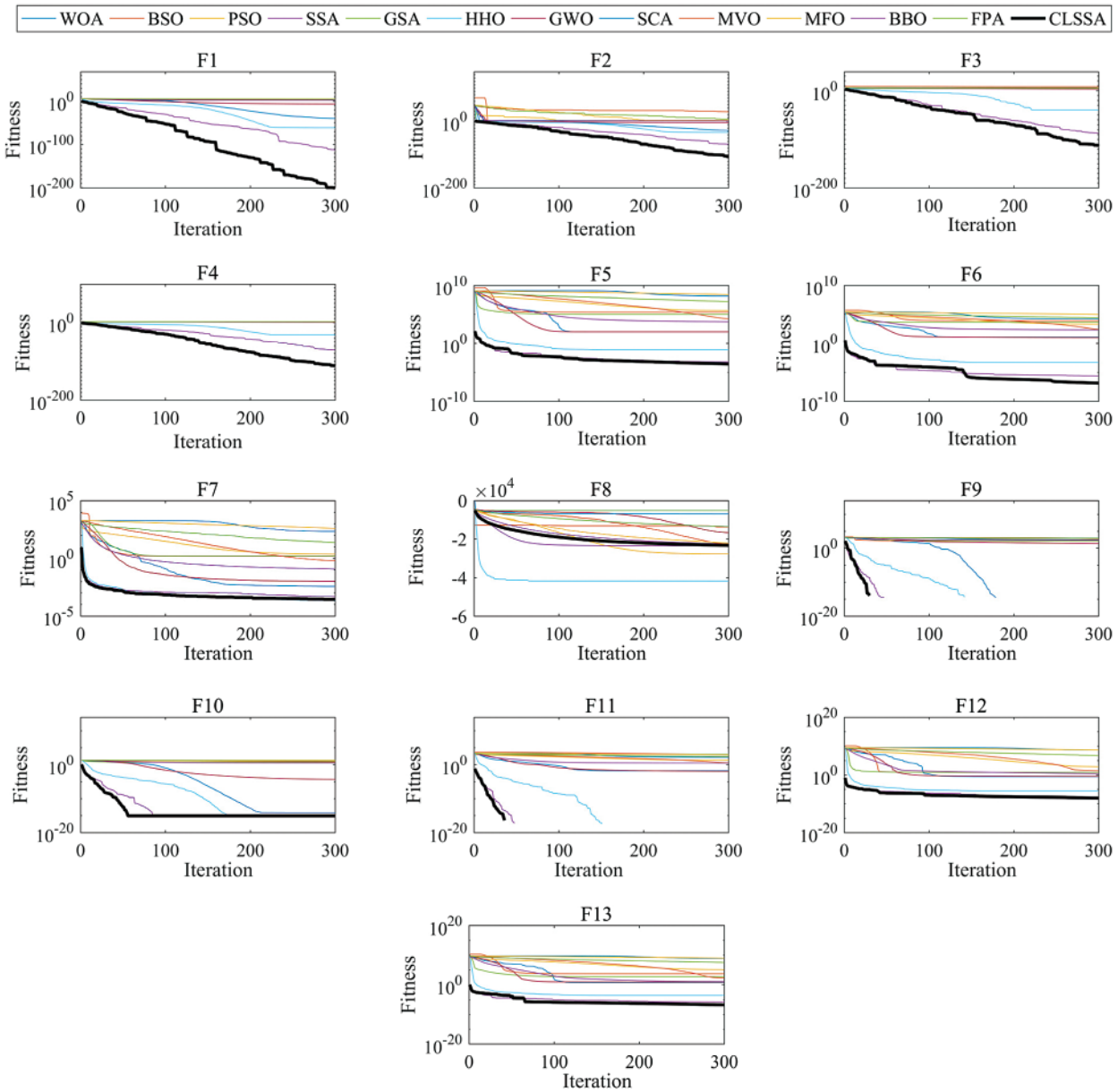

Figure 14: Convergence graphs of 13 algorithms on 13 representative functions

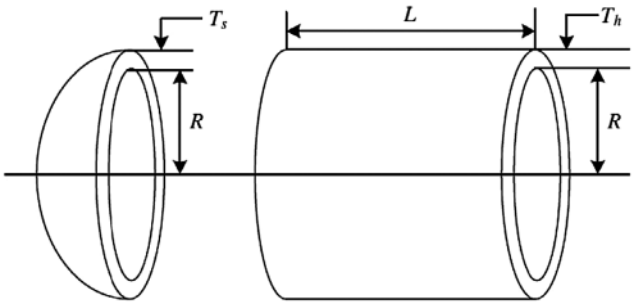

Figure 15: Schematic of the pressure vessel design problem

Table 10: Comparison of the best solutions obtained by various approaches for the pressure vessel design problem

Algorithm		CLSSA	CSDE [42]	HPSO [43]	GA [44]	MBA [45]	BBBO [46]
Optimum value	Th	0.7782	0.8125	0.8125	0.9375	0.7802	1.1250
	Ts	0.3847	0.4375	0.4375	0.5000	0.3856	0.6250
	R	40.3209	42.1000	42.0984	48.3290	40.4292	58.1967
	L	199.9822	176.6000	176.6366	112.6790	198.4694	44.2721
Optimum cost		5885.7092	6059.7100	6059.7143	6410.3811	5889.3216	7206.6400

4.5.2 Tension/Compression Spring Design Problem

The tension/compression spring design problem is a mechanical engineering design optimization problem, which can be used to evaluate the superiority of the algorithm. As shown in Fig. 16, the goal of this problem is to reduce the weight of the spring. It includes four nonlinear inequalities and three continuous variables: wire diameter w(x1), coil average diameter d(x2), coil length or number L(x3). The comparison results are shown in Table 11. The mathematical model of this problem can be described as Eq. (11).

$$\min f(x_1, x_2, x_3) = (x_3 + 2){x_1}^2 x_2 \tag{11}$$

subject to

$$g_1(X) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0$$

$$g_2(X) = \frac{x_2(4x_2 - x_1)}{12566 x_1^3 (x_2 - x_1)} + \frac{1}{5108 x_1^2} - 1 \leq 0$$

$$g_3(X) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0$$

$$g_4(X) = \frac{2(x_1 + x_2)}{3} - 1 \leq 0$$

Variable range : $0.05 \leq x_1 \leq 2,\ 0.25 \leq x_2 \leq 1.3,\ 2.0 \leq x_3 \leq 15.0$

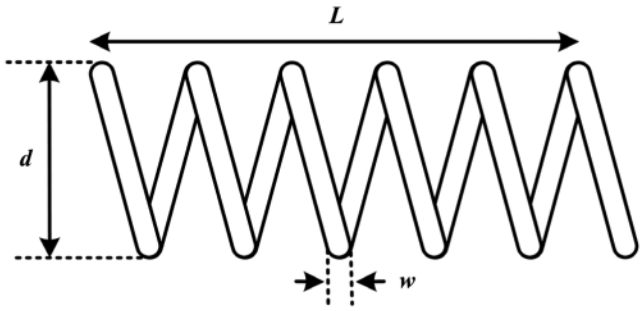

Figure 16: Schematic of tension/compression spring design problem

Table 11: Comparison of the best solutions obtained by various approaches for the tension/compression spring design problem

Algorithm		CLSSA	ALO [47]	GWO	MFO	MVO	GSA
Optimum value	w	0.0518	0.0517	0.0508	0.0521	0.0500	0.0571
	d	0.3592	0.3569	0.3357	0.3661	0.3159	0.4843
	L	11.1441	11.2793	12.6457	10.7587	14.2583	7.6234
Optimum cost		0.0127	0.0127	0.0127	0.0127	0.0128	0.0152

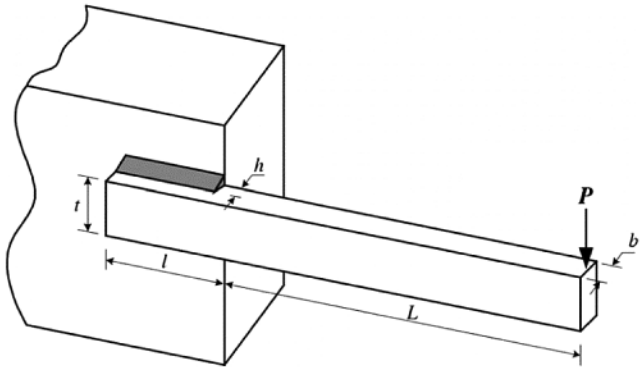

Figure 17: Schematic of welded beam design problem

4.5.3 Welded Beam Design Problem

As shown in Fig. 17, the main purpose of the welded beam design problem is to reduce the manufacturing cost of the welded beam, which mainly involves four variables: the width h (x1) and length l (x2) of the weld zone, the depth t (x3) and the thickness b (x4), and subject to the constraints of bending stress, shear stress, maximum end deflection and load conditions. The comparison results are shown in Table 12. The mathematical model of the problem is described as Eq. (12).

$$\min f(x_1, x_2, x_3, x_4) = 1.10471 {x_1}^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2) \tag{12}$$

subject to

$$g_1(X) = \tau_d - \tau(X) \geq 0$$

$$g_2(X) = \sigma_d - \sigma(X) \geq 0$$

$$g_3(X) = x_4 - x_1 \geq 0$$

$$g_4(X) = P_c(X) - P \geq 0$$

$$g_5(X) = \delta_d - \delta(X) \geq 0$$

where

$$\tau(X) = \sqrt{(\tau'(X))^2 + (\tau''(X))^2 + x_2\tau'(X)\tau''(X)/\sqrt{0.25(x_2^2 + (x_1 + x_3)^2)}}$$

$$\sigma(X) = 50400/x_3^2 x_4$$

$$P_c(X) = 64746.002(1 - 0.0282346x_3)x_3 x_4^3$$

$$\delta(X) = 2.1952/x_3^2 x_4$$

$$\tau'(X) = 6000/(\sqrt{2}x_1 x_2)$$

$$\tau''(X) = \frac{6000(14 + 0.5x_2)\sqrt{0.25(x_2^2 + (x_1 + x_3)^2)}}{2(0.707x_1x_2(x_2^2/12 + 0.25(x_1 + x_3)^2))}$$

In this paper, 15 algorithms are selected and compared with CLSSA. The simulation results show that CLSSA achieves the optimal values in all three engineering problems, which proves that CLSSA is highly competitive.

Table 12: Comparison of the best solutions obtained by various approaches for the welded beam design problem

Algorithm		CLSSA	CDE [48]	HGA [49]	TEO [50]	HHO	hHHO-SCA [51]
Optimum value	h	0.2057	0.2031	0.2057	0.2057	0.2040	0.1900
	l	3.4722	3.5430	3.4709	3.4731	3.5311	3.6965
	t	9.0362	9.0335	9.0396	9.0351	9.0275	9.3863
	b	0.2058	0.2062	0.2057	0.2058	0.2061	0.2041
Optimum cost		1.7251	1.7335	1.7252	1.7253	1.7320	1.7790

5 Conclusions

In this paper, we use three strategies combining chaos theory, logarithmic spiral search and adaptive steps to modify the basic sparrow search algorithm. First, the chaotic mapping is used to generate the values of the parameter R_2. Second, the logarithmic spiral search strategy is used to expand the search of SSA to the surrounding area, thus enhancing the population diversity and avoiding falling into local optimum. In addition, the adaptive step control strategy is proposed to effectively balance the exploitation and exploration of SSA. To evaluate the performance of the proposed CLSSA, 23 classical test functions are used for verification. The simulation results show that it is effective to improve the SSA performance by using chaotic mapping to generate the value

of parameter R_2, in which the iterative mapping has the best impact. Three improvement strategies can improve the performance of SSA. Compared with eleven advanced algorithms, CLSSA has higher convergence accuracy, faster convergence speed and more stable performance. In addition, CLSSA was applied to three engineering optimization problems. The results show that CLSSA has excellent performance in terms of convergence rate and accuracy for structural engineering design problems. In future work, we plan to further improve the performance of CLSSA. We can hybridize CLSSA with other algorithms, such as combining with MBO, EHO, etc., with the same excellent performance, to achieve the effect of $1 + 1 > 2$. We can also further study the impact of SSA parameter settings on performance. In addition, we plan to apply it to solve some real world problems, such as unmanned combat aerial vehicles task allocation and unmanned combat aerial vehicles path planning.

Funding Statement: The author acknowledges funding received from the following science foundations: The Science Foundation of Shanxi Province, China (2020JQ-481, 2021JM-224), Aero Science Foundation of China (201951096002).

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

References

1. Wang, F., Li, Y., Liao, F., Yan, H. (2020). An ensemble learning based prediction strategy for dynamic multi-objective optimization. *Applied Soft Computing, 96,* 106592. DOI 10.1016/j.asoc.2020.106592.
2. Sun, J., Miao, Z., Gong, D., Zeng, X. J., Li, J. et al. (2020). Interval multiobjective optimization with memetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics, 50(8),* 3444–3457. DOI 10.1109/TCYB.2019.2908485.
3. Wang, G. G., Tan, Y. (2019). Improving metaheuristic algorithms with information feedback models. *IEEE Transactions on Cybernetics, 49(2),* 542–555. DOI 10.1109/TCYB.2017.2780274Y.
4. Feng, Y., Deb, S., Wang, G. G., Alavi, A. H. (2021). Monarch butterfly optimization: A comprehensive review. *Expert Systems with Applications, 168,* 114418. DOI 10.1016/j.eswa.2020.114418.
5. Wu, G. (2016). Across neighborhood search for numerical optimization. *Information Sciences, 329,* 597–618. DOI 10.1016/j.ins.2015.09.051.
6. Wu, G., Pedrycz, W., Suganthan, P. N., Mallipeddi, R. (2015). A variable reduction strategy for evolutionary algorithms handling equality constraints. *Applied Soft Computing, 37,* 774–786. DOI 10.1016/j.asoc.2015.09.007.
7. Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning.* USA: Addison-Wesley.
8. Beyer, H. G., Schwefel, H. P. (2002). Evolution strategies-a comprehensive introduction. *Natural Computing, 1(1),* 3–52. DOI 10.1023/A:1015059928466.
9. Yao, X., Liu, Y., Lin, G. (1999). Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation, 3(2),* 82–102. DOI 10.1109/4235.771163.
10. Sarker, R. A., Elsayed, S. M., Ray, T. (2014). Differential evolution with dynamic parameters selection for optimization problems. *IEEE Transactions on Evolutionary Computation, 18(5),* 689–707. DOI 10.1109/TEVC.2013.2281528.
11. Simon, D. (2008). Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation, 12(6),* 702–713. DOI 10.1109/TEVC.2008.919004.
12. Dupanloup, I., Schneider, S., Excoffier, L. G. L. (2002). A simulated annealing approach to define the genetic structure of populations. *Molecular Ecology, 11(12),* 2571–2581. DOI 10.1046/j.1365-294X.2002.01650.x.
13. Rashedi, E., Nezamabadi-pour, H., Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences, 179(13),* 2232–2248. DOI 10.1016/j.ins.2009.03.004.

14. Wei, Z., Huang, C., Wang, X., Han, T., Li, Y. (2019). Nuclear reaction optimization: A novel and powerful physics-based algorithm for global optimization. *IEEE Access, 7,* 66084–66109. DOI 10.1109/Access.6287639.
15. Hatamlou, A. (2013). Black hole: A new heuristic optimization approach for data clustering. *Information Sciences, 222,* 175–184. DOI 10.1016/j.ins.2012.08.023.
16. Yang, X. S. (2010). *Nature-inspired metaheuristic algorithms*. Second Edition, UK: Luniver Press.
17. Eberhart, R., Kennedy, J. (1995). A new optimizer using particle swarm theory. *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, Nagoya, Japan.
18. Colorni, A., Dorigo, M., Maniezzo, V., Varela, F., Bourgine, P. (1992). Distributed optimization by ant colonies. *Toward a practice of autonomous systems: Proceedings of the first european conference on artificial life*, pp. 134–142. Massachusetts.
19. Wang, G. G., Deb, S., Cui, Z. (2019). Monarch butterfly optimization. *Neural Computing and Applications, 31(7),* 1995–2014. DOI 10.1007/s00521-015-1923-y.
20. Wang, G. G. (2018). Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems. *Memetic Computing, 10(2),* 151–164. DOI 10.1007/s12293-016-0212-3.
21. Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M. M. et al. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems, 97,* 849–872. DOI 10.1016/j.future.2019.02.028.
22. Wang, G. G., Deb, S., Coelho, L. D. S. (2015). Earthworm optimisation algorithm: A bio-inspired metaheuristic algorithm for global optimisation problems. *International Journal of Bio-Inspired Computation, 12(1),* 1–22. DOI 10.1504/IJBIC.2018.093328.
23. Li, J., Lei, H., Alavi, A. H., Wang, G. G. (2020). Elephant herding optimization: Variants, hybrids, and applications. *Mathematics, 8(9),* 1415. DOI 10.3390/math8091415.
24. Li, S., Chen, H., Wang, M., Heidari, A. A., Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems, 111,* 300–323. DOI 10.1016/j.future.2020.03.055.
25. Gao, D., Wang, G. G., Pedrycz, W. (2020). Solving fuzzy job-shop scheduling problem using de algorithm improved by a selection mechanism. *IEEE Transactions on Fuzzy Systems, 28(12),* 3265–3275. DOI 10.1109/TFUZZ.91.
26. Tang, A. D., Han, T., Zhou, H., Xie, L. (2021). An improved equilibrium optimizer with application in unmanned aerial vehicle path planning. *Sensors, 21(5),* 1814. DOI 10.3390/s21051814.
27. Chen, S., Chen, R., Wang, G. G., Gao, J., Sangaiah, A. K. (2018). An adaptive large neighborhood search heuristic for dynamic vehicle routing problems. *Computers & Electrical Engineering, 67,* 596–607. DOI 10.1016/j.compeleceng.2018.02.049.
28. Wang, F., Li, Y., Zhou, A., Tang, K. (2020). An estimation of distribution algorithm for mixed-variable newsvendor problems. *IEEE Transactions on Evolutionary Computation, 24(3),* 479–493. DOI 10.1109/TEVC.2019.2932624.
29. Wolpert, D. H., Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1(1),* 67–82. DOI 10.1109/4235.585893.
30. Xue, J., Shen, B. (2020). A novel swarm intelligence optimization approach: Sparrow search algorithm. *Systems Science & Control Engineering, 8(1),* 22–34. DOI 10.1080/21642583.2019.1708830.
31. Wang, G. G., Deb, S., Gandomi, A. H., Zhang, Z., Alavi, A. H. (2016). Chaotic cuckoo search. *Soft Computing, 20,* 3349–3362. DOI 10.1007/s00500-015-1726-1.
32. Wang, G. G., Gandomi, A. H., Alavi, A. H. (2013). A chaotic particle-swarm krill herd algorithm for global numerical optimization. *Kybernetes, 42(6),* 962–978. DOI 10.1108/K-11-2012-0108.
33. Ewees, A. A., Elaziz, M. E. A. (2020). Performance analysis of chaotic multi-verse harris hawks optimization: A case study on solving engineering problems. *Engineering Applications of Artificial Intelligence, 88,* 103370. DOI 10.1016/j.engappai.2019.103370.
34. Wang, G. G., Guo, L., Gandomi, A. H., Hao, G., Wang, H. (2014). Chaotic krill herd algorithm. *Information Sciences, 274,* 17–34. DOI 10.1016/j.ins.2014.02.123.

35. Kaur, G., Arora, S. (2018). Chaotic whale optimization algorithm. *Journal of Computational Design and Engineering, 5(3),* 275–284. DOI 10.1016/j.jcde.2017.12.006.
36. Wang, T., Yang, L., Liu, Q. (2020). Beetle swarm optimization algorithm: Theory and application. *Filomat, 34(15),* 5121–5137. DOI 10.2298/FIL2015121W.
37. Mirjalili, S., Mirjalili, S. M., Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69,* 46–61. DOI 10.1016/j.advengsoft.2013.12.007.
38. Mirjalili, S. (2016). SCA: A sine cosine algorithm for solving optimization problems. *Knowledge Based Systems, 96,* 120–133. DOI 10.1016/j.knosys.2015.12.022.
39. Mirjalili, S., Mirjalili, S. M., Hatamlou, A. (2016). Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Computing and Applications, 27(2),* 495–513. DOI 10.1007/s00521-015-1870-7.
40. Mirjalili, S. (2015). Moth-flame optimization algorithm. *Knowledge Based Systems, 89,* 228–249. DOI 10.1016/j.knosys.2015.07.006.
41. Yang, X. S. (2012). Flower pollination algorithm for global optimization. *Proceedings of the 11th International Conference on Unconventional Computation and Natural Computation*, pp. 240–249. Orleans.
42. Zhang, Z., Ding, S., Jia, W. (2019). A hybrid optimization algorithm based on cuckoo search and differential evolution for solving constrained engineering problems. *Engineering Applications of Artificial Intelligence, 85,* 254–268. DOI 10.1016/j.engappai.2019.06.017.
43. He, Q., Wang, L. (2007). A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization. *Applied Mathematics and Computation, 186(2),* 1407–1422. DOI 10.1016/j.amc.2006.07.134.
44. Deb, K., Goyal, M. (1996). A combined genetic adaptive search (GeneAS) for engineering design. *Journal of Computer Science and Informatics, 26,* 34–45.
45. Sadollah, A., Bahreininejad, A., Eskandar, H., Hamdi, M. (2013). Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. *Applied Soft Computing, 13,* 2592–2612. DOI 10.1016/j.asoc.2012.11.026.
46. Guo, W., Chen, M., Wang, L., Wu, Q. (2016). Backtracking biogeography-based optimization for numerical optimization and mechanical design problems. *Applied Intelligence, 44(4),* 894–903. DOI 10.1007/s10489-015-0732-4.
47. Mirjalili, S. (2015). The ant lion optimizer. *Advances in Engineering Software, 83,* 80–98. DOI 10.1016/j.advengsoft.2015.01.010.
48. Huang, F. Z., Wang, L., He, Q. (2007). An effective co-evolutionary differential evolution for constrained optimization. *Applied Mathematics and Computation, 186(1),* 340–356. DOI 10.1016/j.amc.2006.07.105.
49. Yan, X., Liu, H., Zhu, Z., Wu, Q. (2017). Hybrid genetic algorithm for engineering design problems. *Cluster Computing, 20(1),* 263–275. DOI 10.1007/s10586-016-0680-8.
50. Kaveh, A., Dadras, A. (2017). A novel meta-heuristic optimization algorithm. *Advances in Engineering Software, 110,* 69–84. DOI 10.1016/j.advengsoft.2017.03.014.
51. Kamboj, V. K., Nandi, A., Bhadoria, A., Sehgal, S. (2020). An intensify harris hawks optimizer for numerical and engineering optimization problems. *Applied Soft Computing, 89,* 106018. DOI 10.1016/j.asoc.2019.106018.