sciendo

JAISCR, 2023, Vol. 13, No. 3, pp. 131 – 149
10.2478/jaiscr-2023-0011

A NOVEL VARIANT OF THE SALP SWARM ALGORITHM FOR ENGINEERING OPTIMIZATION

Fuyun Jia[1], Sheng Luo[2], Guan Yin[3], Yin Ye[4,*]

[1]*Department of Public Infrastructure, Henan Medical College*
Zhengzhou 451191 Henan, China

[2]*School of Information Science and Technology, Nantong University*
Nantong 226019 Jiangsu, China

[3]*School of Energy Resources, China University of Geosciences (Beijing)*
Beijing 100083 Beijing, China

[4]*School of Computer Science and Artificial Intelligence, Wuhan University of Technology*
Wuhan 430070 Hubei China

[*]*E-mail: yeyin@whut.edu.cn*

Submitted: 8th November 2022; Accepted: 7th May 2023

Abstract

There are many design problems need to be optimized in various fields of engineering, and most of them belong to the NP-hard problem. The meta-heuristic algorithm is one kind of optimization method and provides an effective way to solve the NP-hard problem. Salp swarm algorithm (SSA) is a nature-inspired algorithm that mimics and mathematically models the behavior of slap swarm in nature. However, similar to most of the meta-heuristic algorithms, the traditional SSA has some shortcomings, such as entrapment in local optima. In this paper, the three main strategies are adopted to strengthen the basic SSA, including chaos theory, sine-cosine mechanism and the principle of quantum computation. Therefore, the SSA variant is proposed in this research, namely SCQ-SSA. The representative benchmark functions are employed to test the performances of the algorithms. The SCQ-SSA are compared with the seven algorithms in high-dimensional functions (1000 dimensions), seven SSA variants and six advanced variants on benchmark functions, the experiment reveals that the SCQ-SSA enhances resulting precision and alleviates local optimal problems. Besides, the SCQ-SSA is applied to resolve three classical engineering problems: tubular column design problem, tension/compression spring design problem and pressure vessel design problem. The design results indicate that these engineering problems are optimized with high accuracy and superiority by the improved SSA. The source code is available in the URL: https://github.com/ye-zero/SCQ-SSA/tree/main/SCQ-SSA.

Keywords: Salp swarm algorithm, meta-heuristic algorithm, chaos theory, sine-cosine mechanism, quantum computation, optimization design of engineering

1 Introduction

Optimization is to find the best solution from the feasible solution set of complex problems. There are a lot of optimization problems in our daily life and industrial production, such as bin packing problems [1] and path planning problems [2] in Cargo transportation, the fixed-outline floorplanning [3] and routing [4] in the integrated circuit, the optimal reactive power dispatch [5] in power system, the maximum power point trackers [6] for solar photovoltaic system, the training optimization of deep convolutional neural network [7], etc. If the objective function of the problem is relatively uncomplicated, the derivative method can be used to solve it. However, the NP-hard problems usually contain many parameters and constraints, or the optimization problems are dynamic and multi-objective [8] as well. The traditional methods optimize problems with the low rate of convergence and poor accuracy, and can not achieve the desired results.

As an artificial intelligence method for intelligent computing, metaheuristic algorithms are surprisingly very popular in many fields of engineering. This popularity is due to the following main features: local optimum suppression, gradient-free search, elasticity and simplicity, the metaheuristic algorithm assumes that the optimization problem is a black box. In other words, meta-heuristic algorithms consider and resolve optimization problems by the inputs, outputs and constraints.

In recent years, several novel algorithms are proposed like Orca Predation Algorithm (OPA, [9]), White Shark Optimizer (WSO, [10]), Artificial Rabbits Optimization (ARO, [11]) as well. Metaheuristic algorithms have been applied widely as a superior tool for solving the optimization problems in science and industry. Particle swarm optimization is designed for multi-station assembly sequence planning [12]. Knowledge mapping optimization method based on the improved ant colony algorithm is proposed in [13]. The optimization objective function of knowledge mapping is constructed and solved by the improved ant colony algorithm. A network security posture prediction model [14] based on the support vector machine optimized by the improved artificial bee colony algorithm (I-ABC) and uses the I-ABC algorithm for SVM.

Salp Swarm Algorithm (SSA) [64] is proposed by Seyedali Mirjalili [15]. as one of the metaheuristic algorithms in 2017. The algorithm is a swarm-inspired algorithm that mimics and mathematically models the behavior of a salp swarm in the ocean. As soon as the algorithm is proposed, it arouses a wide interest among researchers. There are many research papers about the improvement and application of SSA. A brief review of these variants is presented as follows. An efficient adaptive salp swarm algorithm based on two-class fuzzy entropy for multi-stage threshold image segmentation is proposed in [16]. The enhanced salp swarm algorithm for multimodal optimization and fuzzy-based grid frequency controller design is presented in [17]. The salp swarm algorithm is applied to design and analyze text document clustering in [18]. An improved salp swarm algorithm based on the energy audit to optimize the parameters of urban rail substations is presented in [19].

Our main contributions can be summarized as follows:

- We propose the efficient variant SCQ-SSA based on three strategies; chaos theory, quantum computation and sine-cosine mechanism to enhance the basic SSA.
- A number of experiments are carried out to verify the performance of the SCQ-SSA on benchmark functions. The SCQ-SSA is compared with seven algorithms in high-dimensional functions (1000 dimensions), seven SSA variants and six advanced variants, the experimental results show that the SCQ-SSA has fine and effective convergence accuracy.
- The SCQ-SSA is employed to optimize the design of three engineering problems. The weight or cost of three engineering problems is designed with the relative minimum, and the experimental data reveals that the SCQ-SSA is superior to other methods.

The remaining portion of this paper is arranged as follows. Section 2 introduces the optimization goal and presents a brief description of basic SSA. The modified SSA variant and three strategies for improvement are discussed in Section 3. The simulation results of performance verification are provided in Section 4. Three classical engineering

problems are optimized by the SCQ-SSA in Section 5. Finally, the conclusion and future work are enumerated in Section 6.

2 Optimization goal and method

2.1 Optimization goal

The optimization goal of this paper is the nonlinear and constraint problem with a single objective function in engineering. The mathematical model of the engineering problem is to obtain the maximum or minimum with constraints as Eq. (1) and Eq. (2).

$$Max\ or\ Min \quad F(X), X \in S^n \tag{1}$$

$$Subject\ to \begin{cases} G_i(X) \leq 0, i = 1,2,...,g, \\ H_i(X) \geq 0, i = 1,2,...,h, \\ K_i(X) = 0, i = 1,2,...,k, \\ x_i \in [lb_i, ub_i], i = 1,2,...,n. \end{cases} \tag{2}$$

Where $F(X)$ is the objective function of obtaining the optimal value, $X = [x_1, x_2, ..., x_n]$ is the vector of the solution and belongs to the collection S; and S is the n-dimensional search space that satisfies the lower bound lb and the upper bound ub; g and h are the numbers of nonlinear constraints; k represents the number of linear constraints; n is the number of variables. Generally speaking, the constrained problems are transformed into unconstrained problems. The most frequently used technique is the penalty function. The mathematical description is as Eq. (3).

$$\begin{aligned} F(X,P) = F(X) + P\sum_{i=1}^{g} \max(G_i(x), 0) \\ -P\sum_{i=1}^{h} \min(H_i(x), 0) + P\sum_{i=1}^{k} |K_i(x)| \end{aligned} \tag{3}$$

Where $F(X,P)$ is the penalty function; P represents penalty factor.

2.2 Salp swarm algorithm

The slap is a marine invertebrate with a transparent and bucket-shaped body. They resemble jellyfish in shape and movement, move by sucking the surrounding water and squirting it as a reverse propulsion force. Salp swarm algorithm is based on the behavior of the slap when it searches the food, the same phenomenon is mimicked and modeled in the algorithm to get the optimum solution for the optimization problem. Unlike many creatures in nature, the slaps are not distributed in groups, but in the form of a chain. They often form the chains of the slaps.

The mathematical model of SSA divides the chain into leader and followers. The one is called the leader in front of the salp chain, which can be considered as an individual with relatively better fitness value, that is, a relatively better solution to the objective function. whereas the remaining are named the followers. The leader guides the group to move and search for a better solution in the search space; the followers follow each other, indirectly accept the influence of the leader. The leader influences the position update of the next slap. In this way, the leader's influence on followers decreases step by step, and the slap chain preserves its diversity.

Assuming that the optimization problem is m-dimensional and the population size of the slap is n, the search space is an n*m-dimensional solution domain. The m-dimensional upper boundary of the search space is $ub_m = [ub_m^1, ub_m^2, ..., ub_m^n]$, while the m-dimensional lower boundary of search space is $lb_m = [lb_m^1, lb_m^2, ..., lb_m^n]$, there are the optimal solution $F_m = [F_m^1, F_m^2, ..., F_m^n]$ somewhere in the solution space. The default initialization of SSA is random as Eq. (4)

$$X_n^m = rand(n,m) * (ub - lb) + lb \tag{4}$$

In the algorithm, the leader is responsible for searching food, and the position of the leader is updated by Eq. (5).

$$x_j^1 = \begin{cases} F_j + c_1((ub_j - lb_j)c_2 + lb_j), c_3 \leq 0.5 \\ F_j - c_1((ub_j - lb_j)c_2 + lb_j), c_3 > 0.5 \end{cases} \tag{5}$$

Where x_j^1 represents the position of the leader in the *j-th* dimension, F_j shows the position of the optimal solution (food source) in the *j-th* dimension. There are three parameters that affect the algorithm, they are c_1, c_2, c_3; where c_2 and c_3 are random numbers between [0,1], c_2 determines the next step size and c_3 ensures the randomness of the direction change of leader. The major coefficient is c_1, it can make the balance between exploration and

exploitation capabilities; c_1 is calculated as Eq. (6).

$$c_1 = 2e^{-\left(\frac{4l}{L}\right)^2} \tag{6}$$

It can be seen that c_1 decreases as the number of iterations increases, which is reasonable. At the beginning of the iteration, the step size of the search space needs to be larger, so that SSA is easier to jump out of the local optimum. At the end of the iteration, the step size is reduced for convergence. l is the current iteration and L is the maximum number of iterations. Followers don't need to change their direction randomly, they just need to follow the movement of the last slap. The motion of the followers conforms to Newton's laws of motion as Eq. (7).

$$s = v_0 t + \frac{1}{2} a t^2 \tag{7}$$

Where s is the distance of the follower; v_0 is the initial velocity of the follower, $v_0 = 0$; a is the current acceleration of the follower; time t is the difference of the number of iterations, $t = 1$; Eq. (7) satisfys two equations: $a = (v - v_0)/t$ and $v = (x - x_0)/t$, so $s = (x - x_0)/2 = (x_j^i - x_j^{i-1})/2$; and Eq. (7) is derived as Eq. (8).

$$x_j^i = x_j^{i-1} + s = \frac{1}{2}\left(x_j^i + x_j^{i-1}\right) \tag{8}$$

Where $i \geq 2$ and x_j^i shows the position of *i-th* follower salp in *j-th* dimension.

3 The modified salp swarm algorithm

3.1 Initialization with chaos

The population size is fixed in SSA, there are a limited number of agents to perform the search process in the large and unknown solution space. Hence, the searchability of each agent needs to be fully exploited. The initialization of the population is crucial to the search for the optimal. If the random method is used to generate the initial positions of agents, the positions of agents are unevenly distributed or some agents are concentrated in some blocks. However, the default initialization of SSA is done in a random manner. The random initialization can easily produce the poor solution and fall into the local optimum, so the improved method of initializing the population is necessary.

Over the years, Chaos has been applied as a very effective mathematical tool to improve the search performance of metaheuristic algorithms. Chaos state manifests as a dynamic and unstable, stochastic-like behavior in the periodic motion of nonlinear systems. It is characterized by stochasticity, regularity, ergodicity and boundedness. Chaos is commonly used in meta-heuristic algorithms in two ways. One is used to update the location of the population, which makes the next update of the population more random, and traverses the entire search space as much as possible. The ergodicity of chaos can improve the global search ability of the algorithm.such as chaotic differential evolution algorithm [20] and chaotic grasshopper optimization algorithm [21] as well. The other is used for initialization, where the position of the generated population spreads across the search space, such as the chaotic artificial bee colony algorithm [22] and chaotic grey wolf optimization [23] and so on. The modified salp swarm algorithm initializes the population using circle maps as Eq. (9).

$$C_{n+1} = \bmod\left(C_n + b - \left(\frac{a}{2\pi}\right)\sin\left(2\pi C_n\right), 1\right) \tag{9}$$

Where a and b are the control factor of chaos. When $a = 0.5$ and $b = 0.2$, the state of chaos is complete and the variable value is between 0 and 1. The random initialization is changed to Eq. (10).

$$X_n^m = C(n,m) * (ub - lb) + lb \tag{10}$$

Where C is the chaotic matrix; ub is the upper boundary of the search space and lb is the lower boundary of the search space. Fig. 1 shows the perturbation plot of Circle Chaos for 200 iterations starting from the middle value of 0.5. The curve of the solution is distributed between the range of 0 and 1 with regularity. The ergodicity of the Circle chaotic map makes it possible to explore other potential solutions in the space, it enhances the diversity of the population and the ability to jump out of the local optimum.

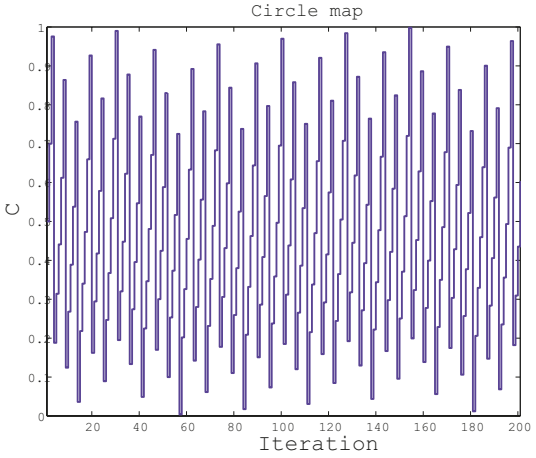

Figure 1. The Circle Map

3.2 Sine-cosine mechanism

The original SSA is similar to most of the meta-heuristic algorithms. There are two possible problems: slow convergence speed and entrapment in local optima. The sine-cosine mechanism is adopted to escape from the local optimum. The idea of the sine-cosine mechanism is derived from Sine-cosine Algorithm (SCA). The position is updated according to sine or cosine function with a certain probability. The four parameters of SCA enhance the randomized behavior of SSA and promotes local optima avoidance. The sine-cosine mechanism is employed as the local search scheme to disturb the position of SSA. The cyclic mode of the sine and cosine functions allows one solution to be repositioned around the other, it guarantees exploitation of SSA. The range of the sine and cosine functions is modified to improve the exploration of SSA. The update of the position is as Eq. (11).

$$x_j^{t+1} = \begin{cases} x_j^t + r_1 \times \sin(r_2) \times \left| r_3 \times F_j - x_j^t \right|, r_4 \leq 0.5 \\ x_j^t + r_1 \times \cos(r_2) \times \left| r_3 \times F_j - x_j^t \right|, r_4 > 0.5 \end{cases} \tag{11}$$

Where x_j^t is the position of the current solution at *t-th* iteration in *j-th* dimension, r_1 is a major parameter to affect the global exploration and local development of the algorithm, r_1 is updated by Eq. (12); r_2 is a random number between 0 and 2π; r_3 is a uniformly distributed random numbers in the range of [0, 2] and r_4 is in [0,1]; r is the threshold, $r = 0.5$; F_j is the position of the best solution.

$$r_1 = a - t\frac{a}{T} \tag{12}$$

Where t is the current iteration, T is the maximum number of iterations, and a is a constant, $a = 2$.

3.3 Quantum computation

Quantum computation is first proposed in the early 1980s. Since then, many researchers have been trying to introduce the concept of quantum computation into traditional methods to improve the performance of computing. Of course, a lot of meta-heuristic algorithms are developed and hybridized with quantum computation to strengthen their advantages and mitigate their weaknesses. A number of quantum-inspired algorithms are proposed, such as adaptive quantum behavior particle swarm optimization [24], quantum grey wolf optimizer [25] and quantum-inspired firefly algorithm [26] as well.

Each dimension of an agent is named a qubit in quantum computation, and there are two basic states: $|0\rangle$ and $|1\rangle$. At any time, the state of a qubit can be considered as a linear combination of two basic states. In general, it is called the superimposed state as Eq. (13).

$$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{13}$$

Where α and β are the complex numbers, they are called probability amplitudes. In other words, the probability of the state $|0\rangle$ is $|\alpha|^2$ and the probability of the state $|1\rangle$ is $|\beta|^2$. The relationship between the two states satisfies Eq. (14).

$$|\alpha|^2 + |\beta|^2 = 1 \tag{14}$$

If only the real number is considered, the relationship makes us think of the sine and cosine functions. Let $\alpha = \cos\theta$ and $\beta = \sin\theta$, where θ is the phase. Then the state of the qubit can be expressed by probability amplitude or phase as Eq. (15).

$$|\varphi\rangle = [\alpha, \beta]^T = [\cos\theta, \sin\theta]^T \tag{15}$$

An m-dimensional agent has m qubits, the state A of an agent can also be represented by probability amplitude or phase as Eq. (16).

$$A = \begin{bmatrix} \alpha_1 & \ldots & \alpha_m \\ \beta_1 & \ldots & \beta_m \end{bmatrix} = \begin{bmatrix} \cos\theta_1 & \ldots & \cos\theta_m \\ \sin\theta_1 & \ldots & \sin\theta_m \end{bmatrix} \tag{16}$$

From Eq. (16) we can know that the possibility of the agent state is 2^m in quantum computation. There is only one definite state for an agent in traditional computing. In contrast, quantum agents do not increase the numbers or dimensions of the population, but potentially enrich the diversity of the population. And the position update of quantum agents is implemented to jump out from the local optimum and improve the solution superiority according to the quantum revolving gate as Eq. (17).

$$\begin{bmatrix} \cos\theta_{ij}^{t+1} \\ \sin\theta_{ij}^{t+1} \end{bmatrix} = \begin{bmatrix} \cos\Delta\theta_{ij}^{t+1} - \sin\Delta\theta_{ij}^{t+1} \\ \sin\Delta\theta_{ij}^{t+1}\cos\Delta\theta_{ij}^{t+1} \end{bmatrix} \begin{bmatrix} \cos\theta_{ij}^{t} \\ \sin\theta_{ij}^{t} \end{bmatrix} \tag{17}$$

Where θ_{ij}^{t} is the phase of the *j-th* dimension of the *i-th* agent in the *t-th* iteration; $\Delta\theta_{ij}^{t+1}$ represents the phase shift of the *j-th* dimension of the *i-th* agent in the $t+1$ iteration. The individual state of agent needs a mapping from quantum state to the solution space as Eq. (18).

$$x_{ij} = \begin{Bmatrix} \cos\theta_{ij}^{2}(ub_j - lb_j) + lb_j, q_1 \leq 0.5 \\ \sin\theta_{ij}^{2}(ub_j - lb_j) + lb_j, q_1 > 0.5 \end{Bmatrix} \tag{18}$$

Where x_{ij} is the postion of the *j-th* dimension of the *i-th* agent; q_1 is a random number between 0 and 1 and Eq. (18) shows the variability and diversity of the positon of search agents.

3.4 The operation of SCQ-SSA

Let's analyze the pseudocode of SCQ-SSA. In the preparation phase, the parameters of the input are denoted and created. The circle chaotic sequences are generated by Eq. (9), which are employed to initialize the slap chains according to Eq. (10). Besides, the initial solution needs to satisfy the upper bound and the upper bound. The fitness value is obtained by the objective function. The related parameters of the leading salp are gotten by Eq. (6) and Eq. (12). The leader updates the position using Eq. (5) and Eq. (11), then the quantum state of agents is converted to the solution space according to Eq. (18). The position of the followers is calculated by Eq. (8) and the best solution is updated in each iteration. If the current solution is superior to the previous solution, the better solution is saved. Afterward, the stop condition for the recursion is checked. At last, the optimum value of the objective function is obtained.

Algorithm 1. Pseudocode of the SCQ-SSA algorithm.

Input: The size of population (N), the maximum number of iteration (t_{max}), the upper bound (ub) and the lower bound (lb) of the search space, the dimension (D) of the objective function.
Output: The optimal solution (F).
for $i = 1 \rightarrow N$ **do**
    for $j = 1 \rightarrow D$ **do**
        The population is initialized by circle maps.
    end for
end for
while $t < t_{max}$ **do**
    Check the upper bound ub and lower bound lb of the search space.
    The fitness value of each search agent is calculated by the objective function.
    Update c_1 using Eq. (6).
    if ($i == 1$) **then**
        Initialize the related parameters of sine-cosine mechanism.
        Update r_1 using Eq. (12).
        Update the position of the leader by Eq. (5) and Eq. (11).
        Convert the quantum state of agents to the solution space by Eq. (18).
    end if
    for $i = 2 \rightarrow N$ **do**
        Update the position of the followers by Eq. (8).
        $i = i + 1$.
    end for
    if (the fitness value gets better) **then**
        Update the fitness value and save the better value.
    end if
    $t = t + 1$.
end while

4 Performance assessment of the SCQ-SSA

The SCQ-SSA discussed in the preceding sections are tested on the representative benchmark functions in this section. The performance of SCQ-SSA is evaluated in comparison with other algorithms. All the experimental algorithms are implemented on a personal computer that has the following specifications: operation system is Windows 10 Pro with a 64-bit; a memory (RAM) of 4 GB; processor type is Intel-Core-i5 with 3.20 GHz; development tools and version is Matlab-R2010a.

4.1 Benchmark functions

In this section, unimodal benchmark functions (f_1-f_7) and multimodal benchmark functions (f_8-f_{13}) are used to test the performance of the algorithms. The mathematical definition and properties of the benchmark functions are presented in Table 1. Where the properties of benchmark functions contain Unimodal and Multimodal. Opt is defined as the optimal value of the function. D is the dimensions of the search space.

4.2 Evaluation indexes

The performance evaluation indexes are as follows:

– *mean value* is the average value obtained after n runs, denoted as Eq. (19).

$$Mean = \frac{x_1 + x_2 + \cdots + x_n}{n} = \frac{\sum_{i=1}^{n} x_i}{n} \tag{19}$$

– *Standard deviation* represents the degree of dispersion among the individuals in the solution set. The formula is Eq. (20).

$$Std = \sqrt{\frac{1}{\mathrm{n}-1} \sum_{i=1}^{n} (x_i - Mean)^2} \tag{20}$$

– *Rank* is to sort the algorithm performance according to the mean and standard deviation. The order of magnitude E is considered as the smallest unit. The average and overall rank are listed at the bottom of the table.

– ε is the level of precision of the objective function. In addition, if the mean value of the function significantly outperforms the precision level, '+' is labeled in the corresponding algorithms; else if the mean value is significantly less than the precision level, '-' is labeled; if mean is not much different from the precision level, '=' is labeled.

4.3 Comparison with the other algorithms in high-dimensional functions

To verify the effectiveness of SCQ-SSA in high-dimensional search space (1000 dimensions), the mean and standard deviation values are obtained by SCQ-SSA and other severn well-known algorithms: Gravitational search algorithm (GSA, [27]) Grey wolf optimizer (GWO, [28]), Cuckoo search (CS, [29]), Whale optimization algorithm (WOA, [30]), Sine cosine algorithm (SCA, [31]), Moth-Flame optimization (MFO, [32]), Arithmetic optimization algorithm (AOA, [33]). In this experiment, when the dimension is 1000, the maximum number of iterations is set to 1000. The population size of each function is 30. Each case runs 30 times independently.

According to the data in Table 2. First of all, the performance of eight algorithms is analyzed by the mean value; SCQ-SSA significantly outperforms other seven algorithms on most functions ($f_3, f_4, f_5, f_6, f_7, f_9, f_{10}, f_{11}, f_{13}$) and even achieves the optimal value 0 on two multimodal functions (f_9 and f_{11}). Then WOA also performs well on eight functions and covers to the optimal value 0 on f_{11}, the other six algorithms obtain the feasible solution. From the mean value, we can know the convergence precision and exploration ability of SCQ-SSA is better than other algorithms.

Next, the standard deviation are used to evaluate the ability of the algorithm, and two additional metrics (ε and *Rank*) are calculated to directly evaluate the performance. we calculate the number of statistics for the three labels:+/=/-. The labels of GWO, CS, SCA, MFO and AOA are 13/0/0, it means that SCQ-SSA is obviously superior to the other five algorithms, followed by GSA (12/0/1) and WOA (9/1/3). Then the average and overall rank of the eight algorithms are discussed. In summary, SCQ-SSA receives the first rank, WOA, AOA, GWO, GSA, CS, SCA, MFO rank second to eighth respectively in terms of mean value and standard deviation. It can be seen that the stability and reliabil-

Table 1. Benchmark functions used in experiments.

Benchmark functions	Properties	Solution space	Opt
$f_1(x) = \sum_{i=1}^{D} x_i^2$	Unimodal	$[-100,100]^D$	0
$f_2(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	Unimodal	$[-10,10]^D$	0
$f_3(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	Unimodal	$[-100,100]^D$	0
$f_4(x) = max\{\lvert x_i \rvert, 1 \le i \le D\}$	Unimodal	$[-100,100]^D$	0
$f_5(x) = \sum_{i=1}^{D-1} [100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2]$	Unimodal	$[-50,50]^D$	0
$f_6(x) = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	Unimodal	$[-100,100]^D$	0
$f_7(x) = \sum_{i=1}^{D} i x_i^4 + random[0,1]$	Unimodal	$[-1.28,1.28]^D$	0
$f_8(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	Multimodal	$[-500,500]^D$	-418.9829*D
$f_9(x) = \sum_{i=1}^{n} (x_i{}^2 - 10\cos(2\pi x_i) + 10)$	Multimodal	$[-5.12,5.12]^D$	0
$f_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e$	Multimodal	$[-32,32]^D$	0
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	Multimodal	$[-600,600]^D$	0
$f_{12}(x) = \frac{\pi}{D}\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})]$ $+(y_n - 1)^2\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$	Munimodal	$[-50,50]^D$	0
$u(x_i,a,k,m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \le a \\ k(-x_i - a)^m, & x < -a \end{cases}$			
$f_{13}(x) = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{D}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)]$ $+(x_n - 1)^2[1 + \sin^2(2\pi x_n)]\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	Munimodal	$[-50,50]^D$	0

Table 2. Comparison between SCQ-SSA and other algorithms in high-dimensional functions.

No.	Index	GSA	GWO	CS	WOA	SCA	MFO	AOA	SCQ-SSA
f_1	Mean	$5.19E{+}02$	$8.72E{-}09$	$1.48E{+}05$	$3.25E{-}142$	$4.05E{+}05$	$2.49E{+}06$	$1.65E{+}00$	$1.70E{-}104$
	Std	$1.77E{+}02$	$3.33E{-}09$	$1.20E{+}04$	$1.03E{-}141$	$1.12E{+}05$	$4.52E{+}04$	$7.17E{-}02$	$7.70E{-}104$
	ε	+	+	+	−	+	+	+	
	Rank	5	3	6	1	7	8	4	2
f_2	Mean	$6.63E{+}00$	$2.32E{+}204$	*INF*	$2.30E{-}100$	*INF*	*INF*	$2.10E{+}01$	$2.14E{-}41$
	Std	$1.92E{+}00$	*INF*	*INF*	$5.85E{-}100$	*INF*	*INF*	$7.46E{-}01$	$3.05E{-}41$
	ε	+	+	+	−	+	+	+	
	Rank	3	5	7	1	7	7	4	2
f_3	Mean	$9.55E{+}03$	$8.50E{+}05$	$5.03E{+}06$	$1.42E{+}08$	$2.44E{+}07$	$1.40E{+}07$	$1.41E{+}02$	$1.00E{-}102$
	Std	$2.07E{+}03$	$2.68E{+}05$	$1.16E{+}06$	$1.10E{+}08$	$4.86E{+}06$	$2.09E{+}06$	$6.78E{+}01$	$2.60E{-}102$
	ε	+	+	+	+	+	+	+	
	Rank	3	4	5	8	7	6	2	1
f_4	Mean	$1.54E{+}01$	$7.68E{+}01$	$3.91E{+}01$	$7.45E{+}01$	$9.96E{+}01$	$9.95E{+}01$	$2.11E{-}01$	$1.87E{-}53$
	Std	$1.26E{+}00$	$5.39E{+}00$	$2.03E{+}00$	$2.58E{+}01$	$9.34E{-}02$	$1.40E{-}01$	$1.13E{-}02$	$1.89E{-}53$
	ε	+	+	+	+	+	+	+	
	Rank	5	7	6	8	3	4	2	1
f_5	Mean	$1.98E{+}04$	$9.97E{+}02$	$4.09E{+}07$	$9.93E{+}02$	$2.94E{+}09$	$1.09E{+}10$	$9.99E{+}02$	$9.98E{+}02$
	Std	$1.28E{+}04$	$2.25E{-}01$	$6.83E{+}06$	$5.99E{-}01$	$9.85E{+}08$	$3.70E{+}08$	$1.34E{-}01$	$2.40E{-}02$
	ε	+	+	+	+	+	+	+	
	Rank	5	3	6	4	7	8	2	1
f_6	Mean	$1.85E{+}03$	$2.09E{+}02$	$1.45E{+}05$	$4.46E{+}01$	$3.65E{+}05$	$2.47E{+}06$	$2.42E{+}02$	$2.50E{+}01$
	Std	$6.47E{+}02$	$2.04E{+}00$	$1.31E{+}04$	$6.50E{+}00$	$1.61E{+}05$	$5.36E{+}04$	$1.96E{+}00$	0
	ε	+	+	+	+	+	+	+	
	Rank	5	3	6	2	7	8	4	1
f_7	Mean	$2.15E{+}00$	$2.21E{-}02$	$3.45E{+}02$	$1.20E{-}03$	$4.28E{+}04$	$1.78E{+}05$	$6.30E{-}05$	$2.89E{-}05$
	Std	$6.17E{-}01$	$5.71E{-}03$	$3.48E{+}01$	$8.11E{-}04$	$9.44E{+}03$	$7.35E{+}03$	$4.09E{-}05$	$2.83E{-}05$
	ε	+	+	+	+	+	+	+	
	Rank	5	4	6	3	7	8	2	1
f_8	Mean	$-8.88E{+}02$	$-9.78E{+}04$	$-3.89E{+}04$	$-3.77E{+}05$	$-2.27E{+}04$	$-1.01E{+}05$	$-3.51E{+}04$	$-1.95E{+}04$
	Std	$2.84E{+}02$	$7.18E{+}03$	$1.45E{+}03$	$5.63E{+}04$	$9.49E{+}02$	$6.69E{+}03$	$2.52E{+}03$	$2.76E{+}02$
	ε	−	+	+	+	+	+	+	
	Rank	1	6	4	8	3	7	5	2
f_9	Mean	$1.38E{+}02$	$1.35E{+}01$	$7.41E{+}03$	$3.64E{-}13$	$1.72E{+}03$	$1.47E{+}04$	$5.12E{-}05$	0
	Std	$1.59E{+}01$	$1.40E{+}01$	$1.71E{+}02$	$1.15E{-}12$	$9.30E{+}02$	$1.17E{+}02$	$7.03E{-}06$	0
	ε	+	+	+	+	+	+	+	
	Rank	5	4	7	2	6	8	3	1
f_{10}	Mean	$3.01E{+}00$	$2.84E{-}06$	$1.31E{+}01$	$3.38E{-}15$	$1.60E{+}01$	$2.02E{+}01$	$9.16E{-}03$	$8.88E{-}16$
	Std	$5.81E{-}01$	$5.16E{-}07$	$3.34E{-}01$	$2.40E{-}15$	$5.23E{+}00$	$2.42E{-}01$	$2.16E{-}04$	0
	ε	+	+	+	+	+	+	+	
	Rank	5	3	6	2	8	7	4	1
f_{11}	Mean	$9.50E{+}01$	$2.87E{-}03$	$9.35E{+}02$	0	$2.83E{+}03$	$1.56E{+}04$	$6.70E{+}03$	0
	Std	$1.38E{+}01$	$6.05E{-}03$	$7.47E{+}01$	0	$9.28E{+}02$	$2.98E{+}02$	$7.33E{+}02$	0
	ε	+	+	+	=	+	+	+	
	Rank	4	3	5	1.5	6	8	7	1.5
f_{12}	Mean	$5.35E{+}00$	$8.58E{-}01$	$1.20E{+}06$	$5.70E{-}02$	$1.00E{+}10$	$2.68E{+}10$	$1.10E{+}00$	$1.19E{-}01$
	Std	$1.57E{+}00$	$2.30E{-}02$	$5.01E{+}05$	$2.63E{-}02$	$1.84E{+}09$	$1.12E{+}09$	$8.14E{-}03$	$6.78E{-}16$
	ε	+	+	+	−	+	+	+	
	Rank	5	3	6	1	7	8	4	2
f_{13}	Mean	$1.28E{+}02$	$6.37E{+}00$	$9.98E{+}02$	$2.07E{+}00$	$3.03E{+}08$	$2.38E{+}08$	$1.00E{+}02$	$2.97E{+}00$
	Std	$2.08E{+}01$	$4.77E{-}01$	$8.30E{+}02$	$6.40E{-}01$	$1.54E{+}08$	$2.15E{+}08$	$3.99E{-}02$	$5.39E{-}02$
	ε	+	+	+	+	+	+	+	
	Rank	5	3	6	2	8	7	4	1
	ε	12/0/1	13/0/0	13/0/0	9/1/3	13/0/0	13/0/0	13/0/0	+/=/−
Rank	Average	4.30	3.92	5.84	3.34	6.38	7.23	3.61	1.34
	Overall	5	4	6	2	7	8	3	1

Table 3. Comparison between SCQ-SSA and SSA variants on benchmark functions.

No.	Index	SSA	GSSA	CSSA	LSSA	WASSA	CASSA	ESSA	SCQ-SSA
f_1	Mean	$6.01E-09$	$3.25E-109$	$1.27E-84$	$1.21E-56$	$4.18E-103$	$1.86E-192$	0	0
	Std	$1.31E-09$	$1.20E-108$	$3.14E-84$	$3.86E-56$	$2.33E-102$	0	0	0
	ε	+	+	+	+	+	+	=	
	Rank	8	4	6	7	5	3	1.5	1.5
f_2	Mean	$1.30E-01$	$4.08E-54$	$1.22E-42$	$3.04E-29$	$1.89E-53$	$5.61E-96$	0	0
	Std	$2.73E-01$	$2.05E-53$	$3.44E-42$	$8.89E-29$	$3.44E-53$	$2.76E-97$	0	0
	ε	+	+	+	+	+	+	=	
	Rank	8	4	6	7	5	3	1.5	1.5
f_3	Mean	$9.33E-04$	$3.88E-106$	$8.04E-83$	$3.37E-55$	$1.22E-103$	$3.42E-19$	0	0
	Std	$2.01E-03$	$2.11E-105$	$4.36E-82$	$1.10E-54$	$5.93E-103$	0	0	0
	ε	+	+	+	+	+	+	=	
	Rank	8	3	5	6	4	7	1.5	1.5
f_4	Mean	$2.83E-01$	$7.76E-56$	$6.56E-42$	$1.54E-28$	$1.98E-53$	$5.52E-97$	0	0
	Std	$3.52E-01$	$2.46E-55$	$2.51E-41$	$7.90E-28$	$7.39E-53$	$8.05E-98$	0	0
	ε	+	+	+	+	+	+	=	
	Rank	8	4	6	7	5	3	1.5	1.5
f_5	Mean	$4.41E+01$	$2.90E+01$	$2.90E+01$	$2.90E+01$	$2.68E+01$	$2.78E+01$	$2.75E+01$	$2.57E+01$
	Std	$3.96E+01$	$2.28E-02$	$3.25E-02$	$2.44E-02$	$1.39E-01$	$8.87E-02$	$1.51E-01$	$1.01E-02$
	ε	+	+	+	+	+	+	+	
	Rank	8	2	4	3	6	5	7	1
f_6	Mean	$6.19E-09$	$9.66E-01$	$9.14E-01$	$9.38E-01$	$8.43E-09$	$4.11E-07$	$6.15E-09$	$5.18E-09$
	Std	$1.34E-09$	$5.77E-01$	$4.96E-01$	$6.20E-01$	$1.41E-09$	$1.05E-07$	$9.52E-10$	$1.05E-10$
	ε	+	+	+	+	+	+	+	
	Rank	3	8	6	7	4	5	2	1
f_7	Mean	$1.83E-02$	$4.16E-04$	$5.02E-04$	$1.10E-03$	$9.57E-03$	$1.41E-05$	$1.14E-05$	$2.55E-06$
	Std	$6.60E-03$	$3.69E-04$	$4.75E-04$	$9.85E-04$	$1.13E-02$	$1.15E-05$	$1.05E-05$	$1.92E-06$
	ε	+	+	+	+	+	+	+	
	Rank	8	4	5	6	7	3	2	1
f_8	Mean	$-2.79E+03$	$-7.46E+03$	$-7.59E+03$	$-7.54E+03$	$-5.55E+02$	$-2.83E+03$	$-2.92E+03$	$-2.56E+03$
	Std	$3.32E+02$	$8.53E+02$	$8.17E+02$	$7.50E+02$	$5.06E+02$	$3.31E+02$	$2.88E+02$	$4.19E+02$
	ε	+	+	+	+	−	+	+	
	Rank	3	6	8	7	1	4	5	2
f_9	Mean	$1.12E+01$	0	0	0	0	0	0	0
	Std	$6.67E+00$	0	0	0	0	0	0	0
	ε	+	=	=	=	=	=	=	
	Rank	2	1	1	1	1	1	1	1
f_{10}	Mean	$2.00E+01$	$8.88E-16$	$8.88E-16$	$8.88E-16$	$2.00E+01$	$2.00E+01$	$2.00E+01$	$8.88E-16$
	Std	$5.84E-04$	0	0	0	$6.05E-14$	$9.64E-05$	$6.39E-04$	0
	ε	+	=	=	=	=	=	=	=
	Rank	4	1	1	1	2	3	5	1
f_{11}	Mean	$2.15E-01$	0	0	0	0	0	0	0
	Std	$9.05E-02$	0	0	0	0	0	0	0
	ε	+	=	=	=	=	=	=	
	Rank	2	1	1	1	1	1	1	1
f_{12}	Mean	$2.46E-01$	$2.34E-02$	$2.64E-02$	$5.46E-02$	$5.78E-12$	$5.12E-10$	$3.07E-12$	$1.66E-02$
	Std	$5.96E-01$	$1.56E-02$	$2.62E-02$	$6.32E-02$	$2.73E-12$	$3.81E-10$	$1.21E-12$	$8.99E-16$
	ε	+	+	+	+	−	−	−	
	Rank	8	5	6	7	2	3	1	4
f_{13}	Mean	$8.24E-04$	$2.43E+00$	$2.14E+00$	$2.62E+00$	$2.15E-03$	$1.92E-03$	$5.49E-04$	$2.70E+00$
	Std	$2.89E-03$	$1.16E+00$	$1.33E+00$	$9.68E-01$	$5.43E-03$	$4.17E-03$	$2.39E-03$	$6.76E-02$
	ε	−	+	+	+	−	−	−	
	Rank	2	8	7	6	4	3	1	5
	ε	12/0/1	10/3/0	10/3/0	10/3/0	7/3/3	8/3/2	4/7/2	+/=/−
Rank	Average	5.53	3.92	4.76	5.07	3.61	3.38	2.38	1.76
	Overall	8	5	6	7	4	3	2	1

Table 4. Comparison between SCQ-SSA and other variants on benchmark functions.

No.	Index	CAOA	DMO	PPSO	HHO-PSO	HHO-GWO	HHO-SCA	SCQ-SSA
f_1	Mean	$6.55E-08$	$2.48E-12$	$2.35E-02$	$7.62E-98$	$4.61E-96$	$1.86E-91$	$1.30E-133$
	Std	$7.56E-08$	$2.14E-12$	$2.73E-02$	$3.11E-97$	$2.50E-95$	$9.49E-91$	$4.70E-133$
	ε	+	+	+	+	+	+	
	Rank	6	5	7	2	3	4	1
f_2	Mean	$2.05E-05$	$5.78E-09$	$1.04E-01$	$2.49E-51$	$1.55E-48$	$2.46E-51$	$2.05E-53$
	Std	$1.86E-05$	$3.03E-09$	$7.25E-02$	$1.20E-50$	$8.42E-48$	$1.11E-50$	$4.79E-53$
	ε	+	+	+	+	+	+	
	Rank	6	5	7	3	4	2	1
f_3	Mean	$2.28E-04$	$3.26E+04$	$2.04E-01$	$7.38E-75$	$1.53E-68$	$8.88E-72$	$1.10E-131$
	Std	$1.57E-04$	$1.02E+04$	$2.69E-01$	$4.02E-74$	$8.36E-68$	$4.86E-71$	$4.40E-131$
	ε	+	+	+	+	+	+	
	Rank	5	7	6	2	4	3	1
f_4	Mean	$1.38E-03$	$2.53E+01$	$2.59E-02$	$1.23E-47$	$3.30E-49$	$8.02E-49$	$5.29E-67$
	Std	$1.59E-03$	$4.44E+00$	$4.04E-02$	$6.73E-47$	$1.21E-48$	$2.83E-48$	$2.71E-66$
	ε	+	+	+	+	+	+	
	Rank	5	7	6	4	2	3	1
f_5	Mean	$2.55E+01$	$1.06E+02$	$2.89E+01$	$7.32E-03$	$1.70E-02$	$1.43E-02$	$2.88E+01$
	Std	$1.98E+00$	$1.38E+02$	$5.03E-01$	$8.53E-03$	$2.16E-02$	$2.02E-02$	$8.15E-02$
	ε	+	+	+	−	−	−	
	Rank	6	7	5	1	3	2	4
f_6	Mean	$3.41E+00$	$3.57E-12$	$3.63E-01$	$1.44E-04$	$1.64E-04$	$2.24E-04$	$1.02E-04$
	Std	$1.56E-01$	$1.99E-12$	$2.39E-01$	$2.50E-04$	$3.08E-04$	$3.38E-04$	$2.37E-04$
	ε	+	−	+	+	+	+	
	Rank	7	1	6	3	4	5	2
f_7	Mean	$5.16E-06$	$3.18E-02$	$2.73E-03$	$1.77E-04$	$1.49E-04$	$1.22E-04$	$2.21E-05$
	Std	$3.28E-06$	$6.07E-03$	$2.22E-03$	$1.74E-04$	$1.15E-04$	$1.10E-04$	$2.35E-03$
	ε	−	+	+	+	+	+	
	Rank	1	7	6	5	4	3	2
f_8	Mean	$-5.94E+03$	$-9.83E+03$	$-9.62E+03$	$-1.25E+04$	$-1.26E+04$	$-1.25E+04$	$-2.59E+03$
	Std	$2.90E+02$	$7.94E+02$	$1.43E+03$	$9.46E-01$	$1.04E+00$	$7.66E-01$	$4.28E+02$
	ε	+	+	+	+	+	+	
	Rank	2	3	4	6	7	5	1
f_9	Mean	$1.22E-08$	$1.72E+02$	$4.31E-01$	0	0	0	0
	Std	$2.82E-08$	$2.36E+01$	$1.83E+00$	0	0	0	0
	ε	+	+	+	=	=	=	
	Rank	2	4	3	1	1	1	1
f_{10}	Mean	$1.12E-05$	$1.11E-06$	$6.06E-02$	$8.88E-16$	$8.88E-16$	$8.88E-16$	$8.88E-16$
	Std	$2.86E-05$	$5.09E-07$	$1.17E-01$	0	0	0	0
	ε	+	+	+	=	=	=	
	Rank	3	2	4	1	1	1	1
f_{11}	Mean	$3.45E-03$	$2.95E-10$	$8.76E-02$	0	0	0	0
	Std	$4.68E-03$	$3.90E-10$	$9.34E-02$	0	0	0	0
	ε	+	+	+	=	=	=	
	Rank	3	2	4	1	1	1	1
f_{12}	Mean	$5.86E-01$	$1.87E-02$	$3.67E-02$	$1.13E-05$	$1.13E-05$	$1.13E-05$	$1.66E-02$
	Std	$1.83E-01$	$5.23E-02$	$5.72E-02$	$1.50E-05$	$1.50E-05$	$1.50E-05$	$1.13E-15$
	ε	+	+	+	−	−	−	
	Rank	5	3	4	1	1	1	2
f_{13}	Mean	$2.94E+00$	$1.71E-03$	$1.64E+00$	$1.13E-04$	$1.13E-04$	$1.13E-04$	$2.73E+00$
	Std	$5.93E-02$	$3.21E-03$	$5.06E-01$	$1.66E-04$	$1.66E-04$	$1.66E-04$	$6.15E-02$
	ε	+	−	+	−	−	−	
	Rank	4	2	5	1	1	1	3
	ε	12/0/1	11/0/2	13/0/0	7/3/3	7/3/3	7/3/3	+/=/-
Rank	Average	4.23	4.23	5.15	2.38	2.76	2.46	1.61
	Overall	5	5	6	2	4	3	1

ity of SCQ-SSA algorithm is better than other algorithms in high-dimensional functions.

4.4 Comparison with the SSA variants

In this subsection, to further investigate the efficiency of the proposed SSA variant, we compared SCQ-SSA with original slap swarm algorithm (SSA, [15]) and six other SSA variants, including Gaussian slap swarm algorithm (GSSA, [34]), Cauchy slap swarm algorithm (CSSA, [34]), Levy slap swarm algorithm (LSSA, [34]), Weight factor and adaptive mutation slap swarm algorithm (WASSA, [35]), Craziness and adaptive slap swarm algorithm (CASSA, [36]), Efficient salp swarm algorithm (ESSA, [37]). In this experiment, the number of dimensions is set to 30, the maximum number of iterations is set to 5000, the population size is 100, and each experiment is run 40 times. The parameter settings of the above algorithms are based on the corresponding original papers

The mean value and standard deviation are used to evaluate the ability of the algorithm in Table 3, a summary of average rank and overall rank are counted in the bottom of Table 3. Inspecting the data, it is seen that the SCQ-SSA receives the first rank, then ESSA also performs well, and CASSA, WASSA, GSSA, CSSA, LSSA, SSA rank third to eighth respectively. In terms of mean value, SCQ-SSA can exhibit fine performance on ten functions (f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_9, f_{10}, f_{11}). Six SSA variants have similar performance as SCQ-SSA on functions (f_9, f_{10}, f_{11}), but it can be obviously noticed that SCQ-SSA achieves the relative optimal solutions on most functions (f_1, f_2, f_3, f_4, f_9, f_{11}) and even converges to the optimal value 0. As can be seen from the overall rank data of mean and standard deviation, the stability and reliability of SCQ-SSA are obviously better than other variants.

4.5 Comparison with the advanced variants

A total of thirteen benchmark functions are used to compare the performances with chaotic arithmetic optimization algorithm (CAOA, [38]), Dwarf mongoose optimization algorithm (DMO, [39]), Pausing particle swarm optimization (PPSO, [40]), Hybrid Harris Hawks optimizer and particle swarm optimization (HHO-PSO, [41]), Hybrid Harris Hawks optimizer and grey wolf optimizer (HHO-GWO, [42]), Hybrid Harris Hawks optimizer and sine cosine algorithm (HHO-SCA, [43]). This experiment is fixed on the population of 50. The maximum number of iterations is set as 1000 in all algorithms; the dimension is fixed at 30 and each experiment is conducted for 30 times.

The comparison results are provided in Table 4. There is a slight increase in the performance of both algorithms on functions (f_1, f_2, f_3, f_4) and SCQ-SSA can even exceed $E-40$ to $E-60$ than the second-ranked algorithm. Especially, SCQ-SSA achieves the relative optimal solutions in most functions and obtain the optimal value 0 on functions (f_9, f_{10}, f_{11}). HHO-PSO, HHO-GWO, HHO-SSA perform good searchability and obtain the fine values on functions (f_5, f_6, f_7, f_{12}, f_{13}), it can be seen from average and overall rank, all three algorithms are 7/3/3, and average rank gap is small, but the best overall rank is SCQ-SSA. It means that SCQ-SS substantially improves the searchability of basic SSA, the precision and stability of the algorithm are significantly enhanced.

5 SCQ-SSA for engineering optimization

In this section, the performance of SCQ-SSA is further verified on three classical engineering problems, namely tubular column design problem, tension/compression spring design problem and pressure vessel design problem. The schematic diagrams of three design problems are drawn on AutoCAD 2010. The optimization model is as follows:

The engineering problems are optimized by SCQ-SSA according to the following steps:

- *Step* 1: The constraint engineering problem is transformed by the penalty function Eq. (3) to the unconstrained objective function.
- *Step* 2: The dimension of the algorithm is equal to the impact factors of the engineering problem.
- *Step* 3: The objective function of the engineering problem is considered as the fitness function of the algorithms. So the SCQ-SSA optimize the objective function and obtain the minimum value.

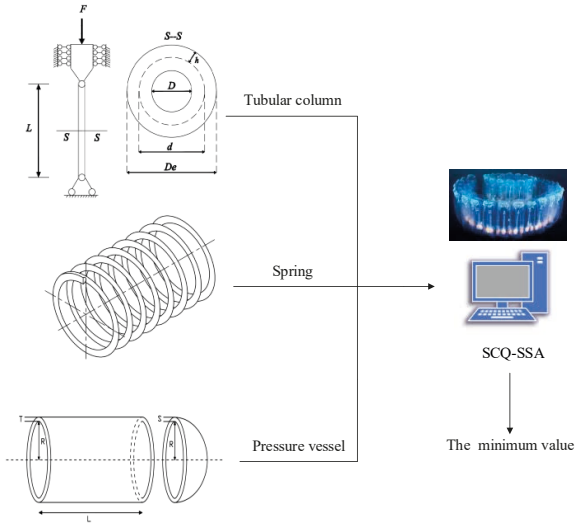

Figure 2. The optimization model.

5.1 Tubular column design problem

Figure 3 shows the design of tubular column with the lowest cost under the condition of bearing pressure load $F = 2500kgf$. The physical parameters of the column is as follow: the elastic modulus $E = 0.85 * 10^6 kgf/cm^2$,the yield stress of the material $\sigma = 500kgf/cm^2$, the density $\rho = 0.0025kgf/cm^3$,The length of the column $L = 250cm$. The constraints are as follows: the stress of the column should be less than the buckling stress (g_1) and the yield stress (g_2), the average diameter of the column d is limited in between $2cm$ (g_3) and $14cm$ (g_4), the thickness of the column (h) is in the range of $0.2cm$ (g_5) and $0.8cm$ (g_6). The mathematical model is as follows:

$$
\begin{aligned}
&Consider \quad \vec{x} = [d,h] = [x_1,x_2] \\
&Minimise \quad F(\vec{x}) = 9.82x_1x_2 + 2x_1 \\
&Subject\ to \quad \begin{cases}
g_1 = \frac{F}{\pi x_1 x_2 \sigma} - 1 \le 0 \\
g_2 = \frac{8FL^2}{\pi^3 E x_1 x_2 (x_1^2 + x_2^2)} - 1 \le 0 \\
g_3 = 2/x_1 - 1 \le 0 \\
g_4 = x_1/14 - 1 \le 0 \\
g_5 = 0.2/x_2 - 1 \le 0 \\
g_6 = x_2/0.8 - 1 \le 0
\end{cases}
\end{aligned}
$$

As indicated in Table 5, the tubular column is optimized by eight algorithms, namely Krill herd algorithm (KH, [44]), Interior search algorithm (ISA, [45]), Cuckoo search (CS, [29]), Hybridizing the electromagnetism-like algorithm (Hybrid EM, [46]), Modified flower pollination algorithm (MFPA, [47]), Competitive search algorithm (CSA, [48]), Hybrid self-adaptive orthogonal genetic algorithm (HSO-GA, [49]), Hybrid-Flash Butterfly Optimization Algorithm (HFBOA, [50]).

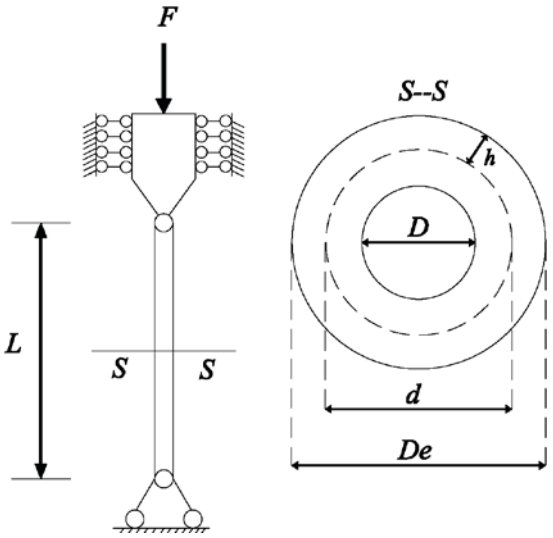

Figure 3. The schematic diagram of tubular column.

Table 5. Comparison results for tubular column design problem.

Index	x_1	x_2	$F(\vec{x})$
KH	5.4512	0.2919	26.5314
ISA	5.4511	0.2919	26.5313
CS	5.4513	0.2919	26.5321
Hybrid EM	5.4510	0.2919	26.5322
MFPA	5.4512	0.2919	26.4999
CSA	5.4511	0.2919	26.5313
HSO-GA	5.4511	0.2919	26.5313
HFBOA	5.4511	0.2919	26.4995
SCQ-SSA	5.4512	0.2919	26.4962

Two key parameters are calculated and the subjective function value is obtained. From the table we can know, SCQ-SSA outperforms eight other algorithms and achieves the lowest cost, and the difference value is relatively minimal. The optimal values of HFBOA and MFPA are below 26.5. then ISA, CSA and HSO-GA have similar performance and the function value is 26.531, followed by KH, CS and Hybrid EM. From the data know that SCQ-SSA does well in tubular column design problem.

5.2 Spring design problem

Spring is an important component in the industrial system, so the tension/compression spring design problem becomes a popular structural design problem. Fig. 4 shows the three main impact factors, namely the diameter of the wire (d), the turns of the coil (N) and the diameter of the coil (D). Taking into account the parameters, the optimization goal is to obtain the minimum weight. The mathematical formulation of this problem is as follows:

$$\begin{aligned}
&\textit{Consider} \quad \vec{x} = [d, D, N] = [x_1, x_2, x_3] \\
&\textit{Minimise} \quad F(\vec{x}) = (x_3 + 2) x_2 x_1^2 \\
&\textit{Subject to} \quad \begin{cases} g_1(\vec{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0 \\ g_2(\vec{x}) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} \leq 0 \\ g_3(\vec{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0 \\ g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0 \end{cases} \\
&\textit{Where} \\
&0.05 \leq x_1 \leq 2, 0.25 \leq x_2 \leq 1.3, 2 \leq x_3 \leq 15
\end{aligned}$$

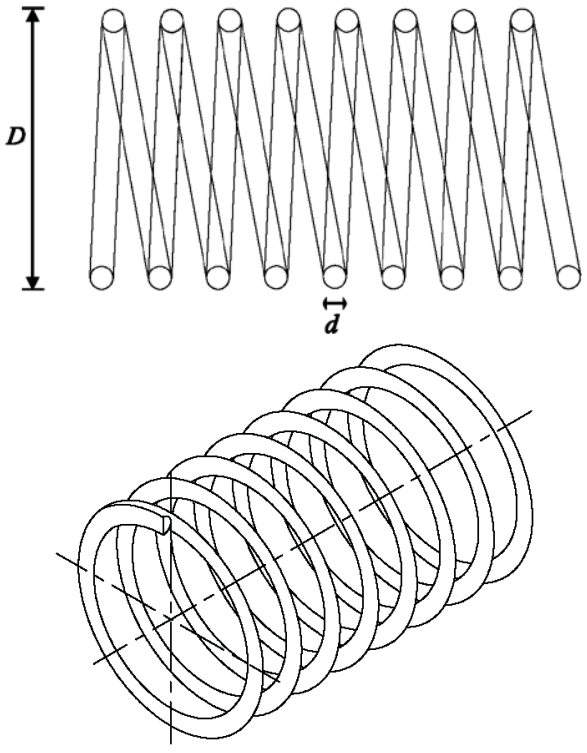

Figure 4. The schematic diagram of spring.

We analyze the structural weight design of tension/compression spring in this paragraph. Table 6 summarizes the comparison results between the proposed SSA variants and Velocity pausing particle swarm optimization (VPPSO, [40]), Elite archives-driven particle swarm optimization (EAPSO, [51]), Termite life cycle optimizer (TLCO, [52]), Sand cat swarm optimization (SCSO, [53]), Exploitation-boosted sine cosine algorithm (EBSCA, [54]), adaptive quadratic interpolation and rounding mechanism sine cosine algorithm (ARSCA, [55]), quantum particle swarm optimization with optimal guided Lévy flight and straight flight (LSFQPSO, [56]), adaptive cooperative foraging and dispersed foraging strategies harris hawks optimization (ADHHO, [57]), Parallel fish migration optimization with compact technology (PCFMO, [58]), efficient salp swarm algorithm (ESSA, [37]).

The minimum weight of the spring and the minimal value are gotten in SCQ-SSA. And ESSA, ADHHO, ARSCA, EAPSO also perform well, the difference between these algorithms and the lowest value of SCQ-SSA is $10E-6$. SCSO obtains the large weight. These results expose that SCQ-SSA handles the tension/compression spring efficiently.

Table 6. Comparison results for spring design problem.

Algorithm	x_1	x_2	x_3	$F(\vec{x})$
VPPSO	0.05250	0.37560	10.26590	0.012700
EAPSO	0.05152	0.35271	11.52783	0.012666
TLCO	0.05168	0.35673	11.28782	0.012702
SCSO	0.05000	0.31750	14.02000	0.012717
EBSCA	0.05161	0.35493	11.39630	0.012668
ARSCA	0.05165	0.35586	11.34009	0.012666
LSFQPSO	0.05151	0.35239	11.55200	0.012672
ADHHO	0.05184	0.36044	11.07410	0.012666
PCFMO	0.05240	0.37416	10.33800	0.012867
ESSA	0.05173	0.35773	11.23048	0.012666
SCQ-SSA	0.05163	0.35542	11.36510	0.012665

5.3 Pressure vessel design problem

The head and tail of the cylindrical pressure vessel are sealed with a hemispherical head respectively in Fig. 5. There are many considerations involved in the minimum weight design of the pressure vessel, mainly including four factors: shell thickness (T), head thickness (S), inner radius (R), cylindrical length of the vessel (L) and L excluding two heads. Therefore, the objective function expression of the pressure vessel design problem is as follows:

$$
\begin{aligned}
&\textit{Consider} \quad \overrightarrow{x} = [T,S,R,L] = [x_1,x_2,x_3,x_4] \\
&\textit{Minimise} \quad F(\overrightarrow{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 \\
&+3.1661x_1^2x_4 + 19.84x_1^2x_3 \\
&\textit{Subject to} \quad \begin{cases} g_1(\overrightarrow{x}) = -x_1 + 0.0193x_3 \leq 0 \\ g_2(\overrightarrow{x}) = -x_2 + 0.00954x_3 \leq 0 \\ g_3(\overrightarrow{x}) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 \\ +1296000 \leq 0 \\ g_4(\overrightarrow{x}) = x_4 - 240 \leq 0 \end{cases} \\
&\textit{Where} \quad 0 \leq x_1, x_2 \leq 99,\ 10 \leq x_3, x_4 \leq 200
\end{aligned}
$$

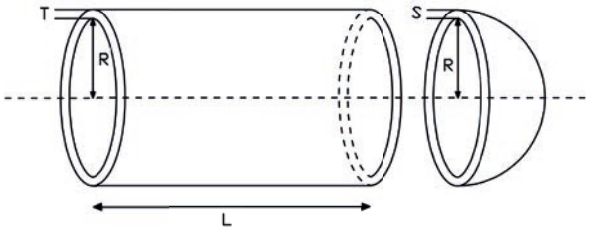

Figure 5. The schematic diagram of pressure vessel.

Table 7. Comparison results for pressure vessel design problem.

Algorithm	x_1	x_2	x_3	x_4	$F(\overrightarrow{x})$
BeSD	0.794	0.412	41.047	191.877	6029.171
HPSO	0.812	0.437	42.098	176.636	6059.714
PHSSA	0.815	0.426	42.091	176.742	6043.986
AOA	0.830	0.416	42.751	169.345	6048.784
PRO	1.125	0.625	58.290	43.692	6050.713
EWOA	0.901	0.452	46.678	127.096	6160.209
ARSCA	0.812	0.437	42.097	176.645	6059.805
SCQ-SSA	0.812	0.403	42.334	173.741	5947.187

As revealed in Table 7, the pressure vessel design of SCQ-SSA is compared with the bezier search differential evolution algorithm (BeSD, [59]), the human behavior-based particle swarm optimization (HPSO, [60]), the hybridized SSA along with the proportional selection scheme (PHSSA, [61]), Arithmetic optimization algorithm (AOA, [33]), Poor and rich optimization algorithm (PRO, [62]), Evolutionary biogeography-based whale optimization method (EWOA, [63]), adaptive quadratic interpolation and rounding mechanism sine cosine algorithm (ARSCA, [55])

It is evident that SCQ-SSA manages to optimize four structural parameters and obtains wonderful results with the minimum weight. The weight is calculated by the other algorithms is over 6000, the weight is relatively large and the design is not desired. In contrast, SCQ-SSA outperforms seven rest algorithms obviously, and the design of the weight is optimized to reduce about 210. It is considered to be a sign of the fine performance of the SCQ-SSA in solving this problem.

6 Conclusion

In this paper, three main strategies are adopted to enhance the standard SSA, namely SCQ-SSA. The initial value is generated by Circle chaos, the sine-cosine mechanism and quantum computation are employed to modify the update of the position. SCQ-SSA is compared with seven algorithms in high-dimensional functions (1000 dimensions), seven SSA variants and six advanced variants to validate the search performance on the benchmark functions. The results demonstrate the superiority of SCQ-SSA in providing good quality of solutions with high convergence performance for most functions. Besides, SCQ-SSA optimizes the design of three classical engineering problems, including tubular column design problem, tension/compression spring design problem and pressure vessel design problem. These data prove the superiority of SCQ-SSA overs the existing approaches in solving these optimization problems.

In future work, we will share more the improvement of meta-heuristic algorithms, such as GOA, SSA and WOA as well. Meta-heuristic algorithms and machine learning algorithms will be integrated to optimize prediction and classification. In addition, the meta-heuristic algorithm will be employed to optimize more engineering problems, for instance, speed reducer design, car side impact design, reinforced concrete beam design.

Acknowledgement

This work was supported in part by the key scientific research projects of colleges and universities of Henan Province under Grant 22B520015, the Training Plan for Young Backbone Teachers of Henan Medical College under Grant 2020010005.

References

[1] Luo Q, Rao Y, Peng D. GA and GWO algorithm for the special bin packing problem encountered in field of aircraft arrangement. Applied Soft Computing, 2022, 114: 108060.

[2] Guo H, Hou X, Cao Z, et al. GP3: Gaussian process path planning for reliable shortest path in transportation networks. IEEE Transactions on Intelligent Transportation Systems, 2022, 23(8):11575-11590.

[3] Shanthi J, Rani D G N, Rajaram S. An Enhanced Memetic Algorithm using SKB tree representation for fixed-outline and temperature driven non-slicing floorplanning. Integration, 2022, 86:84-97.

[4] Li L, Cai Y, Zhou Q. A survey on machine learning-based routing for VLSI physical design. Integration, 2022, 86:51-56.

[5] Muhammad, Yasir and Raja, Muhammad Asif Zahoor and Altaf, Muhammad et al. Design of fractional comprehensive learning PSO strategy for optimal power flow problems. Applied Soft Computing, 2022, 130:109638.

[6] Javed S, Ishaque K. A comprehensive analyses with new findings of different PSO variants for MPPT problem under partial shading. Ain Shams Engineering Journal, 2022, 13(5): 101680.

[7] Ye Y, Huang Q, Rong Y, et al. Field detection of small pests through stochastic gradient descent with genetic algorithm. Computers and Electronics in Agriculture, 2023, 206: 107694.

[8] Deng W, Zhang X, Zhou Y, et al. An enhanced fast non-dominated solution sorting genetic algorithm for multi-objective problems. Information Sciences, 2022, 585: 441-453.

[9] Jiang Y, Wu Q, Zhu S, et al. Orca predation algorithm: A novel bio-inspired algorithm for global optimization problems. Expert Systems with Applications, 2022, 188: 116026.

[10] Braik M, Hammouri A, Atwan J, et al. White Shark Optimizer: A novel bio-inspired meta-heuristic algorithm for global optimization problems. Knowledge-Based Systems, 2022, 243: 108457.

[11] Wang L, Cao Q, Zhang Z, et al. Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems. Engineering Applications of Artificial Intelligence, 2022, 114: 105082.

[12] Shuan-Jun Song and Cheng-Hong Qiu and Long-Guang Peng et al. An Assembly Line Multi-Station Assembly Sequence Planning Method Based on Particle Swarm Optimization Algorithm. Journal of Computers, 2022, 33: 115-125.

[13] Xu S F, Jiang Y N. An Optimization Method of Knowledge Mapping Relationship Based on Improved Ant Colony Algorithm. Journal of Computers, 2022, 33(2): 137-147.

[14] Ke G, Chen R S, Chen Y C, et al. Network Security Situation Prediction Method Based on Support Vector Machine Optimized by Artificial Bee Colony Algorithms. Journal of Computers, 2021, 32(1): 144-153.

[15] Mirjalili S, Gandomi A H, Mirjalili S Z, et al. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. Advances in engineering software, 2017, 114: 163-191.

[16] Mahajan S, Mittal N, Salgotra R, et al. An efficient adaptive salp swarm algorithm using type II fuzzy entropy for multilevel thresholding image segmentation. Computational and Mathematical Methods in Medicine, 2022, 2022.

[17] Nayak S, Kar S K, Dash S S, et al. Enhanced Salp Swarm Algorithm for Multimodal Optimization and Fuzzy Based Grid Frequency Controller Design. Energies, 2022, 15(9): 3210.

[18] Ponnusamy M, Bedi P, Suresh T, et al. Design and analysis of text document clustering using salp swarm algorithm. The Journal of Supercomputing, 2022: 1-17.

[19] Zhang J, Liu W, Tian Z, et al. Urban Rail Substation Parameter Optimization by Energy Audit and Modified Salp Swarm Algorithm. IEEE Transactions on Power Delivery, 2022.

[20] Abdelkader E M, Moselhi O, Marzouk M, et al. An exponential chaotic differential evolution algorithm for optimizing bridge maintenance plans. Automation in Construction, 2022, 134: 104107.

[21] Khalaf K S, Sharif M A, Wahhab M S. Digital Communication Based on Image Security using Grasshopper Optimization and Chaotic Map. International Journal of Engineering, 2022, 35(10): 1981-1988.

[22] Alshammari M E, Ramli M A M, Mehedi I M. Hybrid Chaotic Maps-Based Artificial Bee Colony for Solving Wind Energy-Integrated Power Dispatch Problem. Energies, 2022, 15(13): 4578.

[23] Kohli, Mehak and Arora, Sankalap. Chaotic grey wolf optimization algorithm for constrained optimization problems. Journal of Computational Design and Engineering, 2018, 5(4): 458-472.

[24] W. Ding and C. Lin and M. Prasad. A Layered-Coevolution-Based Attribute-Boosted Reduction Using Adaptive Quantum Behavior PSO and Its Consistent Segmentation for Neonates Brain Tissue. IEEE Transactions on Fuzzy Systems, 2018, 26(3): 1177-1191.

[25] K. Srikanth and L. K. Panwar and B. Panigrahi. Meta-heuristic framework: Quantum inspired binary grey wolf optimizer for unit commitment problem. Computers & Electrical Engineering, 2018, 70: 243-260.

[26] D. Zouache and F. Nouioua and A Moussaoui. Quantum-inspired firefly algorithm with particle swarm optimization for discrete optimization problems. Soft Computing, 2016, 20(7): 2781-2799.

[27] Rashedi E, Nezamabadi-Pour H, Saryazdi S. GSA: a gravitational search algorithm. Information sciences, 2009, 179(13): 2232-2248.

[28] Mirjalili S, Mirjalili S M, Lewis A. Grey wolf optimizer. Advances in engineering software, 2014, 69: 46-61.

[29] Gandomi A H, Yang X S, Alavi A H. Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Engineering with computers, 2013, 29: 17-35.

[30] Mirjalili S, Lewis A. The whale optimization algorithm. Advances in engineering software, 2016, 95: 51-67.

[31] Mirjalili S. SCA: a sine cosine algorithm for solving optimization problems. Knowledge-based systems, 2016, 96: 120-133.

[32] Mirjalili S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. Knowledge-based systems, 2015, 89: 228-249.

[33] Abualigah L, Diabat A, Mirjalili S, et al. The arithmetic optimization algorithm. Computer methods in applied mechanics and engineering, 2021, 376: 113609.

[34] Nautiyal B, Prakash R, Vimal V, et al. Improved salp swarm algorithm with mutation schemes for solving global optimization and engineering problems. Engineering with Computers, 2021: 1-23.

[35] Wu J, Nan R, Chen L. Improved salp swarm algorithm based on weight factor and adaptive mutation. Journal of Experimental & Theoretical Artificial Intelligence, 2019, 31(3): 493-515.

[36] Zhang D, Chen Z, Xin Z, et al. Salp swarm algorithm based on craziness and adaptive. Control and Decision, 2020, 35(9): 2112-2120.

[37] Wang C, Xu R, Ma L, et al. An efficient salp swarm algorithm based on scale-free informed followers with self-adaption weight. Applied Intelligence, 2023, 53(2): 1759-1791.

[38] Aydemir S B. A novel arithmetic optimization algorithm based on chaotic maps for global optimization. Evolutionary Intelligence, 2022: 1-16.

[39] Agushaka J O, Ezugwu A E, Abualigah L. Dwarf mongoose optimization algorithm. Computer methods in applied mechanics and engineering, 2022, 391: 114570.

[40] Shami T M, Mirjalili S, Al-Eryani Y, et al. Velocity pausing particle swarm optimization: a novel variant for global optimization. Neural Computing and Applications, 2023: 1-31.

[41] Sarma R, Bhargava C, Jain S, et al. Application of ameliorated Harris Hawks optimizer for designing of low-power signed floating-point MAC architecture. Neural Computing and Applications, 2021, 33: 8893-8922.

[42] Nandi A, Kamboj V K. A Canis lupus inspired upgraded Harris hawks optimizer for nonlinear, constrained, continuous, and discrete engineering design problem. International Journal for Numerical Methods in Engineering, 2021, 122(4): 1051-1088.

[43] Kamboj V K, Nandi A, Bhadoria A, et al. An intensify Harris Hawks optimizer for numerical and engineering optimization problems. Applied Soft Computing, 2020, 89: 106018.

[44] Gandomi A H, Alavi A H. Krill herd: a new bio-inspired optimization algorithm. Communications in nonlinear science and numerical simulation, 2012, 17(12): 4831-4845.

[45] Gandomi A H. Interior search algorithm (ISA): a novel approach for global optimization. ISA transactions, 2014, 53(4): 1168-1183.

[46] Rocha A M A C, Fernandes E M G P. Hybridizing the electromagnetism-like algorithm with descent search for solving engineering design problems. International Journal of Computer Mathematics, 2009, 86(10-11): 1932-1946.

[47] Meng O K, Pauline O, Kiong S C, et al. Application of modified flower pollination algorithm on mechanical engineering design problem, IOP conference series: materials science and engineering. IOP Publishing, 2017, 165(1): 012032.

[48] Xu Y, Liu H, Xie S, et al. Competitive search algorithm: a new method for stochastic optimization. Applied Intelligence, 2022, 52(11): 12131-12154.

[49] Zhongyang J, Zixing C, Yong W. Hybrid self-adaptive orthogonal genetic algorithm for solving global optimization problems. Journal of Software, 2010, 21(6): 1296-1307.

[50] Zhang M, Wang D, Yang J. Hybrid-flash butterfly optimization algorithm with logistic mapping for solving the engineering constrained optimization problems. Entropy, 2022, 24(4): 525.

[51] Zhang Y. Elite archives-driven particle swarm optimization for large scale numerical optimization and its engineering applications. Swarm and Evolutionary Computation, 2023, 76: 101212.

[52] Minh H L, Sang-To T, Theraulaz G, et al. Termite life cycle optimizer. Expert Systems with Applications, 2023, 213: 119211.

[53] Seyyedabbasi A, Kiani F. Sand Cat swarm optimization: A nature-inspired algorithm to solve global optimization problems. Engineering with Computers, 2022: 1-25.

[54] Li C, Liang K, Chen Y, et al. An exploitation-boosted sine cosine algorithm for global optimization. Engineering Applications of Artificial Intelligence, 2023, 117: 105620.

[55] Yang X, Wang R, Zhao D, et al. An adaptive quadratic interpolation and rounding mechanism sine cosine algorithm with application to constrained engineering optimization problems. Expert Systems with Applications, 2023, 213: 119041.

[56] Liu X, Wang G G, Wang L. LSFQPSO: quantum particle swarm optimization with optimal guided Lévy flight and straight flight for solving optimization problems. Engineering with Computers, 2022, 38(Suppl 5): 4651-4682.

[57] Zhang X, Zhao K, Niu Y. Improved Harris hawks optimization based on adaptive cooperative foraging and dispersed foraging strategies. IEEE Access, 2020, 8: 160297-160314.

[58] Chu S C, Xu X W, Yang S Y, et al. Parallel fish migration optimization with compact technology based on memory principle for wireless sensor networks. Knowledge-Based Systems, 2022, 241: 108124.

[59] Akgüngör A P, Korkmaz E. Bezier Search Differential Evolution algorithm based estimation models of delay parameter k for signalized intersections. Concurrency and Computation: Practice and Experience, 2022, 34(13): e6931.

[60] Liu H, Zhang X W, Liang H, et al. Stability analysis of the human behavior-based particle swarm optimization without stagnation assumption. Expert Systems with Applications, 2020, 159: 113638.

[61] Abualigah L, Shehab M, Diabat A, et al. Selection scheme sensitivity for a hybrid Salp Swarm Algorithm: analysis and applications. Engineering with Computers, 2022, 38(2): 1149-1175.

[62] Moosavi S H S, Bardsiri V K. Poor and rich optimization algorithm: A new human-based and multi populations algorithm. Engineering Applications of Artificial Intelligence, 2019, 86: 165-181.

[63] Tu J, Chen H, Liu J, et al. Evolutionary biogeography-based whale optimization methods with communication structure: Towards measuring the balance. Knowledge-Based Systems, 2021, 212: 106642.

[64] Castelli M, Manzoni L, Mariot L, et al. Salp swarm optimization: a critical review. Expert Systems with Applications, 2022, 189: 116029.

Sheng Luo is currently pursuing the bachelor's degree with the Software engineering, School of Information Science and Technology, Nantong University. His research interests include machine learning and image recognition.
https://orcid.org/0009-0002-1498-9684

Guan Yin received the B.S. degree in land resource management from the Gannan Normal University ,China,in 2021. He is currently pursuing the master's degree with the School of Land Science and Technology, China University of Geosciences. His research interests include machine learning and remote sensing image processing.
https://orcid.org/0009-0002-7853-7767

Yin Ye received the B.S. degree in software engineering from the Jiangxi University of Science and Technology, China, in 2017. He obtained the M.S. degree in computer technology from Fuzhou University, China, in 2020. He is currently pursuing the doctor's degree with the School of Computer Science and Artificial Intelligence, Wuhan University of Technology. He has published four international conference papers and six SCI journal papers. His research interests include meta-heuristic algorithm, intelligent computing, machine learning and engineering application.
https://orcid.org/0000-0002-7653-3669

Fuyun Jia received the B.S. degree in computer science and technology from the Henan University of Finance and Economics, China, in 2010. He obtained his M.S. degree in computer software and theory from Fuzhou University, China, in 2015. He is currently working as a lecturer at Henan Medical College. His research interests include intelligent computing and computer application.
https://orcid.org/0009-0004-9413-1462