Physica A 537 (2020) 122621

Contents lists available at ScienceDirect

Physica A

journal homepage: www.elsevier.com/locate/physa

Enhance chaotic gravitational search algorithm (CGSA) by balance adjustment mechanism and sine randomness function for continuous optimization problems☆

Jianhua Jiang [a,b,*], Xi Yang [a], Xianqiu Meng [a], Keqin Li [c,*]

[a] Department of Data Science, Jilin University of Finance and Economics, Changchun 130117, PR China
[b] Key Laboratory of Symbolic Computation and Knowledge Engineering, Ministry of Education, Jilin University, Changchun, 130012, PR China
[c] Department of Computer Science, State University of New York, New Paltz, NY 12561, USA

ARTICLE INFO

Article history:
Received 26 June 2019
Received in revised form 6 September 2019
Available online 17 September 2019

Keywords:
Optimization algorithm
Gravitational search algorithm
Sine cosine algorithm
Balance adjustment mechanism

ABSTRACT

The gravitational search algorithm (GSA) is a population-based meta-heuristic optimization algorithm which finds the optimal solution by the law of gravity and attraction between objects. However, as the number of iterations increases, the increase of the quality of the agents makes GSA fall into the local optimal solution more easily, which greatly reduces the exploration capability of the algorithm. Although the chaotic gravitational search algorithm (CGSA) uses chaotic maps for improving diversity to solve this problem, it still has problems with the balance of exploration and exploitation. This paper proposes the balance adjustment based chaotic gravitational search algorithm (BA-CGSA), which introduces the sine randomness function and the balance mechanism to solve the above problem. 30 benchmark functions of IEEE CEC 2014 are adopted to evaluate the performance of the proposed algorithm in terms of exploration and exploitation. Meanwhile, a real engineering design problem is used to illustrate the ability of the algorithm to solve practical application problems. The experimental results demonstrate its good performance in continuous optimization problems.

1. Introduction

The optimization refers to the process of finding optimal values for the parameters of a given system from all the possible values to maximize or minimize its output [1]. Optimization method is applied in many fields [2–6]. In general, optimization algorithms can be classified into two main categories: deterministic and stochastic [7]. Due to the drawbacks of the conventional optimization algorithms such as local optimal stagnation, the research on stochastic optimization [8] algorithm has become an academic hotspot in recent years [9–13]. The stochastic optimization algorithm treats the optimization problem as a black box [14], which maximizes or minimizes system outputs to meet requirements through changing inputs and adjusting system parameters. The characteristics of the black box make the stochastic optimization

☆ The authors are grateful to the financial support by theNational Natural Science Foundation of China (no. 61572225), Natural Science Foundation of the Science and Technology Department of Jilin Province, China (no. 20180101044JC), the Social Science Foundation of Jilin Province, China (nos. 2019B68, 2017BS28), the Foundation of the Education Department of Jilin Province, China (nos. JJKH20180465 kJ) and the Foundation of Jilin University of Finance and Economics (no. 2018Z05).
* Corresponding authors.
E-mail addresses: jianhuajiang@yahoo.com (J. Jiang), lik@newpaltz.edu (K. Li).

https://doi.org/10.1016/j.physa.2019.122621
0378-4371/

algorithm highly flexible and easy to be applied in different fields, which essentially benefits from higher local optimal avoidance compared with traditional optimization algorithms.

Inspirations of a new algorithm can be from evolutionary phenomena, the collective behavior of creatures (swarm intelligence techniques), physical rules, and human-related concepts and so on. Some popular algorithms in each of these subclasses are as follows:

- Evolutionary techniques: Genetic Algorithms (GA) [15], Differential Evolution (DE) [16–18], Biogeography-Based Optimization algorithm (BBO) [19], Evolution Strategy (ES) [20], etc.
- Swarm intelligence techniques: Ant Colony Optimization (ACO) [21], Particle Swarm Optimization (PSO) [22], Artificial Bee Colony (ABC) [2,23] and Firefly Algorithm (FA) [24], Sine Cosine Algorithm(SCA) [1], etc.
- Physics-based techniques: Gravitational Search Algorithm (GSA) [25], Chaotic gravitational constants for the Gravitational Search Algorithm (CGSA) [26], Colliding Bodies Optimization (CBO) [27], Black Hole (BH) [28], etc.
- Human-related techniques: League Championship Algorithm (LCA) [29], Mine Blast Algorithm (MBA) [30], Teaching–Learning-Based Optimization (TLBO) [31], etc.

Chaotic gravitational search algorithm (CGSA) was proposed by Seyedali Mirjalili and Amir H. Gandomi in 2017 [26] based on GSA. CGSA uses chaotic functions to increase the perturbation strength to help the agents jump out of the local optimal solution, but the convergence of the algorithm is greatly reduced. Therefore, how to adjust the random mode to find the best balance of disturbance intensity is a question that is worth considering. We can see that the final results of applying 10 kinds of chaotic maps are not the same. Among them, the comprehensive performance of the chaotic map 9, the sinusoidal map, is the best [26]. After examining its characteristics, we boldly suspect that a sine function with similar properties will perform well.

In addition, the algorithm uses the same search strategy throughout the entire iteration, which makes it impossible to complete tasks of different standards in the exploration stage and the exploitation stage efficiently. According to the law of universal gravitation to updates the moving speed and position, CGSA can find the optimal solution. The gravitational force between the objects determines the acceleration of the motion, thus determining the speed and position of the motion at the next moment. Therefore, the acceleration of the motion of the agent point plays a crucial role in the determination of its position. The whole iterative process should be divided into two parts. The exploration phase should enhance the effect of acceleration to help the agent point to explore a larger search range and jump out of the local optimal solution. The search phase should reduce the effect of acceleration, so as to find the optimal solution faster. As mentioned earlier, CGSA does not adjust the effects of acceleration throughout the iteration, which should be an important reason for the lack of convergence of the algorithm.

In summary, the following two questions are raised:

Problem 1. Whether the new stochastic mechanism superimposed by the sine function and the chaotic map helps to optimize finding optimal solutions.

Problem 2. Whether the acceleration adjustment coefficient changing with the stage helps to ameliorate the convergence of the algorithm.

In this paper, we propose the BA-CGSA algorithm to verify the above problems. The major contributions of this paper can be summarized as:

- Optimize the perturbation mode by superimposing the chaotic map and the sine function.
- Use the constraint coefficient k to control the effect of acceleration at different stages.
- The superposition of sine function and a chaotic map performs better than the hybrid of chaotic map and normal random function, exploring the new way of stochastic optimization.

This paper is presented as follows: Firstly, Section 2 introduces the related algorithm; Secondly, the innovative ideas and proposals of the BA-CGSA are presented in Section 3; Thirdly, the comparative experiments and an actual engineering design problem are introduced in Section 4; In Section 5, discussion is given to analyze the performance of BA-CGSA algorithm; Finally, conclusions and future works are demonstrated in the last Section 6.

2. Related work

The proposed BA-CGSA algorithm is inspired by GSA, CGSA, and SCA. The principle of the related algorithms will be introduced in Section 2.

2.1. GSA: gravitational search algorithm

GSA is a new heuristic intelligent optimization algorithm, which was first proposed by Esmat Rashedi in 2009 [25]. Based on Newton's law of gravity and the interaction between particles, GSA estimates the global optimum of the given

search space. In GSA, all agents are considered as objects, and their performances are measured by their masses. Any two agents attract each other by the gravity force, and all of them move towards the heavier masses which represent the better solutions. When the algorithm satisfies the stop criterion, the position of the agent with the largest inertial mass represents the best optimal solution obtained so far.

Now, considering a gravitational system with N agents (masses), and the i_{th} agent position is defined as Eq. (1).

$$X_i = (x_i^1, x_i^2, \ldots x_i^k, \ldots x_i^d), \ for\ i = 1, 2, \ldots, N \tag{1}$$

where d is the number of variables, and x_i^k represents the position of i agent in the k_{th} dimension.

The gravitational constant and the Euclidean distance between agent i and j are calculated by Eqs. (2) and (3).

$$G(t) = G_0 \times e^{(-\alpha \frac{t}{T})} \tag{2}$$

$$R_{ij}(t) = \|X_i(t), X_j(t)\|_2 \tag{3}$$

where G_0 is the initial value for the gravitational constant, α is a constant, t is the current iteration, and T is the maximum number of iterations, the number 2 represents the square, and it means that the value is calculated with Euclidean distance equations.

The gravitational forces between agents are calculated as Eq. (4).

$$F_{ij}^d(t) = G(t)\frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon}(x_j^d(t) - x_i^d(t)) \tag{4}$$

where M_{pi} is the passive gravitational mass of agent i, M_{aj} is the active gravitational mass of agent j, ε is a constant.

The gravitational force between agent i and all the other agents are calculated as Eq. (5).

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} rand_j \times F_{ij}^d(t) \tag{5}$$

where $rand_j$ is a random number in the interval [0, 1].

Then, by the law of gravity, the acceleration and velocities of agent i at time t in the d_{th} direction should be calculated as Eqs. (6) and (7).

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ij}(t)} \tag{6}$$

where M_{ij} is the inertial mass of i_{th} agent,

The next velocity of the agent is considered as a fraction of its current velocity added to its acceleration. Therefore, the next velocity could be calculated as follows:

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t) \tag{7}$$

where $rand_i$ is a random number in the interval [0, 1], $v_i^d(t)$ represents the current velocity of the agent, $a_i^d(t)$ is the current acceleration.

After calculating the acceleration and velocity, the position of the agent can be updated with Eq. (8).

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{8}$$

The mass of the agent represents the fitness value calculated by the fitness function. With the greater mass, the agent moves at a slower speed. It is more important that both gravitation and influence are increased. So a heavier mass means a more efficient agent which has higher attractions and more slowly walk. The relationship between mass and fitness makes the mass of agents need to be updated as Eqs. (9) and (10).

$$m_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)} \tag{9}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \tag{10}$$

where $fit_i(t)$ is the fitness value of the solution i in the t_{th} iteration, $best(t)$ is the best solution in the iteration t, and $worst(t)$ is the worst solution in the iteration t.

After all, the pseudo-code of the GSA algorithm is presented as follows:

Initialize a set of random search agents m
Evaluate the fitness value of each agent by objective functions
while (the iterative criterion is not satisfied)
    Update $G(t)$, $M_i(t)$ at time t for $i = 1, 2, \ldots, N$ by Eqs. (2), (9) and (10)
    Calculate the acceleration and velocity of agents by Eqs. (6) and (7)
    Update the positions of all the agents by Eq. (8)
end while
return the optimal solution

Algorithm 1: Pseudo-code of the basic GSA algorithm

Although GSA has a superior performance compared with PSO [22] and GA [15], this algorithm still has some problems. During the iteration, the mass of the search agent is becoming heavier and heavier by the updating equation of mass (Eqs. (9) and (10)). Ultimately, the searching speed is deteriorated by the increasing iteration. The convergence of the algorithm is affected by the increase of the iteration number.

2.2. CGSA: chaotical gravitational search algorithm

Chaotic gravitational search algorithm (CGSA) is a new optimization algorithm based on GSA. It was proposed by Seyedali Mirjalili and Amir H.Gandomi in 2017 [26]. Although GSA has a super performance compared to PSO [22] and GA [15], it suffers from an essential problem [26]. Due to the direct effect of fitness function on calculating mass, search agents get heavier and heavier over the course of iterations [26]. CGSA introduces chaotic maps to solve these problems.

In order to redefine G, the normalized range is reduced proportionally to the iteration, and its equation is shown in Eq. (11).

$$V(t) = MAX - \frac{t}{T}(MAX - MIN) \tag{11}$$

where T is the maximum number of iterations, t is the current iteration, [MAX,MIN] represents the adaptive interval.

Normalize $C_k(t)$ *from* $[a, b]$ *to* $[0, V(t)]$

$$C_k^{norm}(t) = \frac{(C_k(t) - a) \times (V(t) - 0)}{(b - a)} \tag{12}$$

where k indicates the index of chaotic maps, t is the current iteration, and $[a, b]$ shows the range of chaotic maps. It can be seen in Eq. (12) that we map $[a, b]$ to $[0, V(t)]$ in each iteration while $V(t)$ is decreased by iterations.

The final value of the gravitational constant is updated in Eq. (13).

$$G(t) = C_k^{norm}(t) + G_0 \times \exp^{(-\alpha \frac{t}{T})} \tag{13}$$

where k indicates the index of chaotic maps, t is the current iteration, G_0 is the initial gravitational constant, α is the descending coefficient, and T is the maximum number of iterations.

The chaotic maps selected are listed as follows:

(1) Chebyshev [32]

$$x_{i+1} = cos(icos^{-1}(x_i))$$

The range of this map lies in the interval of $(-1, 1)$.

(2) Circle [33]

$$x_{i+1} = mod(x_i + b - (\frac{a}{2\pi})\sin(2\pi x_i), 1), \qquad a = 0.5,\ b = 0.2,$$

The range of this map lies in the interval of $(0, 1)$.

(3) Gauss/mouse [34]

$$x_{i+1} = \begin{cases} 1, & if\ x_i = 0 \\ \dfrac{1}{mod(x_i, 1)}, & otherwise \end{cases}$$

The range of this map lies in the interval of $(0, 1)$.

(4) Iterative [35]

$$x_{i+1} = \sin(\frac{a\pi}{x_i}), \qquad a = 0.7$$

The range of this map lies in the interval of $(-1, 1)$.
(5) Logistic [35]

$$x_{i+1} = ax_i(1 - x_i), \qquad a = 4$$

The range of this map lies in the interval of $(0, 1)$.
(6) Piecewise [36]

$$x_{i+1} = \begin{cases} \frac{x_i}{P}, & 0 \le x_i < P, P = 0.4 \\ \frac{x_i - P}{0.5 - P}, & P \le x_i < 0.5 \\ \frac{1 - P - x_i}{0.5 - P}, & 0.5 \le x_i < 1 - P \\ \frac{1 - x_i}{P}, & 1 - P \le x_i < 1 \end{cases}$$

The range of this map lies in the interval of $(0, 1)$.
(7) Sine [37]

$$x_{i+1} = \frac{a}{4}\sin \pi x_i, \qquad a = 4$$

The range of this map lies in the interval of $(0, 1)$.
(8) Singer [38]

$$x_{i+1} = \mu(7.86x_i - 23.31x_i^2 + 28.75x_i^3 - 13.302875x_i^4), \qquad \mu = 2.3$$

The range of this map lies in the interval of $(0, 1)$.
(9) Sinusoida [39]

$$x_{i+1} = ax_i^2 \sin(\pi x_i), \qquad a = 2.3$$

The range of this map lies in the interval of $(0, 1)$.
(10) Tent [40]

$$x_{i+1} = \begin{cases} \frac{x_i}{0.7}, & x_i < 0.7 \\ \frac{10}{3}(1 - x_i), & x_i \ge 0.7 \end{cases}$$

The range of this map lies in the interval of $(0, 1)$.

Compared with GSA, CGSA performs better [26], but it still lacks in optimal solution and convergence. CGSA incorporates chaotic functions into the GSA search formula. The random perturbation of the algorithm is increased, which greatly improves the global exploration capability of CGSA. At the same time, however, the convergence and optimal value of the algorithm are also affected. Compared with GSA, the convergence speed of CGSA has not been significantly improved, which is the inadequacy of CGSA.

2.3. SCA: sine cosine algorithm

The SCA proposed by Seyedali Mirjalili in 2016 is a novel intelligent population-based random optimization algorithm for searching global optimal solution [1]. The SCA creates multiple initialed random candidate solutions and optimizes them by using a mathematical model based on sine and cosine functions. Several random and adaptive variables are integrated into this algorithm to balance the exploration and exploitation in the different iterations of the optimization. The results of unimodal test function show that SCA algorithm converges substantially faster than FA [7], BA [41], FPA [42], GSA [25], PSO [22] and GA [15]. The SCA changes its search agents abruptly in the initial stage of optimization and gradually in the final steps of optimization, which guarantees the brilliant performance of exploration, exploitation, local optima avoidance, and convergence of the algorithm. The great characters make SCA perform illustriously in solving challenging real problems, such as the airfoil shape design [1]. The stochastic optimization algorithms consider optimization problems as black boxes, which makes the SCA more flexibility, meaning the stochastic algorithms are readily applicable to problems in different fields [1].

SCA starts the optimization process with a set of random solutions, and then updates the position of each agent as Eqs. (14) and (15)

$$X_i^{t+1} = X_i^t + r_1 \times \sin(r_2) \times \left| r_3 P_i^t - X_i^t \right|, \qquad r_4 < 0.5 \tag{14}$$

$$X_i^{t+1} = X_i^t + r_1 \times cos(r_2) \times \left| r_3 P_i^t - X_i^t \right|, \qquad r_4 \geq 0.5 \tag{15}$$

where X_i^t is the position of the current solution in i_{th} dimension at t_{th} iteration, P_i is the destination point in i_{th} dimension. r_1, r_2, r_3, r_4 are four main parameters in the SCA algorithm. The parameter r_1 defines how far the movement should be towards or outwards the destination. The parameter r_2 decides the movement direction. The parameter r_3 gives random weights for destination. At last, the parameter r_4 equally switches between the sine and cosine components in Eqs. (14) and (15).

In order to balance exploration and exploitation, the parameter r_1 is changed adaptively by the Eq. (16).

$$r_1 = a - t \times \frac{a}{T} \tag{16}$$

where t is the current iteration, T is the maximum number of iterations, and a is a constant, such as $a = 2$. The pseudo-code of the SCA algorithm is presented as Algorithm 2.

Initialize a set of random search agents X
Do
    Evaluate the fitness value of each agent by objective functions
    Update the best solution obtained so far $P = X^*$
    Update parameters r_1, r_2, r_3 and r_4
    Update the positions of search agents by using Eqs. (14) and (15)
While (satisfy the maximum number of iterations)
Return the best solution obtained so far as the global optimum

Algorithm 2: Pseudo-code of the SCA algorithm

The parameter r_1 is one of the important inspirations brought by the SCA algorithm. It decreases linearly from the constant a. This ensures that the search agents can explore at a high rate during the initial iteration and exploit carefully during the final iteration. This mechanism guarantees the balance between exploration and exploitation in the different iterations of the optimization.

The SCA algorithm uses the unique properties of the sine function to explore the search space, which makes the SCA algorithm more simple and efficient, showing outstanding experimental results. The excellent performance of the sine function in the SCA algorithm makes us pay attention to this simple but efficient function. It inspires us to explore the search space by using the sine function when looking for the optimal solution.

3. Methods

3.1. The design of the weighting coefficient of acceleration

Inspired by the parameter r_1 in the SCA algorithm, we add the constraint parameter k to the new proposed algorithm, aiming to obtain better experimental results.

In the SCA algorithm, the parameter r_1 is a linearly decreasing variable, which globally controls the movement length of the agent point, which affects its next iteration position. In the entire iterative process, the parameter r_1 gradually decreases from the maximum value to zero. This allows the agent point to move fast in the early stage and slow in the later. The constraint action of the parameter r_1 helps the algorithm to realize the transition from exploration to exploitation. The parameter r_1 plays a crucial role in the entire SCA algorithm. The equation for calculating the new parameters is shown in Eq. (17).

$$k = c - c \times \frac{iteration}{max\_it} \tag{17}$$

where c is a constant, *iteration* indicates the current iteration, *max_it* is the maximum number of iterations.

The linear constraint parameter k controls the change of the acceleration, which affects the variation of the velocity, and finally, have an impact on the next iteration position of the agent point. After many tests, when the initial value of k is set to 2, the experimental results are best.

3.2. The improvement of random weight of speed

The sine function is fused into the GSA algorithm to optimize the algorithm. Adding the sine function to the velocity update equation of the GSA algorithm makes the result better. The next position of the proxy point is made more reasonably because of the new equation.

The speed update equation of the new algorithm is shown in Eq. (18).

$$v'_{t+1} = \sin(rand \times \pi) \times v_t \tag{18}$$

where *rand* is a random variable in the interval [0, 1].

3.3. The new speed equation

According to Sections 3.1 and 3.2, we can combine Eqs. (17) and (18) to get the final speed update equation in Eq. (19).

$$v_i^{t+1} = \sin(rand \times \pi) \times v_i^t + k \times a \tag{19}$$

where a is the acceleration of the agent i at time t, *rand* is a random variable in the interval [0, 1].

This formula has been improved in acceleration and speed, and the optimal solution can be obtained at a lower cost. We propose the new stochastic optimization algorithm BA-CGSA with the new velocity update equation.

3.4. The BA-CGSA algorithm

By adding the weighting coefficient of acceleration and the random weight of velocity, the speed update equation of the new algorithm is designed more reasonable, which makes it more rational to obtain a next position of the agent point. Therefore the algorithm can find a better optimal solution at a faster convergence speed. In summary, we propose the new algorithm BA-CGSA with the improved velocity updating equations.

The flow chart of BA-CGSA is shown in Fig. 1.

The pseudo-code of BA-CGSA is shown in Algorithm 3.

Initialize a set of random search agents m
Evaluate the fitness value of each agent by objective functions
while (the iterative criterion is not satisfied)
    Renew $M_i(t)$ by Eqs. (9) and (10)
    Update $C_k^{norm}(t)$ and $G(t)$ by Eqs. (12) and (13)
    Assgin the total force of agent i in different directions by Eq. (4)
    Estimate the acceleration of each agent by Eq. (6)
    Evaluate the weighting coefficient of acceleration k by **Eq. (17)**
    Compute the random weight of velocity by **Eq. (18)**
    Calculate the velocity of each agent by **Eq. (19)**
    Attain the positions of all the agents by Eq. (8)
end while
return the optimal solution

Algorithm 3: Pseudo-code of the BA-CGSA algorithm

4. Experiments and analysis

In order to test the performance and efficiency of the proposed BA-CGSA, a variety of experiments are conducted. Due to the stochastic nature of the optimization algorithm, we employ an appropriate and sufficient sets of test functions and case studies to ensure that the superior results of the algorithm do not happen by chance. Firstly, we make a comparative experiment with CGSA by four sets of benchmark functions of IEEE CEC 2014. Secondly, we compare it with four well-known population-based optimization algorithms: SCA [1], PSO [22], ABC [23], TSA [43]. PSO, TSA and, ABC are well-regarded and widely-used algorithms in the field, so using them as comparison algorithms is a good choice and a challenge for the proposed algorithm. SCA is a good population-based optimization algorithm. Although it is a new algorithm that has just been proposed in recent years, it has attracted the attention of many scholars. Based on this, many variants have been proposed [44–49]. All these algorithms are good comparative algorithms to prove the superiority of BA-CGSA for solving the high-dimensional global optimization problems. Thirdly, a challenging real-world engineering optimization problem, the pressure vessel design, is utilized to verify the search-ability of the BA-CGSA. The following subsections illustrate the initial values of the above experiments, show the experimental results respectively, and analysis the results in detail.

All experimental data are coded in the Matlab R2016a environment under Windows 10 operating system. All simulations are running on a computer with Intel Core(TM) i7-6700HQ CPU @ 2.60 GHz with 8G of memory. The benchmark functions we used in the first two experiments are displayed in Table 1.

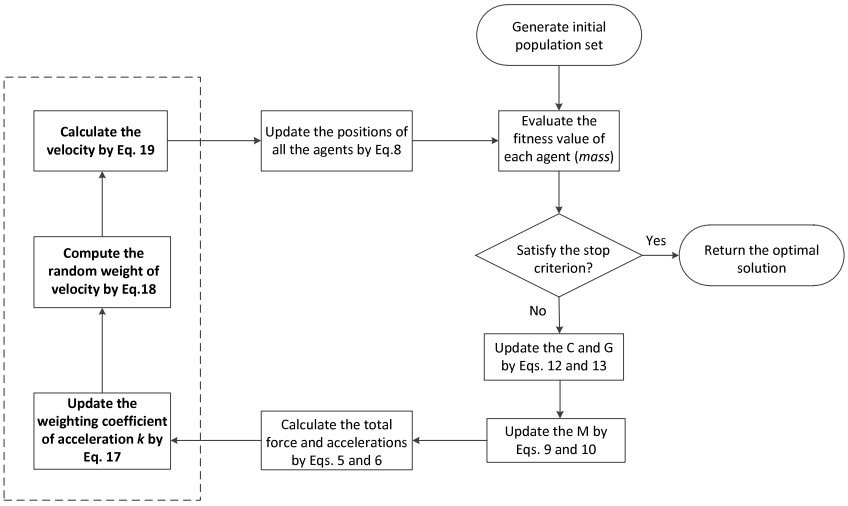


Fig. 1. The flow chart of BA-CGSA.

4.1. Comparison between BA-CGSA and CGSA

We compare with CGSA in terms of convergence speed, optimal solution, stability, and runtime. Four criteria are used for the two algorithms: best, mean, worst, and standard deviation. For a fair comparison, the common parameters are equally set. In each function, the population size (N) is fixed to 30, corresponding to the dimensions 30, 50, and 100, and the maximum number of iterations (*iter_max*) is 500 on all the simulations. This means the maximum number of fitness function evaluations (*FFEs*) is 15,000. Each group of tests runs more than 30 times.

The comparison results of CGSA and BA-CGSA are reported in Table 2, where *best* refers to the best value, *mean* represents the average best values, *worst* denotes the worst value, and the *std* means the standard deviation value. The experimental results of the optimal solutions of BA-CGSA and CGSA in three dimensions ($Dim = 30, 50, 100$) are compared in Table 3, where count represents the number of better solutions.

From Table 2, it can be noticed that the BA-CGSA obtains a better optima (0) under 24 benchmark test functions ($f1$, $f2$, $f4 - f19$, $f21$, $f22$, $f24$, $f25$, $f29$, $f30$) with a fixed number of fitness function evaluations of 15,000. This shows the excellent ability of BA-CGSA. Besides, it can be seen from Table 3 that, as the dimension increases, the number that BA-CGSA gets better solutions gradually increases. This proves BA-CGSA has more obvious advantages in solving high-dimensional stochastic optimization problems.

For further explanation, Fig. 2(f) shows the convergence curves of eight representative test functions for CGSA and BA-CGSA in 30 dimensions. From Fig. 2(f), we can see that BA-CGSA obtains better optimal solutions at a faster convergence rate. The outstanding performances show the excellent ability of the algorithm in terms of convergence and global optimal solution. Especially, BA-CGSA shows a surprising ability of convergence in the combination function $F29$, and the hybrid function $F18$.

To analyze whether BA-CGSA greatly adds the algorithm running time to achieve outstanding performances, we make a comparison between CGSA and BA-CGSA of the execution time (seconds) in three dimensions in Table 4.

As can be seen from Table 4, for all benchmark functions, the average CPU runtime of CGSA and BA-CGSA in three dimensions is very close. In other words, there is no significant difference in execution time between CGSA and BA-CGSA.

4.2. Comparison with other population-based algorithms

To further demonstrate the superiority of BA-CGSA in solving global optimization problems, we compare it with four well-known population-based optimization algorithms: SCA [1], PSO [22], ABC [23], TSA [43]. The dimensions of each test function are set to 100, and each algorithm executes independently over 30 times on each function. In this experiment, the initial population size of the five algorithms is set to 30, and the maximum number of iterations was 500. The comparative results are shown in Tables 5 and 6.

The experimental results show that the comprehensive average ranking of BA-CGSA is the first, which indicates the preeminent ability of it among six algorithms under 30 benchmark functions. In order to demonstrate the superiority of the proposed algorithm, we compare it with other five algorithms respectively in Table 7.

As shown in Table 7, regarding the multiple-problem Wilcoxon's test, the ISCA provides higher R+ values than R- values in all cases. Moreover, according to the Wilcoxon's rank-sum test, the p-values are less than 0.1 and 0.05 in all the cases. This means that the performance of the BA-CGSA is significantly better than all of the algorithms for high-dimensional problems.

Table 1
Benchmark functions of CEC 2014.

Function name	Function formulation
Unimodal functions	Note: M-rotation matrix
High Conditioned Elliptic Function	$F_1(x) = f_1(M(x - o_1)) + 100$
Rotated Bent Cigar Function	$F_2(x) = f_2(M(x - o_2)) + 200$
Rotated Discus Function	$F_3(x) = f_3(M(x - o_3)) + 300$
Multimodal functions	
Shifted and Rotated Rosenbrock's Function	$F_4(x) = f_4(M(\frac{2.048(x-o_4)}{100}) + 1) + 400$
Shifted and Rotated Ackley's Function	$F_5(x) = f_5(M(x - o_5)) + 500$
Shifted and Rotated Weierstrass Function	$F_6(x) = f_6(M(\frac{0.5(x-o_6)}{100})) + 600$
Shifted and Rotated Griewank's Function	$F_7(x) = f_7(M(\frac{600(x-o_7)}{100})) + 700$
Shifted Rastrigin's Function	$F_8(x) = f_8(\frac{5.12(x-o_8)}{100}) + 800$
Shifted and Rotated Rastrigin's Function	$F_9(x) = f_8(M(\frac{5.12(x-o_9)}{100})) + 900$
Shifted Schwefel's Function	$F_{10}(x) = f_9(\frac{1000(x-o_{10})}{100}) + 1000$
Shifted and Rotated Schwefel's Function	$F_{11}(x) = f_9(M(\frac{1000(x-o_{11})}{100})) + 1100$
Shifted and Rotated Katsuura Function	$F_{12}(x) = f_{10}(M(\frac{5(x-o_{12})}{100})) + 1200$
Shifted and Rotated HappyCat Function	$F_{13}(x) = f_{11}(M(\frac{5(x-o_{13})}{100})) + 1300$
Shifted and Rotated HGBat Function	$F_{14}(x) = f_{12}(M(\frac{5(x-o_{14})}{100})) + 1400$
Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	$F_{15}(x) = f_{13}(M(\frac{5(x-o_{15})}{100} + 1)) + 1500$
Shifted and Rotated Expanded Scaffer's F6 Function	$F_{16}(x) = f_{14}(M(x - o_{16}) + 1) + 1600$
Hybrid Functions	
$F17 = f_9(M_1z_1) + f_8(M2z_2) + f_1(M_3z_3) + 1700$	$p = [0.3, 0.3, 0.4]$
$F18 = f_2(M_1z_1) + f_{12}(M2z_2) + f_8(M_3z_3) + 1800$	$p = [0.3, 0.3, 0.4]$
$F19 = f_7(M_1z_1) + f_6(M2z_2) + f_4(M_3z_3) + f_{14}(M_4z_4) + 1900$	$p = [0.2, 0.2, 0.3, 0.3]$
$F20 = f_{12}(M_1z_1) + f_3(M2z_2) + f_{13}(M_3z_3) + f_8(M_4z_4) + 2000$	$p = [0.2, 0.2, 0.3, 0.3]$
$F21 = f_{14}(M_1z_1) + f_{12}(M2z_2) + f_4(M_3z_3) + f_9(M_4z_4) + f_1(M_5z_5) + 2100$	$p = [0.1, 0.2, 0.2, 0.2, 0.3]$
$F22 = f_{10}(M_1z_1) + f_{11}(M2z_2) + f_{13}(M_3z_3) + f_9(M_4z_4) + f_5(M_5z_5) + 2200$	$p = [0.1, 0.2, 0.2, 0.2, 0.3]$
Notes:	
$z_1 = [y_{S_1}, y_{S_2}, \ldots, y_{S_{n1}}]$,	
$z_2 = [y_{S_{n1+1}}, y_{S_{n1+2}}, \ldots, y_{S_{n1+n2}}]$,	
$z_N = [y_{S_{\sum_{i=1}^{N-1} n_i+1}}, y_{S_{\sum_{i=1}^{N-1} n_i+2}}, \ldots, y_{S_D}]$	
$y = x - o_i$, $S = randperm(1 : D)$, $p_i : percentageofg_i(x)$	
$n_1 = \lceil p_1 D \rceil$, $n_2 = \lceil p_2 D \rceil$, ..., $n_{N-1} = \lceil p_{N-1} D \rceil$, $n_N = D - \sum_{i=1}^{N-1} n_i$	
Composition Functions	
$F_{23} = \omega_1 * F'_4(x) + \omega_2 * \left[1e^{-6}F'_1(x) + 100\right] + \omega_3 * \left[1e^{-26}F'_2(x) + 200\right]$ $+\omega_4 * \left[1e^{-6}F'_3(x) + 300\right] + \omega_5 * \left[1e^{-6}F'_1(x) + 400\right] + 2300$	$\sigma = [10, 20, 30, 40, 50]$
$F_{24} = \omega_1 * F'_{10}(x) + \omega_2 * \left[F'_9(x) + 100\right] + \omega_3 * \left[F'_{14}(x) + 200\right] + 2400$	$\sigma = [20, 20, 20]$
$F_{25} = \omega_1 * 0.25F'_{11}(x) + \omega_2 * \left[F'_9(x) + 100\right] + \omega_3 * \left[1e^{-7}F'_1(x) + 200\right] + 2500$	$\sigma = [10, 30, 50]$
$F_{26} = \omega_1 * 0.25F'_{11}(x) + \omega_2 * \left[F'_{13}(x) + 100\right] + \omega_3 * \left[1e^{-7}F'_1(x) + 200\right]$ $+\omega_4 * \left[2.5F'_6(x) + 300\right] + \omega_5 * \left[10F'_7(x) + 400\right] + 2600$	$\sigma = [10, 10, 10, 10, 10]$
$F_{27} = \omega_1 * 10F'_{14}(x) + \omega_2 * \left[10F'_9(x) + 100\right] + \omega_3 * \left[2.5F'_{11}(x) + 200\right]$ $+\omega_4 * \left[25F'_6(x) + 300\right] + \omega_5 * \left[1e^{-6}F'_1(x) + 400\right] + 2700$	$\sigma = [10, 10, 10, 20, 20]$
$F_{28} = \omega_1 * 2.5F'_{15}(x) + \omega_2 * \left[10F'_{13}(x) + 100\right] + \omega_3 * \left[2.5F'_{11}(x) + 200\right]$ $+\omega_4 * \left[5e^{-4}F'_{16}(x) + 300\right] + \omega_5 * \left[1e^{-6}F'_1(x) + 400\right] + 2800$	$\sigma = [10, 20, 30, 40, 50]$
$F_{29} = \omega_1 * F'_{17}(x) + \omega_2 * \left[F'_{18}(x) + 100\right] + \omega_3 * \left[F'_{19}(x) + 200\right] + 2900$	$\sigma = [10, 30, 50]$
$F_{30} = \omega_1 * F'_{20}(x) + \omega_2 * \left[F'_{21}(x) + 100\right] + \omega_3 * \left[F'_{22}(x) + 200\right] + 3000$	$\sigma = [10, 30, 50]$
Notes:	
$\omega_i = \frac{1}{\sqrt{\sum_{j=1}^{D}(x_j - o_{ij})^2)}} exp(-\frac{\sum_{j=1}^{D}(x_j - o_{ij})^2}{2D\sigma_i^2})$	

Table 2
Comparison results of CGSA and BA-CGSA with $D = 100$.

Function	CGSA				BA-CGSA			
	Best	Mean	Worst	St.dev	Best	Mean	Worst	St.dev
F_1	2.8893E+08	5.3130E+08	7.7722E+08	9.9254E+07	6.5492E+07	**1.0241E+08**	1.5839E+08	2.1156E+07
F_2	7.8657E+10	9.4973E+10	1.1434E+11	7.5724E+09	1.3483E+10	**2.1958E+10**	3.5678E+10	4.9011E+09
F_3	2.0656E+05	**2.3277E+05**	2.8380E+05	1.7804E+04	2.1567E+05	2.5393E+05	2.9244E+05	1.8438E+04
F_4	1.0380E+04	1.3713E+04	1.9730E+04	2.0452E+03	1.6900E+03	**2.8053E+03**	4.1873E+03	6.0454E+02
F_5	5.2000E+02	5.2000E+02	5.2000E+02	6.4627E−04	5.2000E+02	**5.2000E+02**	5.2000E+02	5.1493E−04
F_6	7.2492E+02	7.3754E+02	7.4452E+02	4.0650E+00	6.9412E+02	**7.0638E+02**	7.2073E+02	5.4196E+00
F_7	1.5852E+03	1.7594E+03	1.9165E+03	7.7811E+01	8.9217E+02	**9.6607E+02**	1.0397E+03	3.8082E+01
F_8	1.3592E+03	1.3852E+03	1.4268E+03	1.7168E+01	1.2935E+03	**1.3392E+03**	1.3990E+03	2.7287E+01
F_9	1.5577E+03	1.6265E+03	1.7109E+03	3.7473E+01	1.4771E+03	**1.5397E+03**	1.6203E+03	3.9110E+01
F_{10}	1.6269E+04	1.7626E+04	1.9033E+04	7.6059E+02	1.4294E+04	**1.6681E+04**	1.8777E+04	1.0306E+03
F_{11}	1.3605E+04	1.5377E+04	1.8089E+04	1.0419E+03	1.1823E+04	**1.3901E+04**	1.6201E+04	1.1436E+03
F_{12}	1.2000E+03	1.2000E+03	1.2000E+03	8.9681E−03	1.2000E+03	**1.2000E+03**	1.2000E+03	4.9914E−03
F_{13}	1.3048E+03	1.3053E+03	1.3056E+03	2.1333E−01	1.3005E+03	**1.3010E+03**	1.3026E+03	6.8364E−01
F_{14}	1.6440E+03	1.6918E+03	1.7425E+03	2.1163E+01	1.4386E+03	**1.4702E+03**	1.5179E+03	1.9141E+01
F_{15}	9.5534E+04	1.8751E+05	3.2808E+05	5.3476E+04	2.9703E+03	**5.4741E+03**	1.2815E+04	2.3019E+03
F_{16}	1.6449E+03	1.6459E+03	1.6466E+03	3.4590E−01	1.6446E+03	**1.6458E+03**	1.6468E+03	5.7922E−01
F_{17}	1.2917E+07	3.5437E+07	5.9658E+07	1.2173E+07	2.0390E+06	**5.0678E+06**	9.5540E+06	1.5950E+06
F_{18}	2.6485E+03	7.6046E+07	7.7806E+08	1.5761E+08	2.6053E+03	**3.5935E+03**	6.5362E+03	1.0229E+03
F_{19}	2.3266E+03	2.4573E+03	2.6133E+03	6.4427E+01	2.0521E+03	**2.1196E+03**	2.2150E+03	3.7171E+01
F_{20}	1.1506E+05	**1.3750E+05**	1.6075E+05	1.2127E+04	9.8201E+04	1.3888E+05	1.7814E+05	1.4942E+04
F_{21}	3.0941E+06	5.0727E+06	1.0683E+07	1.7720E+06	1.6835E+06	**3.0459E+06**	4.9262E+06	7.1930E+05
F_{22}	4.3767E+03	5.7704E+03	7.1999E+03	7.2039E+02	3.8682E+03	**5.6300E+03**	7.0117E+03	6.9307E+02
F_{23}	2.5009E+03	**2.5350E+03**	3.0451E+03	1.2758E+02	2.5004E+03	2.7470E+03	2.9560E+03	1.5616E+02
F_{24}	2.7126E+03	2.7435E+03	2.7595E+03	1.1615E+01	2.7213E+03	**2.7399E+03**	2.7765E+03	1.1832E+01
F_{25}	2.7000E+03	2.7071E+03	2.7314E+03	9.7614E+00	2.7000E+03	**2.7067E+03**	2.7451E+03	1.1147E+01
F_{26}	2.8001E+03	**2.8002E+03**	2.8003E+03	4.2767E−02	2.8002E+03	2.8003E+03	2.8004E+03	4.9494E−02
F_{27}	8.1079E+03	**9.9263E+03**	1.1313E+04	7.5849E+02	9.2563E+03	1.0892E+04	1.2975E+04	9.5703E+02
F_{28}	9.3969E+03	**1.3358E+04**	1.7126E+04	1.6718E+03	1.1357E+04	1.5600E+04	1.9312E+04	2.0145E+03
F_{29}	3.0825E+06	1.3208E+08	1.2106E+09	3.5580E+08	7.9663E+03	**1.7939E+05**	2.6245E+06	6.4652E+05
F_{30}	2.2406E+06	5.3807E+06	9.8986E+06	1.9573E+06	7.3878E+05	**1.9223E+06**	1.1569E+07	1.9224E+06

4.3. BA-CGSA for engineering design problems

We further verify the performance and efficiency of BA-CGSA by solving a practical engineering problem, the pressure vessel design. This problem is widely discussed and solved in the literature to better demonstrate the effectiveness of the algorithm [50,51]. In this experiment, the initial population size is set to 30 and the maximum number of iteration is set to 500.

The purpose of solving the pressure vessel design problem is to minimize the overall cost of the cylindrical pressure vessel. The problem has four variables: the thickness of the shell (Ts), the thickness of the head (Th), the inner radius (R), and the length of the cylindrical portion that does not take into account the head (L). The schematic diagram of the pressure vessel is given in Fig. 3.

The mathematical formulation of pressure vessel design problem is defined as follows:

$$
\begin{aligned}
&\textit{Consider}\quad X = [x_1, x_2, x_3, x_4] = (T_s, T_h, R, L)\\
&\textit{Minimize}\quad f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1611x_1^2x_4 + 19.84x_1^2x_3\\
&\textit{Subjectto}\quad g_1(X) = -x_1 + 0.0193x_3 \le 0\\
&\qquad\qquad g_2(X) = -x_2 + 0.00954 \le 0\\
&\qquad\qquad g_3(X) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0\\
&\qquad\qquad g_4(X) = x_4 - 240.0 \le 0\\
&\qquad\qquad g_5(X) = -x_1 + 1.1 \le 0\\
&\qquad\qquad g_6(X) = -x_2 + 0.6 \le 0\\
&\textit{Where}\quad 1.1 \le x_1 \le 99, 0.6 \le x_2 \le 99, 10 \le x_3 \le 200, 10 \le x_4 \le 240
\end{aligned}
$$

To cope with the constraints of the PVD problem, a penalty function is introduced. If a constraint is violated, a penalty value is added to the objective function. For g_6 constraint, the penalty value is calculated as follows:

Table 3
Comparison results between BA-CGSA and CGSA in dimensions 30, 50, 100 with 15000 FFEs.

Function	Dim = 30		Dim = 50		Dim = 100	
	CGSA	BA-CGSA	CGSA	BA-CGSA	CGSA	BA-CGSA
F_1	6.4657E+06	**4.4683E+06**	3.7568E+07	**5.8657E+06**	5.3130E+08	**1.0241E+08**
F_2	8.4774E+03	**8.2209E+03**	6.1956E+09	**7.9515E+03**	9.4973E+10	**2.1958E+10**
F_3	**3.3854E+04**	4.1747E+04	**9.7715E+04**	1.1109E+05	**2.3277E+05**	2.5393E+05
F_4	5.0487E+02	**4.9052E+02**	1.2684E+03	**5.7190E+02**	1.3713E+04	**2.8053E+03**
F_5	**5.2000E+02**	**5.2000E+02**	**5.2000E+02**	**5.2000E+02**	**5.2000E+02**	**5.2000E+02**
F_6	6.2714E+02	**6.1554E+02**	6.5706E+02	**6.3584E+02**	7.3754E+02	**7.0638E+02**
F_7	**7.0000E+02**	**7.0000E+02**	7.5585E+02	**7.0000E+02**	1.7594E+03	**9.6607E+02**
F_8	9.4629E+02	**9.0254E+02**	1.0754E+03	**1.0321E+03**	1.3852E+03	**1.3392E+03**
F_9	1.0676E+03	**9.9890E+02**	1.2690E+03	**1.1387E+03**	1.6265E+03	**1.5397E+03**
F_{10}	4.8136E+03	**4.1552E+03**	8.8559E+03	**7.8056E+03**	1.7626E+04	**1.6681E+04**
F_{11}	5.6698E+03	**4.8041E+03**	9.5235E+03	**8.4381E+03**	1.5377E+04	**1.3901E+04**
F_{12}	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**	**1.2000E+03**
F_{13}	1.3004E+03	**1.3003E+03**	1.3006E+03	**1.3005E+03**	1.3053E+03	**1.3010E+03**
F_{14}	**1.4002E+03**	1.4003E+03	1.4007E+03	**1.4003E+03**	1.6918E+03	**1.4702E+03**
F_{15}	1.5206E+03	**1.5056E+03**	1.9276E+03	**1.5425E+03**	1.8751E+05	**5.4741E+03**
F_{16}	1.6137E+03	**1.6136E+03**	**1.6227E+03**	1.6228E+03	1.6459E+03	**1.6458E+03**
F_{17}	2.2480E+05	**2.2419E+05**	8.5995E+05	**8.3776E+05**	3.5437E+07	**5.0678E+06**
F_{18}	**2.1890E+03**	2.3398E+03	5.2610E+03	**3.7618E+03**	7.6046E+07	**3.5935E+03**
F_{19}	**1.9180E+03**	1.9181E+03	1.9546E+03	**1.9415E+03**	2.4573E+03	**2.1196E+03**
F_{20}	**2.3412E+04**	2.7151E+04	**2.7771E+04**	3.1272E+04	**1.3750E+05**	1.3888E+05
F_{21}	**1.5153E+05**	1.6792E+05	8.4063E+05	**7.4048E+05**	5.0727E+06	**3.0459E+06**
F_{22}	3.3100E+03	**3.2108E+03**	4.0501E+03	**4.0323E+03**	5.7704E+03	**5.6300E+03**
F_{23}	**2.5386E+03**	2.6152E+03	**2.5218E+03**	2.6516E+03	**2.5350E+03**	2.7470E+03
F_{24}	**2.6041E+03**	2.6133E+03	**2.6359E+03**	2.6592E+03	2.7435E+03	**2.7399E+03**
F_{25}	2.7036E+03	**2.7028E+03**	**2.7005E+03**	2.7014E+03	2.7071E+03	**2.7067E+03**
F_{26}	**2.7938E+03**	2.7946E+03	**2.8001E+03**	**2.8001E+03**	**2.8002E+03**	2.8003E+03
F_{27}	4.4657E+03	**3.7179E+03**	5.9205E+03	**5.8604E+03**	**1.0238E+04**	1.1151E+04
F_{28}	**5.0664E+03**	6.0102E+03	**8.5416E+03**	1.0075E+04	**1.3358E+04**	1.5600E+04
F_{29}	1.8438E+05	**6.5336E+03**	7.9259E+05	**9.3767E+03**	1.3208E+08	**1.7939E+05**
F_{30}	1.2208E+04	**1.1389E+04**	1.0946E+05	**5.6573E+04**	5.3807E+06	**1.9223E+06**
Count	13	20	10	23	8	24

Let be $x_2 = 0.1$, g_6 constraint function is calculated as 0.5, and the constraint is therefore violated. New $f(X)$ is obtained by using Eq. (20).

$$nF(X) = f(X) + \sum_{i=1}^{6} v_i \tag{20}$$

where v_i is the violation of i_{th} constraint and is calculated as Eq. (21).

$$v_i(X) = \begin{cases} h \times (g_i(X))^2 & if \quad g_i(X) > 0 \\ 0 & otherwise \end{cases} \tag{21}$$

where, h is a high positive constant number.

Briefly, the violation value (Eqs. (20) and (21)) of each constraint is added to objective function value. While the method minimizes the objective function, it is tried to cope with the constraints by using the penalty function (Eq. (21))

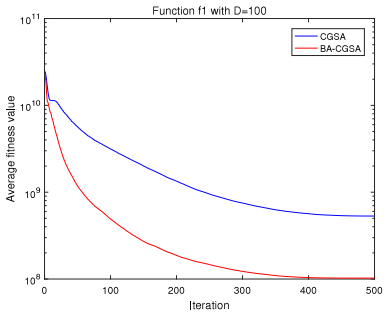
BA-CGSA is applied to solve the pressure vessel design problem and is compared with four other algorithms. The experimental results are displayed in Table 8.

From the table, we can see that the best value, mean value and worst value of the overall cost calculated by BA-CGSA are the best. This shows that BA-CGSA has an outstanding performance in solving this practical engineering problem.

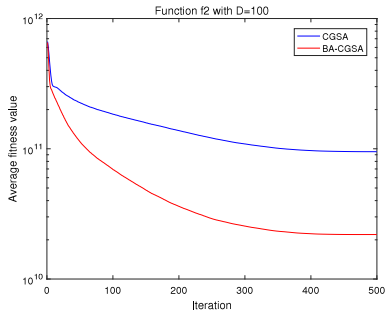
5. Discussion

5.1. The comparison with CGSA

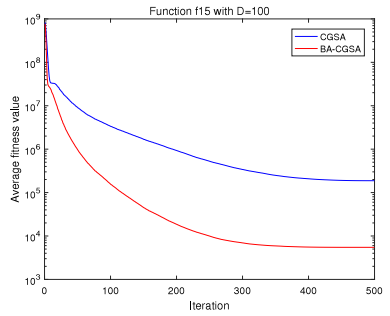
By comparing the experimental data, it can be seen that the average optimal solution of BA-CGSA is better than that of CGSA, which indicates the BA-CGSA algorithm has a stronger ability to find the global optimal solution. Moreover, the convergence of BA-CGSA is better. And these comparisons are more obvious in the high dimension, indicating that BA-CGSA has an advantage in solving high-dimensional global optimization problems. These advantages benefit from the balance adjustment mechanism of the algorithm.



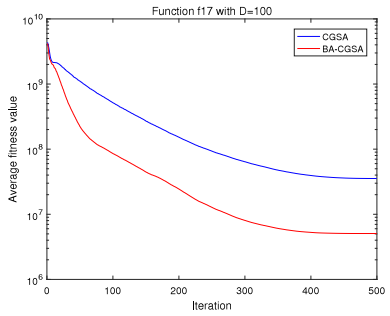
(a) Unimodal functions $F1$



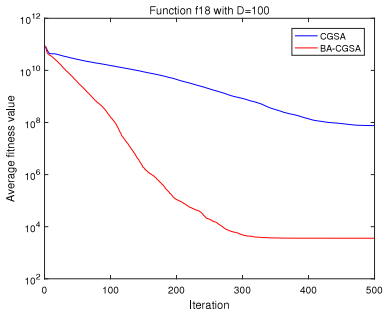
(b) Unimodal functions $F2$



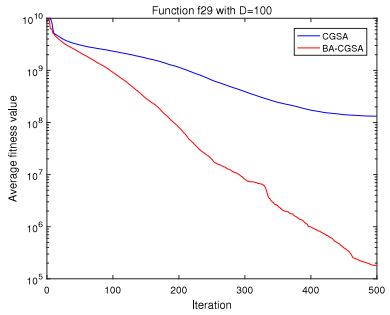
(c) Multimodal functions $F15$



(d) Hybrid Functions $F17$



(e) Hybrid Functions $F18$



(f) Composition Functions $F29$

Fig. 2. Convergence curves of CGSA and BA-CGSA on six representative test functions with D = 100.

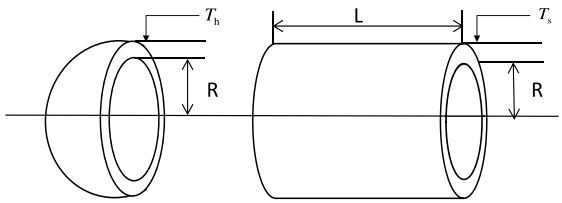


Fig. 3. The pressure vessel problem.

5.2. The comparison with other algorithms

Compared with other algorithms, the global optimal solution of BA-CGSA is better, which indicates that BA-CGSA has a stronger global search-ability. On the one hand, this is due to the balance adjustment mechanism of BA-CGSA. On the other hand, it benefits from the application of chaotic maps. The chaotic map allows the algorithm to jump out of the

Table 4
The mean execution time (seconds) obtained by CGSA and BA-CGSA.

Function	Dim = 30		Dim = 50		Dim = 100	
	CGSA	BA-CGSA	CGSA	BA-CGSA	CGSA	BA-CGSA
F_1	5.4512E+00	5.4937E+00	7.2915E+00	7.3341E+00	1.3540E+01	1.3516E+01
F_2	5.4402E+00	5.4085E+00	7.1654E+00	7.2309E+00	1.3295E+01	1.3325E+01
F_3	4.9438E+00	4.9583E+00	7.2465E+00	7.2904E+00	1.3139E+01	1.3263E+01
F_4	5.2831E+00	5.2088E+00	7.4066E+00	7.5381E+00	1.3288E+01	1.3247E+01
F_5	4.9676E+00	4.9867E+00	7.3740E+00	7.3721E+00	1.3251E+01	1.3346E+01
F_6	8.1424E+00	8.1679E+00	1.2193E+01	1.2253E+01	2.3315E+01	2.3300E+01
F_7	4.9920E+00	5.0093E+00	7.2261E+00	7.2671E+00	1.3314E+01	1.3372E+01
F_8	4.8860E+00	4.9260E+00	7.0842E+00	7.1565E+00	1.2698E+01	1.2813E+01
F_9	5.0403E+00	5.1211E+00	7.2936E+00	7.3165E+00	1.3222E+01	1.3389E+01
F_{10}	5.0739E+00	5.1071E+00	7.2821E+00	7.3506E+00	1.3141E+01	1.3203E+01
F_{11}	5.0331E+00	5.0939E+00	7.4148E+00	7.4639E+00	1.3674E+01	1.3822E+01
F_{12}	5.5422E+00	5.5884E+00	8.2180E+00	8.2814E+00	1.5344E+01	1.5575E+01
F_{13}	4.8857E+00	4.9232E+00	7.1825E+00	7.2440E+00	1.3177E+01	1.3271E+01
F_{14}	4.8873E+00	4.9323E+00	7.2094E+00	7.2259E+00	1.3181E+01	1.3319E+01
F_{15}	4.9080E+00	4.9419E+00	7.2242E+00	7.2787E+00	1.3280E+01	1.3364E+01
F_{16}	4.8915E+00	4.9417E+00	7.2151E+00	7.2788E+00	1.3312E+01	1.3431E+01
F_{17}	4.9431E+00	4.9867E+00	7.3116E+00	7.3608E+00	1.3479E+01	1.3662E+01
F_{18}	4.9002E+00	4.9426E+00	7.1846E+00	7.2471E+00	1.3253E+01	1.3429E+01
F_{19}	5.5179E+00	5.5538E+00	8.3021E+00	8.2849E+00	1.5441E+01	1.5552E+01
F_{20}	4.9320E+00	4.9709E+00	7.2025E+00	7.2574E+00	1.3274E+01	1.3720E+01
F_{21}	4.9338E+00	4.9788E+00	7.2700E+00	7.3446E+00	1.7832E+01	1.8199E+01
F_{22}	5.0013E+00	5.0601E+00	7.3591E+00	7.4214E+00	1.5056E+01	1.5060E+01
F_{23}	5.5472E+00	5.5889E+00	8.4991E+00	8.4581E+00	1.6658E+01	1.6762E+01
F_{24}	5.3198E+00	5.3685E+00	7.9824E+00	8.9096E+00	1.5102E+01	1.5239E+01
F_{25}	5.4424E+00	5.4900E+00	8.1583E+00	8.2227E+00	1.5898E+01	1.6016E+01
F_{26}	8.7852E+00	8.8326E+00	1.4660E+01	1.4710E+01	2.7809E+01	2.7856E+01
F_{27}	8.9832E+00	9.1105E+00	1.4004E+01	1.4053E+01	2.7702E+01	2.7708E+01
F_{28}	5.7872E+00	5.8430E+00	8.9987E+00	8.9808E+00	1.7778E+01	1.7759E+01
F_{29}	5.9846E+00	6.0478E+00	9.2678E+00	9.3317E+00	1.7606E+01	1.7707E+01
F_{30}	5.4322E+00	5.4741E+00	8.3826E+00	8.3705E+00	1.5864E+01	1.5992E+01

Table 5
Comparative experiment between BA-CGSA and other five algorithms.

Function	Index	ABC	PSO	SCA	TSA	CGSA9	BA-CGSA
F_1	Mean	1.7403E+10	5.2790E+08	4.8569E+09	2.3423E+09	5.3130E+08	1.0241E+08
	St.dev	2.7425E+09	1.6344E+08	1.1237E+09	4.0093E+08	9.9254E+07	2.1156E+07
F_2	Mean	5.2050E+11	1.0831E+10	2.3311E+11	5.3276E+10	9.4973E+10	2.1958E+10
	St.dev	4.8308E+10	2.4309E+09	1.5142E+10	5.6705E+09	7.5724E+09	4.9011E+09
F_3	Mean	1.3879E+06	4.3928E+05	4.0007E+05	3.4496E+05	2.3277E+05	2.5393E+05
	St.dev	4.5118E+05	5.6238E+04	3.9111E+04	2.6095E+04	1.7804E+04	1.8438E+04
F_4	Mean	2.5641E+05	2.3810E+03	5.5530E+04	9.1280E+03	1.3713E+04	2.8053E+03
	St.dev	3.1229E+04	4.6383E+02	7.5324E+03	1.1120E+03	2.0452E+03	6.0454E+02
F_5	Mean	5.2144E+02	5.2136E+02	5.2140E+02	5.2137E+02	5.2000E+02	5.2000E+02
	St.dev	2.7942E−02	5.4074E−02	3.1855E−02	2.3788E−02	6.4627E−04	5.1493E−04
F_6	Mean	7.6318E+02	7.1394E+02	7.6037E+02	7.4441E+02	7.3754E+02	7.0638E+02
	St.dev	2.5255E+00	8.9064E+00	4.4573E+00	2.0694E+00	4.0650E+00	5.4196E+00
F_7	Mean	6.2414E+03	7.8324E+02	3.0400E+03	1.1617E+03	1.7594E+03	9.6607E+02
	St.dev	3.7524E+02	1.8727E+01	1.5524E+02	4.5464E+01	7.7811E+01	3.8082E+01
F_8	Mean	2.7573E+03	1.3492E+03	2.1812E+03	1.8837E+03	1.3852E+03	1.3392E+03
	St.dev	9.5698E+01	6.5532E+01	5.5462E+01	4.4901E+01	1.7168E+01	2.7287E+01
F_9	Mean	3.2297E+03	1.8819E+03	2.4087E+03	2.0962E+03	1.6265E+03	1.5397E+03
	St.dev	1.3604E+02	1.3332E+02	6.1323E+01	3.7420E+01	3.7473E+01	3.9110E+01
F_{10}	Mean	3.3688E+04	1.7707E+04	3.1920E+04	2.9600E+04	1.7626E+04	1.6681E+04
	St.dev	5.2128E+02	1.4056E+03	9.4642E+02	1.1552E+03	7.6059E+02	1.0306E+03

local optimal solution, which solves the problem of falling into local optimum. Thus, the algorithm can find the global optimal solution more efficiently.

5.3. The influence of k on the agent motion

Through the comparative experiments, we can find that the convergence of the BA-CGSA algorithm is greatly improved after applying the linear change weight coefficient of k. This is because the linear coefficient k plays a vital role in the balance of the proxy point when it moves from the exploration phase to the exploitation phase.

Table 6
Comparative experiment between BA-CGSA and other five algorithms.

Function	Index	ABC	PSO	SCA	TSA	CGSA9	BA-CGSA
F_{11}	Mean	3.5376E+04	3.1848E+04	3.3436E+04	3.2206E+04	1.5377E+04	1.3901E+04
	St.dev	5.9350E+02	1.1259E+03	5.9900E+02	5.6044E+02	1.0419E+03	1.1436E+03
F_{12}	Mean	1.2051E+03	1.2045E+03	1.2049E+03	1.2046E+03	1.2000E+03	1.2000E+03
	St.dev	2.5794E−01	5.3180E−01	2.8762E−01	2.7738E−01	8.9681E−03	4.9914E−03
F_{13}	Mean	1.3136E+03	1.3007E+03	1.3080E+03	1.3034E+03	1.3053E+03	1.3010E+03
	St.dev	7.6111E−01	1.5515E−01	3.3757E−01	2.6827E−01	2.1333E−01	6.8364E−01
F_{14}	Mean	2.9428E+03	1.4300E+03	2.0560E+03	1.5363E+03	1.6918E+03	1.4702E+03
	St.dev	1.3051E+02	9.7165E+00	5.0560E+01	1.1587E+01	2.1163E+01	1.9141E+01
F_{15}	Mean	5.2634E+08	1.8947E+04	1.1097E+07	6.1557E+05	1.8751E+05	5.4741E+03
	St.dev	1.6465E+08	9.4540E+03	5.7009E+06	1.7223E+05	5.3476E+04	2.3019E+03
F_{16}	Mean	1.6483E+03	1.6470E+03	1.6476E+03	1.6470E+03	1.6459E+03	1.6458E+03
	St.dev	1.7329E−01	5.4856E−01	2.9078E−01	1.8778E−01	3.4590E−01	5.7922E−01
F_{17}	Mean	1.9032E+09	8.5632E+07	7.1323E+08	2.2282E+08	3.5437E+07	5.0678E+06
	St.dev	3.1131E+08	3.4618E+07	1.5508E+08	3.9546E+07	1.2173E+07	1.5950E+06
F_{18}	Mean	1.5521E+09	1.9660E+07	1.4366E+10	3.0777E+03	7.6046E+07	3.5935E+03
	St.dev	2.5296E+08	4.7298E+07	3.0186E+09	5.3076E+02	1.5761E+08	1.0229E+03
F_{19}	Mean	3.9892E+03	2.1413E+03	4.4727E+03	2.1560E+03	2.4573E+03	2.1196E+03
	St.dev	3.6732E+02	3.1038E+01	5.1031E+02	2.8290E+01	6.4427E+01	3.7171E+01
F_{20}	Mean	1.8893E+07	3.1250E+05	9.1164E+05	2.5099E+05	1.3750E+05	1.3888E+05
	St.dev	7.7440E+06	9.9400E+04	5.5546E+05	3.8851E+04	1.2127E+04	1.4942E+04
F_{21}	Mean	8.4195E+08	3.6206E+07	2.8164E+08	9.0861E+07	5.0727E+06	3.0459E+06
	St.dev	1.7734E+08	1.2849E+07	7.1775E+07	2.1960E+07	1.7720E+06	7.1930E+05
F_{22}	Mean	1.3247E+04	6.4944E+03	9.3433E+03	7.0275E+03	5.7704E+03	5.6300E+03
	St.dev	1.4901E+03	9.3896E+02	6.5651E+02	2.3619E+02	7.2039E+02	6.9307E+02
F_{23}	Mean	5.8250E+03	2.7116E+03	4.2680E+03	2.7676E+03	2.5350E+03	2.7470E+03
	St.dev	3.5864E+02	1.9842E+01	3.0836E+02	1.1538E+01	1.2758E+02	1.5616E+02
F_{24}	Mean	4.1634E+03	2.9169E+03	3.1704E+03	2.9902E+03	2.7435E+03	2.7399E+03
	St.dev	1.5691E+02	1.5207E+01	4.3602E+01	1.2939E+01	1.1615E+01	1.1832E+01
F_{25}	Mean	3.6754E+03	2.8567E+03	3.0730E+03	3.0482E+03	2.7071E+03	2.7067E+03
	St.dev	1.6379E+02	2.9711E+01	8.5742E+01	2.8835E+01	9.7614E+00	1.1147E+01
F_{26}	Mean	2.7426E+03	2.8413E+03	3.1035E+03	3.0008E+03	2.8002E+03	2.8003E+03
	St.dev	1.2716E+01	9.8607E+00	2.3215E+02	2.2698E+01	4.2767E−02	4.9494E−02
F_{27}	Mean	7.1218E+03	5.8617E+03	7.4101E+03	6.3860E+03	9.9263E+03	1.0892E+04
	St.dev	5.5832E+01	1.8384E+02	1.4790E+02	8.6634E+01	7.5849E+02	9.5703E+02
F_{28}	Mean	2.4634E+04	1.4264E+04	2.5946E+04	2.0721E+04	1.3358E+04	1.5600E+04
	St.dev	2.4713E+03	1.9108E+03	1.5930E+03	1.6849E+03	1.6718E+03	2.0145E+03
F_{29}	Mean	1.0351E+05	2.5627E+07	1.9776E+09	2.2393E+07	1.3208E+08	1.7939E+05
	St.dev	3.8889E+04	1.0518E+08	3.8429E+08	6.1478E+06	3.5580E+08	6.4652E+05
F_{30}	Mean	1.7096E+05	9.8418E+05	6.4348E+07	2.6842E+06	5.3807E+06	1.9223E+06
	St.dev	4.9455E+04	7.9040E+05	2.0717E+07	7.9446E+05	1.9573E+06	1.9224E+06
Average ranking		3.5333	5.3000	2.5333	5.1333	2.7667	1.7333
Total ranking		4	6	2	5	3	1

Table 7
Results of the multiple-problem Wilcoxon's test and Wilcoxon's rank-sum test for BA-CGSA and other algorithms on 30 test functions.

Algorithm	Better	Equal	Worst	R^+	R^-	p	$\alpha = 0.1$	$\alpha = 0.05$
BACGSA versus ABC	26	0	4	402	63	4.8969E−04	Yes	Yes
BACGSA versus PSO	21	0	9	328	137	4.9498E−02	Yes	Yes
BACGSA versus SCA	29	0	1	450	15	7.6909E−06	Yes	Yes
BACGSA versus TSA	28	0	2	435	30	3.1123E−05	Yes	Yes
BACGSA versus CGSA9	22	2	6	373	92	5.2775E−03	Yes	Yes

Table 8
Results obtained by BA-CGSA and other algorithms for pressure vessel design problem.

Algorithm	TSA	BA-CGSA	SCA	ABC	PSO
Best Cost	7067.7511	**7019.0328**	7323.4213	7203.8915	7205.1078
Mean Cost	7146.0100	**7029.5108**	7534.1766	7221.7900	7236.9200
Worst Cost	7261.5711	**7090.7252**	8050.1037	7265.7685	7309.3154
Std.Dev	51.6371	19.8275	188.0822	15.3953	21.9913
x1 of Best Cost	1.1039	1.1013	1.1028	1.1250	1.1257
x2 of Best Cost	0.6284	0.6000	0.6001	0.6250	0.6257
x3 of Best Cost	56.3499	57.0530	57.0788	58.0728	58.0142
x4 of best Cost	55.2178	50.6656	50.6020	45.1283	45.5515

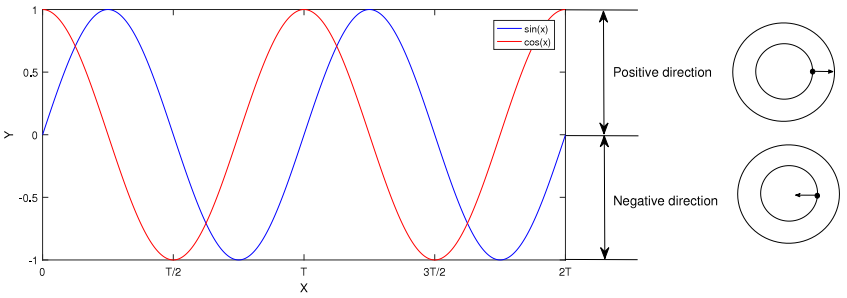


Fig. 4. Sine function affects the direction of agent point motion.

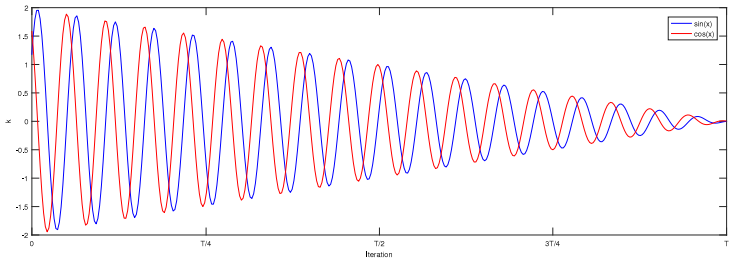


Fig. 5. Sine and cosine functions affected by the weighting factor k.

The value of k is linearly reduced from 2 to 0, which helps the algorithm to complete the conversion of phases. In the initial exploration phase, the value of k is linearly reduced from 2 to 1, amplifying the effects of agents, especially the better ones. This design makes the agent explore the area of the search space as large as possible in the early phase, thus avoiding sticking into the local optimization. In the later exploitation phase, the k value decreases linearly from 1 to 0, gradually slowing down the speed of searching, so that the agents can explore around the positions of good-performed agents obtained in the earlier phase. This ensures that the agent will not jump out of the critical area easily where has obtained good results, increasing the possibility of achieving better global optimal solutions.

5.4. The influence of the sine function on the agent motion

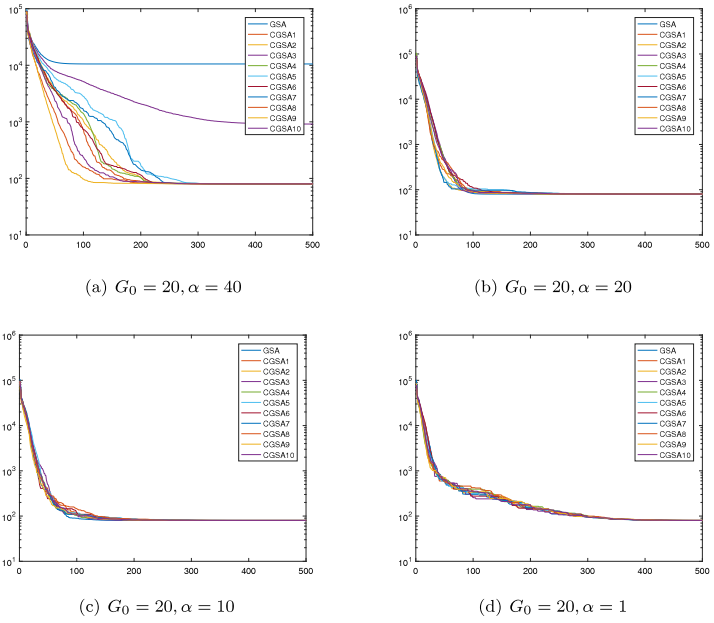
The sine function fluctuates regularly between $[-1, 1]$ so that the next movement direction of the agent is not invariable. When the value of the sine function is positive, the agent point moves to the positive direction of the current point. When it is negative, the agent point moves to the opposite. The mechanism of the motion direction of the agent point affected by the sine function is shown in Fig. 4.

The parameter k and the sine function control the next moving direction and step size of the agent points, forming the balance adjustment mechanism of the algorithm. In the early stage of the experiment, the value of the balance adjustment mechanism fluctuates regularly between $[-2, -1]$ and $[1, 2]$, emphasizing the exploration. In the later phase of the experiment, the value of the balance adjustment mechanism fluctuates regularly between $[-1, 1]$, emphasizing exploitation. The superposition principle of the k factor and the sine function is shown in Fig. 5.

6. Conclusion and future work

CGSA combines chaotic maps with GSA to enhance the perturbation of the algorithm so as to solve the problem of falling into the local optimal solution. But this enhances the instability of the algorithm and brings some disadvantages. Inspired by the sine function and the balance mechanism of SCA, we propose the improved algorithm BA-CGSA based on CGSA. Compared with other algorithms, BA-CGSA has advantages in the following four aspects:

- **Balance:** Applying the linear variable weight coefficient k to balance the two stages of exploration and exploitation. So the searching speed of agents over the whole iterative process can be controlled globally, which realizes the balance adjustment mechanism of the algorithm.
- **Directivity:** The directivity of the sine function makes the searching process of the algorithm more reasonable, improving the convergence of the algorithm.
- **Chaotics:** The chaotic mapping functions are preserved to avoid the algorithm falling into the local optimal solution.
- **Flexibility:** Ten kinds of chaotic maps allow users to choose a more suitable algorithm according to different situations, which makes the algorithm more flexible.



(a) $G_0 = 20, \alpha = 40$ (b) $G_0 = 20, \alpha = 20$

(c) $G_0 = 20, \alpha = 10$ (d) $G_0 = 20, \alpha = 1$

Fig. 6. The influence of the value of α for benchmark function $F1$.

Although we have obtained the outstanding performance of BA-CGSA, there are still some problems that need further research. The gravity constant G is the main global control parameter, which defines the intensity of gravitational forces between the search agents. Therefore the determination of G is crucial to the algorithm. CGSA redefines the calculation formula of G by using the chaotic function, and has achieved good improvement results. But the selection of the constants G_0 and α in the formula still needs to be discussed further. In order to make the differences between CGSA and the proposed algorithm, the same 24 benchmark functions tested in CGSA are evaluated for its fairness. In future work, some more benchmark functions can be applied. The influence of constant α is shown in Fig. 6 by using the benchmark function $F1$ as an example.

References

[1] S. Mirjalili, SCA: a sine cosine algorithm for solving optimization problems, Knowl. Based Syst. 96 (96) (2016) 120–133.
[2] J. Jiang, Y. Feng, J. Zhao, et al., DataABC: A fast ABC based energy-efficient live VM consolidation policy with data-intensive energy evaluation model, Future Gener. Comput. Syst. 74 (2017) 132–141.
[3] J. Jiang, D. Wu, Y. Chen, K. Li, Complex network oriented artificial bee colony algorithm for global bi-objective optimization in three-echelon supply chain, Appl. Soft Comput. 76 (2019) 193–204.
[4] B. Yang, S. Yang, J. Zhang, D. Li, Optimizing random searches on three-dimensional lattices, Physica A 501 (2018) 120–125.
[5] Y. Yang, L. Tu, K. Li, T. Guo, Optimized inter-structure for enhancing the synchronizability of interdependent networks, Physica A 521 (2019) 310–318.
[6] M. Moradi, S. Parsa, An evolutionary method for community detection using a novel local search strategy, Physica A 523 (2019) 457–475.
[7] X.S. Yang, Firefly algorithm, stochastic test functions and design optimisation, Int. J. Bio-inspired Comput. 2 (2) (2010) 78–84.
[8] J.C. Spall, Intrduction to Stochastics Search and Optimization: Estimation, Simulation, and Control, John Wiley & Sons, Inc., 2003.
[9] I. Boussaid, J. Lepagnot, P. Siarry, A survey on optimization metaheuristics, Inform. Sci. 237 (2013) 82–117.
[10] J.A. Parejo, A. Ruizcortés, S. Lozano, P. Fernández, Metaheuristic optimization frameworks: a survey and benchmarking, Soft Comput. 16 (2012) 527–561.
[11] A. Zhou, B. Qu, H. Li, S. Zhao, P.N. Suganthan, Q. Zhang, Multiobjective evolutionary algorithms: a survey of the state of the art, Swarm Evol. Comput. 1 (1) (2011) 32–49.
[12] H. Marouani, Y. Fouad, Particle swarm optimization performance for fitting of levy noise data, Physica A 514 (2019) 708–714.
[13] J. Tang, R. Zhang, Y. Yao, F. Yang, Z. Zhao, R. Hu, Y. Yuan, Identification of top-k influential nodes based on enhanced discrete particle swarm optimization for influence maximization, Physica A 513 (2019) 477–496.
[14] S. Droste, T. Jansen, I. Wegener, Upper and lower bounds for randomized search heuristics in black-box optimization, Theory Comput. Syst. 39 (2006) 525–544.
[15] J.H. Holland, J.S. Reitman, Cognitive systems based on adaptive algorithms, ACM SIGART Bull. 63 (63) (1977) 49.
[16] W. Yong, L. Hanxiong, H. Tingwen, L. Long, Differential evolution based on covariance matrix learning and bimodal distribution parameter setting, Appl. Soft Comput. 18 (2014) 232–247.
[17] Y. Wang, Z. Cai, Q. Zhang, Differential evolution with composite trial vector generation strategies and control parameters, IEEE Trans. Evol. Comput. 15 (2011) 55–66.
[18] Y. Wang, Z. Cai, Q. Zhang, Enhancing the search ability of differential evolution through orthogonal crossover, Inform. Sci. 185 (2012) 153–177.
[19] D. Simon, Biogeography-based optimization, IEEE Trans. Evol. Comput. 12 (2008) 702–713.

[20] K. Deb, An efficient constraint handling method for genetic algorithms, Comput. Methods Appl. Mech. Engrg. 186 (2) (2000) 311–338.
[21] M. Dorigo, L.M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, IEEE Trans. Evol. Comput. 1 (1) (1997) 53–66.
[22] C. Tsai, K. Huang, C. Yang, M. Chiang, A fast particle swarm optimization for clustering, Soft Comput. 19 (2) (2015) 321–338.
[23] D. Karaboga, C. Ozturk, A novel clustering approach: artificial bee colony (ABC) algorithm, Appl. Soft Comput. 11 (1) (2011) 652–657.
[24] X.S. Yang, Firefly algorithm, stochastic test functions and design optimisation, Int. J. Bio-inspired Comput. 2 (2) (2010) 78–84.
[25] E. Rashedi, H. Nezamabadipour, S. Saryazdi, GSA: A gravitational search algorithm, Inform. Sci. 179 (13) (2009) 2232–2248.
[26] S. Mirjalili, A.H. Gandomi, Chaotic gravitational constants for the gravitational search algorithm, Appl. Soft Comput. 53 (2017) 407–419.
[27] A. Kaveh, V. Mahdavi, A hybrid CBO-PSO algorithm for optimal design of truss structures with dynamic constraints, Appl. Soft Comput. 34 (2015) 260–273.
[28] A. Hatamlou, Black hole: a new heuristic optimization approach for data clustering, Inform. Sci. 222 (2013) 175–184.
[29] A.H. Kashan, League Championship Algorithm (LCA): an algorithm for global optimization inspired by sport championships, Appl. Soft Comput. 16 (2014) 171–200.
[30] A. Sadollah, A. Bahreininejad, H. Eskandar, M. Hamdi, Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems, Inform. Sci. 13 (2013) 2592–2612.
[31] R. Rao, V. Savsani, D. Vakharia, Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems, Comput. Aided Design 43 (3) (2011) 303–315.
[32] N. Wang, L. Liu, L. Liu, Genetic Algorithm in Chaos, Or Transactions, 2001.
[33] L. Yang, T. Chen, Application of Chaos in genetic algorithms, Commun. Theor. Phys. 38 (2) (2002) 168–172.
[34] V. Jothiprakash, R. Arunkumar, Optimization of hydropower eeservoir using evolutionary algorithms coupled with Chaos, Water Res. Manag. 27 (7) (2013) 1963–1979.
[35] Z. Guo, B. Cheng, M. Ye, B. Cao, Self-adaptive chaos differential evolution, Int. Conf. Natural Comput. (2006) 972–975.
[36] S. Saremi, S.M. Mirjalili, S. Mirjalili, Chaotic krill herd optimization algorithm, Proc. Technol. 12 (2014) 180–185.
[37] G. Wang, L. Guo, A.H. Gandomi, G. Hao, H. Wang, Chaotic Krill Herd algorithm, Inform. Sci. 274 (274) (2014) 17–34.
[38] H. Peitgen, H. Jurgens, D. Saupes, Chaos and Fractals, Springer-Verlag, 1992.
[39] Y. Li, S. Deng, D. Xiao, A novel Hash algorithm construction based on chaotic neural network, Neural Comput. Appl. 20 (1) (2011) 133–141.
[40] E. Ott, Chaos in Dynamical Systems, Cambridge University Press, 2002.
[41] X.S. Yang, Bat algorithm for multi-objective optimisation, Int. J. Bio-inspired Comput. 3 (5) (2011) 267–274.
[42] A. Abran, P.N. Robillard, Function points analysis: an empirical study of its measurement processes, IEEE Trans. Softw. Eng. 22 (12) (1996) 895–910.
[43] M.S. Kiran, TSA: Tree-Seed Algorithm for continuous optimization, Expert Syst. Appl. 42 (19) (2015) 6686–6698.
[44] N.K. Singh, S. Singh, A novel hybrid GWO-SCA approach for optimization problems, Eng. Sci. Technol. Int. J. 20 (6) (2017) 1586–1601.
[45] M.A. Elaziz, D. Oliva, S. Xiong, An improved opposition-based sine cosine algorithm for global optimization, Expert Syst. Appl. 90 (2017) 484–500.
[46] J. Wang, W. Yang, P. Du, T. Niu, A novel hybrid forecasting system of wind speed based on a newly developed multi-objective sine cosine algorithm, Energy Convers. Manage. 163 (2018) 134–150.
[47] A.I. Hafez, H.M. Zawbaa, E. Emary, A.E. Hassanien, Sine Cosine Optimization Algorithm for Feature Selection, 2016, pp. 1–5.
[48] R. Sindhu, R. Ngadiran, Y.M. Yacob, N.A.H. Zahri, M. Hariharan, Sine–Cosine algorithm for feature selection with elitism strategy and new updating mechanism, Neural Comput. Appl. 28 (10) (2017) 2947–2958.
[49] A. Attia, R.A. Sehiemy, H.M. Hasanien, Optimal power flow solution in power systems using a novel Sine-Cosine algorithm, Int. J. Electr. Power Energy Syst. 99 (2018) 331–343.
[50] W. Long, T. Wu, X. Liang, S. Xu, Solving high-dimensional global optimization problems using an improved sine cosine algorithm, Expert Syst. Appl. (2019) 108–126.
[51] G.C. Onwubolu, B.V. Babu, New Optimization Techniques in Engineering, Springer Berlin Heidelberg, 2004, pp. 150–153.