Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105 (Online)
Volume **6**, Number **3**, August **2021**

Adaptive Particle Swarm Optimization Using Scale-Free Network Topology

Wei Li

School of Information Engineering
JiangXi University of Science and Technology
Ganzhou, China
liwei@jxust.edu.cn

Bo Sun

School of Information Engineering
JiangXi University of Science and Technology
Ganzhou, China
sunbo@mail.jxust.edu.cn

Ying Huang

School of Mathematical and Computer Science
Gannan Normal University
Ganzhou, China
nhwshy@whu.edu.cn

Soroosh Mahmoodi

Soroosh Khorshid Iranian Co
Abyek Industrial Zone
Qazvin Province, Iran
soroosh.mahmoodi@yahoo.com

Received May 2021; revised June 2021

Abstract. *Particle swarm optimization (PSO) algorithm has some advantages, such as fewer control parameters, and faster convergence speed. It shows a considerable optimization performance for solving numerical optimization and various application problems in reality. However, the PSO algorithm usually possesses some defects such as low convergence accuracy and easy to fall into local optimum while resolving some complex issues. An adaptive particle swarm optimization algorithm using scale-free network topology is proposed. Based on the characteristics of scale-free network topology with power-law distribution, this novel algorithm can construct a corresponding neighborhood for each particle. Then, it selects the elite particles from the community be participated in the particle evolution process and consider full play to the guiding role of elite particles within the population search process. In addition, a new adaptive weight strategy and an introduction to the differential evolution operation for achieving a balance ability to the global and local exploration within the search process are proposed. To verify the performance of the proposed algorithm experimentally, eighteen benchmark functions are employed. Experimental results show that the proposed adaptive particle swarm optimization algorithm using scale-free network topology has efficient robustness which competitive solutions can be obtained.*
Keywords: Particle swarm optimization; Scale-free network topology; Inertia weight; Self-adaption; Neighbor.

1. **Introduction.** With the exploitation of industrial manufacturing and computational intelligence, many problems in real life have become a promising research issue, and the traditional accurate algorithm is more and more challenging to meet the needs of optimization performance. Swarm intelligence optimization algorithm, has received more and more research because of its high efficiency and stability in solving complex problems. For example, particle swarm optimization algorithm, ant colony algorithm, firefly algorithm, artificial bee colony algorithm and so on. Where particle swarm optimization algorithm is a population intelligent optimization algorithm proposed by Kennedy and Eberhart in 1995, which is based on the research of birds foraging behavior [1]. Since the algorithm was proposed, it has been favored by many scholars because of its unique group search behavior and efficient and stable optimization performance. Particle swarm optimization algorithm has been widely used in humanities, engineering, chemistry, medicine, high-end physics, and other scientific fields, such as nonlinear constrained discrete variable optimization problem in the field of engineering design [2], complex scheduling optimization problems such as high-dimensional, nonlinear, multi constraint in the area of power system [3], and robot trajectory planning in the area of robot intelligent control questions [4, 5].

The research shows that the performance of PSO depends on the setting of control parameters, such as population size, inertia weight, cognitive coefficient, and so on. To get better weight parameters, Shi and Eberhart [6] proposed to use fixed inertia weight to balance the global exploration and local exploitation ability of the population. However, this strategy often makes the algorithm have poor local exploitation ability. Therefore, Shi and Eberhart [7] proposed a strategy to make the inertia weight change linearly in each generation. The experimental results show that the weight decreases linearly from the initial value of 0.9 to 0.4, the optimization performance of the algorithm is greatly improved. Liu et al. [8] proposed an inertia weight adaptive strategy based on the relationship between the fitness and average fitness of particles. If the fitness of particles is lower than the average fitness, it reduces the effect of the previous velocity and makes them tend to exploitation. On the contrary, it increases the effect of the previous velocity and makes them tend to explore. Tanweer et al. [9] proposed a self-tuning particle swarm optimization algorithm. By controlling the inertia weight of each particle in the evolution process, the inertia weight of the optimal particle increases and the inertia weight of other particles decreases, To improve the local exploitation ability of the optimal particle, and at the same time, make other particles have higher local-global exploration ability. Ratnoweera et al. [10] proposed a particle swarm optimization algorithm with time-varying acceleration coefficient, i.e., in the process of evolution, the individual cognitive and social cognitive coefficients are linear, individual cognitive coefficient decreases, social cognitive coefficient increases, and then realizes the transition from global exploration to local exploitation. Adam et al. [11] conducted experiments on eight particle swarm optimization algorithms under different population sizes and dimensions, and analyzed the impact of population size on algorithm performance in the face of various problems. Xu [12] introduced the average absolute value of all particle velocities, and proposed a strategy of dynamically adjusting inertia weight through feedback control. Zhou and Shi [13] proposed a method to adaptively adjust inertia weight based on particle velocity information, and used velocity information to replace position information based on APSO [14]. Agrawal and Tripathi [15] adopted an adaptive inertia weight strategy based on a binomial probability distribution for global optimization, which improved the accuracy and convergence speed of the particle swarm optimization algorithm. Kang et al. [16] proposed a non-inertial particle swarm optimization with elite mutation-Gaussian process regression (NIPSO-GPR) to optimize the hyper-parameters of GPR. NIPSO-GPR

can adaptively obtain hyper-parameters of GPR via uniform non-inertial velocity update formula and adaptive elite mutation strategy. Wu et al. [17] proposed ant colony system to data mining algorithm takes the multi-threshold constraint to secure and sanitize patents' records in different lengths, which is applicable in a real medical situation.

In order to solve the problems of premature convergence and easy to fall into the local optimum of particle swarm optimization algorithm, researchers have done lots of improvement work. Li et al. [18] proposed a multi-population collaborative particle swarm optimization algorithm based on dynamic piecewise mean learning strategy and multi-dimensional comprehensive learning strategy, which adopted multi-dimensional comprehensive learning strategy to accelerate the convergence rate of solution, and introduced a differential mutation operator to increase the diversity of particle swarm. Meng et al. [19] based on the improved particle swarm optimization algorithm and cross search optimization algorithm, updated the sequence of particle swarm optimization, enhanced the global convergence ability of population through horizontal crossover, and enhanced the diversity of particle swarm optimization through the vertical crossover. Wang et al. [20] proposed an incentive evolutionary game model for stimulating cooperation among nodes. By constructing our game model in routers, normal nodes and abnormal nodes are encouraged to participate in network collaboration by self-evolution of gaming. Then after evolution, the entire network can reach a general cooperative state of nodes. Wu et al. [21] proposed a new concept of minimal support for solving this issue. In compliance with a given threshold function, the proposed approach would set a tighter threshold for an object containing several items. Yang et al. [22] proposed a multigroup multistrategy SCA algorithm. The algorithm executes multiple populations in parallel, and each population executes a different optimization strategy. Information is exchanged among populations through intergenerational communication. Zhang et al. [23] proposed a short-term traffic flow prediction algorithm of quantum genetic algorithm learning vector quantization (QGA-LVQ) neural network to forecast the changes of traffic flow. Utilizing the global optimization ability of quantum genetic algorithm (QGA), it is combined with LVQ neural network to overcome some shortcomings of LVQ neural network, including sensitive to initial weights and prone to local minima. Kang et al. [24] proposed a solution, which utilizes data mining technique with clustering concept, by gathering the current feedback data that have from our SPN (Smart Protection Network). It can form these data in groups with similarity, and by deploying these data to client side, to achieve the reduction of traffic usage.

Although the researchers speed up the convergence of the algorithm by parameter adaptation and population diversity, there are still problems that the experience of particle swarm in the search process is not fully utilized. In order to consider full play to the experience learned by elite particles in the population, and better balance the exploration and exploitation ability of particle swarm optimization in the search process, a novel adaptive particle swarm optimization algorithm using scale-free network topology (SFAPSO) is proposed. Where the scale-free network topology is used to construct the population neighborhood, the velocity differential strategy is used to update the particle velocity, and a new inertia weight adaptive is used to balance the global exploration and local exploitation ability. Experimental results show that the proposed algorithm can make full use of the experience of elite particles in evolutionary process, and effectively improve the search ability and convergence accuracy of the algorithm. The main novel aspects of this paper are summarized as follows:

1. An elite particle selection mechanism is designed. Based on the power-law distribution of scale-free network topology, the corresponding community is constructed for each

particle, and elite particles are selected from the community to participate in the particle search process.

2. An update strategy with velocity differential is proposed. Two particles are selected from the neighborhood for differential operation, and then introduced into the velocity update formula and consider full play to the guiding role of elite particles in the population learning process.

3. A new adaptive strategy of inertia weight is introduced for achieving a balance ability to the global exploration and local exploitation within the search process.

The rest of this paper is organized as follows: The standard particle swarm optimization algorithm is described in Section 2, as well as scale-free network topology related knowledge. The details of our proposed SFAPSO algorithm are shown in Section 3, including scale-free network topology construction, velocity differential update, and the improvement of inertia weight adaptive. The experimental results and analysis on several benchmark functions are described in Section 4, and the conclusion is provided in Section 5.

2. **Related Works.**

2.1. **Particle swarm optimization.**

Particle swarm optimization (PSO) is a stochastic optimization algorithm based on population search. In this algorithm, each particle is composed of three parts, including particle velocity vector X_{ij}=(V_{i1},V_{i2},...,V_{iD}), position vector X_{ij}= (X_{i1},X_{i2},...,X_{iD}), and the historical optimal position X_{pbest}=(X_{pbest1}, X_{pbest2},...,X_{pbestD}). In addition, in the iterative process of the algorithm, each particle also stores the global optimal position of any particle in its neighborhood X_{gbest}=(X_{gbest1}, X_{gbest2},...,X_{gbestD}). The update equation of velocity and position of particle i at t + 1 is as follows:

$$V_{ij}^{t+1} = V_{ij}^{t} + c_1 \times r_1(pbest_{ij}^{t} - X_{ij}^{t}) + c_2 \times r_2(gbest - X_{ij}^{t}) \quad i = 1, 2...N, j = 1, 2...D \tag{1}$$

$$X_{ij}^{t+1} = X_{ij}^{t} + V_{ij}^{t+1} \quad i = 1, 2...N, j = 1, 2...D \tag{2}$$

Where, i denotes the index of the particle, j denotes the dimension of the particle, t represents the current iteration number of the algorithm, c_1 and c_2 are acceleration coefficients, r_1 and r_2 are random numbers with uniform distribution between [0, 1]. In order to control the influence of the previous velocity of particles on the current velocity, an inertia weight ω is usually introduced to the velocity in equation (1), as shown in equation (3):

$$V_{ij}^{t+1} = \omega \times V_{ij}^{t} + c_1 \times r_1(pbest_{ij}^{t} - X_{ij}^{t}) + c_2 \times r_2(gbest - X_{ij}^{t}) \tag{3}$$

Where, ω as an important parameter of particle swarm optimization algorithm, ω can control the exploitation and exploration ability of the algorithm. If the inertia weight is large, the global exploration ability of the algorithm is strong, and the particle is easy to find the global optimal solution. If the inertia weight is small, the local search ability of the algorithm is strong, and the particles are easy to converge.

The basic flow of particle swarm optimization algorithm is shown in FIGURE 1. Firstly, a population of particles is initialized randomly, including the position and velocity of the particles. Secondly, the fitness value of the position of each particle is evaluated, and the historical optimal position of the particle itself and the global optimal position of the population is obtained by comparison. In the iterative process of the algorithm, the velocity and position of the particle are updated according to the current velocity and position of the particle and the historical optimal position of the particle and its neighborhood to the current position until the termination condition is satisfied.

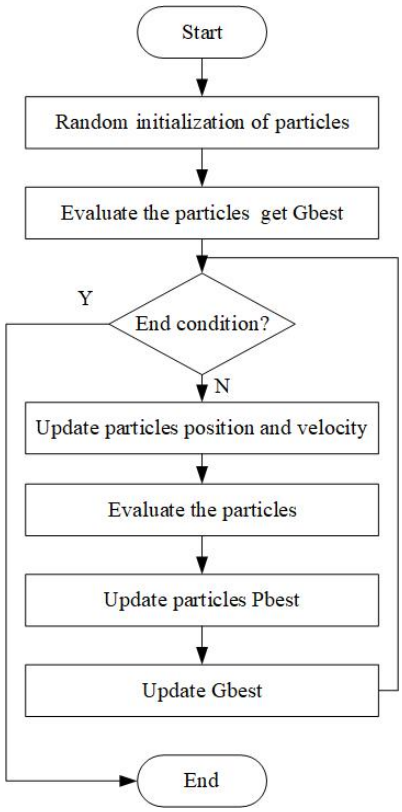

FIGURE 1. Particle swarm optimization algorithm

Although particle swarm optimization algorithm has been developed a lot, there are still some problems to be solved. For example, make better use of the two kinds of particle experience: The previous optimal particle position and the global optimal particle position. Therefore, in this paper, velocity differential strategy and inertia weight adaptive strategy based on scale-free network topology are proposed, which will be described in detail in Section 3.

2.2. **Scale-Free network topology.** Scale-free network topology is very common in daily life, such as server and client, Internet of things [25], social network [26], aviation network [27], etc. The most basic property of scale-free network topology model is that a node with degree k obeys the power-law distribution, and the network obeys the power-law distribution, i.e., the probability density of nodes in the network obeys the power function distribution, which is usually expressed by the following equation:

$$P(k) = k^{-\lambda} \tag{4}$$

Where, $P(k)$ denotes the probability distribution of node degree, k denotes the number of connections, and power exponent λ represents a parameter of network structure.

The biggest advantage of particle swarm optimization algorithm is that it can share information among particles. As can be seen from FIGURE 2, the elite particles are in the hub position and have a greater influence on the information interaction within the population. In comparison the ordinary particles far away from the elite particles have less influence on the population. FIGURE 2 describes the characteristics of scale-free network topology: A few nodes (blue nodes) have more degrees, which are called elite nodes, while most nodes (white nodes) have fewer degrees, which are called ordinary nodes. FIGURE 3 is the function image of equation (4), which can be seen intuitively λ with the increase of k value, $P(k)$ decreases.

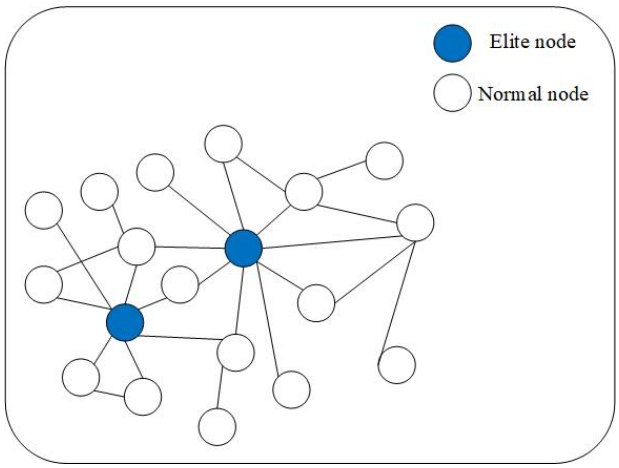

FIGURE 2. Scale-free network topology

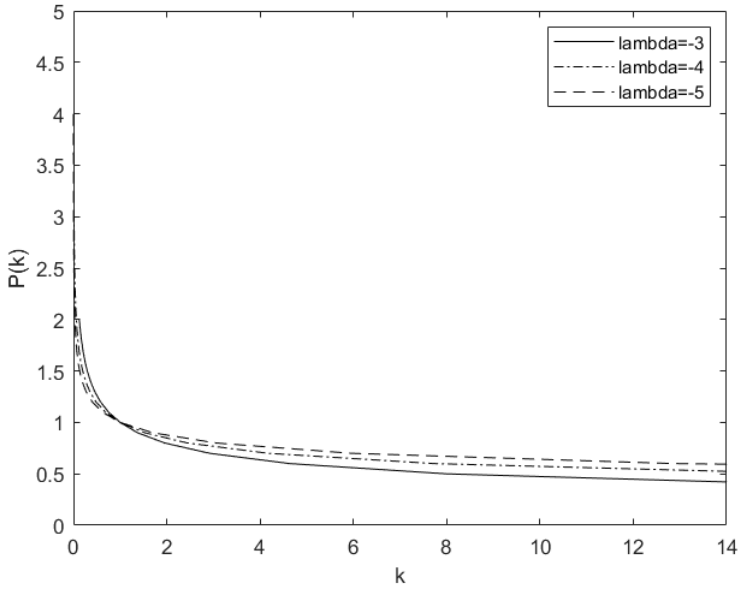

FIGURE 3. Power-law distribution function

In 1999, Barabási et al. proposed a scale-free network topology model based on a preferential attachment mechanism, namely Barabási-Albert (BA) model [18]. The model usually includes the following two steps: Firstly, select several elite particles to form a small network. Secondly, the remaining nodes are connected to the network through the cycle, and the probability of access to the nodes with several heights is high. As can be seen from FIGURE 4, the node with degree 10 is more attractive to the node i to be accessed. This paper takes BA model as an example to describe the construction process of scale-free network topology.

The neighborhood based information interaction constructed by scale-free network topology can reduce the impact of ordinary particle interaction on the whole population, thus speeding up the convergence speed of the population. Therefore, it is very advantageous to use scale-free network topology to optimize the whole search process of PSO.

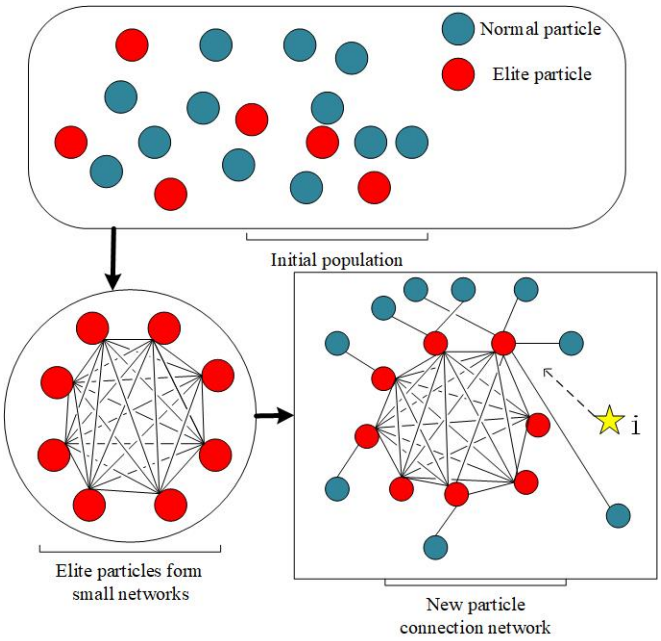

FIGURE 4. Scale-free network topology construction process

3. The proposed SFAPSO algorithm.

3.1. Update strategy with velocity differential.

Based on the analysis of scale-free network topology, a velocity differential strategy using scale-free network topology and differential evolution is proposed. Each particle selects elite particles from the constructed topological neighborhood be participated in its own velocity update, and consider full play to the information interaction ability of elite particles. The details of velocity differential strategy is as follows:

Firstly, the scale-free network topology is constructed using the BA model. The Euclidean distance between each particle in the population and the global optimal particle is calculated. The nearest 10% particles of the population are selected as elite particles. The initial network is constructed by full connection. The degree proportion of each particle is calculated by equation (5). In the iterative process, a roulette selection algorithm is used to connect the remaining particles into the network. The degree proportion and cumulative probability equation of particles are as follows:

$$P_i = \frac{degree_i}{\sum_{j=1}^{n} degree_j} \tag{5}$$

$$W_i = \sum_{j=1}^{i} P_j \tag{6}$$

Where, P_i denotes the proportion of node i degree to all degrees in the network, degree i is the degree of node i, and W_i represents the probability of accumulation.

After the scale-free network topology is constructed, each particle selects the elite particle from its neighborhood as its current optimal position. According to the fitness value, two particles are selected from the neighborhood by the roulette selection algorithm to participate in the velocity update. Particle fitness ratio and cumulative probability can

be described as follows:

$$Q_i = \frac{f(x_i)}{\sum_{j=1}^{n} f(x_j)} \tag{7}$$

$$S_i = \sum_{j=1}^{i} Q_j \tag{8}$$

In this paper, we denote an equilibrium factor g to equalize the differential velocity update and the classical velocity update. After determining the optimal position of each particle, the equilibrium factor g is used to determine how to update the velocity. The value of equilibrium factor g will be described in detail in the experimental part, and the velocity update equation can be described as follows:

$$V_{ij}^{t+1} = \omega_i \times V_{ij}^{t} + c_1 \times r_1(Fpbest_{ij}^{t} - X_{ij}^{t}) + c_2 \times r_2(gbest - X_{ij}^{t}) + \alpha \times (Fpbest_{R1j}^{t} - Fpbest_{R2j}^{t}) \tag{9}$$

Where, $Fpbest_{ij}^{t}$ denotes the elite particles selected from the domain, α is the difference coefficient, $R1$ and $R2$ represent two particles randomly selected from the community. Details of the velocity differential update strategy using scale-free network topology are as follows:

Strategy1: Update strategy with velocity differential

```
Initialization: Population size N, network initial node number m0, equilibrium factor g;
Calculate and sort the distance between particles and global optimal particles;
Select the first m0 elite particles as the initial nodes to form the network;
Calculate the degree of each node in the network: Degree = m0-1;
For i=m0+1:N
   Use roulette selection algorithm to connect the particle into the network;
   Add 1 to the degree of particle i and connected particle;
End
For i=1:N
   Find out the index indexFP corresponding to the node connected to each node;
   Use roulette selection algorithm selects R1 and R2;
End
For i=1:N
   Select the particle with the smallest fitness value in the neighborhood as the current optimal position;
End
For i=1:N
   if rand>g
      Execute the improved velocity update equation;
   else
      Execute the classical velocity update equation;
   End
End
```

3.2. **Adaptive strategy of inertia weight.** In this paper, we use the community based method to select the elite individuals as the current optimal position in the velocity differential strategy, i.e., in search process, the particle with larger degree is more likely to be affected by the elite, and the particle with smaller degree is less likely to be affected by the elite. Based on the above analysis, we conclude that controlling the inertia weight of each particle is more suitable than the strategy of linearly decreasing the inertia weight, which is improved based on the inertia weight adaptive proposed by Liu et al. [4].

Liu et al. [4] introduced the average fitness and minimum fitness of particles to achieve the "protection" effect on the optimal particles. The inertia weight calculation equation is as follows:

$$\omega_i(t) = \begin{cases} \omega_{\min} + \frac{(\omega_{\max} - \omega_{\min})(f_i(t) - f_{\min}(t))}{f_{avg}(t) - f_{\min}(t)} & if \ \ f_i(t) \le f_{avg}(t) \\ \omega_{\max} & if \ \ f_i(t) > f_{avg}(t) \end{cases} \tag{10}$$

Where, $\omega_t(t)$ denotes the inertia weight of the ith particle of the t generation, ω_{min} and ω_{max} denote the minimum and maximum inertia weight, $f_i(t)$ denotes the fitness value of the ith particle of the t generation, $f_{avg}(t)$ represents the average fitness value of the particle, and $f_{min}(t)$ is the global optimal fitness value of the particle. In order to make full use of the experience of elite particles in the community, this paper improves equation (10). The fitness value of particles is enhanced to the optimal fitness value after neighborhood selection, and the updated inertia weight calculation equation is expressed as follows:

$$\omega_i(t) = \begin{cases} \omega_{\min} + \frac{(\omega_{\max}-\omega_{\min})(f_{pbest}(t)-f_{\min}(t))}{f_{avg}(t)-f_{\min}(t)} & if\ \ f_{pbest}(t) \le f_{avg}(t) \\ \omega_{\max} & if\ \ f_{pbest}(t) > f_{avg}(t) \end{cases} \tag{11}$$

Strategy 2: Adaptive strategy of inertia weight

```
Input: ω_max=1.2, ω_min=0.2, f_avg, f_pbest, f_gbest
Output: ω
For i=1:N
    if f_pbest <f_avg do
        ω_i=ω_min-(ω_max-ω_min)*(f_pbest-f_gbest)/f_avg-f_gbest;
    else
        ω_i=ω_max;
    End
End
```

3.3. **SFAPSO Algorithm.** Based on the above design of velocity differential strategy and inertia weight adaptive strategy, a novel adaptive particle swarm optimization algorithm using scale-free network topology.

Algorithm 1: Adaptive particle swarm optimization using scale-free network topology

```
Initialization: Population size: N,dimension: D, α=0.5, g=0.6;
Evaluate particle fitness f(x_i), f_gbest;
While FEs < maxFEs
    Use Strategy 2 to give each particle inertia weight;
    Use Strategy 1 to build scale-free network topology and update velocity and location;
    Evaluate particle fitness;
    Update global optimal particles;
End
```

The process of the algorithm is described as follows:

Step 1: Initialization population and parameters: N=100, D=30, c_1=c_2=2.0, α=0.5, g=0.6.

Step 2: Calculated the fitness value $f(x_i)$ and the optimal value f_{gbest} of each particle.

Step 3: Use Eq. (11) to give each particle inertia weight.

Step 4: Constructed the scale-free network topology and determine the optimal position of each particle.

Step 5: When $g > 0.6$, use Eq. (9) to update the particle velocity, and use Eq. (2) to update the position.

Step 6: Calculated the fitness and global optimal fitness of particles.

Step 7: Loop steps 2 through 7 until the number of evaluations equals the maximum number of evaluations.

4. **Experimental results.** The paper is carried out on a computer with Win_64 bit, Intel (R) core (TM) i5-6300hq CPU @ 2.30ghz and 8GB ram. Using Matlab r2019a to carry out the relevant experimental work, and the results are discussed and analyzed. The experiment is divided into three parts. In the first part, the value of equilibrium factor g is analyzed. In the second part, the optimal combination of experimental parameters and SFAPSO algorithm is explored. In the third part, the SFAPSO algorithm is compared with the other five well-known particle swarm optimization algorithms. In this experiment, a set of eighteen benchmark functions are used, and all functions information can be obtained from the website [29]. Where, $f1$ - $f6$ are unimodal test functions, and $f7$ - $f18$ are multimodal test functions. To improve the efficiency of the experiment, experiment one and experiment two selected eight benchmark functions (four unimodal test functions and four multimodal test functions) from the eighteen benchmark functions.The details of the benchmark functions are shown in TABLE 1.

4.1. **Value of equilibrium factor.** In order to judge the influence of the value of equilibrium factor on the experiment and determine its optimal value, the experimental parameters in this section are $N = 100$, $D = 30$, and the maximum number of evaluations are $D * 10000$. Each value of equilibrium factor g is calculated 30 times separately, and the mean value and standard deviation are recorded. Considering that the equilibrium factor is for equalization velocity differential update and classical velocity update, this section takes 0.5 as the center value, and g is 0.3, 0.4, 0.5, 0.6 and 0.7. TABLE 2 clearly shows the value of g when the experimental results are good.

Table 1. Benchmark functions

	Test Functions	Search Range
$f1$	Brown function	[-1, 4]
$f2$	Exponential function	[-1, 1]
$f3$	Powell Sum function	[-1, 1]
$f4$	Zakharov function	[-5, 10]
$f5$	Step function	[-100, 100]
$f6$	Rotated hyper-ellipsoid function	[-100, 100]
$f7$	Happy Cat function	[-2, 2]
$f8$	Periodic function	[-10, 10]
$f9$	Quaritic function	[-1.28, 1.28]
$f10$	Xin-She Yang N.2 function	[-2π, 2π]
$f11$	Six Hump Camel bsck function	[-5, 5]
$f12$	Hartmann function	[0, 1]
$f13$	Norwegian function	[-1.1, 1.1]
$f14$	Branin function	[-5, 15]
$f15$	Easom function	[-100, 100]
$f16$	Goldstein and Price function	[-2, 2]
$f17$	Shubert function function	[-10, 10]
$f18$	Michalewicz function	[0, π]

From the ranking of the experimental results in TABLE 2, it can be seen that each value can perform well in $f1$ - $f3$, the performance of g value 0.5 in $f8$ function is better than 0.6, and the performance of g value 0.6 in $f4$ and $f5$ function is better than 0.5. From the mean ranking, the experimental results of g values 0.5 and 0.6 are similar. Still, the standard deviation ranking of 0.6 is better than 0.5, which indicates that the

TABLE 2. Equilibrium factor

	0.3		0.4		0.5		0.6		0.7	
	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
$f1$	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
Rank	1	1	1	1	1	1	1	1	1	1
$f2$	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
Rank	1	1	1	1	1	1	1	1	1	1
$f3$	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
Rank	1	1	1	1	1	1	1	1	1	1
$f4$	3.69E-03	5.89E-03	2.07E-03	2.29E-03	1.73E-03	1.78E-03	**1.68E-03**	1.95E-03	1.27E-02	1.71E-02
Rank	4	4	3	3	2	1	1	2	5	5
$f5$	1.96E-02	2.29E-02	1.74E-02	1.68E-02	1.29E-02	2.07E-02	**1.26E-02**	**1.44E-02**	2.35E-02	1.88E-02
Rank	4	5	3	2	2	4	1	1	5	3
$f6$	**1.00E+00**	1.79E-03	**1.00E+00**	1.37E-03	**1.00E+00**	**3.87E-04**	**1.00E+00**	7.52E-04	**1.00E+00**	1.72E-03
Rank	1	5	1	4	1	1	1	2	1	3
$f7$	5.64E-04	4.20E-04	**4.92E-04**	**3.35E-04**	7.48E-04	5.09E-04	5.95E-04	4.09E-04	1.80E-03	1.60E-03
Rank	2	3	1	1	4	4	3	2	5	5
$f8$	4.51E-12	1.06E-12	4.54E-12	8.30E-13	**4.48E-12**	1.02E-12	4.61E-12	1.08E-12	4.79E-12	1.46E-12
Rank	2	2	3	1	1	3	4	4	5	5
Average rank	2	2.75	1.75	1.75	1.63	2	1.62	1.75	3	3
Final rank	4		2		3		1		5	

evolution process of particle swarm is more stable when the g value is 0.6. On the whole, the experimental effect is the best when the equalization factor g is 0.6, so the equilibrium factor g is 0.6 in SFAPSO algorithm.

4.2. **Optimal strategy combination.** In this paper, eight benchmark functions are used to test five different combinations of the SFAPSO algorithm under the condition that the initial population size is 100 and the problem dimension is 30. The combination of the SFAPSO algorithm is based on the construction of scale-free network topology. Combination one uses classical velocity update equation, combination two uses velocity differential update strategy, combination three uses inertia weight adaptive and classical velocity update equation, and combination four uses inertia weight adaptive and velocity differential update strategy, combination five is a velocity differential update strategy with adaptive inertia weight and introducing equilibrium factor g. TABLE 3 shows the selected benchmark function and value range, and TABLE 4 shows the parameter details of each combination.

TABLE 3. Benchmark functions

	Test Functions	Search Range
$f1$	Brown function	[-1, 4]
$f2$	Exponential function	[-1, 1]
$f3$	Powell Sum function	[-1, 1]
$f4$	Zakharov function	[-5, 10]
$f5$	Happy Cat function	[-2, 2]
$f6$	Periodic function	[-10, 10]
$f7$	Quaritic function	[-1.28, 1.28]
$f8$	Xin-She Yang N.2 function	[-2π, 2π]

TABLE 5 shows the mean and standard deviation of each combination running 30 times independently. TABLE 5 shows that combination three, combination four and combination five can get the optimal value on $f1$ - $f3$. Combination one can get the

TABLE 4. Different combination parameter settings

	Parameter Settings
1	ω=0.85, c_1=c_2=2.0
2	ω=0.85, c_1=c_2=2.0, α=0.5
3	ω_{max}=1.2, ω_{min}=0.2, c_1=c_2=2.0
4	ω_{max}=1.2, ω_{min}=0.2, c_1=c_2=2.0, α=0.5
5	ω_{max}=1.2, ω_{min}=0.2, c_1=c_2=2.0, α=0.5, g=0.6

TABLE 5. Strategy mix comparison

	1		2		3		4		5	
	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
$f1$	6.50E+00	4.60E+00	8.90E+00	1.19E+01	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**0.00E+00**
$f2$	1.31E-02	7.18E-02	3.70E-05	9.04E-05	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**0.00E+00**
$f3$	1.09E-09	3.96E-09	5.85E-10	1.76E-09	0.00E+00	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**	**0.00E+00**
$f4$	1.40E+01	2.51E+01	7.18E+00	1.57E+01	5.98E-02	2.05E-02	6.40E-03	7.81E-03	**1.68E-03**	**1.95E-03**
$f5$	1.40E-01	9.31E-02	1.22E-01	6.38E-02	6.93E-02	1.84E-02	2.31E-02	**1.26E-02**	**1.26E-02**	1.44E-02
$f6$	1.07E+00	1.49E-01	1.22E+00	3.19E-01	1.01E+00	2.04E-02	1.00E+00	**3.28E-04**	**1.00E+00**	7.52E-04
$f7$	5.56E-01	2.94E+00	4.81E-01	1.42E+00	1.02E-03	8.85E-04	2.33E-03	1.77E-03	**5.95E-04**	**4.09E-04**
$f8$	**4.45E-12**	**9.10E-13**	4.95E-12	2.75E-12	5.30E-12	4.91E-12	5.06E-12	1.35E-12	4.61E-12	1.08E-12

better results than others on $f8$. the experimental results of combination five on $f4$ - $f7$ are better than those of other combinations. Through the overall experimental results, combination five has a significant effect in the experiment, i.e., the SFAPSO algorithm proposed in this paper.

4.3. **Comparison of algorithm results.** To illustrate the effectiveness of the SFAPSO algorithm, a set of eighteen benchmark functions are tested in 30 and 50-dimensions respectively, and compared with five well-known PSO variants. Where includes SPSO [30], CLPSO [31], BLPSO [32], SLPSO [33], ACPSO [34]. To ensure the fairness of algorithm comparison, the population size N of each algorithm is set to 100, D is 30 and 50-dimensions respectively, and the maximum number of evaluations are D * 10000. The rest parameters are set according to the corresponding paper. To avoid the influence of error, each algorithm runs 30 times independently, and records the optimal value, average value, and standard deviation. TABLE 6 clearly describes the parameter settings of each algorithm, in which, in the SFAPSO algorithm, ω_{max} is 1.2, ω_{min} is 0.2, c_1 and c_2 are 2.0, The value of α is 0.5 and g is 0.6.

TABLE 6. Comparison algorithm parameter settings

Algorithm	Parameter Settings
SPSO	ω=0.85, c_1=c_2=2.05
CLPSO	ω=0.9-0.2, c=1.49445, $gapm$=5, V_{max}=0.2*$range$
BLPSO	ω=0.9-0.2, c=1.49445, I=E=1, V_{max}=0.2*$range$
SLPSO	M=100, m=M+floor(d/10), c_3=d/M*0.01
ACPSO	ω=0.9-0.4, c_1=c_2=1.49445, $swarmNum$=3, $alpha$=0.1, $beta$=0.1, V_{max}=0.2*$range$
SFAPSO	ω_{max}=1.2, ω_{min}=0.2, c_1=c_2=2.0, α=0.5, g=0.6

From TABLE 7, i.e., 30-dimensional data, it can be concluded that SFAPSO has good exploitation ability in $f1$, $f2$, and $f3$ functions, SLPSO, ACPSO, and SFAPSO can converge to the optimal values in $f5$, $f11$, and $f17$ functions, and ACPSO has better

TABLE 7. Results of comparison algorithms on benchmark functions (30-D)

		SPSO	BLPSO	CLPSO	SLPSO	ACPSO	SFAPSO
	Optimal	4.85E+01	8.01E-64	4.77E+01	2.79E-71	3.37E-06	**0.00E+00**
$f1$	Mean	1.09E+02	1.02E-62	1.24E+02	1.21E-69	8.43E-06	**0.00E+00**
	Std.	2.43E+01	3.13E-62	4.59E+01	1.67E-69	4.46E-06	**0.00E+00**
	Rank	5	3	6	2	4	1
	Optimal	7.71E-01	0.00E+00	3.77E-01	0.00E+00	1.02E-07	**0.00E+00**
$f2$	Mean	8.24E-01	0.00E+00	5.27E-01	7.77E-17	3.76E-07	**0.00E+00**
	Std.	2.31E-02	0.00E+00	4.48E-02	5.17E-17	2.05E-07	**0.00E+00**
	Rank	6	1	5	3	4	1
	Optimal	8.50E-03	2.20E-117	2.41E-04	1.82E-175	7.52E-33	**0.00E+00**
$f3$	Mean	3.82E-02	3.07E-112	2.08E-03	6.52E-158	3.49E-26	**0.00E+00**
	Std.	1.52E-02	1.39E-111	9.39E-04	2.41E-157	1.17E-25	**0.00E+00**
	Rank	6	3	5	2	4	1
	Optimal	2.73E+02	1.17E-05	1.38E+02	5.39E+00	**4.23E-10**	1.69E-05
$f4$	Mean	3.33E+02	3.64E-05	3.22E+02	1.08E+01	**1.27E-09**	1.68E-03
	Std.	3.49E+01	2.46E-05	6.14E+01	4.65E+00	**6.45E-10**	1.95E-03
	Rank	6	2	5	4	1	3
	Optimal	2.74E+04	6.91E+03	6.17E+03	0.00E+00	0.00E+00	**0.00E+00**
$f5$	Mean	3.46E+04	9.06E+03	1.53E+04	0.00E+00	0.00E+00	**0.00E+00**
	Std.	2.80E+03	1.12E+03	2.82E+03	0.00E+00	0.00E+00	**0.00E+00**
	Rank	6	4	5	1	1	1
	Optimal	2.83E+04	1.00E+04	1.89E+04	**7.95E-06**	6.11E-04	8.05E-05
$f6$	Mean	3.86E+04	1.41E+04	2.92E+04	**3.64E-04**	1.95E-03	1.94E-03
	Std.	4.56E+03	2.07E+03	6.42E+03	**3.54E-04**	1.11E-03	1.63E-03
	Rank	6	4	5	1	3	2
	Optimal	3.12E-01	6.67E-03	3.68E-01	3.74E-02	7.26E-04	**7.77E-16**
$f7$	Mean	4.86E-01	8.76E-03	5.38E-01	6.02E-02	**1.40E-03**	1.26E-02
	Std.	6.39E-02	1.40E-03	9.11E-02	1.33E-02	**5.17E-04**	1.44E-02
	Rank	5	2	6	4	1	3
	Optimal	5.59E+00	1.16E+00	5.17E+00	1.00E+00	1.02E+00	**1.00E+00**
$f8$	Mean	6.90E+00	1.24E+00	6.53E+00	1.00E+00	1.03E+00	**1.00E+00**
	Std.	4.57E-01	4.04E-02	6.12E-01	**1.60E-16**	9.80E-03	7.52E-04
	Rank	6	4	5	1	3	1
	Optimal	1.87E+01	2.29E-03	3.41E+00	5.05E-03	9.44E-02	**3.83E-05**
$f9$	Mean	3.17E+01	4.22E-03	7.27E+00	1.01E-02	4.38E-01	**5.95E-04**
	Std.	4.91E+00	1.09E-03	1.68E+00	2.53E-03	2.88E-01	**4.09E-04**
	Rank	6	2	5	3	4	1
	Optimal	3.32E-09	3.84E-12	1.41E-08	5.34E-12	**3.53E-12**	3.51E-12
$f10$	Mean	1.76E-07	4.29E-12	2.72E-07	8.50E-12	**3.62E-12**	4.61E-12
	Std.	1.61E-07	2.22E-13	3.38E-07	1.40E-12	**1.11E-13**	1.08E-12
	Rank	5	2	6	4	1	3
	Optimal	8.51E-06	1.33E-04	1.11E-15	0.00E+00	0.00E+00	**0.00E+00**
$f11$	Mean	4.09E-04	2.87E-03	8.06E-09	0.00E+00	0.00E+00	**0.00E+00**
	Std.	3.00E-04	2.38E-03	1.50E-08	0.00E+00	0.00E+00	**0.00E+00**
	Rank	5	6	4	1	1	1
	Optimal	1.56E-04	6.93E-04	2.23E-07	2.13E-07	2.13E-07	**2.13E-07**
$f12$	Mean	1.25E-03	8.48E-03	6.51E-06	2.13E-07	2.58E-02	**2.13E-07**
	Std.	7.80E-04	6.54E-03	1.23E-05	0.00E+00	1.41E-01	**0.00E+00**
	Rank	4	5	3	1	6	1
	Optimal	6.72E-01	2.94E-01	3.37E-01	2.15E-01	8.28E-02	**8.30E-03**
$f13$	Mean	7.63E-01	3.70E-01	5.01E-01	2.23E-01	1.31E-01	**5.61E-02**
	Std.	3.79E-02	3.91E-02	7.86E-02	**7.97E-03**	2.01E-02	6.83E-02
	Rank	6	4	5	3	2	1
	Optimal	1.23E-05	3.67E-05	3.58E-07	3.58E-07	3.58E-07	**3.58E-07**
$f14$	Mean	4.66E-04	1.01E-02	2.24E-06	3.58E-07	3.58E-07	**3.58E-07**
	Std.	4.06E-04	1.97E-02	8.76E-06	0.00E+00	0.00E+00	**0.00E+00**
	Rank	5	6	4	1	1	1
	Optimal	5.88E-04	2.73E-02	5.86E-11	0.00E+00	0.00E+00	**0.00E+00**
$f15$	Mean	9.23E-02	2.82E-01	2.96E-01	3.33E-02	0.00E+00	**0.00E+00**
	Std.	8.85E-02	2.16E-01	4.42E-01	1.83E-01	0.00E+00	**0.00E+00**
	Rank	4	5	6	3	1	1
	Optimal	5.76E-04	5.09E-03	1.00E+00	0.00E+00	0.00E+00	**0.00E+00**
$f16$	Mean	7.54E-03	9.12E-02	2.50E-06	0.00E+00	2.70E+00	**0.00E+00**
	Std.	6.04E-03	1.12E-01	7.17E-06	0.00E+00	1.48E+01	**0.00E+00**
	Rank	4	5	3	1	6	1

	Optimal	4.13E-03	1.75E-01	3.73E-10	0.00E+00	0.00E+00	**0.00E+00**
$f17$	Mean	8.03E-02	3.33E+00	2.65E-04	0.00E+00	0.00E+00	**0.00E+00**
	Std.	7.09E-02	2.39E+00	9.73E-04	0.00E+00	0.00E+00	**0.00E+00**
	Rank	5	6	4	1	1	1
	Optimal	1.00E-03	1.52E-03	6.22E-05	1.35E-06	0.00E+00	**0.00E+00**
$f18$	Mean	6.79E-02	6.22E-02	1.57E-02	2.32E-05	5.41E-14	**0.00E+00**
	Std.	5.70E-02	6.65E-02	1.51E-02	2.51E-05	1.06E-13	**0.00E+00**
	Rank	6	5	4	3	2	1
	Average Rank	5.33	3.83	4.78	2.17	2.56	1.39
	Final Rank	6	4	5	2	3	1

TABLE 8. Results of comparison algorithms on benchmark functions (50-D)

		SPSO	BLPSO	CLPSO	SLPSO	ACPSO	SFAPSO
	Optimal	3.89E+02	3.42E+02	2.14E+02	8.10E-81	2.28E-03	**0.00E+00**
$f1$	Mean	1.16E+03	8.91E+02	9.42E+02	2.28E-79	1.03E-02	**0.00E+00**
	Std.	8.41E+02	4.20E+02	8.60E+02	4.25E-79	3.59E-03	**0.00E+00**
	Rank	6	4	5	2	3	**1**
	Optimal	9.59E-01	5.14E-01	7.41E-01	1.11E-16	2.00E-04	**0.00E+00**
$f2$	Mean	9.77E-01	6.09E-01	8.01E-01	1.11E-16	4.42E-04	**0.00E+00**
	Std.	5.18E-03	3.82E-02	3.38E-02	0.00E+00	1.34E-04	**0.00E+00**
	Rank	6	4	5	2	3	**1**
	Optimal	1.24E-02	1.42E-04	2.60E-04	1.24E-194	8.13E-28	**0.00E+00**
$f3$	Mean	6.27E-02	7.41E-04	2.57E-03	4.03E-176	1.72E-24	**0.00E+00**
	Std.	2.59E-02	4.69E-04	2.25E-03	0.00E+00	3.93E-24	**0.00E+00**
	Rank	6	4	5	2	3	1
	Optimal	6.39E+02	7.15E+02	4.96E+02	1.66E+02	**1.03E-05**	1.41E-03
$f4$	Mean	1.09E+03	1.14E+09	7.42E+02	2.06E+02	**3.31E-05**	1.29E-02
	Std.	1.11E+03	2.59E+09	1.85E+02	2.44E+01	**1.54E-05**	8.98E-03
	Rank	5	6	4	3	1	2
	Optimal	5.76E+04	1.52E+04	2.49E+04	**0.00E+00**	5.00E+00	0.00E+00
$f5$	Mean	7.44E+04	1.86E+04	3.32E+04	**3.33E-02**	1.01E+01	6.67E-01
	Std.	6.12E+03	1.96E+03	3.53E+03	**1.83E-01**	2.83E+00	8.44E-01
	Rank	6	4	5	1	3	2
	Optimal	7.49E+04	2.39E+04	5.46E+04	3.62E-01	2.23E+00	**1.12E-01**
$f6$	Mean	1.03E+05	3.67E+04	8.10E+04	1.73E+02	7.03E+00	**3.74E-01**
	Std.	1.19E+04	7.60E+03	1.54E+04	9.12E+02	3.03E+00	**1.46E-01**
	Rank	6	4	5	3	2	1
	Optimal	4.92E-01	5.87E-01	5.31E-01	1.11E-01	4.28E-03	**3.13E-03**
$f7$	Mean	6.46E-01	8.68E-01	6.51E-01	1.43E-01	**6.37E-03**	2.97E-02
	Std.	5.95E-02	1.21E-01	7.27E-02	1.88E-02	**1.65E-03**	1.67E-02
	Rank	4	6	5	3	1	2
	Optimal	1.28E+01	1.43E+01	1.16E+01	1.00E+00	1.33E+00	**1.00E+00**
$f8$	Mean	1.39E+01	1.64E+01	1.31E+01	1.00E+00	1.56E+00	**1.00E+00**
	Std.	5.46E-01	7.26E-01	8.37E-01	**0.00E+00**	8.31E-02	4.46E-03
	Rank	5	6	4	1	3	1
	Optimal	1.15E+02	1.15E+00	2.25E+01	1.56E-02	2.69E-01	**3.51E-05**
$f9$	Mean	1.51E+02	4.16E+00	3.38E+01	2.56E-02	6.58E-01	**4.62E-04**
	Std.	1.79E+01	1.86E+00	6.22E+00	5.28E-03	3.21E-01	**4.36E-04**
	Rank	6	4	5	2	3	1
	Optimal	4.63E-13	3.97E-11	1.14E-12	2.63E-20	3.42E-20	**1.21E-20**
$f10$	Mean	2.41E-11	4.27E-10	9.26E-11	3.36E-20	5.55E-20	**1.49E-20**
	Std.	3.53E-11	4.18E-10	8.77E-11	4.09E-21	1.27E-20	**2.66E-21**
	Rank	4	6	5	2	3	1
	Optimal	4.18E-06	2.14E-05	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**
$f11$	Mean	3.23E-04	1.50E-03	2.92E-11	0.00E+00	0.00E+00	**0.00E+00**
	Std.	3.34E-04	1.74E-03	6.15E-11	0.00E+00	0.00E+00	**0.00E+00**
	Rank	5	6	4	1	1	1
	Optimal	2.90E-04	4.10E-04	2.13E-07	2.13E-07	2.13E-07	**2.13E-07**
$f12$	Mean	1.10E-03	7.52E-03	4.43E-07	2.13E-07	9.54E-02	**2.13E-07**
	Std.	6.39E-04	6.87E-03	7.80E-07	0.00E+00	5.23E-01	**0.00E+00**
	Rank	4	5	3	1	6	1
	Optimal	9.43E-01	5.01E-01	7.22E-01	3.25E-01	2.87E-01	**2.47E-02**
$f13$	Mean	9.83E-01	5.87E-01	8.45E-01	3.42E-01	3.19E-01	**2.32E-01**
	Std.	1.14E-02	5.36E-02	5.97E-02	**7.25E-03**	1.95E-02	1.39E-01
	Rank	6	4	5	3	2	1

	Optimal	2.24E-05	5.05E-05	3.58E-07	3.58E-07	3.58E-07	**3.58E-07**
$f14$	Mean	3.54E-04	4.24E-03	4.76E-07	3.58E-07	3.58E-07	**3.58E-07**
	Std.	3.04E-04	5.05E-03	4.90E-07	0.00E+00	0.00E+00	**0.00E+00**
	Rank	5	6	4	1	1	1
	Optimal	1.72E-03	2.34E-02	3.29E-13	0.00E+00	0.00E+00	**0.00E+00**
$f15$	Mean	7.30E-02	2.29E-01	2.19E-01	6.67E-02	0.00E+00	**0.00E+00**
	Std.	6.08E-02	2.19E-01	4.08E-01	2.54E-01	0.00E+00	**0.00E+00**
	Rank	4	6	5	3	1	1
	Optimal	1.45E-04	8.90E-04	1.40E-12	0.00E+00	0.00E+00	**0.00E+00**
$f16$	Mean	9.94E-03	2.90E-02	6.32E-09	0.00E+00	2.70E+00	**0.00E+00**
	Std.	1.17E-02	3.08E-02	1.30E-08	0.00E+00	1.48E+01	**0.00E+00**
	Rank	4	5	3	1	6	1
	Optimal	1.45E-03	1.77E-02	0.00E+00	0.00E+00	0.00E+00	**0.00E+00**
$f17$	Mean	7.83E-02	1.47E+00	1.53E-06	0.00E+00	0.00E+00	**0.00E+00**
	Std.	7.13E-02	1.33E+00	7.71E-06	0.00E+00	0.00E+00	**0.00E+00**
	Rank	5	6	4	1	1	1
	Optimal	2.15E-06	4.65E-05	3.71E-05	4.05E-07	7.11E-15	**0.00E+00**
$f18$	Mean	1.62E-04	7.02E-04	1.24E-03	1.25E-05	**1.43E-12**	2.08E-11
	Std.	1.34E-04	6.13E-04	1.08E-03	1.19E-05	**1.97E-12**	1.14E-10
	Rank	4	5	6	3	1	2
	Average Rank	5.11	5.06	4.56	1.94	2.33	1.22
	Final Rank	6	5	4	2	3	1

convergence effect in $f4$ function than other algorithms, because the optimal value of $f4$ function, i.e., zakharovfcn function tend to be smooth, The improved inertia weight adaptive strategy used in SFAPSO algorithm gives each particle a specific weight, which will lead to the elite particles search stagnation when they exploit near the optimal value. The convergence of SFAPSO on the $f6$ function is slightly worse than that of SLPSO, but it is better than the other four comparison algorithms. Although the mean value of the $f7$ function is not better than other algorithms, the minimum value is better, from which we can see that the exploitation ability of SFAPSO on the $f7$ function is effective. The convergence of ACPSO and SFAPSO on the $f10$ function is almost the same, which shows excellent competitiveness.

By analyzing the 50-dimensional data in TABLE 8, we can see that SFAPSO still has excellent exploitation ability in $f1$, $f2$, and $f3$ functions, which can show that SFAPSO still has remarkable exploitation ability in the high-dimensional space of unimodal function. SFAPSO and SLPSO can converge to the optimal value on $f5$ and $f8$ functions, but their stability is slightly worse than SLPSO. The main reason is that the learning objectives of social learning mechanism proposed by SLPSO is the whole particle swarm. Hence, their stability is better on some functions, while SFAPSO convergence accuracy and stability on other functions are better than the other five comparative algorithms.The functions of $f11$, $f15$, $f16$, and $f17$ functions can converge to the actual optimal value regardless of the minimum value or the mean value, which shows that SFAPSO algorithm has a certain competitiveness in the accuracy of solving multimodal problems. At the same time, the standard deviation shows that SFAPSO algorithm has superior stability.

It is easy to get that SFAPSO, SLPSO and ACPSO converge to the optimal value on the $f8$, $f11$, and $f17$ multimodal functions by comparing the 30 and 50-dimensional data. But SLPSO and ACPSO are not converging to the optimal value in the $f12$, $f15$, and $f18$ multimodal functions, which shows that SFAPSO has good performance in multimodal optimization. The average and final ranking of the SFAPSO algorithm in 30 and 50-dimensions are the first. The final rank shows that the SFAPSO algorithm has good competitiveness in convergence accuracy and stability.

In addition, to illustrate the stability of the algorithm, the box graph is used to analyze the results of eighteen benchmark functions running independently from 30 and

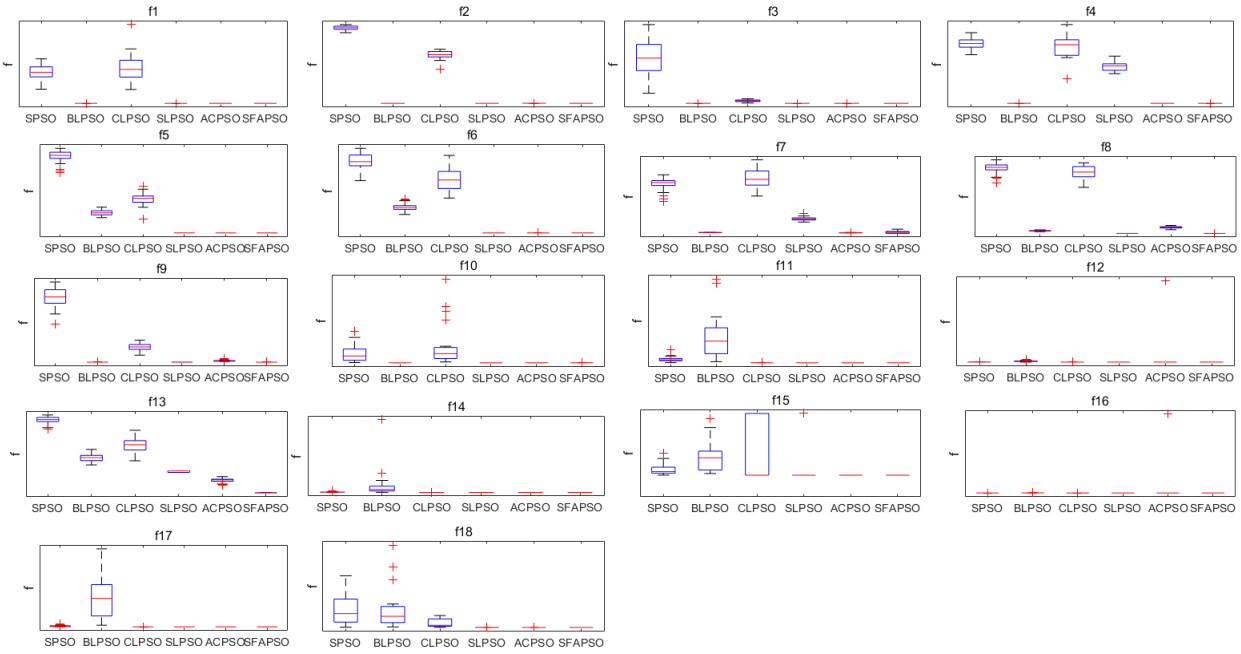

FIGURE 5. The box graphs of comparison algorithms (30-D)

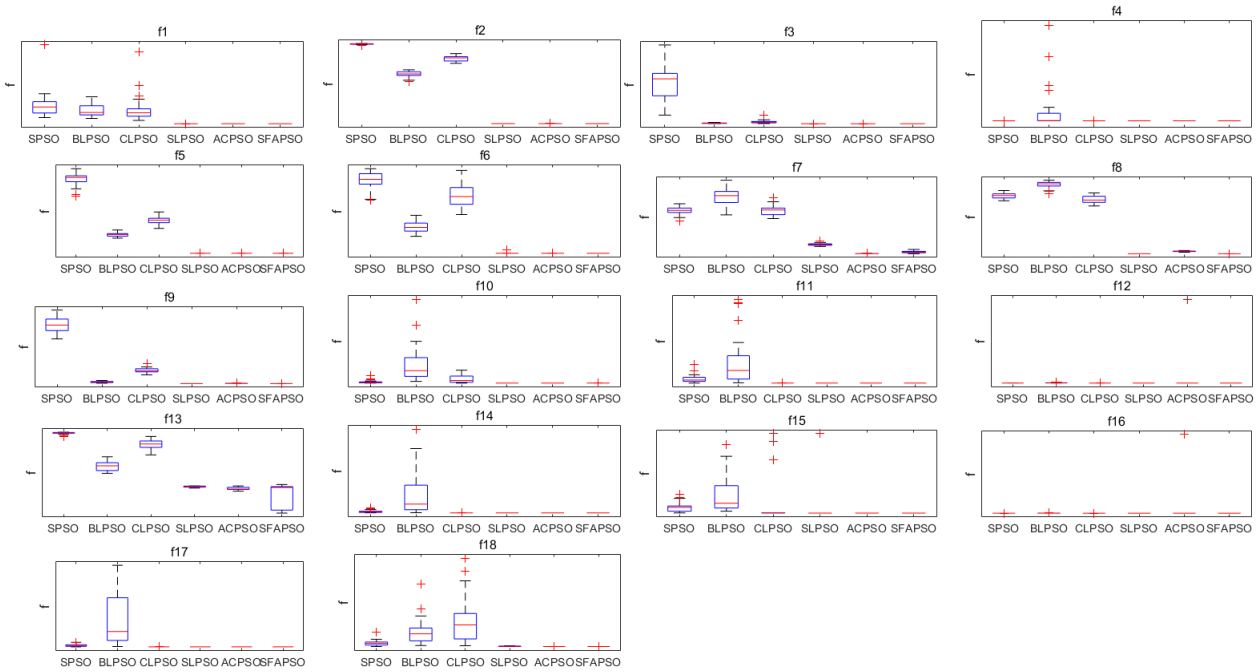

FIGURE 6. The box graphs of comparison algorithms (50-D)

50-dimensions 30 times. The results are shown in FIGURE 5 and FIGURE 6. The upper horizontal line of the rectangular box in the figure is the upper quartile of the data. The lower horizontal line is the lower quartile, and the red horizontal line in the rectangle represents the median. If there is only one horizontal line, the result of the algorithm is the exavt 30 times. In the $f7$ function, i.e., Happy cat function, it is slightly worse than ACPSO, because there are many minimum points near its optimal value, so it still can not play the advantages of SFAPSO. Still, it is relatively stable in other functions, which

indicates that SFAPSO effectively balances the exploration and exploitation capabilities while ensuring the accuracy. Combined with the ranking of 30 and 50-dimensions and the stability analysis of the box graph, it can be concluded that SFAPSO is better than other algorithms on the whole and shows good performance.

5. **Conclusions.** In order to solve the problem that particle swarm optimization (PSO) algorithm falls into local optimum when solving some problems, an adaptive PSO algorithm using scale-free network topology is proposed. In the SFAPSO algorithm, the update strategy with velocity differential is designed based on the neighborhood relationship, which takes advantage of the power-law distribution of scale-free network topology and consider full play to the advantage of elite particle information sharing. In addition, by introducing an improved inertia weight adaptive strategy, each particle is given a specific inertia weight to balance the global exploration and local exploitation ability. The SFAPSO algorithm proposed in this paper is compared with five well-known PSO variants. The experimental results show that the SFAPSO algorithm has efficient robustness which competitive solutions can be obtained.

However, the SFAPSO algorithm can't get a satisfactory solution effectively in solving some problems with more local extremum. Therefore, the subsequent work will focus on diversity exploitation to further improve the global search ability of the algorithm.

6. **Acknowledgement.** This work was supported by the National Natural Science Foundation of China (Grant Nos. 62066019 and 61903089), the JiangXi Provincial Natural Science Foundation (Grant Nos. 20202BABL202020 and 20202BAB202014), the National Key Research and Development Program (Grant Nos. 2020YFB1713700 and 2020YFB1713705).

REFERENCES

[1] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, *Mhs95 Sixth International Symposium on Micro Machine & Human Science IEEE*, vol. 1, pp. 39-43. 2002.
[2] Z. Liu, T. Nishi, Multipopulation ensemble particle swarm optimizer for engineering design problems, *Mathematical Problems in Engineering*, vol. 2020, 1450985, 2020.
[3] El. Sehiemy, A. Ragab, F. Selim, B. Bentouati, M. A. Abido, A novel multi-objective hybrid particle swarm and salp optimization algorithm for technical-economical-environmental operation in power systems, *Energy*, vol. 193, 116817, 2020.
[4] S. Tian, Y. Li, Y. Kang, J. Xia, Multi-robot path planning in wireless sensor networks based on jump mechanism PSO and safety gap obstacle avoidance, *Future Generation Computer Systems*, vol. 118. pp. 37-47, 2020.
[5] X. C. Pu, J. J. Li, Y. Zhang, An improved PSO algorithm for robot multi-goal path planning, *International Journal of Science*, vol. 4, no. 3, 2017.
[6] Y. Shi, R. Eberhart, A modified particle swarm optimizer, *IEEE World Congress on Computational Intelligence*, pp. 69-73, 1998.
[7] Y. Shi, R. Eberhart, Empirical study of particle swarm optimization, *Congress on Evolutionary Computation-CEC99*, vol. 3, pp. 1945–50, 1999.
[8] B. Liu, L. Wang, Y. H. Jin, F. Tang, D. X. Huang, Improved particle swarm optimization combined with chaos, *Solitons and Fractals*, vol. 25, no. 5, pp. 1261–1271, 2005.
[9] M. Tanweer, S. Suresh, N. Sundararajan, Self regulating particle swarm optimization algorithm, *Information Sciences*, vol. 294, pp. 182–202, 2015.
[10] A. Ratnaweera, S. Halgamuge, H. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients *IEEE Transactions on Evolutionary Computation*, vol. 8, pp. 240–255, 2004.
[11] A. P. Piotrowskia, J. J. Napiorkowskia, A. E. Piotrowskab, Population size in particle swarm optimization, *Swarm and Evolutionary Computation*, vol. 58, 100718, 2020.
[12] G. Xu, An adaptive parameter tuning of particle swarm optimization algorithm, *Applied Mathematics and Computation*, vol. 219, no. 9, pp. 4560-4569, 2013.

[13] Z. Zhou, Y. H. Shi , Inertia weight adaption in particle swarm optimization algorithm, *Lecture Notes in Computer Science*, vol. 6728, no. 1, pp. 71-79, 2011.
[14] Z. Zhan, J. Zhang, Y. Li, H. S. Chung, Adaptive particle swarm optimization, *IEEE Transactions on Systems, Man, Cybern. Part-B*, vol. 39, no. 6, pp. 1362–1381, 2009.
[15] A. Agrawal, S. Tripathi, Particle swarm optimization with probabilistic inertia weight, *Advances in Intelligent Systems and Computing*, vol. 741, pp. 239-248, 2019.
[16] L. Kang, R. S. Chen, N. Xiong, Y. C. Chen, Y. X. Hu, C. M. Chen, Selecting hyper-parameters of gaussian process regression based on non-inertial particle swarm optimization in Internet of things, *IEEE Access*, vol. 7, pp. 59504-59513, 2019.
[17] J. M. T. Wu, G. Srivastava, J. C. W. Lin, Q. Teng, A multi-threshold ant colony system-based sanitization model in shared medical environments, *ACM Transactions on Internet Technology*, vol. 21, no. 49, pp. 1-26, 2021.
[18] W. Li, X. Meng, Y. Huang, Z. Fu, Multipopulation cooperative particle swarm optimization with a mixed mutation strategy, *Information Sciences*, vol. 529, pp. 179-196, 2020.
[19] A. B. Meng, Z. Li, H. Yin, S. Z. Chen, Z. Z. Guo, Accelerating particle swarm optimization using crisscross search, *Information Sciences*, vol. 329. pp. 52-72, 2016.
[20] E. K. Wang, C. M. Chen, S. M. Yiu, MM Hassan, M. Alrubaian, G. Fortino, Incentive evolutionary game model for opportunistic social networks, *Future Generation Computer Systems*, vol. 102, pp. 14-29, 2020.
[21] J. M. T. Wu, G. Srivastava, S. Tayeb, J. C. W. Lin, A PSO-based sanitization process with multi-thresholds model, *Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event*, 2021.
[22] Q. Y. Yang, S. C. Chu, Pan, J. Shyang, C. M. Chen, Sine cosine algorithm with multigroup and multistrategy for solving CVRP, *Mathematical Problems in Engineering*, https://doi.org/10.1155/2020/8184254, 2020.
[23] F. Q. Zhang, T. Y. Wu, Y. Wang, R. Xiong, G. Y. Ding, P. Mei, L. Y. Liu, Application of quantum genetic optimization of LVQ neural network in smart city traffic network prediction, *IEEE Access*, vol. 8, pp. 104555-104564, 2020.
[24] L. L. Kang, R. S. Chen, Y. C. Chen, C. C. Wang, X. G. Li, T. Y. Wu, Using cache optimization method to reduce network traffic in communication systems based on cloud computing, *IEEE Access*, vol. 7, no. 1, pp. 124397-124409, 2019.
[25] D. Chemodanova, F. Espositob, P. Calyama, A. Sukhovcd, REBATE: a epulsive-based traffic engineering protocol for dynamic scale-free networks, *Future Generation Computer Systems*, vol. 108, pp. 624-635, 2020.
[26] D. Liu, V. Fodor, L. K. Rasmussen, Will scale-free popularity develop scale-free geo-social networks?, *IEEE Transactions on Network Science and Engineering*, vol. 6, no. 3, pp. 587-598, 2019.
[27] H. Wen, X. P. Fan; H. F. Zhang, A. H. Chen, Research on scale-free network congestion control, *Journal of Chinese Computer Systems*, vol. 34, no. 11, pp. 2482-2486, 2013.
[28] A. L. Barabási, R. Albert, Emergence of scaling in random networks, *science*, vol. 286, no. 15439, pp. 509-512, 1999.
[29] http://www.sfu.ca/ ssurjano/optimization.html.
[30] M. Clerc, Multipopulation cooperative particle swarm optimization with a mixed mutation strategy, *Standard Particle Swarm Optimisation Available*, vol. 529, pp. 179-196, 2011.
[31] J. J. Liang, A. Qin, P. N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281-295, 2006.
[32] X. Chen, H. Tianfield, C. Mei, W. Du, G. Liu, Biogeography-based learning particle swarm optimization, *Soft Computing*, vol. 21, no. 24, pp. 7519-7541, 2017.
[33] R. Cheng, Y. Jin, A social learning particle swarm optimization algorithm for scalable optimization, *Information Sciences*, vol. 291, pp. 43-60, 2015.
[34] H. Mohammad, R. M. Mohammad, M. E. Mohammad, Adaptive cooperative particle swarm optimizer, *Applied Intelligence*, vol. 39, pp. 397-420, 2013.