Mathematical Biosciences and Engineering

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

MBE, 18(6): 7464–7489.
DOI: 10.3934/mbe.2021369
Received: 10 June 2021
Accepted: 23 August 2021
Published: 31 August 2021

Research article

A multi-sample particle swarm optimization algorithm based on electric field force

Shangbo Zhou[1,2,*], **Yuxiao Han**[1,2], **Long Sha**[1,2] **and Shufang Zhu**[1,2]

[1] College of Computer Science, Chongqing University, Chongqing 400044, China

[2] Key Laboratory of Dependable Service Computing in Cyber Physical Society, Ministry of Education, Chongqing 400030, China

* **Correspondence:** Email: shbzhou@cqu.edu.cn.

Abstract: Aiming at the premature convergence problem of particle swarm optimization algorithm, a multi-sample particle swarm optimization (MSPSO) algorithm based on electric field force is proposed. Firstly, we introduce the concept of the electric field into the particle swarm optimization algorithm. The particles are affected by the electric field force, which makes the particles exhibit diverse behaviors. Secondly, MSPSO constructs multiple samples through two new strategies to guide particle learning. An electric field force-based comprehensive learning strategy (EFCLS) is proposed to build attractive samples and repulsive samples, thus improving search efficiency. To further enhance the convergence accuracy of the algorithm, a segment-based weighted learning strategy (SWLS) is employed to construct a global learning sample so that the particles learn more comprehensive information. In addition, the parameters of the model are adjusted adaptively to adapt to the population status in different periods. We have verified the effectiveness of these newly proposed strategies through experiments. Sixteen benchmark functions and eight well-known particle swarm optimization algorithm variants are employed to prove the superiority of MSPSO. The comparison results show that MSPSO has better performance in terms of accuracy, especially for high-dimensional spaces, while maintaining a faster convergence rate. Besides, a real-world problem also verified that MSPSO has practical application value.

Keywords: particle swarm optimization; electric field force; comprehensive learning; segmented learning; parameter adaptation

1. Introduction

In recent years, traditional optimization algorithms cannot optimally solve many complex problems in real life. On the contrary, the metaheuristic algorithm can give a feasible solution to the

problem within a limited range and approach the optimal solution as much as possible. Therefore, it has gradually become the first choice for solving these complex problems and is used to deal with multi-dimensional, single-object or multi-objective, continuous, or combinatorial optimization problems. Metaheuristic algorithms deal with issues by abstracting some phenomena in nature and animals into algorithms. Some representative algorithms have also appeared in this field, such as genetic algorithm (GA), particle swarm optimization (PSO), ant colony optimization (ACO), symbiotic organisms search (SOS), simulated annealing (SA), salp swarm algorithm (SSA), and so on. To improve the algorithm's performance further, the hybrid metaheuristic algorithm that combines the advantages of various algorithms has also received extensive attention. We can't extravagantly expect an algorithm to solve all optimization problems. Therefore, researchers always upgrade metaheuristic algorithms for specific issues. Kaplan et al. [1] introduced GA in the field search of SA, which improves the search performance and effectively solves the excitation current estimation of synchronous motors. To solve complex multimodal functions and high-dimensional problems, Mohamed et al. [2] introduced the concept of attraction and repulsion into the imperial competitive algorithm (ICA). They verified the effectiveness of the algorithm through a multi-objective engineering problem. Çelik et al. [3] proposed an improved slap swarm algorithm (mSSA) to avoid the premature convergence of SSA. They also offered an improved stochastic fractal search (ISFS) algorithm, which replaced the first update process of SFS with a chaos-based strategy to effectively deal with the automatic generation control problem of the power system [4]. Lin et al. [5] proposed a comprehensive algorithm combining PSO and the estimate distribution algorithm (EDA) to solve the maximum segmentation problem. In addition, considering the shortcomings of SOS premature, ISOS [6] combined the advantages of quasi-oppositional based learning (QOBL) and chaotic local search (CLS), which balances exploration and exploitation. hSOS-SA [7] integrated the SA into the SOS algorithm to improve the convergence accuracy of the algorithm. Similarly, Singh et al. [8] proposed a hybrid metaheuristic algorithm (HSSAPSO), using the velocity phase of PSO in SSA, which reduced the risk of PSO falling into a local optimum. The abovementioned researches review that the metaheuristic algorithms have significantly developed.

Among the developing methods in the field of metaheuristic algorithms, the PSO algorithm has attracted wide attention from researchers and practitioners for its advantages of easy implementation, strong adaptability, and low complexity. PSO [9] is a metaheuristic algorithm proposed by Kennedy and Eberhart to simulate the predation behavior of birds. The algorithm compares the problem's search space to the flight space of birds, and each bird is abstracted into a particle to represent a candidate solution to the problem. According to the fitness value, all particles are randomly searched domain to find a better solution. Therefore, the algorithm relies on a random process similar to evolution. Randomness makes the PSO algorithm generate uncertainty. Hence, researchers usually adopt new strategies to make the particles move purposefully, thereby upgrading the PSO algorithm. Uncertainty modeling has also made some progress over the years. Given the uncertainty of gene expression data, Taylan et al. [10] introduced stochastic differential equations into uncertainty modeling for the first time. Kropat et al. [11] used an interval algorithm to overcome the problem that given data is susceptible to noise. Since the data for practical problems are usually discrete, CMARS [12] applied the framework of conic quadratic programming to improve multivariate adaptive regression splines (MARS). Özmen et al. [13] further improved the CMARS algorithm through a robust optimization technique to deal with data uncertainty. A robust and flexible regulatory network regression model was introduced in the literature [14] to determine unknown system parameters from uncertain data. Semialgebraic sets

were used to express the uncertain state of genes and environmental factors in Reference [15], solving the gene-environment network's data uncertainty, which improved the model prediction accuracy. The above researchers have adopted effective methods to address data uncertainty and reduce errors. As a simple but powerful method, PSO is superior to deterministic algorithms in solving some specific problems. And PSO has shown good performance in many practical optimization tasks and has been used in many research fields, including Function Optimization [16–18], Classification prediction [19, 20], Neural network training [21–23], Feature selection [24–26], and Image encryption [27, 28].

Although PSO is simple and easy to implement, it still has the problems of easily falling into a local optimum and slow convergence speed. To overcome these problems, many researchers have proposed various PSO algorithm variants. To enhance the searchability of the algorithm, Liang et al. [29] offered a dynamic grouping particle swarm optimization (DMSPSO) algorithm, which divides the population into multiple subgroups. The members of the group exchange information for better local exploration, and at the same time, frequently reorganize and change the learning samples to realize the information exchange among the populations. HCLDMS-PSO [30] introduced a non-uniform mutation operator in the PSO to enhance the global search ability and adjusts the global best position through the Gaussian mutation operator, reducing the risk of falling into the local optimum. Zhang et al. [31] added two constraint factors to the PSO through migration learning, namely the source domain factor and the target domain factor, to balance the PSO algorithm's search ability and search efficiency.

Furthermore, to avoid premature convergence, CMPSOWV [32] discarded the velocity component in the particle velocity update formula and introduced a constraint processing method to guide the particle search space with the best personal position and the global best position. Chen et al. [33] also introduced chaos mapping in the PSO algorithm and adjusted the inertia weight through the sinogram, balancing local exploitation and global exploration effectively. To better deal with multi-modal functions, Zhang et al. [34] introduced a local optimal topology (LOT) based on the comprehensive learning particle swarm optimization (CLPSO) algorithm, which expanded the search space of particles and improved the convergence speed of the algorithm. To increase the convergence speed of the algorithm, Zhu et al. [35] proposed a multi-ion particle swarm optimization algorithm (MIONPSO) based on repulsive force and attractive force. They introduced the concept of charge in the PSO algorithm and divided the population into multiple sub-populations, then the optimal solution of each group guides the update of individuals. In addition, papers [36, 37] both adopted various strategies in the PSO algorithm to make the particles search better in the feasible domain space, which improves the accuracy of the solution.

Based on the above research, we can find that most of the articles on PSO algorithms have put forward novel concepts to increase the diversified behavior of particles, thereby reducing the risk of falling into a local optimum. Nevertheless, the matching update strategies of many new PSO algorithms are not perfect, and they cannot take the convergence speed and accuracy into account simultaneously. Additionally, when solving complex optimization problems, a single improvement strategy has no advantage in improving convergence accuracy. On the contrary, adopting a variety of improvement strategies can enhance the diversity of the algorithm and achieve higher convergence accuracy.

To further improve the performance of the PSO algorithm, inspired by MIONPSO, this paper proposes a multi-sample particle swarm optimization algorithm based on electric field force. MSPSO introduces the concept of the electric field into the PSO algorithm and makes it match more complete strategies. We construct multiple learning samples to improve the performance of the particle swarm

algorithm in solving unimodal and multimodal problems. Lots of experiments have verified the effectiveness of these additional strategies. We further evaluated the performance of MSPSO through practical cases of design problems of multiphase codes for spread spectrum pulse radar. And the main contributions of this paper can be summarized as follows.

- To show the diverse behavior of particles, we regard the feasible region of the population as an electric field, and the particles suffer the electric field force of other charged particles in the electric field.
- We propose an electric field force-based comprehensive learning strategy to construct the attractive sample and the repulsive sample to guide the particle movement purposefully. The interaction of the two pieces directs the particles to areas that are more conducive to exploration.
- We use the historical experience information of elite particles and general particles obtained by the tournament mechanism to update the corresponding weight coefficients adaptively, which enhances the diversity of the population.
- To reduce the risk of falling into a local optimum, we construct a global learning sample by a segment-based weighted learning strategy so that particles can learn more helpful information from the elite particles.

The rest of the paper is organized as follows. Section 2 introduces related works. Section 3 describes the details of MSPSO. The performance of newly introduced strategies is experimentally verified in Section 4. In Section 5, sixteen benchmark functions and eight well-known PSO variants are employed to verify the effectiveness of MSPSO. Finally, Section 6 summarizes the relevant conclusions.

2. Related work

2.1. Basic PSO

PSO is a random optimization algorithm based on population, in which each particle represents a potential problem solution. In the D-dimensional space, each particle has two attributes, the velocity vector $\boldsymbol{V}_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,D})$ and the position vector $\boldsymbol{X}_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$. $\boldsymbol{X}_i$ is the candidate solution of the problem, and $\boldsymbol{V}_i$ represents the search direction and step size of i^{th} particle. Each individual adjusts its trajectory according to its own best historical experience $\boldsymbol{pbest}_i = (pb_{i,1}, pb_{i,2}, \ldots, pb_{i,D})$ and the best overall experience in history $\boldsymbol{gbest} = (g_1, g_2, \ldots, g_D)$ in the feasible domain space. The speed and position update rules are defined as Equations (2.1) and (2.2) respectively.

$$v_{i,j}^{t+1} = \omega v_{i,j}^{t} + c_1 r_1 \left(pbest_{i,j}^{t} - x_{i,j}^{t}\right) + c_2 r_2 \left(gbest_{j}^{t} - x_{i,j}^{t}\right), \tag{2.1}$$

$$x_{i,j}^{t+1} = x_{i,j}^{t} + v_{i,j}^{t+1}, \tag{2.2}$$

where $j = 1, 2, \ldots, D$, ω represents the inertia weight which determines the proportion of the previous speed; c_1 and c_2 are acceleration learning factors, respectively representing the weights learned from $\boldsymbol{pbest}_i$ and $\boldsymbol{gbest}$; r_1 and r_2 are random numbers in the interval $[0, 1)$.

2.2. Other improved PSO algorithms

The PSO algorithm mainly improves from four directions, including parameter adjustment, learning modes adjustment, topology change, and hybrid algorithm. The proposed MSPSO mainly covers the

two improvement directions of adjusting parameters and changing particle learning modes, so other approaches will not be discussed.

For adjusting parameters, Shih and Eberhart [38] proposed a strategy of linearly reducing the inertia weight from 0.9 to 0.4, changing the step length of the particle's velocity component in different evolutionary periods. XPSO [37] also adopted this method which effectively improves the performance of the algorithm. However, the strategy of linear weight reduction does not perform well on the objective function with multiple optimal solutions. To solve the above problem, Farooq et al. [39] proposed a phased update strategy. In the first half of the iteration, the inertia weight was linearly reduced from 0.9 to 0.4. The same method is repeated in the second half of the iteration. Chatterjee et al. [40] proposed a new nonlinear function to adaptively change the inertia weight, which effectively balances local exploitation and global exploration. Similarly, the sigmoid activation function in the neural network is used to change the inertia weight in HCLDMS-PSO [30] non-linearly. The abovementioned researches reveal that the adaptive adjustment of parameters will benefit the evolution of the population.

For changing particle learning modes, Liang et al. [41] proposed CLPSO, which encourages each particle to learn from different particles of different dimensions to obtain more comprehensive information. An opposition-based learning competitive particle swarm optimizer (OBL-CPSO) [42] introduced two mechanisms of oppositional learning and competitive learning. Competitive learning allows particles with poor fitness to learn from particles with good fitness, and particles with moderate fitness are updated through oppositional learning. To solve the problem of dimensionality disaster, Shi et al. [43] proposed a segment-based learning strategy, which randomly divides the dimension into several segments. At the same time, a predominant learning strategy allows elite particles to guide each dimension segment. Additionally, MPCPSO [36] introduced two new strategies: a dynamic segment-based mean learning strategy (DSMLS) and a multidimensional comprehensive learning strategy (MDCLS), which effectively improved the performance of the algorithm. DSMLS realizes the information exchange between the elite population and the ordinary population, and MDCLS accelerates the convergence speed. XPSO [37] extends the social learning part of each particle from one sample to two samples so that the particles learn from the global best particle and the local best particle. The abovementioned researches show that the dynamic selection of learning samples effectively maintains the diversity of the population, which is conducive to solving complex multimodal problems.

3. The proposed MSPSO algorithm

In this part, we introduce the proposed MSPSO in detail. Section 3.1 gives the particle model, and Section 3.2 explains the electric field force-based comprehensive learning strategy. The segment-based weighted learning strategy is introduced in Section 3.3, and Section 3.4 lists the framework of the MSPSO algorithm.

3.1. Learning model based electric field force

In MSPSO, the feasible region of particles is seen as an electric field, and every particle is regarded as an electric charge. Hence, the electric field has a powerful effect on the particles. To put it simply, when the particle velocity is updated, the electric field force around the particle will affect it. We hope that the particle swarm can exhibit diverse behaviors by calculating the electric field force between particles, which is beneficial to the evolution of the population. At the same time, we propose an elec-

tric field force-based comprehensive learning strategy to construct attractive and repulsive samples and utilize the historical knowledge of particles to adjust the weight coefficients adaptively. Additionally, to increase the convergence accuracy of the algorithm, we adopt a segment-based weighted learning strategy to construct a global learning sample. The proposed MSPSO is introduced in detail as follows. The velocity update rule of positively charged particles:

$$\boldsymbol{V}_i^{t+1} = \omega^t \cdot \boldsymbol{V}_i^t + \alpha_1^t \cdot (\boldsymbol{PE}_1^t - \boldsymbol{X}_i^t) - \beta_1^t \cdot \left(\boldsymbol{PG}_1^t - \boldsymbol{X}_i^t\right) + c \cdot r \cdot \left(\boldsymbol{GM}^t \cdot f_{ig} - \boldsymbol{X}_i^t\right), \tag{3.1}$$

The velocity update rule of negatively charged particles:

$$\boldsymbol{V}_i^{t+1} = \omega^t \cdot \boldsymbol{V}_i^t + \alpha_2^t \cdot (\boldsymbol{PE}_2^t - \boldsymbol{X}_i^t) - \beta_2^t \cdot \left(\boldsymbol{PG}_2^t - \boldsymbol{X}_i^t\right) + c \cdot r \cdot \left(\boldsymbol{GM}^t \cdot f_{ig} - \boldsymbol{X}_i^t\right), \tag{3.2}$$

Update the position according to Equation (3.3).

$$\boldsymbol{X}_i^{t+1} = \boldsymbol{X}_i^t + \boldsymbol{V}_i^{t+1}, \tag{3.3}$$

where $\boldsymbol{V}_i^t = (v_{i,1}, v_{i,2}, \ldots, v_{i,D})$ is the velocity of the i^{th} particle at the t^{th} time, and $\boldsymbol{X}_i^t = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$ is the position of the i^{th} particle at the t^{th} time. ω^t is t^{th} inertia weight. $\boldsymbol{PE}_1^t$ and $\boldsymbol{PG}_1^t$ represent t^{th} attractive sample and repulsive sample of positively charged particles, respectively. $\boldsymbol{PE}_2^t$ and $\boldsymbol{PG}_2^t$ denote t^{th} attractive sample and repulsive sample of negatively charged particles, respectively. α_1^t, β_1^t, α_2^t, and β_2^t represent t^{th} weight coefficient of the corresponding learning sample. $\boldsymbol{GM}^t$ denotes t^{th} global learning sample and f_{ig} is the electric field force of $\boldsymbol{GM}^t$ on the current update particle $\boldsymbol{X}_i^t$.

The particle's velocity update equation consists of four parts. The first part is the velocity of the particle itself. Attraction is reflected in the second part. To reduce the risk of falling into a local optimum, we select elite particles to construct an attractive sample by EFCLS. The third part is repulsion. For particles with poor fitness, their trajectory may not find the optimal value, and the particles should be urged to explore the optimal value in other directions. Similarly, we choose general particles to construct the repulsive sample by EFCLS. The last part is the global learning sample built by the SWLS to guide particles' movement and increase the convergence accuracy of the algorithm.

The inertia weight ω plays a vital role in the PSO algorithm. A larger ω in the early stage can enhance the global exploration ability, while a smaller ω in the later stage is beneficial to local exploitation. Existing studies have shown that during the evolution of the population, the dynamic change of the inertia weight can obtain a better optimization result than the fixed value. In MSPSO, a nonlinear decreasing inertia weight based on the sigmoid function proposed by reference [30] is adopted. The calculation method of ω is defined as Equation (3.4).

$$\omega^t = \omega_{\max} - \frac{\omega_{\max} - \omega_{\min}}{1 + e^{-5 * \frac{t}{T}}}, \tag{3.4}$$

where $\omega_{\max}$ is the maximum inertia weight, and $\omega_{\min}$ is the minimum inertia weight, t is the current iteration number. T represents the maximum number of iterations.

3.2. Electric field force-based comprehensive learning strategy

In MSPSO, all particles are in an electric field. Particles with opposite charges attract each other, and particles with the same charge repel each other. To purposefully guide particles to a better area, we

enable the attraction of particles with opposite charges to lead the particles to move in the direction of the elite particles, thereby accelerating the convergence speed. At the same time, the repulsion of the particles with the same charge drives the poorer particles to explore other better directions, reducing the risk of falling into the local optimum. Therefore, we select elite particles and general particles through the tournament mechanism and construct attractive sample $\boldsymbol{PE}$ and repulsive sample $\boldsymbol{PG}$ based on the electric field force.

Firstly, We screen out the elite particles and general particles of the positively and negatively charged subpopulation through the tournament mechanism. The tournament mechanism is to classify the particles of the population according to fitness ranking. The specific method is as follows. We arrange all particles of the positively charged subpopulation in ascending order of fitness. After sorting, the set of positively charged particles $POS = \{\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_{n_1}\}$ satisfies $f(\boldsymbol{X}_1) \leq f(\boldsymbol{X}_2) \leq \cdots \leq f(\boldsymbol{X}_{n_1})$, where f denotes fitness function, n_1 is the number of positively charged particles. Then we select the first n_e particles as the elite particles to create set S_1. And the last n_g particles are used as general particles in the positively charged subpopulation to form set S_2. Similarly, we arrange all the particles of the negatively charged subpopulation in ascending order of fitness. After sorting, the set of negatively charged particles $NEG = \{\boldsymbol{X}_{n_1+1}, \boldsymbol{X}_{n_1+2}, \ldots, \boldsymbol{X}_{n_1+n_2}\}$ satisfies $f(\boldsymbol{X}_{n_1+1}) \leq f(\boldsymbol{X}_{n_1+2}) \leq \cdots \leq f(\boldsymbol{X}_{n_1+n_2})$, where n_2 represent the number of negatively charged particles. Then we select the first n_e particles as the elite particles to create the set S_3. The last n_g particles are used as the general particles of the negatively charged subpopulation to form the set S_4. Finally, we obtained the following four sets.

elite particle set of the positively charged subpopulation $S_1 = \{\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_{n_e}\}$,
general particle set of the positively charged subpopulation $S_2 = \left\{\boldsymbol{X}_{n_1-n_g+1}, \boldsymbol{X}_{n_1-n_g+2}, \ldots, \boldsymbol{X}_{n_1}\right\}$,
elite particle set of the negatively charged subpopulation $S_3 = \{\boldsymbol{X}_{n_1+1}, \boldsymbol{X}_{n_1+2}, \ldots, \boldsymbol{X}_{n_1+n_e}\}$,
general particle set of the negatively charged subpopulation $S_4 = \left\{\boldsymbol{X}_{n_1+n_2-n_g+1}, \boldsymbol{X}_{n_1+n_2-n_g+2}, \ldots, \boldsymbol{X}_{n_1+n_2}\right\}$.

In physics, we calculate the magnitude of the force between two charges through Coulomb's law, it is proportional to the distance between the charges. In the early stage of population evolution, the distance between particles is so far that the force is small. On the contrary, particles are more concentrated in the later stage, and the distance between the particles is small. Therefore, we calculate the electric field force based on the distance between the particles. The electric field force of any two charged particles $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$ is defined as Equation (3.5).

$$f_{ij} = \varepsilon \cdot e^{-d_{ij}}, \tag{3.5}$$

where f_{ij} denotes the eletric field force of $\boldsymbol{X}_j$ to $\boldsymbol{X}_i$, d_{ij} represents the distance of $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$. ε is a real number distributed in $(0, 1)$, avoiding too large search space for particles.

The distance of any two particles $\boldsymbol{X}_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$ and $\boldsymbol{X}_j = \left(x_{j,1}, x_{j,2}, \ldots, x_{j,D}\right)$ is calculated by Euclidean distance:

$$d_{ij} = \sqrt{\sum_{k=1}^{D} \left(x_{i,k} - x_{j,k}\right)^2}, \tag{3.6}$$

where D is the dimension of a particle, $x_{i,k}$ represents the value of $\boldsymbol{X}_i$ in the k^{th} dimension, and $x_{j,k}$ denotes the value of $\boldsymbol{X}_j$ in the k^{th} dimension.

This paper adopts the same method to build $\boldsymbol{PE}_1$ and $\boldsymbol{PE}_2$, but the selected population is different. $\boldsymbol{PG}_1$ and $\boldsymbol{PG}_2$ are also constructed in the same way. Therefore, we take positively

charged particles as an example to illustrate building $\boldsymbol{PE}_1 = (PE_{1,1}, PE_{1,2}, \ldots, PE_{1,D})$ and $\boldsymbol{PG}_1 = (PG_{1,1}, PG_{1,2}, \ldots, PG_{1,D})$.

n_e elite particles are selected from the negatively charged subpopulation to construct the attractive sample $\boldsymbol{PE}_1$ through the electric field force-based comprehensive learning strategy, directing the particles to a better area. The n_e elite particles constitute the set S_3. For the currently updated positively charged particle $\boldsymbol{X}_i$, we calculate the electric field force f_{ij} of each elite particle $\boldsymbol{X}_j$ in S_3 to $\boldsymbol{X}_i$, where j represents the serial number of the elite particle in S_3.

$$PE_{1,d} = \frac{1}{n_e} \sum_{\boldsymbol{X}_j \in S_3} pbest_{j,d} \cdot f_{ij}, \tag{3.7}$$

where $d = 1, 2, \ldots, D$, $\boldsymbol{PE}_{1,d}$ is the value of $\boldsymbol{PE}_1$ in the d^{th} dimension, $pbest_{j,d}$ represents d^{th} dimension of the best historical of the elite particle $\boldsymbol{X}_j$ in S_3. f_{ij} is the electric field force of $\boldsymbol{X}_j$ to $\boldsymbol{X}_i$.

Similarly, n_g general particles are selected from the positively charged subpopulation to construct the repulsive sample $\boldsymbol{PG}_1$ through the electric field force-based comprehensive learning strategy. The n_g general particles constitute the set S_2. For the currently updated positively charged particle $\boldsymbol{X}_i$, we calculate the electric field force f_{ik} of each elite particle $\boldsymbol{X}_k$ in S_2 to $\boldsymbol{X}_i$, where K represents the serial number of the general particle in S_2.

$$PG_{1,d} = \frac{1}{n_g} \sum_{\boldsymbol{X}_k \in S_2} pbest_{k,d} \cdot f_{ik}, \tag{3.8}$$

where $d = 1, 2, \ldots, D$, $\boldsymbol{PG}_{1,d}$ is the value of $\boldsymbol{PG}_1$ in the d^{th} dimension, $pbest_{k,d}$ represents d^{th} dimension of the best historical of the general particle $\boldsymbol{X}_k$ in S_2. f_{ik} is the electric field force of the $\boldsymbol{X}_k$ to $\boldsymbol{X}_i$.

Algorithm 1 lists the pseudo code of EFCLS, taking positively charged particles as an example. And it is worth noting that when we construct $\boldsymbol{PE}_2$, $\boldsymbol{X}_i$ comes from S_1. When we build $\boldsymbol{PG}_2$, $\boldsymbol{X}_k$ comes from S_4.

Algorithm 1 electric field force-based comprehensive learning strategy

```
Sort positively charged particles and negatively charged particles respectively by fitness value
for i = 1 → n_1 do
    for X_j in S_3 do
        Calculate the distance d_ij between X_i and X_j by Eq (3.6)
        Calculate the electric field force f_ij of X_j to X_i by Eq (3.5)
    end for
    Construct the attractive sample PE_1 by Eq (3.7)
    for X_k in S_2 do
        Calculate the distance d_ik between X_i and X_k by Eq (3.6)
        Calculate the electric field force f_ik of X_k to X_i by Eq (3.5)
    end for
    Construct the repulsive sample PG_1 by Eq (3.8)
end for
```

In MSPSO, once the particle is updated, the selection of elite particles and general particles will change according to the fitness value. To enhance the diversity of the population, we adaptively adjust the weight coefficients $\alpha_1, \beta_1, \alpha_2$ and β_2 by the historical information of the particles.

We assign a weight coefficient w_i to each particle and initialize w_i by a random real number generated by $\mathcal{N}\left(0.1, \sigma^2\right)$. Besides, real numbers randomly generated from Gaussian distribution $\mathcal{N}\left(\mu_1, \sigma^2\right)$, $\mathcal{N}\left(\mu_2, \sigma^2\right)$, $\mathcal{N}\left(\mu_3, \sigma^2\right)$ and $\mathcal{N}\left(\mu_4, \sigma^2\right)$ are assigned to each α_2, β_1, α_1 and β_2. μ_1, μ_2, μ_3 and μ_4 are updated according to the historical knowledge of elite particles and general particles of the subpopulations as Equation (3.9).

$$\mu_k = \lambda \cdot \frac{1}{|S_k|} \sum_{X_i \in S_k} w_i, \quad k = 1, 2, 3, 4, \tag{3.9}$$

where λ represents the degree to which μ_k $(k = 1, 2, 3, 4)$ learns from the historical knowledge of the corresponding subpopulation. S_k $(k = 1, 2, 3, 4)$ represents the four sets that we have filtered through the tournament mechanism. $|S_k|$ denotes the number of particles in S_k. We update w_i by an actual random number generating by its corresponding $\mathcal{N}\left(\mu_k, \sigma^2\right)$. Based on experience, we set σ to 0.1.

3.3. Segment-based weighted learning strategy

Although the attraction and repulsion in the velocity update equation can disperse the particles and facilitate exploration, the particles may fall into the locally optimal values of the two subpopulations. Therefore, to reduce the risk of falling into the local optimum, we construct a global learning sample $\boldsymbol{GM}$ by SWLS to guide particle motion. Algorithm 2 lists the pseudo-code of SWLS.

Algorithm 2 segment-based weighted learning strategy

```
Initialize the number of segments of particle dimensions m = D/10
Sort particles in ascending order by fitness value, and select some top particles to form an elite population
for j = 1 → m do
    E randomly select k particles from the elite population
    for d = j → j + 10 do
        Construct the global elite sample GM_d by Eq (3.10)
    end for
end for
```

Firstly, The dimension of the global learning sample is divided into m segments evenly, and we let each segment follow the elite particles of the entire population to construct the learning sample. Specifically, we sort all particles in ascending order through the tournament mechanism and select a part of top particles to form an elite population. For each segment, we select k particles randomly from the elite population. Then we use a weighting strategy to construct the global learning sample $\boldsymbol{GM} = (GM_1, GM_2, \ldots, GM_D)$.

$$GM_d = \sum_{i=1}^{k} \frac{f(E_i)}{\sum_{j=1}^{k} f\left(E_j\right) + \gamma} E_{i,d}, \tag{3.10}$$

where $d = 1, 2, \ldots, D$, GM_d represents the value of $\boldsymbol{GM}$ in the d^{th} dimension. E is the set of selected k elite particles, and E_j denotes the j^{th} elite particle $\boldsymbol{X}_j$ in E. $E_{i,d}$ is the value of E_i in the d^{th} dimension. The fitness function is denoted by f, and γ is a small positive number that prevents the denominator from being zero.

In this way, the particles can obtain more comprehensive information, which is conducive to global exploration, increasing the probability of finding the global best value. Based on SWLS, we calculate the electric field force of the newly constructed sample on the currently updated particles, which represents the influence of the newly created global learning example $\boldsymbol{GM}$ on the particles in the electric field.

3.4. Algorithm framework

By integrating the above strategies, Algorithm 3 shows the implementation process of MSPSO.

Algorithm 3 MSPSO algorithm

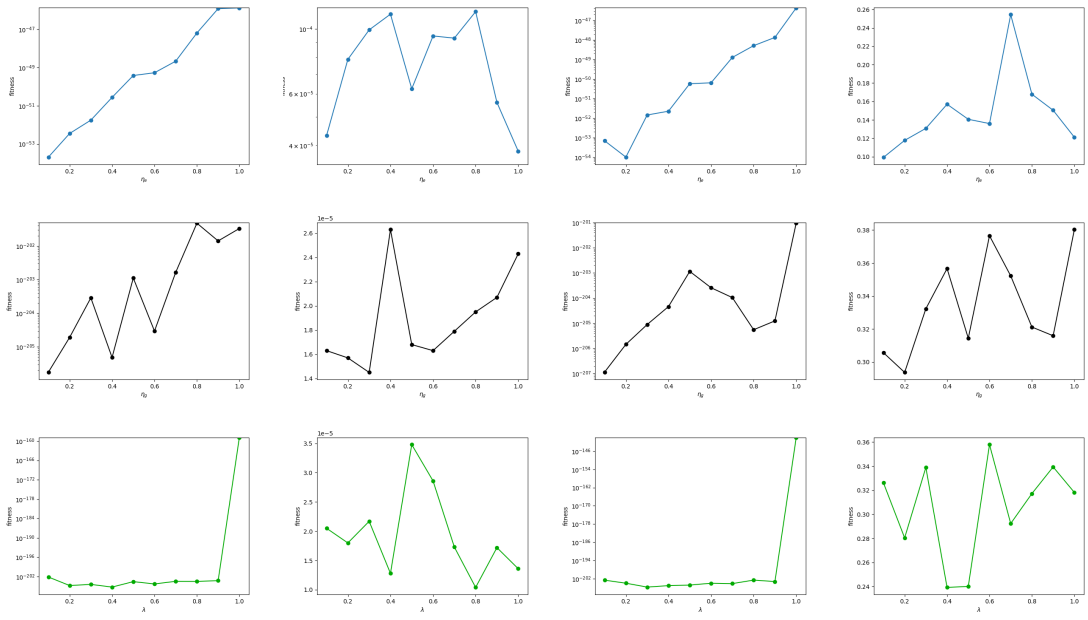
/*Initialization*/
Determine the size of two subpopulations: $n_1 = n_2 = n/2$, the first n_1 particles are positively charged particles, and the remaining n_2 particles are negatively charged particles
for $i = 1 \rightarrow n$ **do**
    Randomly initialize $\boldsymbol{V}_i$ and $\boldsymbol{X}_i$
    Evaluate $f(\boldsymbol{X}_i)$; $\boldsymbol{pbest}_i = \boldsymbol{X}_i$
end for
Initialize all particles' weight coefficient w_i, set t=1
/*Main Loop*/
while t<$Maxiter$ **do**
    Compute ω^t by Eq (3.4)
    Compute μ_1, μ_2, μ_3 and μ_4 by Eq (3.9)
    Update $\alpha_1, \beta_1, \alpha_2, \beta_2$
    Construct the global elite sample $\boldsymbol{GM}^t$ by SWLS
    for $i = 1 \rightarrow n_1$ **do**
        Construct $\boldsymbol{PE}_1^t$ and $\boldsymbol{PG}_1^t$ by EFCLS
        Compute the electric field force f_{ig} of $\boldsymbol{GM}^t$ to $\boldsymbol{X}_i$ by Eq (3.5)
        Update $\boldsymbol{V}_i$ by (3.1)
    end for
    for $i = (n_1 + 1) \rightarrow (n_1 + n_2)$ **do**
        Construct $\boldsymbol{PE}_2^t$ and $\boldsymbol{PG}_2^t$ by EFCLS
        Compute the electric field force f_{ig} of $\boldsymbol{GM}^t$ to $\boldsymbol{X}_i$ by Eq (3.5)
        Update $\boldsymbol{V}_i$ by (3.2)
    end for
    Update all particles' $\boldsymbol{X}_i$ by (3.3)
    Update all particles' weight coefficient w_i and $\boldsymbol{pbest}_i$
    $t = t + 1$
end while

4. Performance of proposed strategies

In this section, we conducted experiments to evaluate the performance of MSPSO. Firstly, the performance of the MSPSO algorithm optimizes with parameter adjustment. Then the diversity of the algorithm search space is analyzed through experiments, and the effectiveness of the new proposed strategies is verified.

4.1. Parameter sensitivity

We carry out experiments to analyze the influence of the parameters n_e, n_g and λ involved in MSPSO on the algorithm's performance. We set the number of particles in the positively charged subpopulation to be the same as the number of particles in the negatively charged subpopulation, $n_1 = n_2 = n/2$. For simplicity, we set $n_e = \eta_e \cdot n/2$, $n_g = \eta_g \cdot n/2$. The experimental setting is that the population size n=60 and the maximum number of function evaluation $MaxFes$=1000D. We change the parameter from 0 to 1, and the step amplitude is 0.1. One unimodal function (Schwefel's P2.21) and three multimodal functions (Quartic, Alpine, Penalized 1) are employed to evaluate the results. The evaluation standard is the mean value after 30 independent runs, and the tests are performed on the problem of 50D. Note that to analyze the influence of each parameter on the performance of the algorithm separately, we set a default value for each parameter, namely $\eta_e = 0.5$, $\eta_g = 0.5$, $\lambda = 0.5$.



(a)Schwefel's P2.21 (b) Quartic (c) Alpine (d) Penalized 1

Figure 1. Sensitivity of η_e, η_g and λ.

(1) Sensitivity of η_e

η_e represents the proportion of elite particles that are attractive to the current update particle. And its function is to guide particles to a better area. A smaller η_e will increase the convergence speed of the algorithm, but it may also cause the particles to ignore the information of other better particles.

Although a larger η_e can obtain more information from the oppositely charged particles, it may cause the particles to evolve in a worse direction. From the first row of Figure 1, we can see that a smaller η_e is more conducive to improving the accuracy, so we set η_e to 0.1.

(2) Sensitivity of η_g

η_g represents the proportion of general particles that repel the current update particle. And its function is to disperse the particle swarm and drive the poorer particles to explore other directions, increasing the ability of global exploration. A smaller η_g may increase the convergence speed of the algorithm but may cause the particles to gather too much and weaken the global exploring ability. A larger η_g would make the particles more dispersed but may change the direction of movement of better particles. According to the experimental results in the second row of Figure 1, η_g is better in the range of $[0.1, 0.2]$, and we set η_g to 0.1.

(3) Sensitivity of λ

λ represents the degree to which the weight coefficients of attraction and repulsion learn from the historical information of elite particles and general particles. If λ is small, the knowledge of elite particles or general particles has less influence on the adjustment of μ_k, which may cause some useful historical information of elite particles or general particles to be ignored. The greater the λ, the greater the dependence of μ_k on elite or general experience. Experimental results based on the third row of Figure 1, $\lambda = 0.4$ is adopted in this work.

According to the above experiment and analysis, we will set η_e, η_g and λ to 0.1, 0.1 and 0.4 in the subsequent experiments.

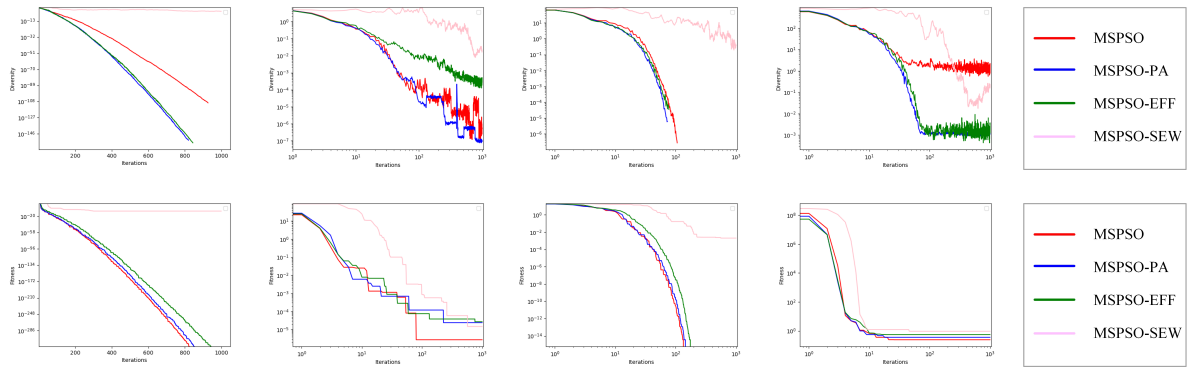
4.2. Diversity analysis

MSPSO introduces electric field into PSO and proposes two learning strategies to improve the performance of the algorithm, namely, an electric field force-based comprehensive learning strategy and a segment-based weighted learning strategy. Additionally, we also adopted a parameter adaptation strategy to update the attractive weight coefficient and the repulsive weight coefficient. To verify the effectiveness of the proposed strategies, we selected four representative test functions for the population diversity experiment: a unimodal function (Different power) and three multimodal functions (Quartic, Ackley, Penalized 1). The diversity experiment settings are as follows: the overall size n=60, the problem dimension D=50, and the maximum number of iterations $MaxFes$=1000D. Population diversity is defined according to references [30].

$$\text{diversity}(n) = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\sum_{j=1}^{D}\left(x_{i,j} - \bar{x}_j\right)^2}, \tag{4.1}$$

where $x_{i,j}$ represents the position of the i^{th} particle in the j^{th} dimension, and $\bar{x}_j$ represents the average value of all positions of particles in j^{th} dimension.

MSPSO-EFF, MSPSO-PA, and MSPSO-SEW represent the algorithm that the electric field force, parameter adaptation, and segment-based weighted learning strategy are removed from MSPSO. The experimental comparison results of the diversity are shown in the first row of Figure 2. In addition, to further analyze the different effects of these strategies on the algorithm, we also experimented on the accuracy, as shown in the second row of Figure 2.



(a)Different power (b) Quartic (c) Ackley (d) Penalized 1

Figure 2. Comparison results on the new strategies.

As shown in Figure 2, MSPSO-SEW delivers the best performance in terms of the diversity on the four test functions. However, MSPSO-SEW sacrifices the convergence accuracy of the algorithm and cannot provide the best results in the accuracy of the solution. On the contrary, the loss of diversity of MSPSO-PA is faster than other algorithms. Although MSPSO does not obtain the best performance in terms of diversity, it is always better than MSPSO-EFF except for the Quartic function, which shows that the electric field force is beneficial to enhance the diversity of the population. Additionally, compared with MSPSO-SEW, MSPSO-EFF, and MSPSO-PA, MSPSO has fewer advantages on convergence speed but always gets the highest accuracy.

We can get some preliminary conclusions about the newly introduced strategies based on the above experimental results. Firstly, from the diversity, electric field force and the adaptive update of weight coefficients both play an essential role. Secondly, the three strategies are all beneficial to improve the accuracy of the algorithm.

5. Experimental results and analysis

5.1. Benchmark functions and baseline algorithms

To evaluate the performance of the MSPSO algorithm from multiple aspects, we selected sixteen widely used benchmark functions and the detailed information is shown in Table 1. The third and fourth columns of Table 1 respectively represent the search range and global best value of the benchmark function. The benchmark functions include two categories: unimodal function (f_1-f_8) and multimodal functions (f_9-f_{16}). Among the multimodal functions, f_{11}-f_{16} are relatively more complicated than f_9-f_{10}.

We selected eight well-known evolutionary algorithms based on particle swarm optimization algorithm to verify the superiority of the MSPSO algorithm, including PSO, DMSPSO, CLPSO, OBL-CPSO, CLPSO-LOT, MPCPSO, XPSO, and MIONPSO. And the specific information about the parameter settings of each algorithm is shown in Table 2. Note that each algorithm's parameter settings and experimental settings are the same as those of the corresponding original text for the fairness of comparison.

Table 1. Benchmark functions.

Name	Benchmark Functions	Search Range	f_m
Sphere	$f_1(x) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$	0
Schwefel's P2.21	$f_2(x) = \max(\lvert x_i \rvert), i = 1, 2, \cdots, D$	$[-100, 100]^D$	0
Schwefel's P2.22	$f_3(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	$[-10, 10]^D$	0
Different power	$f_4(x) = \sum_{i=1}^{D} \lvert x_i \rvert^{i+1}$	$[-100, 100]^D$	0
Bent cigar	$f_5(x) = x_1^2 + 10^6 \sum_{i=2}^{D} x_i^2$	$[-100, 100]^D$	0
Discus	$f_6(x) = 10^6 x_1^2 + \sum_{i=2}^{D} x_i^2$	$[-100, 100]^D$	0
Zakharov	$f_7(x) = \sum_{i=1}^{D} x_i^2 + (\sum_{i=1}^{D} 0.5 x_i^2)^2 + (\sum_{i=1}^{D} 0.5 x_i)^4$	$[-15, 10]^D$	0
Rosenbrock	$f_8(x) = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-30, 30]^D$	0
Quartic	$f_9(x) = \sum_{i=1}^{D} i x_i^4 + random[0, 1]$	$[-1.28, 1.28]^D$	0
Alpine	$f_{10}(x) = \sum_{i=1}^{D} \lvert x_i \sin x_i + 0.1 x_i \rvert$	$[-10, 10]^D$	0
Schwefel's P2.26	$f_{11}(x) = \sum_{i=1}^{D} -(x_i \sin(\sqrt{\lvert x_i \rvert}))$	$[-500, 500]^D$	-12569.5
Rastrigin	$f_{12}(x) = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$	$[-5.12, 5.12]^D$	0
Ackley	$f_{13}(x) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - \exp(\sum_{i=1}^{D} \cos 2\pi x_i) + 20 + e$	$[-32, 32]^D$	0
Griewank	$f_{14}(x) = 1/4000 \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} x_i / \sqrt{i} + 1$	$[-600, 600]^D$	0
Penalized 1	$f_{15}(x) = \frac{\pi}{D}\left\{10\sin^2(\pi y_i) + \sum_{i=1}^{D-1}(y_i - 1)^2 \cdot \left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_D - 1)^2\right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$, where $y_i = 1 + 0.25(x_i + 1)$, $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, x_i > a \\ 0, -a \le x_i \le a \\ k(-x_i - a)^m, x_i < -a \end{cases}$	$[-50, 50]^D$	0
Penalized 2	$f_{16}(x) = 0.1\left\{\sin^2 \pi 3 y_i + \sum_{i=1}^{D}(y_i - 1)^2 \cdot \left[1 + \sin^2(3\pi y_{i+1})\right] + (x_n - 1)^2\left[1 + \sin^2(2\pi y_D)\right]\right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$, where $y_i = 1 + 0.25(x_i + 1)$, u as $Penalized1$	$[-50, 50]^D$	0

Table 2. The details of baseline algorithms.

Algorithm	Year	Parameter settings
PSO [44]	1998	$\omega = 0.7298, c_1 = c_2 = 1.49445$
DMSPSO [29]	2005	$\omega = 0.7298, c_1 = c_2 = 1.49445, R = 10, L = 100$
CLPSO [41]	2006	$\omega = [0.4, 0.9], c = 1.49445, m = 7$
OBL-CPSO [42]	2016	$\omega = 0.2 \sim 0.7$
CLPSO-LOT [34]	2019	$\omega = [0.4, 0.9], c = 1.49445, m = 7, g = 30$
MPCPSO [36]	2020	$\omega = 0.7298, c_1 = c_2 = 1.49445, F = 0.5, t = 0.5$
XPSO [37]	2020	$n = 50, \omega = [0, 4, 0.9], \eta = 0.2, stag_{max} = 5, p = 0.5$
MIONPSO [35]	2021	$n = 200, c = 1.49445, SN = 50, R = 10, \alpha = 0.2, \beta = 0.04$

5.2. Comparison on solutions accuracy

The experimental settings are as follows: the population size n=60, the maximum number of function evaluation $MaxFes$=1000D, each algorithm runs 30 times independently. We adopt the evaluation indicators commonly used in the PSO algorithm: average and standard deviation. Additionally, to show the performance characteristics of each algorithm on different dimensions, we conducted experiments in 30D, 50D, and 100D.

Table 3, Table 4, and Table 5 show the results of different algorithms on different dimensions and benchmark functions, including the running time of the CPU.

Table 3. Comparison results of all algorithms on benchmark functions (30D).

		MSPSO	MIONPSO	XPSO	MPCPSO	CLPSO-LOT	OBL-CPSO	CLPSO	DMSPSO	PSO
f_1	Mean	**0.00E+00**	**0.00E+00**	1.88E-14	1.30E-62	1.24E+00	9.93E-32	3.17E+01	9.10E+00	3.09E-20
	Std	**0.00E+00**	**0.00E+00**	7.08E-14	4.36E-62	6.63E+00	3.65E-31	1.50E+01	3.64E+00	4.01E-20
	Time(s)	11.03	18.94	10.89	42.03	53.77	1.48	52.64	2.46	0.38
f_2	Mean	4.10E-198	**0.00E+00**	3.20E-04	1.63E-32	3.41E+01	2.89E-16	4.29E+01	2.86E+00	1.32E-01
	Std	0.00E+00	**0.00E+00**	4.96E-04	2.61E-32	7.07E+00	1.48E-15	4.96E+00	8.55E-01	1.00E-01
	Time(s)	10.99	19.52	10.85	40.85	55.26	1.44	54.17	2.50	0.36
f_3	Mean	1.32E-197	**0.00E+00**	2.91E-07	3.42E-33	9.17E-03	9.06E-17	6.58E-01	1.25E+00	3.52E-08
	Std	0.00E+00	**0.00E+00**	6.53E-07	6.12E-33	6.11E-03	2.84E-16	3.03E-01	4.06E-01	6.29E-08
	Time(s)	11.18	19.94	10.91	40.46	53.41	1.79	52.69	2.70	0.54
f_4	Mean	**0.00E+00**	1.28E+21	2.79E-07	2.78E+24	9.99E+28	3.36E-47	5.87E+27	1.57E+07	3.33E+12
	Std	**0.00E+00**	6.75E+21	1.06E-06	1.23E+25	5.33E+29	1.46E-46	1.73E+28	5.04E+07	1.80E+13
	Time(s)	14.40	41.72	14.60	45.81	56.02	8.19	58.02	6.74	3.48
f_5	Mean	**0.00E+00**	**0.00E+00**	3.54E-02	1.10E+09	3.46E+04	2.69E-26	2.62E+07	7.56E+06	3.33E+02
	Std	**0.00E+00**	**0.00E+00**	1.91E-01	2.02E+08	7.82E+04	1.13E-25	1.56E+07	2.89E+06	1.80E+03
	Time(s)	11.03	18.19	10.89	41.98	54.36	1.66	54.42	2.64	0.46
f_6	Mean	**0.00E+00**	1.52E+03	1.08E-03	4.18E-32	5.26E-02	9.92E-31	2.47E+02	2.66E+01	9.54E-18
	Std	**0.00E+00**	9.33E+02	5.12E-03	2.23E-31	1.30E-01	5.09E-30	2.36E+02	9.28E+00	3.84E-17
	Time(s)	11.00	17.77	11.08	41.16	55.72	1.76	52.92	2.70	0.47
f_7	Mean	**0.00E+00**	1.17E+02	6.25E-17	1.25E-64	4.55E+00	3.52E+02	6.20E+00	3.70E-01	3.44E+00
	Std	**0.00E+00**	3.18E+01	1.78E-16	5.03E-64	3.85E+00	7.18E+01	2.23E+00	1.21E-01	1.85E+01
	Time(s)	11.80	21.99	11.57	41.90	59.73	4.67	53.98	3.40	1.12
f_8	Mean	2.88E+01	2.90E+01	**2.51E+01**	2.71E+01	9.60E+02	2.81E+01	1.94E+04	2.50E+02	3.71E+01
	Std	2.98E-02	1.16E-03	**4.27E-01**	2.05E-01	1.54E+03	3.70E-01	9.59E+03	1.18E+02	3.23E+01
	Times	16.45	53.71	16.44	47.26	60.28	12.17	58.41	9.29	5.67
f_9	Mean	**1.47E-05**	1.63E-02	3.12E-03	1.45E-03	9.37E-02	1.92E-05	1.96E-01	9.06E-03	7.93E-03
	Std	**1.46E-05**	9.37E-03	1.33E-03	1.49E-03	5.40E-02	2.11E-05	5.96E-02	2.91E-03	2.75E-03
	Time(s)	11.77	23.35	12.00	42.99	52.05	3.38	55.42	3.62	1.33
f_{10}	Mean	4.98E-198	**0.00E+00**	2.11E-04	1.09E-33	2.95E+00	1.12E-17	1.27E+00	1.40E-01	5.18E-08
	Std	0.00E+00	**0.00E+00**	3.19E-04	3.39E-33	9.23E-01	2.80E-17	4.15E-01	8.01E-02	1.76E-07
	Time(s)	11.04	18.12	10.96	41.13	51.79	1.73	55.74	2.60	0.53
f_{11}	Mean	-2.17E+03	-5.98E+03	-7.61E+03	**-1.25E+04**	-8.09E+03	-3.18E+03	-8.15E+03	-7.18E+03	-6.75E+03
	Std	3.31E+02	5.36E+02	2.25E+03	**6.57E+01**	3.08E+02	9.77E+02	3.31E+02	8.10E+02	8.68E+02
	Time(s)	11.07	17.81	11.44	41.89	51.04	1.81	57.05	2.58	0.50
f_{12}	Mean	**0.00E+00**	**0.00E+00**	1.37E-01	4.74E-16	1.69E+01	**0.00E+00**	4.33E+01	1.90E+01	4.63E+01
	Std	**0.00E+00**	**0.00E+00**	7.39E-01	1.65E-15	4.33E+00	**0.00E+00**	9.11E+00	3.72E+00	1.24E+01
	Time(s)	10.97	18.66	11.01	47.17	49.79	2.04	54.65	2.70	0.63
f_{13}	Mean	**0.00E+00**	7.11E-15	3.41E-10	7.58E-15	2.66E+00	2.37E-16	3.46E+00	2.18E+00	4.98E-01
	Std	**0.00E+00**	0.00E+00	5.85E-10	6.27E-15	1.51E+00	8.86E-16	6.09E-01	3.61E-01	6.95E-01
	Time(s)	11.50	21.34	11.39	51.93	52.98	2.83	52.25	3.25	1.01
f_{14}	Mean	**0.00E+00**	**0.00E+00**	5.36E-02	**0.00E+00**	1.58E-01	**0.00E+00**	1.60E+00	1.09E+00	1.04E-02
	Std	**0.00E+00**	**0.00E+00**	1.61E-01	**0.00E+00**	1.91E-01	**0.00E+00**	3.27E-01	4.40E-02	2.16E-02
	Time(s)	14.88	44.28	15.52	49.33	56.73	10.08	55.81	7.33	4.30
f_{15}	Mean	3.24E-01	1.55E+00	**1.84E-07**	3.38E-04	3.33E+00	2.02E-01	4.23E+01	2.25E-01	5.53E-02
	Std	1.06E-01	6.45E-02	**5.85E-07**	1.02E-04	5.09E+00	4.91E-02	7.02E+01	1.95E-01	1.44E-01
	Time(s)	16.98	60.12	17.03	49.10	60.03	14.09	58.79	10.14	6.24
f_{16}	Mean	2.13E+00	5.51E+00	**7.35E-04**	1.17E-01	5.95E-01	2.59E+00	4.22E+00	9.67E-01	3.66E-03
	Std	4.96E-01	6.43E-01	**2.74E-03**	4.68E-01	1.14E+00	3.19E-01	2.58E+00	3.81E-01	1.03E-02
	Time(s)	26.06	123.38	27.23	59.60	70.05	33.51	71.04	21.66	15.10

(1) Unimodal functions (f_1-f_8)

It can be seen from Table 3 that when MSPSO solves the problem of 30D, it shows a better mean value and standard deviation on most unimodal functions. Except for the functions f_2, f_3 and f_8, MSPSO can find the optimal value of other unimodal functions, and it also has strong stability. On functions f_2 and f_3, the accuracy of MSPSO is slightly lower than MIONPSO, but better than the other

Table 4. Comparison results of all algorithms on benchmark functions (50D).

		MSPSO	MIONPSO	XPSO	MPCPSO	CLPSO-LOT	OBL-CPSO	CLPSO	DMSPSO	PSO
f_1	Mean	**0.00E+00**	**0.00E+00**	2.13E-10	3.09E-64	1.94E-01	2.80E-32	9.27E-01	1.02E+02	9.59E-08
	Std	**0.00E+00**	**0.00E+00**	9.02E-10	8.52E-64	1.76E-01	9.54E-32	4.91E-01	3.53E+01	2.58E-07
	Times	15.22	18.86	17.40	67.17	107.23	1.53	85.96	3.50	0.44
f_2	Mean	1.99E-198	**0.00E+00**	1.21E-03	4.27E-33	3.42E+01	1.30E-17	4.64E+01	7.86E+00	5.30E+00
	Std	0.00E+00	**0.00E+00**	2.25E-03	6.15E-33	5.94E+00	5.65E-17	2.70E+00	1.39E+00	9.98E-01
	Times	14.98	19.04	17.70	67.49	107.36	1.48	85.18	2.44	0.41
f_3	Mean	2.20E-198	**0.00E+00**	1.59E-04	3.16E-34	1.11E-01	1.86E-15	2.14E-01	6.27E+00	6.91E-03
	Std	0.00E+00	**0.00E+00**	3.50E-04	4.54E-34	3.61E-02	7.38E-15	3.75E-02	1.14E+00	9.49E-03
	Times	15.07	20.36	17.48	67.52	106.16	1.89	87.51	2.74	0.60
f_4	Mean	**0.00E+00**	4.95E+36	3.16E+05	2.54E+27	3.44E+28	7.23E-43	1.65E+33	6.25E-01	3.33E+28
	Std	**0.00E+00**	2.63E+37	1.69E+06	1.32E+28	1.85E+29	3.90E-42	8.40E+33	1.95E+00	1.80E+29
	Times	21.42	56.22	24.02	73.94	124.31	13.57	90.84	9.67	5.84
f_5	Mean	**0.00E+00**	**0.00E+00**	1.68E+01	1.14E+09	1.71E+05	1.74E-26	8.11E+05	9.50E+07	2.02E-01
	Std	**0.00E+00**	**0.00E+00**	8.95E+01	2.68E+08	1.32E+05	8.16E-26	3.15E+05	2.98E+07	4.42E-01
	Times	15.69	19.74	17.44	67.82	105.71	1.70	84.22	2.69	0.52
f_6	Mean	**0.00E+00**	3.87E+03	7.66E-01	2.43E-29	1.38E-01	3.07E-27	2.03E+00	2.86E+02	3.40E-06
	Std	**0.00E+00**	2.02E+03	1.37E+00	1.31E-28	5.11E-02	1.46E-26	1.00E+00	8.78E+01	1.64E-05
	Times	15.64	19.50	17.84	66.37	105.17	1.71	84.58	2.62	0.53
f_7	Mean	**0.00E+00**	3.06E+02	4.13E-10	1.35E-65	1.42E+00	1.04E+03	2.23E+00	6.76E+00	1.51E+02
	Std	**0.00E+00**	7.55E+01	1.09E-09	5.17E-65	5.57E-01	2.43E+02	5.91E-01	1.74E+00	1.90E+02
	Times	15.64	23.62	18.13	67.09	107.24	3.04	87.26	3.61	1.17
f_8	Mean	4.87E+01	4.90E+01	4.55E+01	**2.72E+01**	3.50E+02	4.82E+01	1.02E+03	2.45E+03	1.00E+02
	Std	4.61E-02	3.76E-04	4.02E-01	**2.35E-01**	1.33E+02	3.37E-01	4.92E+02	1.05E+03	4.32E+01
	Times	24.22	80.51	27.19	76.74	118.10	22.21	100.35	14.25	9.66
f_9	Mean	**1.60E-05**	2.09E-02	5.70E-03	1.22E-03	1.32E-01	1.71E-05	2.22E-01	3.88E-02	2.46E-02
	Std	**1.45E-05**	1.35E-02	2.97E-03	1.08E-03	2.77E-02	2.03E-05	4.53E-02	1.25E-02	1.07E-02
	Times	16.58	30.26	19.62	69.16	104.76	5.01	90.74	4.67	2.20
f_{10}	Mean	3.88E-199	**0.00E+00**	1.71E-03	2.00E-34	6.15E+00	3.80E-17	2.99E+00	1.37E+00	4.36E-03
	Std	0.00E+00	**0.00E+00**	1.99E-03	4.12E-34	2.99E+00	7.95E-17	8.70E-01	4.74E-01	8.39E-03
	Times	15.05	19.66	17.52	65.92	104.91	1.79	87.64	2.64	0.59
f_{11}	Mean	-3.24E+03	-7.98E+03	-8.84E+03	-1.25E+04	-1.20E+04	**-7.40E+05**	-1.22E+04	-1.11E+04	-1.10E+04
	Std	6.01E+02	1.07E+03	3.54E+03	2.12E+02	4.60E+02	**3.55E+06**	3.58E+02	1.20E+03	9.82E+02
	Times	15.19	19.53	18.27	67.04	102.54	1.78	84.37	2.63	0.57
f_{12}	Mean	**0.00E+00**	**0.00E+00**	7.32E-09	4.14E-16	2.89E+01	**0.00E+00**	9.66E+01	5.73E+01	8.01E+01
	Std	**0.00E+00**	**0.00E+00**	1.47E-08	1.35E-15	8.48E+00	**0.00E+00**	1.07E+01	1.03E+01	1.52E+01
	Times	15.12	20.34	17.57	71.85	99.92	2.02	85.52	2.80	0.70
f_{13}	Mean	**0.00E+00**	7.11E-15	3.79E-06	7.46E-15	1.83E+00	1.18E-16	1.67E+00	3.74E+00	2.19E+00
	Std	**0.00E+00**	0.00E+00	7.64E-06	4.53E-15	1.55E+00	6.38E-16	3.20E-01	3.88E-01	8.50E-01
	Times	15.52	23.00	18.05	67.31	105.04	2.90	85.03	3.56	1.12
f_{14}	Mean	**0.00E+00**	**0.00E+00**	1.90E-01	2.59E-17	2.69E-01	**0.00E+00**	9.61E-01	1.97E+00	3.06E-02
	Std	**0.00E+00**	**0.00E+00**	2.03E-01	1.40E-16	1.36E-01	**0.00E+00**	9.62E-02	3.01E-01	4.43E-02
	Times	21.50	62.78	25.19	79.22	111.73	15.99	91.82	12.13	6.89
f_{15}	Mean	3.19E-01	1.40E+00	**7.33E-05**	3.22E-04	8.63E-02	3.42E-01	7.31E-01	1.73E+00	4.34E-01
	Std	8.57E-02	6.81E-02	**1.01E-04**	8.90E-05	1.68E-01	4.85E-02	5.25E-01	7.54E-01	7.55E-01
	Times	25.42	92.85	28.01	78.90	118.17	23.04	97.69	15.90	10.65
f_{16}	Mean	3.98E+00	9.34E+00	1.26E-02	2.85E-02	3.02E-01	4.96E+00	8.10E-01	1.07E+01	**9.52E-03**
	Std	8.36E-01	1.08E+00	1.37E-02	1.67E-02	3.60E-01	6.85E-01	2.71E-01	3.09E+00	**1.19E-02**
	Times	41.15	206.59	45.55	96.25	139.44	54.44	117.62	35.18	26.35

seven algorithms. All algorithms have similar performance on the function f_8 while XPSO performs best, and no optimal value is found. Besides, MSPSO also shows similar performance to the 30D problem with high dimensions (Table 4 and Table 5), almost finding the optimal value. And it won first

place five times both on 50D and 100D.

(2) Multimodal functions (f_9-f_{16})

The results in Table 3 indicate that for the eight multimodal functions, MSPSO can easily jump out of the local optimal values on functions f_{12}, f_{13} and f_{14}. None of the algorithms finds the optimal value on function f_9, but MSPSO achieves the highest accuracy. Although MSPSO does not reach the global best value on function f_{10}, it has a small difference from the MIONPSO, and is better than the other

Table 5. Comparison results of all algorithms on benchmark functions (100D).

		MSPSO	MIONPSO	XPSO	MPCPSO	CLPSO-LOT	OBL-CPSO	CLPSO	DMSPSO	PSO
f_1	Mean	**0.00E+00**	**0.00E+00**	5.8647E-07	5.46E-60	2.46E+02	1.40E-27	2.01E+03	1.31E+03	1.05E+01
	Std	**0.00E+00**	**0.00E+00**	1.6439E-06	2.94E-59	1.75E+02	7.53E-27	4.46E+02	2.02E+02	1.98E+01
	Times	24.91	23.63	33.54	128.20	205.09	1.59	174.23	2.63	0.56
f_2	Mean	7.10E-199	**0.00E+00**	1.62E-01	7.19E-32	5.26E+01	2.67E-16	6.95E+01	1.37E+01	2.04E+01
	Std	0.00E+00	**0.00E+00**	8.63E-01	1.35E-31	1.04E+01	1.35E-15	2.64E+00	1.35E+00	1.99E+00
	Times	25.55	22.96	33.52	127.58	207.50	1.67	174.20	2.69	0.53
f_3	Mean	1.93E-197	**0.00E+00**	4.24E-03	1.26E-32	9.39E+00	5.19E-15	2.66E+01	3.19E+01	4.61E+00
	Std	0.00E+00	**0.00E+00**	9.01E-03	4.07E-32	4.46E+00	1.32E-14	2.79E+00	3.12E+00	2.35E+00
	Times	25.87	24.14	33.66	127.61	204.80	1.98	174.48	2.92	0.73
f_4	Mean	**0.00E+00**	1.45E+83	9.47E+28	1.09E+103	2.54E+108	7.67E-51	3.78E+112	5.29E-05	3.33E+106
	Std	**0.00E+00**	7.79E+83	5.10E+29	5.86E+103	1.34E+109	2.64E-50	2.02E+113	1.72E-04	1.80E+107
	Times	37.89	96.83	47.00	141.20	231.71	26.96	193.48	17.58	11.51
f_5	Mean	**0.00E+00**	1.33E+01	4.06E+01	5.53E+09	2.08E+08	1.26E-24	2.03E+09	1.20E+09	1.45E+07
	Std	**0.00E+00**	7.18E+01	9.91E+01	3.96E+08	9.54E+07	5.88E-24	4.57E+08	2.24E+08	4.33E+07
	Times	25.73	23.58	33.59	129.36	205.84	1.72	173.17	2.73	0.63
f_6	Mean	**0.00E+00**	8.80E+03	7.93E+01	1.05E-46	2.08E+02	7.16E-27	3.10E+03	3.07E+03	3.02E+01
	Std	**0.00E+00**	3.82E+03	4.10E+01	5.65E-46	1.08E+02	3.85E-26	5.46E+02	5.33E+02	8.20E+01
	Times	25.70	23.67	34.05	128.51	206.93	1.76	172.96	2.79	0.65
f_7	Mean	**0.00E+00**	9.46E+02	4.56E-07	7.64E-65	2.39E+02	4.03E+03	3.40E+02	1.29E+02	9.21E+02
	Std	**0.00E+00**	2.63E+02	1.38E-06	1.97E-64	5.93E+01	8.93E+02	4.31E+01	3.05E+01	3.61E+02
	Times	26.05	28.02	33.92	130.75	209.66	3.09	174.00	3.73	1.33
f_8	Mean	9.85E+01	9.90E+01	**9.58E+01**	9.72E+01	4.01E+04	9.84E+01	8.15E+05	8.52E+04	6.66E+02
	Std	5.70E-02	1.76E-03	**6.41E-01**	2.34E-01	4.01E+04	2.71E-01	2.09E+05	2.63E+04	4.61E+02
	Times	43.94	149.08	53.27	149.15	227.81	40.76	196.59	27.57	19.68
f_9	Mean	**1.40E-05**	3.66E-02	1.64E-02	1.61E-03	8.55E-01	1.55E-05	2.50E+00	3.28E-01	1.56E-01
	Std	**1.30E-05**	3.14E-02	7.67E-03	1.75E-03	2.48E-01	1.86E-05	3.53E-01	7.57E-02	3.32E-02
	Times	28.56	48.02	37.46	134.62	204.35	9.25	177.02	7.24	4.30
f_{10}	Mean	**8.29E-198**	1.62E-03	1.14E-02	1.59E-33	2.73E+01	2.79E-15	3.51E+01	1.24E+01	1.30E+00
	Std	**0.00E+00**	8.42E-03	1.08E-02	2.93E-33	1.01E+01	1.45E-14	3.60E+00	2.05E+00	1.36E+00
	Times	25.19	25.23	33.77	128.85	203.92	1.86	169.35	2.83	0.74
f_{11}	Mean	-4.56E+03	-1.13E+04	-1.41E+04	**-4.18E+04**	-2.05E+04	-2.44E+04	-2.10E+04	-1.79E+04	-2.01E+04
	Std	8.04E+02	1.63E+03	7.81E+03	**1.77E+02**	7.73E+02	9.82E+04	5.67E+02	2.32E+03	1.91E+03
	Times	25.27	25.01	34.60	129.76	191.74	1.88	168.23	2.73	0.72
f_{12}	Mean	**0.00E+00**	**0.00E+00**	1.52E-04	6.51E-16	2.00E+02	**0.00E+00**	4.50E+02	2.54E+02	1.74E+02
	Std	**0.00E+00**	**0.00E+00**	6.59E-04	2.88E-15	3.86E+01	**0.00E+00**	2.28E+01	3.02E+01	2.19E+01
	Times	24.69	25.88	33.19	139.98	186.87	2.12	169.19	2.88	0.86
f_{13}	Mean	**0.00E+00**	7.11E-15	6.71E-05	9.12E-15	6.02E+00	**0.00E+00**	8.83E+00	6.29E+00	4.16E+00
	Std	**0.00E+00**	0.00E+00	2.03E-04	8.44E-15	2.53E+00	**0.00E+00**	5.39E-01	4.25E-01	1.07E+00
	Times	25.32	28.79	33.96	126.64	194.72	3.05	168.16	3.41	1.24
f_{14}	Mean	**0.00E+00**	**0.00E+00**	9.99E-01	4.07E-17	3.62E+00	**0.00E+00**	4.06E+01	1.27E+01	9.42E-01
	Std	**0.00E+00**	**0.00E+00**	2.11E-01	1.27E-16	1.74E+00	**0.00E+00**	9.52E+00	1.67E+00	4.95E-01
	Times	38.60	113.30	48.74	149.56	210.19	28.63	181.40	19.10	15.16
f_{15}	Mean	3.42E-01	1.30E+00	**7.31E-03**	3.67E-03	3.64E+02	5.46E-01	3.35E+03	8.09E+00	4.06E+00
	Std	9.73E-02	2.85E-02	**2.44E-03**	7.74E-04	1.81E+03	4.45E-02	3.79E+03	2.37E+00	1.46E+00
	Times	46.11	174.66	54.43	149.66	226.42	44.88	193.01	29.62	22.45
f_{16}	Mean	8.59E+00	1.98E+01	**8.47E-01**	1.53E+00	1.18E+02	1.01E+01	2.03E+02	9.71E+01	2.46E+01
	Std	1.53E+00	3.16E-02	**2.35E-01**	2.82E+00	8.32E+01	6.89E-01	3.89E+01	1.76E+01	1.84E+01
	Times	77.33	393.10	89.59	183.75	280.93	114.62	255.71	69.21	56.40

seven algorithms. However, MSPSO performs poorly on function f_{15} and f_{16}. The reason is that the global learning sample guides the movement of particles and makes their convergence speed faster,

which causes particles to fall into the local optimum. In addition, it can be seen from Table 4 and Table 5 that MSPSO has won first place 4 times and 5 times in the 50D and 100D multimodal functions, respectively. The results show that as the dimension increases, the performance of MSPSO is getting better and better.

To further illustrate the comprehensive performance of comparison algorithms, in terms of the accuracy of the solution, Table 6 performs a Friedman test of mean values with a significance level of $\alpha = 0.05$ on sixteen benchmark functions of all algorithms on different dimensions. The result shows that MSPSO takes first place on unimodal functions with an obvious advantage and ranks third on multimodal functions. Although MSPSO is slightly worse than MPCPSO and OBL-CPSO in handling multimodal problems, it still ranks first overall. MSPSO has the best overall performance on benchmark functions, followed by MPCPSO and OBL-CPSO.

Table 6. Friedman test of mean values on benchmark functions of different dimensions.

Average Rank	Algorithm	Ranking	Unimodal		Multimodal	
			Algorithm	Ranking	Algorithm	Ranking
1	MSPSO	2.51	MSPSO	1.73	MPCPSO	2.90
2	MPCPSO	3.30	MPCPSO	3.71	OBL-CPSO	3.21
3	OBL-CPSO	3.58	OBL-CPSO	3.96	MSPSO	3.29
4	XPSO	4.17	XPSO	4.13	XPSO	4.21
5	MINOPSO	4.40	MIONPSO	4.27	MIONPSO	4.52
6	PSO	5.60	PSO	5.63	PSO	5.58
7	CLPSO-LOT	6.67	CLPSO-LOT	6.75	CLPSO-LOT	6.58
8	DMSPSO	6.96	DMSPSO	6.83	DMSPSO	7.08
9	CLPSO	7.81	CLPSO	8.00	CLPSO	7.63

The above experimental results show that the performance of MSPSO is better than the other eight comparison algorithms. And it can effectively deal with high-dimensional unimodal and multimodal optimization problems, which have high robustness and convergence accuracy. The better performance of MSPSO benefits from the following advantages. Firstly, the interaction of attraction and repulsion disperses particles instead of gathering them at one point, enhances the capability of global exploration. The attraction between particles exchanges the information of the two subpopulations, making the particles co-evolve in a better direction. Secondly, the attractive sample and repulsive sample constructed by EFCLS make particles move purposefully and lead them to a better area. At the same time, the global learning sample created by SWLS enables particles to obtain more comprehensive information of the elite particles, which is conducive to global exploration. Thirdly, the adaptive adjustment of the weight coefficients enhances the diversity of the population, which is beneficial to the evolution of the population. Finally, the non-linear decrease of the inertia weight makes particles focus on global exploration in the early stage and local exploitation later, which improves the accuracy of convergence.

5.3. Statistical significance test of experimental results

Table 3, Table 4, and Table 5 record the CPU time consumed by each algorithm in sixteen test functions, and Table 7 shows the comprehensive ranking results of CPU time. The top three are PSO, DMSPSO, and OBL-CPSO. And MSPSO is close behind.

Figure 3 is the comprehensive performance diagram of MSPSO and other comparison algorithms. It is a comprehensive ranking diagram based on accuracy and CPU time on benchmark functions. The closer to the lower-left corner, the better the performance of the algorithm. From Figure 3, we can

Table 7. Ranking of CPU time consumption on different dimensions.

	MSPSO	MIONPSO	XPSO	MPCPSO	CLPSO-LOT	OBL-CPSO	CLPSO	DMSPSO	PSO
30D	4.44	6.44	4.44	6.81	8.44	2.50	8.31	2.63	1.00
50D	3.94	6.31	4.94	6.81	8.94	2.50	7.94	2.63	1.00
100D	4.31	5.25	5.56	6.88	8.94	2.50	7.94	2.63	1.00
Ave rank	4.23	6.00	4.98	6.83	8.77	2.50	8.06	2.63	1.00
Final rank	4	6	5	7	9	3	8	2	1

see that MSPSO and OBL-CPSO are the best among all algorithms and show similar performance. Although the accuracy of MSPSO is higher than that of OBL-CPSO, the CPU takes longer. This result indicates that MSPSO has a higher solution accuracy and performs well in algorithm complexity.

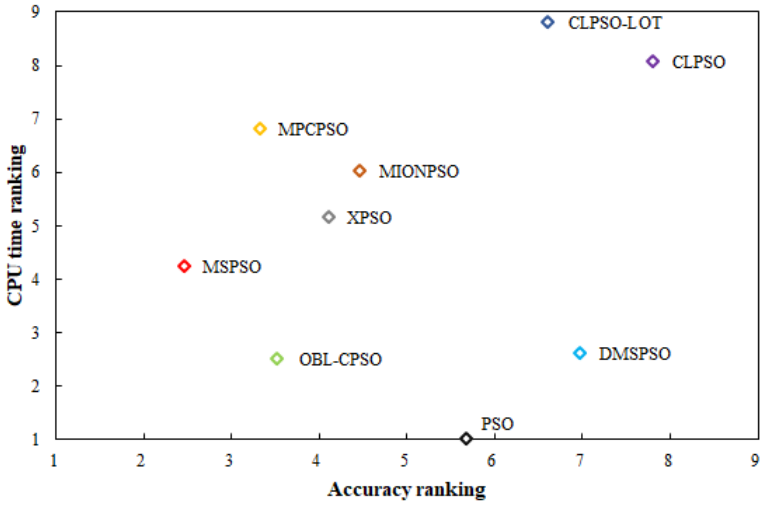


Figure 3. Comprehensive performance of all algorithms.

To further prove the superiority of the performance of the proposed MSPSO, a Nemenyi test was performed based on the Friedman test, shown in Figure 4. CD denotes the critical difference. It can be seen from Figure 4 that the performance of the MSPSO algorithm is significantly higher than that of PSO, CLPSO-LOT, DMSPSO, and CLPSO. Compared with MPCPSO, OBL-CPSO, XPSO, and MIONPSO, MSPSO has the best performance on the Friedman test, but there is no significant difference.

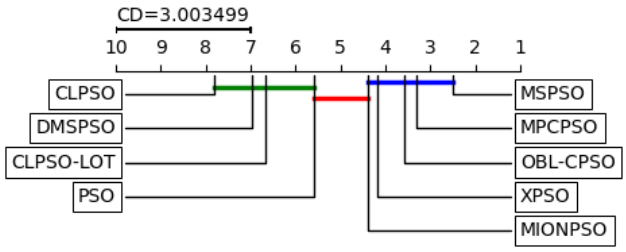


Figure 4. Friedman rankings-based Nemenyi test result.

In addition, we use the Wilcoxon signed-rank test with a significant level of α=0.05 to represent the

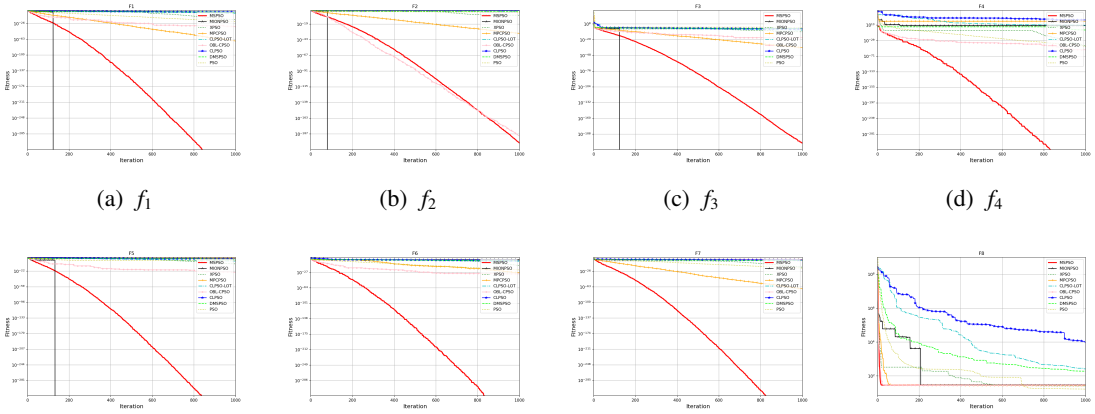
magnitude of the difference between MSPSO and the baseline algorithms on different dimensions, as shown in Table 8. It can be seen from Table 8 that MSPSO has a significant improvement compared with CLPSO-LOT, CLPSO, DMSPSO, and PSO. It is worth noting that there is no significant difference between MSPSO and PSO on the problem of 30D. However, the significant level of MSPSO and PSO changes from 0.1 to 0.01 with increased dimension. There is no statistically significant difference between MSPSO and MIONPSO, XPSO, MPCPSO, OBL-CPSO. However, MSPSO shows better results than these baseline algorithms on most of the test functions. Therefore, the statistical data of Friedman test, Nemenyi test, and Wilcoxon signed rank-test verify the consistent validity of MSPSO on the benchmark problems. And MSPSO performs better in high-dimensional spaces.

Table 8. Wilcoxon Signed Rank Test of benchmark functions.

D	MSPSO	MIONPSO	XPSO	MPCPSO	CLPSO-LOT	OBL-CPSO	CLPSO	DMSPSO	PSO
30	P-value	0.084	0.605	0.427	0.013	0.300	0.004	0.017	0.070
50	P-value	0.084	0.352	0.352	0.039	0.084	0.023	0.006	0.026
100	P-value	0.050	0.379	0.570	0.008	0.101	0.008	0.005	0.009

5.4. Analysis of the algorithm on convergence

The dynamic convergence curve of each benchmark function in the complete iteration cycle is employed to illustrate the convergence effect. We select the convergence of the first 1000 rounds to observe the difference in the convergence speed of each algorithm, as shown in Figure 5 and Figure 6.



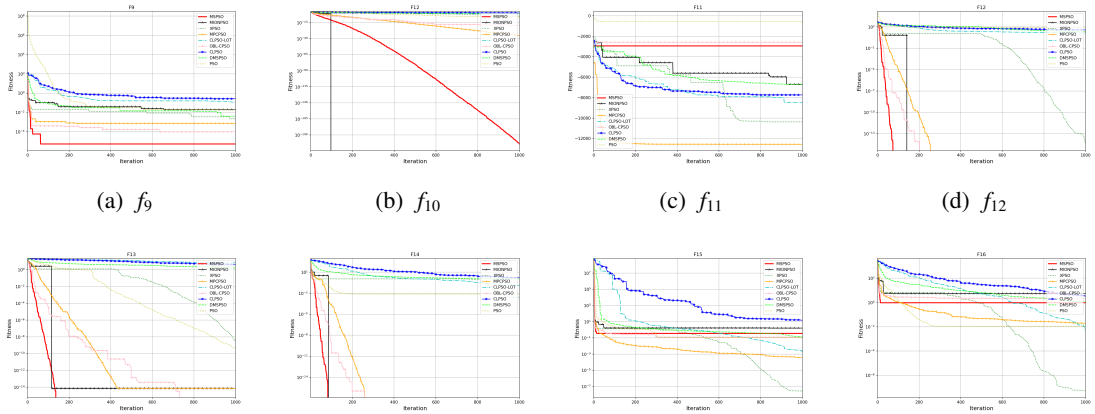
(a) f_1 (b) f_2 (c) f_3 (d) f_4

(e) f_5 (f) f_6 (g) f_7 (h) f_8

Figure 5. Convergence curves of all algorithms on unimodal functions.

For the eight unimodal functions shown in Figure 5, MSPSO maintains a high convergence speed and has a high convergence accuracy. Other PSO algorithms most converge faster in the early stage, then tend to be slow. In the initial stage of function f_8, although MSPSO has a high convergence speed, the convergence accuracy is relatively poor. As shown in Figure 6, for most multimodal functions, MSPSO has obvious advantages in terms of convergence speed and convergence accuracy. But on the functions f_{11}, f_{15} and f_{16}, although MSPSO converges faster in the early stage and then tends to be flat, it does not converge to the global optimal solution.

Based on the above analysis, MSPSO, compared with other PSO algorithms, has the best convergence speed and accuracy on unimodal functions and multimodal functions, mainly due to the fol-



(a) f_9 (b) f_{10} (c) f_{11} (d) f_{12}

(e) f_{13} (f) f_{14} (g) f_{15} (h) f_{16}

Figure 6. Convergence curves of all algorithms on multimodal functions.

lowing reasons. Firstly, the learning samples constructed by the two strategies proposed by MSPSO guide the movement of particles no longer blind, avoiding particles from exploring in other poor directions, and accelerate the convergence speed. At the same time, the learning samples guide particles to possible exploration and development areas, reducing the risk of falling into the local optimum. Additionally, the introduction of the electric field force and the adaptive update of parameters increase the diversity of the population and make the convergence speed faster.

5.5. MSPSO performance for a real-word problem

In this section, MSPSO will verify its performance against a practical optimization problem that is widely used. Dukic et al. [45] proposed a design method of multiphase code for spread spectrum pulse radar(SSPR) based on the characteristics of the non-periodic autocorrelation function, which is used in the design of multiphase pulse compression code. Gil-López et al. [46] formulated SSRP as a nonlinear maximum-minimum optimization problem, which is defined as follows.

$$\begin{aligned}
&\text{global } {}_{x\in X}min f(x) = max\left\{\varphi_1(x), \ldots, \varphi_{2m}(x)\right\}, X = \left\{(x_1, \ldots, x_n) \in R^n \mid 0 \le x_j \le 2\pi, j = 1, \ldots, n\right\}, \\
&\quad \text{where } m = 2n - 1, \text{ and} \\
&\quad \varphi_{2i-1}(x) = \sum_{j=i}^{n} \cos\left(\sum_{k=|2i-j-1|+1}^{j} x_k\right), i = 1, \ldots, n, \\
&\quad \varphi_{2i}(x) = 0.5 + \sum_{j=i+1}^{n} \cos\left(\sum_{k=|2i-j|+1}^{j} x_k\right), i = 1, \ldots, n-1, \\
&\quad \varphi_{m+i}(x) = -\varphi_i(x), i = 1, \ldots, m.
\end{aligned} \tag{5.1}$$

Table 9 shows the comparison results of MSPSO and other algorithms in solving the SSRP coding design problem. The fifth row of Table 9 represents the final ranking of all algorithms on the SSRP problem of different dimensions. Although the performance of MSPSO on SSRP is slightly lower than MPCPSO, it is better than the other seven algorithms. The experiment result shows that the proposed

MSPSO also has a better performance in solving practical problems and has a practical application value.

Table 9. Comparison results on SSRP coding design problem.

	MSPSO	MIONPSO	XPSO	MPCPSO	CLPSO-LOT	OBL-CPSO	CLPSO	DMSPSO	PSO
30D	4.98E-06	1.80E-03	1.40E-03	**8.55E-16**	4.32E-04	1.84E-04	7.45E-02	4.95E-02	3.21E-03
50D	1.32E-05	3.08E-03	1.31E-03	**2.80E-15**	3.28E-04	2.86E-04	9.52E-02	6.76E-02	6.75E-03
100D	2.69E-07	5.12E-03	2.10E-03	**1.45E-14**	4.73E-04	4.30E-04	1.56E-01	1.43E-01	7.19E-03
Ave rank	2.00	6.00	5.00	**1.00**	4.00	3.00	9.00	8.00	7.00
Final Rank	2	6	5	**1**	4	3	9	8	7

5.6. Comparison with other metaheuristic algorithms

This section further compares MSPSO with other recently released metaheuristic algorithms, namely MOMICA [2] and mSSA [3]. MOMICA is an improved ICA algorithm, which is conducive to solving the multimodal problem. mSSA is an improved SSA, which solves optimization problems more widely. The unimodal functions cannot evaluate the ability of the algorithm to fall into the local optimum well. Therefore, we chose three unimodal functions (f_1, f_3, f_8) and five more complex multimodal functions ($f_{12}, f_{13}, f_{14}, f_{15}, f_{16}$) for experiments. The experimental parameter configurations of MOMICA and mSSA are the same as the original text, and the experimental results can be found in the respective original texts. Table 10 lists the comparison results of algorithms independently running 30 times on the 30D problem. In addition, we performed the Friedman test and Wilcoxon signed-rank test

Table 10. Comparisons between MSPSO and other two metaheuristic algorithms.

		MSPSO	MOMICA	mSSA
f_1	Mean	**0.00E+00**	6.72E-28	**0.00E+00**
	Std	**0.00E+00**	6.34E-15	**0.00E+00**
	Rank	1	3	1
f_3	Mean	1.32E-197	7.16E-17	**1.07E-210**
	Std	0.00E+00	3.39E-08	**0.00E+00**
	Rank	2	3	1
f_8	Mean	2.88E+01	**2.67E+01**	2.84E+01
	Std	2.98E-02	**2.65E-02**	2.97E-01
	Rank	3	1	2
f_{12}	Mean	**0.00E+00**	2.98E-01	**0.00E+00**
	Std	**0.00E+00**	7.45E+00	**0.00E+00**
	Rank	1	3	1
f_{13}	Mean	**0.00E+00**	1.06E-13	8.88E-16
	Std	**0.00E+00**	1.02E-06	2.04E-31
	Rank	1	3	1
f_{14}	Mean	**0.00E+00**	4.00E-03	**0.00E+00**
	Std	**0.00E+00**	1.32E-04	**0.00E+00**
	Rank	1	3	1
f_{15}	Mean	3.24E-01	5.30E-02	**1.35E-11**
	Std	1.06E-01	2.00E-02	**4.04E-12**
	Rank	3	2	1
f_{16}	Mean	2.13E+00	6.54E-01	**1.60E-10**
	Std	4.96E-01	9.38E-06	**5.61E-11**
	Rank	3	2	1
Avg rank		1.88	2.50	1.25
Friedman		2.06	2.50	1.44
p-value		-	0.779	0.138

on the experimental results, as shown in the penultimate row and last row of Table 10. The third-to-last row of Table 10 represents the average rank of the algorithm on the eight test functions.

From Table 10, we can see that MSPSO and mSSA are effective for unimodal problems and achieve good performance on multimodal functions. The Wilcoxon signed-rank test shows that there is no significant difference between MSPSO and the other two algorithms. However, mSSA offers the best performance on test functions in terms of solution accuracy and Friedman test results. The proposed MSPSO follows closely behind. It is worth noting that MSPSO is inferior to the other two algorithms on the multimodal functions f_{15} and f_{16}. The reason may be that the global learning sample guides the particles to move too fast, which increases the probability of falling into the local optimum. The experimental results show that the proposed MSPSO needs to be further improved to improve the ability to jump out of the local optimum, which is more conducive to solving complex optimization problems.

6. Conclusions

In this study, we propose a multi-sample particle swarm optimization algorithm based on electric field force. In the proposed MSPSO, electric field force-based comprehensive learning strategy and segment-based weighted learning strategy are employed to construct learning samples, enhancing the ability of global exploration. Additionally, the adaptive changes of inertia weights and weight coefficients strengthen the diversity of the population and help the particles get rid of the local optimum. The experimental results of MSPSO and eight PSO algorithms demonstrate the superiority of MSPSO performance. It can obtain faster convergence speed and higher convergence accuracy when dealing with unimodal and multimodal problems. MSPSO is also effective in solving practical problems.

Nevertheless, the proposed MSPSO algorithm still has some problems, and further research and improvement are needed. On the one hand, although the MSPSO algorithm can obtain a faster convergence rate, the accuracy of the multimodal function is not ideal. For example, the optimal value cannot be found on the multimodal function f_{11}. Secondly, adjusting the parameters through the normal distribution function has a certain degree of randomness, which affects the algorithm's accuracy. Therefore, our subsequent work will further explore the local exploitation features of MSPSO, such as matching appropriate strategies for the specific dimension of particles to improve search efficiency. On the other hand, the application selected in this article is relatively single. Next, we will extend the application of the proposed MSPSO algorithm.

Acknowledgement

This work was supported by NSFC Grant Nos. 61701060 and 61801067, Guangxi Colleges and Universities Key Laboratory of Intelligent Processing of Computer Images and Graphics Project No. GIIP1806, and the Science and Technology Research Project of Higher Education of Hebei Province (Grant No. QN2019069), and Chongqing Key Lab of Computer Network and Communication Technology (CY-CNCL-2017-02).

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. O. Kaplan, E. Elik, Simplified model and genetic algorithm based simulated annealing approach for excitation current estimation of synchronous motor, *Adv. Electr. Comput. Eng.*, **18** (2018), 75–84.
2. N. Mohamed, N. Bilel, A. S. Alsagri, A multi-objective methodology for multi-criteria engineering design, *Appl. Soft Comput.*, **91** (2020), 106204.
3. E. Çelik, N. Öztürk, Y. Arya, Advancement of the search process of salp swarm algorithm for global optimization problems, *Expert Syst. Appl.*, **182** (2021), 115292.
4. E. Çelik, Improved stochastic fractal search algorithm and modified cost function for automatic generation control of interconnected electric power systems, *Eng. Appl. Artif. Intell.*, **88** (2020), 103407.
5. G. Lin, J. Guan, An integrated method based on PSO and EDA for the max-cut problem, *Comput. Intell. Neurosci.*, **2016** (2016), 3420671.
6. E. Çelik, A powerful variant of symbiotic organisms search algorithm for global optimization, *Eng. Appl. Artif. Intell.*, **87** (2020), 103294.
7. E. Çelik, N. Öztürk, A hybrid symbiotic organisms search and simulated annealing technique applied to efficient design of PID controller for automatic voltage regulator, *Soft Comput.*, **22** (2018), 8011–8024.
8. N. Singh, S. B. Singh, E. H. Houssein, Hybridizing salp swarm algorithm with particle swarm optimization algorithm for recent optimization functions, *Evol. Intell.*, (2020), 1–34.
9. J. Kennedy, R. Eberhart, Particle swarm optimization, in *Proceedings of ICNN'95-international conference on neural networks*, IEEE, (1995), 1942–1948.
10. P. Taylan, B. Akteke-Ozturk, Mathematical and data mining contributions to dynamics and optimization of gene-environment networks, *Int. J. Theor. Phys.*, **4** (2007), 115–146.
11. E. Kropat, G. W. Weber, B. Akteke-Öztürk, Eco-finance networks under uncertainty, in *Proceedings of the international conference on engineering optimization*, (2008).
12. G. W. Weber, İ. Batmaz, G. Köksal, P. Taylan, CMARS: A new contribution to nonparametric regression with multivariate adaptive regression splines supported by continuous optimization, *Inverse Probl. Sci. Eng.*, **20** (2012), 371–400.
13. A. Özmen, G. W. Weber, İ. Batmaz, E. Kropat, RCMARS: Robustification of CMARS with different scenarios under polyhedral uncertainty set, *Commun. Nonlinear Sci. Numer. Simul.*, **16** (2011), 4780–4787.
14. A. Özmen, E. Kropat, G. W. Weber, Robust optimization in spline regression models for multi-model regulatory networks under polyhedral uncertainty, *Optimization*, **66** (2017), 2135–2155.
15. E. Kropat, G. W. Weber, E. B. Tirkolaee, Foundations of semialgebraic gene-environment networks, *J. Dynam. Games*, **7** (2020), 253.
16. R. K. Agrawal, B. Kaur, P. Agarwal, Quantum inspired Particle Swarm Optimization with guided exploration for function optimization, *Appl. Soft Comput.*, **102** (2021), 107122.

17. Y. Du, F. Xu, A hybrid multi-step probability selection particle swarm optimization with dynamic chaotic inertial weight and acceleration coefficients for numerical function optimization, *Symmetryn*, **12** (2020), 922.

18. D. Tian, X. Zhao, Z. Shi, Chaotic particle swarm optimization with sigmoid-based acceleration coefficients for numerical function optimization, *Swarm Evol. Comput.*, **51** (2019), 100573.

19. C. Wu, F. Yang, Y. Wu, R. Han, Prediction of crime tendency of high-risk personnel using C5. 0 decision tree empowered by particle swarm optimization, *Math. Biosci. Eng.*, **16** (2019), 4135–4150.

20. M. Zhu, K. Wu, Y. Zhou, Z. Wang, J. Qiao, et al., Prediction of cooling moisture content after cut tobacco drying process based on a particle swarm optimization-extreme learning machine algorithm, *Math. Biosci. Eng.*, **18** (2021), 2496–2507.

21. P. Singh, S. Chaudhury, B. K. Panigrahi, Hybrid MPSO-CNN: Multi-level Particle Swarm optimized Hyperparameters of Convolutional Neural Network, *Swarm Evol. Comput.*, **63** (2021), 100863.

22. J. Rojas-Delgado, R. Trujillo-Rasúa, Training Neural Networks by Continuation Particle Swarm Optimization, in *International Workshop on Artificial Intelligence and Pattern Recognition*, Springer, (2018), 59–67 .

23. T. L. Dang, Y. Hoshino, Hardware/software co-design for a neural network trained by particle swarm optimization algorithm, *Neural Process Lett.*, **49** (2019), 481–505.

24. L. M. Abualigah, A. T. Khader, E. S. Hanandeh, A new feature selection method to improve the document clustering using particle swarm optimization algorithm, *J. Comput. Sci.*, **25** (2018), 456–466.

25. M. A. Tawhid, K. B. Dsouza, Hybrid binary bat enhanced particle swarm optimization algorithm for solving feature selection problems, *Appl. Comput. Inform.*, **16** (2018), 117–136.

26. F. Kılıç, Y. Kaya, S. Yildirim, A novel multi population based particle swarm optimization for feature selection, *Knowl. Based Syst.*, **219** (2021), 106894.

27. X. Wang, Y. Li, Chaotic image encryption algorithm based on hybrid multi-objective particle swarm optimization and DNA sequence, *Opt. Lasers Eng.*, **137** (2021), 106393.

28. H. T. Yau, T. H. Hung, C. C. Hsieh, Bluetooth based chaos synchronization using particle swarm optimization and its applications to image encryption, *Sensors*, **12** (2012), 7468–7484.

29. J. J. Liang, P. N. Suganthan, Dynamic multi-swarm particle swarm optimizer, in *Proceedings 2005 IEEE Swarm Intelligence Symposium*, IEEE, (2005), 124–129.

30. S. Wang, G. Liu, M. Gao, S. Cao, A. Guo, J. Wang, Heterogeneous comprehensive learning and dynamic multi-swarm particle swarm optimizer with two mutation operators, *Inf. Sci.*, **540** (2020), 175–201.

31. Q. Zhang, H. G. Li, An improved least squares SVM with adaptive PSO for the prediction of coal spontaneous combustion, *Math. Biosci. Eng.*, **16** (2019), 3169–3182.

32. K. M. Ang, W. H. Lim, N. A. M. Isa, S. S. Tiang, C. H. Wong, A constrained multi-swarm particle swarm optimization without velocity for constrained optimization problems, *Expert Syst. Appl.*, **140** (2020), 112882.

33. K. Chen, F. Zhou, A. Liu, Chaotic dynamic weight particle swarm optimization for numerical function optimization, *Knowl. Based Syst.*, **139** (2018), 23–40.
34. K. Zhang, Q. Huang, Y. Zhang, Enhancing comprehensive learning particle swarm optimization with local optima topology, *Inf. Sci.*, **471** (2019), 1–18.
35. S. Zhu, S. Zhou, J. Shang, L. Wang, B. Qiang, A multiion particle swarm optimization algorithm based on repellent and attraction forces, *Concurr. Comput.*, **33** (2021), e5979.
36. W. Li, X. Meng, Y. Huang, Z. H. Fu, Multipopulation cooperative particle swarm optimization with a mixed mutation strategy, *Inf. Sci.*, **529** (2020), 179–196.
37. X. Xia, L. Gui, G. He, B. Wei, Y. Zhang, F. Yu, H. Wu, Z. H. Zhan, An expanded particle swarm optimization based on multi-exemplar and forgetting ability, *Inf. Sci.*, **508** (2020), 105–120.
38. Y. Shi, R. C. Eberhart, Parameter selection in particle swarm optimization, in *International conference on evolutionary programming*, Springer, (1998), 591–600.
39. M. U. Farooq, A. Ahmad, A. Hameed, Opposition-based initialization and a modified pattern for Inertia Weight (IW) in PSO, in *2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, IEEE, (2017), 96–101.
40. A. Chatterjee, P. Siarry, Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization, *Comput. Oper. Res.*, **33** (2006), 859–871.
41. J. J. Liang, A. K. Qin, P. N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, *IEEE Trans. Evol. Comput.*, **10** (2006), 281–295.
42. J. Zhou, W. Fang, X. Wu, J. Sun, S. Cheng, An opposition-based learning competitive particle swarm optimizer, in *2016 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, (2016), 515–521.
43. Q. Yang, W. N. Chen, T. Gu, H. Zhang, J. D. Deng, Y. Li, J. Zhang, Segment-Based Predominant Learning Swarm Optimizer for Large-Scale Optimization, *IEEE Trans. Cybern.*, **47** (2017), 2896–2910.
44. Y. Shi, R. Eberhart, A modified particle swarm optimizer, in *IEEE world congress on computational intelligence*, IEEE, (1998), 69–73.
45. M. L. Dukic, Z. S. Dobrosavljevic, A method of a spread-spectrum radar polyphase code design, *IEEE J. Sel. Areas Commun.*, **8** (1990), 743–749.
46. S. Gil-López, J. Del Ser, S. Salcedo-Sanz, Á. M. Pérez-Bellido, J. Marı, J. A. Portilla-Figueras, et al., A hybrid harmony search algorithm for the spread spectrum radar polyphase codes design problem, *Expert Syst. Appl.*, **39** (2012), 11089–11093.

AIMS Press