Mathematical Biosciences and Engineering

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

MBE, 19(5): 5241–5268.
DOI: 10.3934/mbe.2022246
Received: 06 December 2021
Revised: 18 February 2022
Accepted: 09 March 2022
Published: 23 March 2022

Research article

Lévy flight-based inverse adaptive comprehensive learning particle swarm optimization

Xin Zhou[1,2], Shangbo Zhou[1,2,*], Yuxiao Han[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: In the traditional particle swarm optimization algorithm, the particles always choose to learn from the well-behaved particles in the population during the population iteration. Nevertheless, according to the principles of particle swarm optimization, we know that the motion of each particle has an impact on other individuals, and even poorly behaved particles can provide valuable information. Based on this consideration, we propose Lévy flight-based inverse adaptive comprehensive learning particle swarm optimization, called LFIACL-PSO. In the LFIACL-PSO algorithm, First, when the particle is trapped in the local optimum and cannot jump out, inverse learning is used, and the learning step size is obtained through the Lévy flight. Second, to increase the diversity of the algorithm and prevent it from prematurely converging, a comprehensive learning strategy and Ring-type topology are used as part of the learning paradigm. In addition, use the adaptive update to update the acceleration coefficients for each learning paradigm. Finally, the comprehensive performance of LFIACL-PSO is measured using 16 benchmark functions and a real engineering application problem and compared with seven other classical particle swarm optimization algorithms. Experimental comparison results show that the comprehensive performance of the LFIACL-PSO outperforms comparative PSO variants.

Keywords: particle swarm optimization; inverse learning; Lévy flight; comprehensive learning

1. Introduction

As many real-world optimization problems become more complex, there is always a need for better optimization algorithms. By observing flying dragonflies, humans found the key to unlocking the mystery of helicopters. By following the light trail of fireflies, the cold light mining lamps have been illuminated. Researchers once again turned their attention to nature to explore intelligent optimization algorithms to solve the optimization problem. An algorithm inspired by natural

biological phenomena or developed by exploiting the intelligent behavior of a group of organisms is called a swarm intelligence algorithm. Commonly used swarm intelligence optimization algorithms mainly include particle swarm optimization algorithm (PSO) [1], whale optimization algorithm (WOA) [2], firefly algorithm (FA) [3], ant lion algorithm (ALA) [4], cuckoo algorithm (CA) [5], genetic algorithm (GA) [6], monarch butterfly optimization [7], earthworm optimization algorithm (EWA) [8], elephant herding optimization (EHO) [9] moth search (MS) algorithm [10], and other optimization algorithms. Among the algorithms mentioned above, particle swarm optimization algorithms are popular because of their fast convergence, few parameters, and easy programming implementation.

Particle swarm optimization (PSO) is a population-based metaheuristic algorithm that was proposed by Eberhart et al. [11] together with Kennedy et al. [1] in 1995. The particle swarm algorithm originated from the research on the feeding behavior of bird flocks. Its core idea is to use the information sharing among the individuals in the population so that the motion of the whole population produces an evolutionary process from disorder to order in the solution space of the problem to obtain the optimal solution.

A large number of intelligent algorithms based on social intelligent behavior have been extensively researched in the past few decades, through the study of natural creatures, and applied to various optimization fields [12]. However, the conventional optimization methods sometimes cannot get a satisfactory solution at a reasonable time. Consequently, a large number of nature-inspired metaheuristics algorithms have been introduced to tackle various complicated problems [13]. For example, for the disadvantage that PSO tends to converge prematurely, Zhou et al. [14] proposes a multi-sample particle swarm optimization algorithm based on electric field force. Feng et al. [15] presents a novel binary monarch butterfly optimization (BMBO) method intended for addressing the 0–1 knapsack problem (0–1 KP). Feng et al. [16] presents generalized opposition-based learning (OBL) monarch butterfly optimization with Gaussian perturbation (OMBO), in which OBL strategy is used on half individuals of the population in the late stage of evolution, and Gaussian perturbation acts on the part of the individuals with poor fitness in each evolution. Liu et al. [17] proposes an artificial bee colony algorithm based on dynamic penalty function and Lévy flight (DPLABC) for COPs, Lévy flight with the logistic map is applied in the employed bee phase in this paper. Guo et al. [18] presents an improved krill herd (IKH) approach to solve global optimization problems. The method uses a new Lévy flight distribution and elitism scheme to update the KH motion calculation. Hanafi et al. [19] proposed a new PSO, called hierarchical PSO, which uses a dynamic tree hierarchy based on the performance of each particle in the population to define the neighborhood structure. Lu et al. [20] proposed a novel multi-group particle swarm optimizer driven by delayed activation (DA) strategy and exclusion mechanism, called enhanced multi-group collaborative particle swarm optimizer (EMCPSO). The EMCPSO aims to take advantage of multi-group technology to overcome the problem of premature convergence of standard PSO. Zhang et al. [21] proposed a novel particle swarm optimization based on the prey-predator relationship to reduce the probability of the algorithm falling into local optimum by introducing the old deletion mechanism. Liang et al. [22] developed a novel PSO named comprehensive learning PSO, which utilizes a new learning strategy to maintain the swarm diversity and thereby prevent premature convergence in solving multi-modal problems. In the comprehensive learning particle swarm optimization (CLPSO), each dimension of a particle determines the learning object according to the learning probability. Xia et al. [23] proposed an

extended particle swarm optimization with multiple paradigms and forgetting capability, which increases the diversity of particles by introducing forgetting capability. Lynn et al. [24] proposed a new comprehensive learning-based particle swarm optimization algorithm called heterogeneous integrated learning particle swarm optimization (HCLPSO). Zhang et al. [25] proposed a comprehensive learning particle swarm optimization algorithm based on Bayesian iterative methods, which is called Bayesian comprehensive learning particle swarm optimization (BCLPSO). The posterior probabilities obtained using Bayesian public iteration can inherit the historical information of the particles that may be utilized, thus maintaining the diversity of the population to prevent premature convergence. Shi et al. [26] introduced the concept of inertia weights (ω) to balance the global search and local search capabilities, which significantly improved the performance of PSO. Clerc et al. [1] also proposed another related parameter, called the shrinkage factor (χ), to prevent premature convergence. Xia et al. [27] proposed an extended particle swarm optimization with multiple examples and a forgetting ability to increase the diversity of particles by introducing the forgetting ability. Wang et al. [28] proposes a novel and robust hybrid metaheuristic optimization method to solve numerical optimization problems by adding differential evolution (DE) mutation operators to the accelerated particle swarm optimization (APSO) algorithm. Mirjalili et al. [29] combines the strengths of both particle swarm optimization (PSO) and gravitational search algorithm (GSA) to propose a novel hybrid optimization algorithm PSOGSA. The PSOGSA is a novel hybrid optimization algorithm, and it outperforms both PSO and GSA in terms of improved exploration and exploitation. Kahouli et al. [30] coupled genetic algorithm and particle swarm algorithm to optimize network reconfiguration in the distribution system to improve reliability and reduce power loss. Lin et al. [31] proposed the global genetic learning particle swarm optimization with diversity enhancement by ring topology (GGL-PSOD) to improve the GL-PSO's performance. Naderi et al. [32] developed a new hybrid algorithm to solve the transmission expansion planning problem in the grid, which is combined with shuffled frog leaping algorithm, particle swarm optimization, and teaching learning-based optimization. James et al. [33] proposed an efficient stock market forecasting model combining artificial intelligence networks and particle swarm optimization. Mohammadi et al. [34] proposed a hybrid particle swarm optimization-differential evolution algorithm combined with a multilayer perceptron for suspended sediment load estimation.

However, the drawback of the PSO algorithm is that when the particles search for the optimal solution, they may ignore the potential optimal solution region due to the influence of the learning paradigm, which leads them to fall into local optimum and difficult to jump out. And with the development of artificial intelligence techniques, the classical PSO variants provide a wide range of intelligence and convenience for humans and lead to many complex optimization problems. These optimization problems tend to be high-dimensional, multi-constrained, multimodal, and other trends. Therefore, we should study and improve global search performance when applying particle swarm algorithms to solve these complex optimization problems.

To improve the performance of particle swarm algorithms in solving complex optimization problems and better balance exploration and exploitation, this paper proposes a Lévy flight-based inverse adaptive comprehensive learning particle swarm optimization (LFIACL-PSO). In the LFIACL-PSO algorithm, we introduce an inverse learning mechanism, which allows particles to change their search direction and learn from the worst position in their history when they fall into local optimum and cannot jump out, learning steps obtained through random variables with Lévy

distribution. Secondly, to prevent the algorithm from converging prematurely and the particle diversity from decreasing, the comprehensive learning strategy and Ring-type topology are introduced more particle learning paradigm. Finally, we combine Lévy flight and inverse learning mechanism for updating the velocity of particles, which can effectively prevent its disadvantage of easily falling into the local optimum.

The rest of this paper is organized as follows: Section 2 mainly briefly introduces classic PSO and Ring-type topology. The implementation of LFIACL-PSO is described in detail in Section 3. Section 4 is the experimental part of this paper; we choose 16 benchmark functions and seven well-known comparison algorithms for experimental verification and analysis. Finally, relevant conclusions are outlined in section Section 5.

2. Related work

In this part, we mainly introduce the classic PSO algorithm and Ring-type topology and set the stage for subsequent content.

2.1. Classic PSO

In a D-dimensional scale-search space with a population of m birds, each bird is seen as a particle, and each particle can be viewed as a point in the space. The state attributes of the ith, $i = (1, 2, 3, \ldots, m)$ particle in the ith iteration is described by a velocity vector V_i and a position vector X_i. During the iteration of the population, the PSO will continuously generate and update the best historical position of the ith particle, i.e., $PB_i = (PB_{i,1}, PB_{i,2}, \ldots, PB_{i,D})$ as well as the best position of the whole population, i.e., $GB = (GB_1, GB_2, \ldots, GB_D)$. The rules of position updating in the traditional PSO algorithm are given as Eqs (2.1) and (2.2).

$$V_{i,d}^k = V_{i,d}^{k-1} + u_{PB_i} \cdot r_{PB_{i,d}} \cdot \left(PB_{i,d}^{k-1} - X_{i,d}^{k-1}\right) + u_{GB_i} \cdot r_{GB_d} \cdot \left(GB_d^{k-1} - X_{i,d}^{k-1}\right), \tag{2.1}$$

$$X_{i,d}^k = X_{i,d}^{k-1} + V_{i,d}^k. \tag{2.2}$$

The speed update Eq (2.1) includes three parts: $V_{i,d}^{k-1}$ is the speed of the previous iteration of the particle; $u_{PB_i} \cdot r_{PB_{i,d}} \cdot (PB_{i,d} - X_{i,d}^{k-1})$ is the cognitive part, which represents the particle's thinking; $u_{GB_i} \cdot r_{GB_d} \cdot (GB_d - X_{i,d}^{k-1})$ is the social part, which represents the information sharing and cooperation between particles. Where $V_i = (V_{i,1}, V_{i,2}, \ldots, V_{i,D})$ and $X_i = (X_{i,1}, X_{i,2}, \ldots, X_{i,D})$ represent the speed and position of the ith particle in the population, respectively; u_{PB_i} and u_{GB_i} denote the acceleration coefficients, adjust the maximum step length of learning ; $r_{PB_{i,d}}$ and r_{GB_d} are the random number in the range [0,1]. Generally, the speed change range of the dth dimension is limited to $[-V_{min,d}, V_{max,d}]$, and the position change range is $[X_{min,d}, X_{max,d}]$, that is, if the speed and position exceed the boundary value during iteration, then the particle velocity is $V_{i,d}^k = V_{max,d}$, position is $X_{i,d}^k = X_{max,d}$.

To improve the ability of the particle to jump out of the local optimum, Shi et al. [26] introduced the inertia weight ω by introducing it into Eq (2.1), as shown in Eq (2.3).

$$V_{i,d}^k = \omega V_{i,d}^{k-1} + u_{PB_i} \cdot r_{PB_{i,d}} \cdot \left(PB_{i,d}^{k-1} - X_{i,d}^{k-1}\right) + u_{GB_i} \cdot r_{GB_d} \cdot \left(GB_d^{k-1} - X_{i,d}^{k-1}\right), \tag{2.3}$$

in addition, Clerc et al. [35] added a constraint factor α to Eq (2.2) to control the velocity, as shown in Eq (2.4).

$$X_{i,d}^{k} = X_{i,d}^{k-1} + \alpha V_{i,d}^{k}. \tag{2.4}$$

2.2. Ring-type topology

In the canonical PSO, each particle has only one learning paradigm in each learning section. However, in real life, people choose to obtain information from more than one person. Furthermore, many studies have shown that children with multiple learning samples have better word learning abilities compared to those who learn from only one sample [36–38]. For this reason, we choose GB and RB_i as the two examples of the *ith* particle social learning part in this study. Specifically, select the Ring-type topology to update the particles at the RB_i. Updating the learning paradigm with Ring-type topology allows the algorithm to reduce the probability of entering a premature trap after being seduced by the local optimal point when performing local development tasks. When the number of times the particle fitness value is not updated is greater than the set maximum number of times, it is not updated for the particle to reselect neighbor particles. Update Velocity by Eq (2.5).

$$\begin{aligned} V_{i,d}^{k} = {} & \omega V_{i,d}^{k-1} + u_{PB_i} \cdot r_{PB_{1,d}} \cdot \left(PB_{i,d}^{k-1} - x_{i,d}^{k-1}\right) + \\ & u_{RB_i} \cdot r_{RB_{2,d}} \cdot \left(RB_{i,d}^{k-1} - x_{i,d}^{k-1}\right) + u_{GB_i} \cdot r_{GB_{3,d}} \cdot \left(GB_{d}^{k-1} - x_{i,d}^{t-1}\right). \end{aligned} \tag{2.5}$$

3. The proposed LFIACL-PSO algorithm

Although the particle algorithm converges quickly and is prone to implementation, it also suffers from premature convergence and loss of particle diversity. To address the shortcomings of particle swarm algorithms, this paper proposes a Lévy flight-based inverse adaptive comprehensive learning particle swarm optimization (LFIACL-PSO).

3.1. Comprehensive learning strategy

Unlike the standard particle swarm update rule, the comprehensive learning strategy improves the learning pattern of the particles themselves in the PSO algorithm by using the historical best information of all other particles to update the velocity of the particles. This strategy allows the diversity of the population to be preserved to prevent premature convergence. The comprehensive learning strategy speed update formula is shown in Eq (3.1).

$$V_{i,d}^{k} = \omega V_{i,d}^{k-1} + u_{PB_i} \cdot r_{PB_{1,d}} \cdot \left(PB_{f_i(d)}^{k-1} - x_{i,d}^{k-1}\right), \tag{3.1}$$

where $f_i(d) = [f_i(1), f_i(2), \ldots, f_i(D-1), f_i(D)]$ determines which particle's $Pbest_{i,d}$ should follow the *ith* particle for each dimension (d), can be the corresponding dimension of any particle's including its own, and the decision depends on the probability Pc_i, referred to as the learning probability, which can take different values for different particles. For each dimension of particle i, we generate a random number. If the random number is greater than Pc_i, the corresponding dimension will learn from the dimension of its *pbest*. Otherwise, it will learn from another particle *pbest* [22]. The learning probability Pc_i of this particle is defined as shown in Eq (3.2).

$$Pc_{\mathrm{i}} = \alpha + \beta * \frac{(\exp(10(i-1)/(PS-1)) - 1)}{(\exp(10) - 1)}, \tag{3.2}$$

where $\alpha = 0.05, \beta = 0.45$, PS denotes the size of the population. The search range of the particles is affected by the boundary $[X_{min}, X_{max}]$. When the boundary is exceeded, the fitness value and position of the particle are not updated. In addition, to improve the best position of the particle itself when it is updated, define the refresh gap m. If the particle iterates m times and the fitness value does not change, update Pc_i.

In this paper, when we use the comprehensive learning strategy as an updated paradigm for the self-awareness part of the particle, then the speed update equation is shown in Eq (3.3), which is obtained by transforming Eqs (2.5) and (3.1).

$$V_{i,d}^{k} = \omega V_{i,d}^{k-1} + u_{PB_i} \cdot r_{PB_{i,d}} \cdot \left(PB_{f_i(d)}^{k-1} - x_{i,d}^{k-1}\right) + u_{RB_i} \cdot r_{RB_{2,d}} \cdot \left(RB_{i,d}^{k-1} - x_{i,d}^{k-1}\right) + u_{GB_i} \cdot r_{GB_{3,d}} \cdot \left(GB_{d}^{k-1} - x_{i,d}^{t-1}\right). \tag{3.3}$$

3.2. CLPSO based on Lévy flight inverse adaptive method

3.2.1. Inverse learning

The classic particle swarm algorithm has all particles moving towards their optimal historical position and the population optimal historical position throughout the iterative process of finding the optimal place. However, in searching for food, every once in a while, birds share information about their distance from food. Therefore, it can be judged that each bird in the population affects the other individuals. The information about each generation of the particles is recorded, and by making full use of this information, it is possible to improve the algorithm's ability to find the best. When a particle gets trapped in a local optimum and stalls, let that particle learn toward its individual historical worst position to quickly guide these particles to escape from the optimum and improve the search capability of the algorithm.

In the process of algorithm iteration, if it is detected that the individual optimal position of the particle is not updated within the set number of iterations, it can be judged that the particle is caught in a local optimum. At this time, inverse learning is used for it, and the learning method of other particles remains unchanged. The worst position of the individual history of the *ith* particle is denoted as $Worst_i = \left(w_{i,1}, w_{i,2}, \ldots, w_{ij}, \ldots, w_{i,D}\right)$. Then the velocity update equation of the particle in inverse learning is Eq (3.4).

$$\begin{aligned} V_{i,d}^{k} = {} & \omega V_{i,d}^{k-1} + u_{PB_i} \cdot r_{PB_{i,d}} \cdot (PB_{f_i(d)}^{k-1} - X_{i,d)}^{k-1}) + u_{RB_i} \cdot r_{RB_{2,d}} \cdot (RB_{i,d}^{k-1} - x_{i,d}^{k-1}) + u_{GB_i} \cdot r_{GB_{3,d}} \cdot (GB_{d}^{k-1} - X_{i,d}^{k-1}) \\ & + I \cdot r_{W_{4,d}} \cdot (x_{i,d}^{k-1} - Worst_{i,d}^{k-1}), \end{aligned} \tag{3.4}$$

where I denotes the inverse learning factor, $Worst_{i,d}$ denotes the value of the *ith* particle at that moment, the worst position of the individual history in the *dth* dimension.

3.2.2. Lévy flight

Lévy Flight, named after the French mathematician Paul Lévy refers to a random walk where the probability distribution of the step size is heavy-tailed. It means that particles moving according to Lévy flight occasionally perform big steps, interspersed with frequent small steps. Many biological phenomena are characterized by a Lévy distribution, such as the flight path of flies. In recent years,

the Lévy distribution has been applied to the field of optimization, such as the cuckoo algorithm [39]. The use of Lévy flight can improve the global search ability of the algorithm and prevent the reduction of population diversity, so the use of Lévy flight in combination with particle swarm algorithm can effectively avoid its shortcomings of prone falling into local optimum. Enhancement of the global optimization-seeking capability of algorithms.

Because of the complexity of the Lévy distribution, the Mantegna algorithm is commonly used for simulation. The equation for the step size $Flight_{size}$ of the simulated Lévy flight is shown in Eq (3.5).

$$\text{Flight}_{\text{size}} = \frac{\mu}{|v|^{1/\gamma}}, \tag{3.5}$$

where $\gamma = 1.5$ is the parameter of the Lévy distribution, μ and v are random numbers obeying a normal distribution as shown in Eq (3.6).

$$\begin{cases} \mu \sim N\left(0, \tau_\mu^2\right) \\ v \sim N\left(0, \tau_v^2\right), \end{cases} \tag{3.6}$$

where

$$\begin{cases} \tau_u = \left\{ \frac{\Gamma(1+\gamma)\sin(\pi\gamma/2)}{\Gamma(1+\gamma/2)\gamma 2^{(\gamma-1)/2}} \right\}^{1/\gamma} \\ \tau_v = 1, \end{cases} \tag{3.7}$$

so, the particle motion step PS_{stepsize} is shown by the Eq (3.8),

$$PS_{\text{stepsize}} = \rho \times Flight_{\text{size}}, \tag{3.8}$$

where ρ is an adjustment factor, which adjusts the step size according to different specific problems so that the step size of the flight is not too large or too small and thus stays in the proper range. Figure 1 shows the image of the Lévy flight of 1000 steps simulated by Mantegna's algorithm in MATLAB.

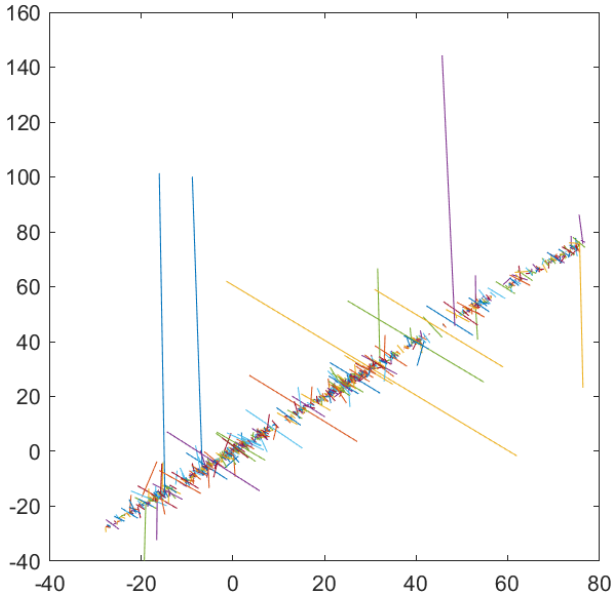


Figure 1. Lévy flight image.

Finally, combining the Lévy flight with the inverse learning mechanism and using the Lévy distribution to obtain the step size of the inverse learning, the velocity update equation for LFIACL-PSO is obtained as Eq (3.9).

$$\begin{aligned} V_{i,d}^{k} = \omega V_{i,d}^{k-1} + u_{PB_i} \cdot r_{PB_{i,d}} \cdot \left(PB_{f_i(d)}^{k-1} - x_{i,d}^{k-1}\right) + u_{RB_i} \cdot r_{RB_{2,d}} \cdot \left(RB_{i,d}^{k-1} - x_{i,d}^{k-1}\right) + u_{GB_i} \cdot r_{GB_{3,d}} \\ \left(GB_{d}^{k-1} - x_{i,d}^{t-1}\right) + I \cdot r_{P_{4,d}} \cdot \left(x_{i,d}^{k-1} - \text{ Worst }_{i,d}^{k-1}\right) \times PS_{\text{ size}}, \end{aligned} \tag{3.9}$$

where $Worst_{i,d}^{k-1}$ is the value of the *dth* dimension of the historical worst position of the *ith* particle.

3.2.3. Adaptive acceleration factor

Use the acceleration coefficients to regulate the maximum step size of particle learning. In typical particle swarm optimization algorithms, when the two acceleration coefficients are not equal, the algorithm is more likely to maintain the speed of convergence and the effectiveness of the search effect. Therefore, the value of the acceleration coefficients is also one of the main focuses of research in particle swarm optimization algorithms. The existing research on the acceleration coefficient mainly has two aspects: the setting based on the empirical value, and the other is the adaptive adjustment through the update iteration of the particle. For instance, the classified state by HMM is used to adapt PSO with both acceleration parameters, and population size [40]. Kiani et al. [41] used a tangential chaos strategy to bootstrap the acceleration coefficients to find the optimal solution. Inspired by the idea of adaptive regulation, in this paper, each particle acceleration coefficient is adaptively adjusted according to the historical information of elite particles in the population. Set the number of elite particles be 0.5 times the total number of particles. When the *ith* particle adjusts its acceleration coefficient, we use the three values generated by the normal distribution functions $N(\delta_1, \theta_1^2), N(\delta_2, \theta_2^2)$ and $N(\delta_3, \theta_3^2)$ assigned to u_{PB_i}, u_{RB_i} and u_{GB_i} respectively. The values of δ_1, δ_2 and δ_3 are updated by updating the ones shown in Eq (3.10).

$$\delta_i = (1 - \lambda) \times \delta_i + \lambda \times \psi, 1 \leq i \leq 3, \tag{3.10}$$

where λ denotes a learning weight for knowledge from elite particles; $\psi = average(\delta_i^{elite})$, δ_i^{elite} represents a set of δ_i of the elites; $average(\delta_i^{elite})$ is the average value of δ_i^{elite}. In addition, from Eq (3.10), the value of λ indicates the degree of influence of the elite particle's experience δ_i on the next generation. When $\lambda = 0$, the information of the elite particle has no effect on the adjustment of δ_i, and when $\lambda = 1$, the new δ_i will rely only on the historical knowledge of the elite particle.

Usually, the sum of two acceleration coefficients in the particle swarm optimization algorithm is 4.0 [42]. Therefore, in the initial evolutionary stage, setting $\delta_1 = \delta_3 = \delta_3 = 1.33$, $\theta_1 = \theta_2 = \theta_3 = 0.1$. Finally, the value of u_{PB_i}, u_{RB_i} and u_{GB_i} can be independently obtained by $N(\delta_1, \theta_1^2), N(\delta_2, \theta_2^2)$ and $N(\delta_3, \theta_3^2)$.

3.3. The general framework of the LFIACL-PSO algorithm

With the combination of the above components, Algorithm 1 shows the pseudo-code for LFIACL-PSO. Algorithm 1 is first initialized with relevant parameters such as population size PS, search space D, maximum number of iterations $Iter\_Max$, velocity V_i and position X_i of particles. Computer population initial fitness value $fit(X_i)$, with the initial fitness value as the current local

optimum value $P\_fit(i)$ and individual worst value $Worst(i)$ for each particle, and the best initial fitness value as the current global optimum value $G\_fit(i)$.

Algorithm 1 The pseudo-code of the LFIACL-PSO algorithm.

Input: The initial position vector $X_i = [x_{i,1}x_{i,2}, x_{i,3}, \ldots, x_{i,D}]$, velocity vector $V_i = [v_{i,1}v_{i,2}, v_{i,3}, \ldots, v_{i,D}]$.
Output: The whole particle's best position.

1: /*Initialization */
2: Initialization PS, D, $Iter\_Max$, $m = 5$, flag$(i) = 0$, $\delta_1 = \delta_1 = \delta_3 = 1.33$, $\theta_1 = \theta_2 = \theta_3 = 0.1$;
3: Computer population initial fitness value $fit(X_i)$, with the initial fitness value as the current local optimum value $P\_fit(i)$ and individual worst value $Worst(i)$ for each particle, and the best initial fitness value as the current global optimum value $G\_fit(i)$.
4: /* LOOP */
5: **for** k = 1 to $Iter\_Max$ **do**
6: **for** i = 1 to PS **do**
7: Update $V_{i,d}^k$ and $X_{i,d}^k$ by Eqs (3.3) and (2.2)
8: **if** fit$(X_i) <$ $P\_$ fit(i) **then**
9: $P\_fit(i) = fit(X_i)$
10: flag$(i) = 0$
11: **if** $fit(X_i) < G\_fit(i)$ **then**
12: $G\_fit(i) = fit(X_i)$
13: **end if**
14: **else**
15: flag$(i) =$ flag$(i) + 1$
16: **if** fit$(X_i) >$ Worst(i) **then**
17: $Worst(i) = fit(X_i)$
18: **end if**
19: Gain μ, v
20: $Flight_{\text{size}} = \frac{\mu}{|v|^{1/\gamma}}$
21: $PS_{\text{size}} = \rho \times Flight_{\text{size}}$
22: **if** flag$(i) > m$ **then**
23: Update $V_{i,d}^k$ and $X_{i,d}^k$ by Eqs (3.9) and (2.2)
24: Update Pc_i by Eq (3.2)
25: Update $PB_{f_i(d)}^{k-1}$, u_{PB_i}, u_{RB_i}, u_{GB_i}
26: Reselect a $RB_{i,d}^{k-1}$
27: **end if**
28: **end if**
29: **end for**
30: **end for**

When the current fitness value of the particle is smaller than the local optimum of the particle, $P\_fit(i)$ is updated and flag$(i) = 0$; similarly, $G\_fit(i)$ is updated. If the current fitness value of the particle is greater than $P\_fit(i)$, setting flag$(i) =$ flag$(i) + 1$, as shown in lines 7–14 of the pseudo code. If the fitness value is greater than the worst fitness value of *ith* position, $Worst(i)$ is updated, as shown

in lines 15,16 of the pseudo code. When flag(i) > m, update the velocity and position by Eqs (3.9) and (2.2), and conversely, update the velocity and position by equations (3.3) and (2.2), as shown in lines 7 and 23 of the pseudo code, update Pc_i by Eq (3.2) and $PB^{k-1}_{f_i(d)}$ and the associated parameters, as shown in lines 24–26 of the pseudo code.

4. Experimental studies

This chapter first analyzes the parameter sensitivity of the parameters introduced in the LFRACL-PSO algorithm. Then, the functional characteristics and roles of LFIACL-PSO analyzed by population diversity experiments. After that, the performance of the LFRACL-PSO algorithm is compared with seven typical PSO variants placed on sixteen benchmark functions in terms of solution accuracy, statistical significance test, CPU time consumption, and convergence speed in turn. Finally, to verify the practical application capability of the LFRACL-PSO algorithm, a spread spectrum radar polyphase code design engineering problem is used for testing and verification. The experimental environment is mainly implemented in windows 10 OS, Intel (R) Core (TM) i5-9600KF@3.70 GHz processor using python language programming, the python version is 3.6, and NumPy library version is 1.19.5.

Tables 1 and 2 list the details of the 16 benchmark functions and the seven comparison algorithms. The 16 functions selected in this paper include 6 unimodal functions ($f_1 - f_6$) and 10 multimodal functions ($f_7 - f_{16}$). Among them, ($f_{10} - f_{16}$) are more complex and difficult to handle multimodal functions.

4.1. Sensitivity of parameters

To analyze how the parameters λ, $flag(i)$, and I affect the performance of LFIACL-PSO. In this part, we set a series of experiments to analyze the sensitivity of the parameters.

First of all, initial values are assigned to each parameter separately, such as $\lambda = 0.2$, $flag(i) = 10$, and $I = 1.4$. When a parameter is analyzed, the rest of the parameters are fixed to their initial values. For example, when testing the efficiency of the parameter λ, the parameter $flag(i) = 10$ and $I = 1.4$. We selected five benchmark functions for experimental analysis, including two unimodal functions f_1, f_6 (Sphere, Rosenbrock) and three multimodal functions f_9, f_{13}, f_{14} (Alpine, Ackley, Griewank). To make sure the optimal values of parameters λ, $flag(i)$ and I in the LFIACL-PSO algorithm, the basic experimental setups are as follows, the search space: $D = 30$, population size: $N = 40$, maximum number of function evaluations: $MaxFEs = 500$. The result of each experiment is the average of 30 independent runs and takes different values of parameters λ, $flag(i)$ and I for the experiments. The results are shown in Tables 3–5, and the best results are bolded. Where Mean, Std, and Min represents the mean, standard deviation, and minimum values of the calculated 30 times results, respectively.

4.1.1. Sensitivity of λ

The parameter λ determines the degree of influence of the experience-based information of the elite particles on a particular particle. When the value of λ is small, the particles obtain more information from the elite particles, which leads to a rapid convergence of the population. When λ is larger, the particle acquires more information from its own historical experience and enhances the particle exploration ability. The experimental results are shown in Table 3. From the experimental

Table 1. Sixteen benchmark functions.

	Name	Benchmark Functions	Search Range	f_{min}
Unimodal	$Sphere(f_1)$	$f_1(x) = \sum_{i=1}^{D} x_i^2,$	$[-100, 100]^D$	0
	$Schwefel'sP1.2(f_2)$	$f_2(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j\right)^2$	$[-100, 100]^D$	0
	$Schwefel'sP2.21(f_3)$	$f_3(x) = max(\lvert x_i\rvert), i = 1, 2, ..., D,$	$[-100, 100]^D$	0
	$Schwefel'sP2.22(f_4)$	$f_4(x) = \sum_{i=1}^{D} \lvert x_i\rvert + \prod_{i=1}^{D} \lvert x_i\rvert,$	$[-10, 10]^D$	0
	$Bentcigar(f_5)$	$f_5(x) = x_1^2 + 10^6 \sum_{i=2}^{D} x_i^2,$	$[-1.28, 1.28]^D$	0
	$Rosenbrock(f_6)$	$f_6(x) = \sum_{i=1}^{D} 100 \times \left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2,$	$[-10, 10]^D$	0
Multimodal	$Schwefel(f_7)$	$f_7(x) = 418.982887273 \times \mathrm{D} - \sum_{i=1}^{D} x_i \sin\left(\sqrt{\lvert x_i\rvert}\right),$	$[-500, 500]^D$	0
	$Dminima(f_8)$	$f_8(x) = 78.332331408 + \sum_{i=1}^{D} \left(x_i^4 - 16x_i^2 + 5x_i\right)/D,$	$[-5, 5]^D$	0
	$Alpine(f_9)$	$f_9(x) = \sum_{i=1}^{D} \lvert x_i \sin x_i + 0.1x_i\rvert,$	$[-10, 10]^D$	0
	$SchwefelP2.26(f_{10})$	$f_{10}(x) = \sum_{i=1}^{D} -\left(x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)\right),$	$[-500, 500]^D$	-12569.5
	$Rastrigin(f_{11})$	$f_{11}(x) = \sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	$[-5.12, 5.12]^D$	0
	$NoncontinuousRastrigin(f_{12})$	$f_{12}(x) = \left(\sum_{i=1}^{D} y_i^2 - 100\cos(2\pi y_i) + 10\right),$ $y_i = \begin{cases} X_i & ,\lvert X_i\rvert < 0.5 \\ \frac{\text{round}(2X_i)}{2} & ,\text{otherwise} \end{cases}$	$[-5, 5]^D$	0
	$Ackley(f_{13})$	$f_{13}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\sum_{i=1}^{D} \cos 2\pi x_i\right) + 20 + e,$	$[-32, 32]^D$	0
	$Griewank(f_{14})$	$\mathrm{f}_{14}(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \frac{x_i}{\sqrt{i}} + 1,$	$[-600, 600]^D$	0
	$Penalized1(f_{15})$	$f_{15}(x) = \frac{\pi}{D}\left\{10\sin^2(\pi y_i) + \sum_{i=1}^{D-1} (y_i - 1)^2 \cdot \left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_D - 1)^2\right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4),$ 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
	$Penalized2(f_{16})$	$f_{16}(x) = 0.1\left\{\sin^2 \pi 3y_i + \sum_{i=1}^{D} (y_i - 1)^2 \cdot \left[1 + \sin^2(3\pi y_{i+1})\right] + (x_n - 1)^2\left[1 + \sin^2(2\pi y_D)\right]\right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$ where $y_i = 1 + 0.25(x_i + 1), (x_i, a, k, m) = \begin{cases} k(x_i - a)^m, x_i > a \\ 0, -a \le x_i \le a \\ k(-x_i - a)^m, x_i < -a \end{cases}$	$[-50, 50]^D$	0

Table 2. Baseline algorithms.

	Algorithm	Year	Parameter settings
1	PSO [26]	1998	$\omega = 0.7298, c_1 = c_2 = 1.49445$
2	KPSO [35]	1999	$K = 0.7298, c_1 = c_2 = 1.49445$
3	DMSPSO [43]	2005	$\omega = 0.7298, c_1 = c_2 = 1.49445, R = 10, L = 100$
4	CLPSO [22]	2006	$\omega = [0.4, 0.9], c = 1.49445, m = 7$
5	TSLPSO [44]	2019	$\omega = 0.4 \sim 0.9, c_1 = c_2 = 1.5, c_3 = 0.5 \sim 2.5$
6	CLPSO-LOT [45]	2019	$\omega = [0.4, 0.9], c = 1.49445, m = 7, g = 30$
7	XPSO [27]	2020	$\omega = [0.4, 0.9], c_1 = c_2 = c_3 = 1.35, Stag_{Max} = 5, \eta = 0.2, p = 0.2$

Table 3. Selection of parameter λ.

λ		f_1	f_6	f_9	f_{13}	f_{14}
0.1	Mean	1.99E-10	2.68E+01	1.67E-03	7.55E-01	1.31-02
	Std	1.65E-10	1.35E+01	1.77E-03	7.52E-01	1.59E-02
	Min	1.67E-11	2.11E+01	1.25E-05	2.05E-06	5.08E-11
0.2	Mean	1.83E-10	**2.37E + 01**	2.98E-03	**7.50E − 01**	**9.01E − 03**
	Std	1.12E-10	**1.07E + 01**	3.78E-03	**6.95E − 01**	**1.32E − 02**
	Min	5.45E-11	**1.04E + 01**	4.51E-06	**2.02E − 06**	**8.69E − 11**
0.3	Mean	2.01E-10	2.46E+01	**1.42E − 03**	7.89E-01	1.04E-02
	Std	1.88E-10	1.02E+01	**1.52E − 03**	6.69E-01	1.21E-02
	Min	3.41E-11	1.33E+01	**6.10E − 06**	2.10E-06	4.83E-11
0.4	Mean	**1.24E − 10**	2.71E+01	1.92E-03	8.74E-01	1.47E-02
	Std	**6.51E − 11**	1.47E+01	2.02E- 03	7.12E-01	1.93E-02
	Min	**1.30E − 11**	1.72E+01	5.52E-05	1.87E-06	1.10E-10

results, we can see that setting λ in the range of $[0.2, 0.4]$ is beneficial to improve the search efficiency of LFIACL-PSO for all test problems. In this paper, $\lambda = 0.2$ was used for all experiments.

4.1.2. Sensitivity of $flag(i)$

The value of the parameter $flag(i)$ determines when the particle updates the new selection speed, position, and neighbor update method. When the value of $flag(i)$ is small, the particles will change the velocity update put direction and neighbor particles frequently, which may disturb the search direction of the particles. When $flag(i)$ is larger, it is beneficial to keep the search trail of particles. However, if the particle falls into an optimal local solution, a larger $flag(i)$ will prevent the particle from reselecting the learning paradigm and jumping out of the optimal solution. The comparison results shown in the Table 4 indicate that the test function values gradually become better when $flag(i) \in [2, 4]$. However, the performance decreases when $flag(i) = 6$, and the performance of LFIACL-PSO gets better when $flag(i) \in [8, 10]$. When $flag(i)$ becomes too high, the performance deteriorates. Therefore, based on the above experimental analysis consideration, $flag(i) = 5$ is used in this paper.

4.1.3. Sensitivity of I

The value of the parameter I determines how much information the particle obtains from the worst position in its history. When I is small, the particle receives less information from its worst location, and inversely, more information is acquired. Determine the optimal value of I by experimenting with different values of I. The results are shown in Table 5, when $I = 1.4$, the optimal value of the test function is obtained. When $I \in [1.6, 1.8]$, the performance of LFIACL-PSO decreases. Therefore, set the parameter $I = 1.4$ in the following experiments based on the above analysis.

In addition, in this paper, we use the adaptive update method to obtain the values of u_{PB_i}, u_{RB_i}, and u_{GB_i}, and I adopts a given constant. The main reason is that I is a parameter value newly introduced in this paper. Therefore, we need to perform experimental analysis on I to obtain its optimal value.

Table 4. Selection of parameter $flag(i)$.

$flag(i)$		f_1	f_6	f_9	f_{13}	f_{14}
2	Mean	2.06E-09	7.05E+02	2.52E-02	1.76E+01	1.16E-02
	Std	2.55E-09	2.49E+03	4.90E-02	6.22E+00	1.66E-02
	Min	1.81E-10	8.63E+00	6.63E-05	1.16E+00	5.20E-11
4	Mean	**1.63E − 10**	2.85E+01	1.67E-03	**6.64E − 01**	**1.03E − 02**
	Std	**1.12E − 10**	1.51E+01	1.54E-03	**7.40E − 01**	**1.19E − 02**
	Min	**3.71E − 11**	1.57E+01	7.98E-06	**1.67E − 06**	**3.35E − 11**
6	Mean	1.29E-09	7.46E+02	1.59E-01	3.59E+00	1.26E-02
	Min	9.04E-11	1.61E+01	9.72E-05	1.56E-03	9.38E-11
8	Mean	1.96E-10	2.52E+01	2.73E-03	7.38E-01	1.03E-02
	Std	1.50E-10	1.06E+01	2.60E-03	6.64E-01	1.15E-02
	Min	2.22E-11	1.80E+01	4.10E-05	2.47E-06	1.94E-10
10	Mean	1.56E-10	2.57E+01	**1.45E − 03**	9.40E-01	1.46E-02
	Std	1.86E-10	1.22E+01	**1.75E − 03**	7.50E-01	1.96E-02
	Min	1.75E-11	1.00E+01	**4.52E − 06**	2.46E-06	2.53E-11
12	Mean	1.91E-10	**2.49E + 01**	2.55E-03	6.78E-01	1.15E-02
	Std	1.32E-10	**9.77E + 00**	3.77E-03	7.00E-01	1.21E-02
	Min	1.98E-11	**1.58E + 01**	9.09E-06	1.92E-06	1.08E-10

Table 5. Selection of parameter I.

I		f_1	f_6	f_9	f_{13}	f_{14}
0.4	Mean	1.55E-10	3.26E+01	2.71E-03	7.14E-01	1.32E-03
	Std	1.46E-10	2.10E+01	4.29E-03	7.20E-01	1.73E-03
	Min	1.47E-11	1.32E+01	8.06E-06	1.60E-06	5.48E-11
0.6	Mean	1.82E-10	**2.54E + 01**	2.23E-03	6.04E-01	8.45E-03
	Std	1.37E-10	**1.01E + 01**	2.83E-03	6.27E-01	9.08E-03
	Min	3.47E-11	**1.80E + 01**	4.78E-06	1.97E-06	7.47E-11
0.8	Mean	1.25E-10	3.08E+01	1.80E-03	8.86E-01	9.18E-03
	Std	8.32E-11	1.70E+01	1.95E-03	7.29E-01	1.32E-02
	Min	3.00E-11	1.34E+01	2.64E-05	1.99E-06	5.19E-11
1.0	Mean	1.36E-10	3.29E+01	4.20E-03	7.74E-01	1.29E-02
	Std	9.66E-11	1.99E+01	8.48E-03	7.11E-01	1.41E-02
	Min	2.38E-11	2.04E+01	1.77E-05	1.33E-06	4.44E-11
1.2	Mean	2.16E-10	2.75E+01	1.84E-03	7.90E-01	7.96E-03
	Std	2.23E-10	1.42E+01	1.73E-03	7.77E-01	1.16E-02
	Min	2.04E-11	1.71E+01	2.64E-05	2.63E-06	9.34E-11
1.4	Mean	**6.72E − 12**	2.66E+01	**4.13E − 05**	**7.29E − 07**	**8.53E − 03**
	Std	**4.75E − 12**	3.31E+01	**3.55E − 05**	**3.94E − 08**	**9.94E − 03**
	Min	**1.97E − 12**	1.55E+01	**5.81E − 06**	**6.90E − 07**	**1.40E − 10**
1.6	Mean	1.94E-10	2.63E+01	1.75E-03	8.04E-01	1.34E-02
	Std	1.75E-10	1.30E+01	1.94E-03	6.53E-01	1.66E-02
	Min	1.30E-11	1.54E+01	1.32E-05	1.16E-06	9.98E-11

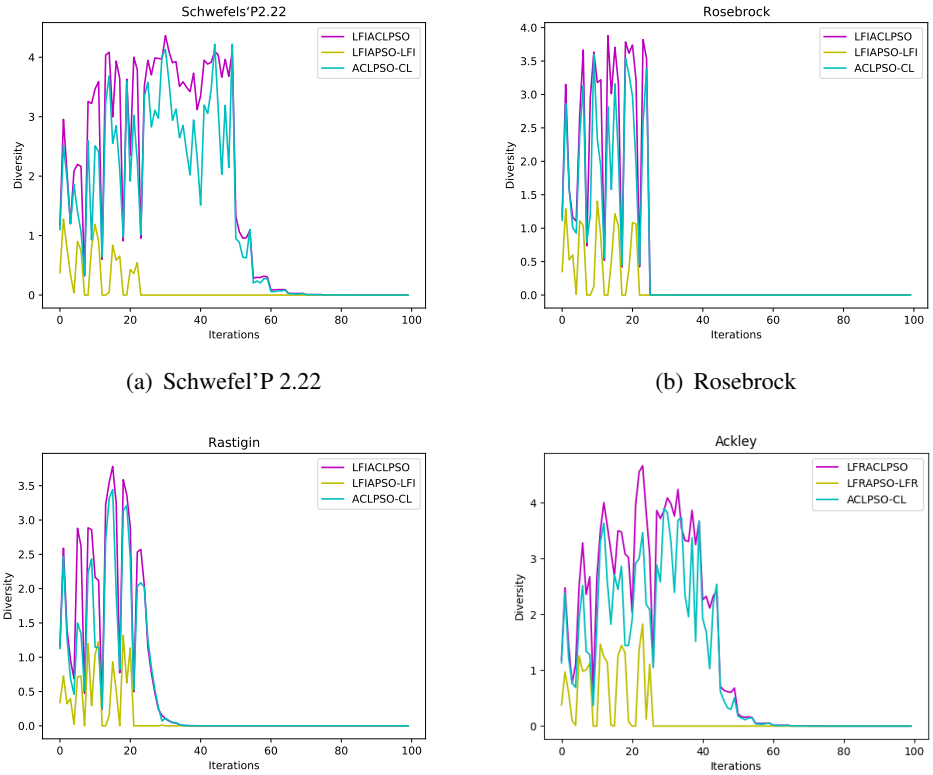
4.2. Diversity analysis

Swarm diversity is used to determine whether the population is being explored or developed [46]. The diversity of particle swarm optimization algorithms can be simply defined as the degree of dispersion of particles in the swarm. This dispersion can be defined around a specific center point. It can also be determined based on the particle's position or velocity. In the study of this article, the definition of swarm diversity is as follows Eq (4.1):

$$\text{Diversity(PS)} = \frac{1}{PS \cdot |L|} \cdot \sum_{i=1}^{PS} \sqrt{\sum_{j=1}^{D} \left(x_{i,j} - \bar{x}_j\right)^2}, \tag{4.1}$$

$$\bar{x}_j = \frac{\sum_{i=1}^{PS} x_{i,j}}{PS}, \tag{4.2}$$

where PS is the population size, $|L|$ is the length of the longest diagonal in the searching space, D is the dimensionality of the problem, $x_{i,j}$ denotes the position of the ith particle on the jth dimension and $\bar{x}_j$ denotes the average value of $x = [x_1, x_2, \ldots, x_{PS}]$ on the jth dimension.



(a) Schwefel'P 2.22 (b) Rosebrock

(c) Rastrigin (d) Ackley

Figure 2. Comparative results of algorithm diversity.

A set of experiments are carried out on the four representative benchmark functions, in which Schwefel's P 2.22 and Rosenbrock are unimodal functions, Rastrigin and Ackley are multimodal functions. The Schwefel's P 2.22 function with $[-10, 10]$ and global minimum given by

$f(0,0,\ldots,0) = 0$; the Rosenbrock function with $[-2.048, 2.048]$ and global minimum given by $f(0,0,\ldots,0) = 0$; the Rastrigin function with $[-5.12, 5.12]$ and global minimum given by $f(0,0,\ldots,0) = 0$ and the Ackley function with $[-30, 30]$ and global minimum given by $f(0,0,\ldots,0) = 0$. The experimental results are shown in Figure 2, where LFIAPSO-CL and ACLPSO-LFI respectively represent the algorithms for removing comprehensive learning strategies and Lévy flight and inverse learning from LFIACL-PSO.

Figure 2 depicts the diversity curves presented by the LFIACL-PSO algorithm on the four benchmark functions. The population size is 40, the search space is 30 dimensions, and the maximum number of iterations of the swarm is 500. As can be seen from the figure, the LFIACL-PSO algorithm maintains good diversity over the entire population space.

4.3. Comparison of solutions accuracy

This part mainly verifies the solution accuracy of the LFIACL-PSO algorithm and the seven comparison algorithms. To ensure the fairness of the experiments, First, the parameter settings of the seven comparison algorithms were consistent with those in the reference article. Second, to reduce the impact caused by the randomness of the algorithm, each algorithm runs 30 times independently. Third, select the mean (Mean), standard deviation (Std), minimum (Min) values of the fitness function values as evaluation criteria. Fourth, to verify the performance characteristics of each algorithm in different dimensions, the search space of particles is set to 20, 30 and 50 dimensions. During the experiments, the population size and the maximum number of iterations were 40 and 1000, respectively.

Tables 6–8 lists the statistical results of benchmark functions on different dimensions and includes the mean CPU time usage of each algorithm.

4.3.1. Unimodal functions

Tables 6–8 details the experimental results obtained for the six unimodal functions ($f_1 - f_6$) for the LFIACL-PSO algorithm proposed in this paper and the other seven comparison algorithms tested in different dimensions. The performance of the algorithms is measured by the number of best results obtained on the mean value. The best results by the different algorithms on the six unimodal functions are bolded. From the comparison results in Table 6, it is clear that on the six unimodal functions ($f_1 - f_6$). When the particle dimension is 20, the LFIACL-PSO yields the optimal solution on the benchmark functions $f_2 - f_6$, and the PSO obtains the optimal solution on f_1. When the particle dimensions are 30 and 50 dimensions, from Tables 7 and 8, we know that the LFIACL-PSO algorithm obtains optimal solutions on the unimodal functions $f_1, f_2, f_4 - f_6$, and the XPSO gains the optimal solution on f_3. The experimental comparison results show that LFIACL-PSO outperforms the other comparison algorithms in the six unimodal functions. The performance of LFIACL-PSO benefits from the following three advantages. First, compared to legacy PSOs, LFIACL-PSO has four exemplars which can give a particle more chance to obtain useful information. In addition, the Ring-type neighbor topology causes the particles to reorient their search at the end, improving the exploration ability of the population. Finally, the inverse learning mechanism and Lévy flight are applied to the particle swarm algorithm. When the particles fall into stagnation due to local optimum, the particles are allowed to learn from the worst position of their individual histories to quickly guide these particles to escape

from local optimum, enhance the diversity of the population, and improve the search capability of the algorithm, thus effectively preventing the disadvantage that the particle swarm algorithm easily falls into local optimum.

4.3.2. Multimodal functions

In reality, most multimodal functions contain many locally optimal solutions, making the algorithm prone to fall into the local optimum trap and converge prematurely. Tables 6–8 lists the experimental results obtained by testing the LFIACL-PSO algorithm with seven comparison algorithms using multimodal functions ($f_7 - f_{16}$) in different dimensions. Select the best results obtained on the mean to measure the algorithm performance. The best results obtained by the different algorithms on the ten multimodal functions are shown in bold. The comparison results presented in Tables 6–8 indicate that KPSO, DMPSO, TSLPSO performed much worse than LFIACL-PSO, mainly because the latter utilized much valuable historical knowledge from different samples, which facilitated the maintenance of population diversity. From the comparison results in Table 6, it is clear that the LFIACL-PSO algorithm obtains optimal values for the multimodal functions $f_9, f_{11} - f_{16}$. The CLPSO-LOT, CLPSO, and TLPSO algorithms obtain optimal values on the multimodal functions f_7, f_8, and f_{10}, respectively. The mean values of f_{12} in CLPSO and CLPSO-LOT are equal to LFIACL-PSO, but their standard deviations are large; therefore, the stability of the LFIACL-PSO algorithm is better than other comparative algorithms. When the search space is 30 dimensions, as can be seen from the comparison results in Table 7, the LFIACL-PSO algorithm obtains a total of 7 best performances on ten multimodal functions $f_7 - f_{16}$, and PSO, TLPSO, CLPSO, and XPSO all acquire one best performance. When the search space is 50 dimensions, as known from the comparison results in Table 8, the LFIACL-PSO algorithm obtains a total of six optimal solutions on $f_7 - f_{16}$ and the suboptimal solution is f_{12}. The PSO and DMPSO obtain the optimal solution in function f_7. The TLPSO algorithm and CLPSO and XPSO obtain optimal solutions on functions f_{10}, f_{12}, f_{15}, respectively. Overall, LFIACL-PSO has the most competitive performance among these classical comparison algorithms, as it achieves the best average value on 6 of the ten multimodal problems.

4.4. Statistical significance test of experimental results

To further illustrate the comprehensive performance of all comparison algorithms, Friedman's test with significance level $alpha = 0.05$ was performed in this section, the results of which are shown in Tables 9 and 10. Tables 9 and 10 represent the results of Friedman's test on unimodal and multimodal functions for each of the eight algorithms in different dimensions, respectively. From Table 9, we know that LFIACL-PSO obtained the best overall performance for unimodal functions $f_1 - f_6$, followed by XPSO, PSO, and CPSO-LOT. From Table 10, the LFIACL-PSO acquires the best overall performance for multimodal functions $f_7 - f_{16}$, followed by CPSO-LOT, XPSO, PSO, and CLPSO. Finally, regarding the accuracy of the solution, Table 11 shows the final average ranking of all algorithms on 16 benchmark functions on different dimensions. From Table 11, the overall performance of LFIACL-PSO is the best, followed by XPSO and PSO.

Furthermore, a graphical approach was used to show the key differences (CDs) in Friedman's rankings, and the results are shown in Figure 4. The CD is calculated as in Eq (4.3). Suppose the difference between the average sequential values of the two algorithms exceeds the critical value

Table 6. Comparison results of 16 benchmark functions on all algorithms (20D).

		PSO	KPSO	DMSPSO	CLPSO	TSLPSO	CLPSO-LOT	XPSO	LFIACL-PSO
f_1	Mean	**4.20E − 28**	1.23E-05	2.66E+00	9.48E+00	2.66E+00	1.05E-01	1.30E-03	6.20E-22
	Std	**8.30E − 28**	4.82E-05	1.56E+00	6.05E+00	1.56E+00	5.41E-01	4.50E-03	4.87E-22
	Min	**1.63E − 30**	8.28E-18	6.56E-01	1.36E+00	6.56E-01	3.75E-14	2.29E-07	6.48E-23
	Time(s)	0.14	0.16	0.04	16.82	3.92	12.45	2.79	13.65
f_2	Mean	1.14E-04	2.50E+01	4.42E+01	2.59E+03	4.14E+03	1.10E+03	9.92E-04	**1.20E − 06**
	Std	1.13E-04	3.32E+01	2.65E+01	7.06E+02	2.01E+04	4.66E+02	3.10E-03	**1.21E − 06**
	Min	5.65E-06	1.98E-01	8.61E+00	1.64E+03	7.32E-02	4.67E+02	1.85E-27	**5.80E − 08**
	Time(s)	1.60	2.62	0.66	16.64	33.28	14.46	4.69	18.35
f_3	Mean	9.88E-05	2.54E+00	1.48E+00	3.35E+01	5.44E+01	2.95E+01	1.01E-04	**4.94E − 06**
	Std	1.17E-04	1.66E+00	4.58E-01	6.62E+00	1.82E+01	5.40E+00	1.21E-04	**1.41E − 05**
	Min	7.77E-06	2.73E-01	8.94E-01	1.83E+01	2.59E+01	1.81E+01	5.85E-38	**7.47E − 09**
	Time(s)	0.13	0.19	0.05	12.98	3.78	12.95	2.73	13.86
f_4	Mean	7.77E-01	2.25E-01	5.17E-01	1.63E-01	1.31E+01	3.84E-04	1.30E-10	**3.81E − 11**
	Std	3.14E+00	4.92E-01	1.31E-01	8.72E-02	4.20E+01	6.30E-04	3.37E-10	**1.26E − 10**
	Min	1.79E-14	1.29E-03	2.00E-01	3.85E-02	5.17E-02	2.12E-05	1.13E-18	**3.03E − 13**
	Time(s)	0.20	0.23	0.06	12.01	5.00	12.90	2.83	13.91
f_5	Mean	6.69E-03	2.46E-02	4.77E-03	9.52E-02	1.51E-02	5.55E-02	1.65E-15	**1.54E − 15**
	Std	3.02E-03	1.85E-02	1.94E-03	3.97E-02	7.98E-03	3.07E-02	4.06E-15	**2.27E − 15**
	Min	1.77E-03	5.81E-03	1.22E-03	2.85E-02	4.24E-03	1.41E-02	7.66E-24	**6.11E − 18**
	Time(s)	0.82	0.41	0.10	12.19	8.76	12.52	4.13	13.87
f_6	Mean	1.47E+01	4.01E+01	2.67E+01	2.33E+02	8.21E+01	1.07E+02	1.54E+01	**3.20E + 00**
	Std	2.19E+01	3.49E+01	1.81E+01	1.38E+02	7.57E+01	1.14E+02	2.86E-01	**2.72E + 00**
	Min	1.24E-03	5.82E+00	1.61E+01	6.74E+01	8.08E-01	1.14E+01	1.48E+01	**2.45E − 02**
	Time(s)	1.96	1.54	0.39	13.08	29.12	14.12	5.19	17.02
f_7	Mean	4.86E+03	9.03E+03	1.74E+04	4.78E+03	3.29E+06	**4.71E+03**	4.79E+03	5.18E+03
	Std	1.92E+02	2.22E+02	2.33E+02	2.25E+02	2.15E+05	**3.30E+02**	2.22E+02	2.47E+02
	Min	4.40E+03	8.36E+03	1.68E+04	4.13E+03	1.15E+05	**3.92E+03**	4.34E+03	4.68E+03
	Time(s)	2.41	3.41	0.85	14.27	46.59	14.94	6.81	20.01
f_8	Mean	1.42E+03	1.42E+03	1.42E+03	**1.41E+03**	1.43E+03	1.42E+03	1.42E+03	1.42E+03
	Std	2.51E+00	3.45E+00	2.32E+00	**1.20E+00**	1.58E+00	1.27E+00	2.38E+00	2.57E+00
	Min	1.42E+03	1.42E+03	1.42E+03	**1.41E+03**	1.43E+03	1.42E+03	1.42E+03	1.42E+03
	Time(s)	3.28	3.65	0.91	14.86	57.69	15.47	9.38	21.56
f_9	Mean	4.93E-09	2.65E-02	6.07E-02	3.93E-01	1.73E+01	9.50E-01	1.49E-05	**4.14E-09**
	Std	1.83E-08	3.89E-02	4.40E-02	2.06E-01	1.96E+01	3.46E-01	4.20E-05	**2.22E-08**
	Min	3.84E-14	5.49E-04	1.66E-02	1.09E-01	1.65E-03	3.43E-01	4.06E-10	**7.85E-14**
	Time(s)	0.3	0.3	0.07	11.93	4.92	12.25	6.89	13.78
f_{10}	Mean	−4.21E+02	−4.47E+03	−4.97E+03	−5.78E+03	**−7.67E+68**	−5.76E+03	−4.27E+03	−4.65E+03
	Std	4.48E+01	5.27E+02	5.13E+02	2.38E+02	**2.99E+69**	2.41E+02	8.31E+02	5.82E+02
	Min	−5.32E+02	−5.54E+03	−6.24E+03	−6.37E+03	**−1.48E+70**	−6.20E+03	−6.80E+03	−5.77E+03
	Time(s)	0.26	0.28	0.07	12.02	7.63	12.27	4.54	13.83
f_{11}	Mean	3.77E+01	2.69E+01	1.09E+01	2.04E+01	8.76E+01	7.09E+00	1.48E-15	**0.00E+00**
	Std	1.27E+01	9.17E+00	2.41E+00	5.76E+00	6.00E+01	3.16E+00	4.79E-15	**0.00E+00**
	Min	1.49E+01	1.19E+01	7.24E+00	1.13E+01	2.49E+01	1.74E+00	0.00E+00	**0.00E+00**
	Time(s)	0.27	0.36	0.09	12.19	5.73	12.01	3.09	13.53
f_{12}	Mean	−1.59E+03	−1.72E+03	−1.76E+03	**−1.78E+03**	−1.22E+03	**−1.78E+03**	−1.64E+03	**−1.78E+03**
	Std	8.94E+01	2.69E+01	2.61E+01	**5.31E+00**	3.59E+02	**2.26E+00**	1.98E+02	**1.13E+00**
	Min	−1.76E+03	−1.77E+03	−1.78E+03	**−1.79E+03**	−1.69E+03	**−1.79E+03**	−1.80E+03	**−1.79E+03**
	Time(s)	4.34	5.25	1.31	16.32	70.6	16.37	3.29	23.67
f_{13}	Mean	1.24E-01	2.30E+00	1.40E+00	2.68E+00	1.53E+01	1.91E+00	4.33E-12	**2.33E-12**
	Std	3.76E-01	9.01E-01	6.54E-01	6.40E-01	7.28E+00	1.26E+00	1.47E-11	**1.35E-12**
	Min	7.11E-15	7.49E-07	2.11E-01	9.23E-01	2.45E+00	8.76E-03	3.55E-15	**6.32E-13**
	Time(s)	0.47	0.55	0.14	12.18	8.27	12.71	3.18	14.16
f_{14}	Mean	2.09E-02	2.72E-02	8.92E-01	1.17E+00	2.76E-01	1.06E-01	6.69E-02	**1.56E-02**
	Std	1.44E-02	2.98E-02	1.36E-01	1.71E-01	7.28E-01	1.63E-01	1.37E-01	**1.57E-02**
	Min	0.00E+00	9.99E-16	5.00E-01	9.33E-01	4.33E-15	3.53E-03	1.11E-16	**0.00E+00**
	Time(s)	1.32	1.4	0.35	13.07	22.8	13.37	3.99	16.18
f_{15}	Mean	5.70E-02	9.07E-01	7.49E-02	4.32E+01	2.53E-01	2.71E+00	6.58E-09	**9.16E-25**
	Std	1.42E-01	1.26E+00	1.03E-01	1.51E+02	3.90E-01	4.88E+00	2.42E-08	**1.19E-24**
	Min	2.38E-31	2.19E-11	3.23E-03	1.98E+00	1.01E-15	3.40E-07	1.17E-15	**5.70E-26**
	Time(s)	1.94	2.09	0.52	13.53	35.79	17.13	4.47	17.65
f_{16}	Mean	1.83E-03	4.92E-02	3.11E-01	1.06E+00	2.30E+01	2.21E-02	7.33E-04	**1.04E-23**
	Std	4.09E-03	6.17E-02	1.56E-01	1.06E+00	1.19E+02	5.19E-02	2.74E-03	**2.07E-23**
	Min	3.25E-31	3.25E-08	8.38E-02	1.38E-01	2.53E-16	2.03E-07	1.74E-13	**3.36E-25**
	Time(s)	4.67	5.23	1.31	16.88	76.02	16.19	6.36	23.37

Table 7. Comparison results of 16 benchmark functions on all algorithms (30D).

		PSO	KPSO	DMSPSO	CLPSO	TSLPSO	CLPSO-LOT	XPSO	LFIACL-PSO
f_1	Mean	1.90E-16	1.63E+00	2.19E+01	3.77E-01	2.27E+02	1.88E-04	1.71E-15	**1.27E-20**
	Std	9.22E-16	3.63E+00	9.23E+00	4.87E-01	8.28E+02	2.57E-04	4.35E-15	**1.31E-20**
	Min	4.62E-20	3.48E-02	7.91E+00	6.54E-03	1.32E-06	7.16E-06	7.72E-64	**2.51E-21**
	Time(s)	0.17	0.37	3.5	49	6.64	18.4	4.22	19.79
f_2	Mean	1.19E+00	6.86E+02	4.27E+02	6.70E+03	4.52E+03	2.70E+03	5.12E+01	**2.17E-01**
	Std	8.65E-01	5.42E+02	1.55E+02	1.23E+03	2.99E+03	5.11E+02	6.62E+01	**4.41E-01**
	Min	1.04E-01	8.00E+01	1.36E+02	4.48E+03	2.85E+02	1.47E+03	4.64E-14	**6.04E-03**
	Time(s)	2.52	5.68	13.38	56.41	81.07	21.64	6.63	26.17
f_3	Mean	2.68E-01	1.08E+01	4.25E+00	3.23E+01	7.55E+01	2.63E+01	**9.98E-04**	3.87E-01
	Std	1.37E-01	3.08E+00	1.31E+00	4.84E+00	1.57E+01	5.28E+00	**1.77E-03**	2.72E-01
	Min	5.88E-02	5.91E+00	2.43E+00	2.24E+01	3.60E+01	1.75E+01	**1.03E-18**	6.94E-02
	Time(s)	0.23	0.35	3.48	49.55	6.22	18.75	4.24	19.73
f_4	Mean	8.29E+01	2.47E+00	2.19E+00	2.69E-02	7.73E+00	2.15E-03	5.88E-06	**8.44E-07**
	Std	1.10E+02	1.54E+00	5.87E-01	2.12E-02	5.94E+00	2.00E-03	1.45E-05	**2.30E-05**
	Min	1.90E-07	4.96E-01	1.38E+00	5.59E-03	2.76E-01	6.17E-04	5.33E-24	**5.74E-11**
	Time(s)	0.39	0.49	3.75	49.94	8.32	18.67	4.28	19.74
f_5	Mean	4.89E-03	1.21E+06	1.92E+07	2.65E+05	5.58E+04	6.37E-02	8.50E-04	**5.78E-09**
	Std	7.04E-03	3.25E+06	7.97E+06	3.69E+05	2.79E+05	2.67E-02	1.86E-03	**2.87E-08**
	Min	1.32E-04	3.78E+03	7.58E+06	8.08E+03	2.78E-02	3.64E-02	5.04E-14	**3.94E-15**
	Time(s)	0.7	1.15	5.08	50.41	19.07	19.72	4.71	20.14
f_6	Mean	4.10E+01	1.03E+02	8.44E+01	1.67E+02	4.57E+02	9.18E+01	2.58E+01	**1.55E+01**
	Std	2.97E+01	6.06E+01	3.38E+01	7.11E+01	6.24E+02	3.02E+01	3.97E-01	**3.38E+00**
	Min	6.46E+00	2.32E+01	4.43E+01	4.86E+01	2.87E+01	2.94E+01	2.48E+01	**7.94E+00**
	Time(s)	2.26	5.06	12.54	55.02	70.34	21.66	6.25	24.96
f_7	Mean	**3.91E+03**	8.13E+03	8.12E+03	8.26E+03	3.67E+06	8.25E+03	8.19E+03	8.69E+03
	Std	**2.38E+0**	2.80E+02	3.38E+02	2.55E+02	4.02E+04	2.91E+02	2.98E+02	2.80E+02
	Min	**3.34E+03**	7.37E+03	7.24E+03	7.42E+03	4.02E+04	7.77E+03	7.04E+03	8.02E+03
	Time(s)	4.35	9.17	19.28	61.76	119.42	22.78	8.04	29.13
f_8	Mean	2.21E+03	2.21E+03	2.21E+03	**2.20E+03**	2.22E+03	2.20E+03	2.21E+03	**2.20E+03**
	Std	2.13E+00	2.70E+00	2.08E+00	**3.66E-01**	2.85E+00	1.06E+00	1.61E+00	**2.10E+00**
	Min	2.20E+03	2.20E+03	2.20E+03	**2.20E+03**	2.21E+03	2.20E+03	2.20E+03	**2.20E+03**
	Time(s)	4.86	10.4	21.79	61.75	138.36	23.82	8.59	31.66
f_9	Mean	1.09E-01	7.72E-01	4.23E-01	4.13E-01	2.10E+01	2.64E+00	2.23E-04	**2.21E-04**
	Std	4.08E-01	9.84E-01	3.16E-01	2.01E-01	3.25E+01	9.40E-01	3.82E-04	**6.53E-04**
	Min	3.47E-08	3.53E-02	8.69E-02	8.76E-02	2.19E-02	1.01E+00	4.20E-07	**2.87E-10**
	Time(s)	0.21	0.51	3.8	50.07	8.13	18.24	4.32	19.79
f_{10}	Mean	−6.28E+02	−6.47E+03	−6.71E+03	−7.97E+03	**−2.53E+67**	−7.93E+03	−5.43E+03	−6.42E+03
	Std	6.53E+01	8.29E+02	7.22E+02	3.05E+02	**1.31E+68**	2.76E+02	1.33E+03	9.31E+02
	Min	−7.33E+02	−8.17E+03	−8.42E+03	−8.78E+03	**−7.32E+68**	−8.71E+03	−9.31E+03	−8.17E+03
	Time(s)	0.21	0.51	3.75	50.11	7.8	18.22	4.51	19.72
f_{11}	Mean	8.34E+01	4.57E+01	2.56E+01	2.95E+01	1.49E+02	6.57E+00	4.36E+00	**0.00E+00**
	Std	2.57E+01	1.51E+01	6.74E+00	6.12E+00	9.19E+01	3.28E+00	2.17E+01	**0.00E+00**
	Min	4.58E+01	2.60E+01	1.41E+01	1.67E+01	6.67E+01	1.67E-01	0.00E+00	**0.00E+00**
	Time(s)	0.25	0.65	4.03	49.73	9.52	17.81	5.17	19.46
f_{12}	Mean	−2.03E+03	−2.57E+03	−2.52E+03	−2.66E+03	−1.50E+03	−2.68E+03	−5.00E+00	**-2.83E+03**
	Std	2.70E+02	5.64E+01	1.37E+02	6.13E+00	6.87E+02	2.16E+00	0.00E+00	**3.56E+01**
	Min	−2.36E+03	−2.64E+03	−2.66E+03	−2.67E+03	−2.52E+03	−2.68E+03	−5.00E+00	**−2.87E+03**
	Time(s)	5.46	14.95	30.12	70.47	175.21	24.95	5.34	34.02
f_{13}	Mean	1.89E+00	4.17E+00	2.85E+00	1.10E+00	2.00E+01	7.33E-01	1.81E+01	**6.22E-01**
	Std	3.48E+00	1.00E+00	3.53E-01	6.35E-01	1.62E+00	8.23E-01	4.77E+00	**7.46E-01**
	Min	9.28E-10	1.34E+00	2.01E+00	5.50E-02	1.20E+01	2.48E-03	0.00E+00	**9.00E-12**
	Time(s)	0.38	0.96	4.71	50.05	13.85	18.91	4.47	20.15
f_{14}	Mean	8.36E-03	4.23E-01	1.23E+00	4.78E-01	3.20E+00	1.12E-02	1.16E-01	**7.34E-03**
	Std	1.31E-02	3.67E-01	8.24E-02	2.63E-01	5.54E+00	9.71E-03	2.16E-01	**1.21E-02**
	Min	0.00E+00	1.15E-02	1.10E+00	1.28E-01	1.59E-05	1.09E-03	8.34E-04	**0.00E+00**
	Time(s)	1.61	4.22	11	53.67	52.05	19.61	5.86	23.56
f_{15}	Mean	1.00E-01	5.54E+00	5.53E-01	9.96E-01	9.17E+05	1.05E-01	**5.95E-06**	4.15E-02
	Std	1.99E-01	3.82E+00	2.67E-01	9.37E-01	4.24E+06	2.41E-01	**2.16E-05**	1.02E-01
	Min	1.51E-19	8.12E-01	7.05E-02	1.72E-02	7.74E-07	2.37E-06	**2.26E-09**	1.16E-23
	Time(s)	2.44	5.91	14.09	56.43	85.28	21.04	6.81	25.96
f_{16}	Mean	8.77E-03	7.12E+00	2.21E+00	9.14E-02	5.10E+01	2.59E-03	2.31E-03	**1.93E-03**
	Std	1.55E-02	6.56E+00	7.88E-01	8.37E-02	8.50E+01	7.10E-03	8.20E-03	**4.86E-03**
	Min	1.91E-19	2.57E-01	9.48E-01	5.37E-03	4.29E-02	3.22E-05	1.58E-07	**1.21E-22**
	Time(s)	5.61	14.14	30.03	66.93	188.98	24.61	10.39	34.77

Table 8. Comparison results of 16 benchmark functions on all algorithms (50D).

		PSO	KPSO	DMSPSO	CLPSO	TSLPSO	CLPSO-LOT	XPSO	LFIACL-PSO
f_1	Mean	5.82E-05	2.02E+02	2.00E+02	1.27E+01	1.91E+04	1.29E+00	1.14E-07	**1.74E-13**
	Std	1.84E-04	1.04E+02	5.20E+01	7.50E+00	8.73E+03	1.83E+00	5.68E-07	**3.39E-13**
	Min	8.73E-08	1.88E+01	9.22E+01	3.59E+00	6.97E+03	4.47E-02	1.79E-76	**6.15E-16**
	Time(s)	0.18	0.37	3.51	49.61	6.38	18.6	6.01	13.62
f_2	Mean	2.41E+02	5.04E+03	2.14E+03	3.23E+04	2.60E+05	1.32E+04	2.30E+03	**1.59E+02**
	Std	2.90E+02	2.11E+03	5.07E+02	3.45E+03	8.86E+04	2.26E+03	1.05E+03	**1.17E+02**
	Min	6.81E+01	1.88E+03	9.31E+02	2.64E+04	1.24E+05	9.71E+03	3.74E+02	**2.57E+01**
	Time(s)	4.11	5.73	13.37	57.02	78.54	22.04	9.56	17.9
f_3	Mean	3.96E+00	2.13E+01	9.46E+00	4.86E+01	9.34E+01	3.60E+01	**2.65E-03**	7.25E+00
	Std	8.34E-01	3.37E+00	1.44E+00	3.04E+00	5.41E+00	7.42E+00	**4.05E-03**	1.75E+00
	Min	2.76E+00	1.36E+01	5.67E+00	4.13E+01	7.61E+01	2.01E+01	**1.82E-27**	3.51E+00
	Time(s)	0.17	0.35	3.48	50.12	6.07	18.74	5.99	13.49
f_4	Mean	3.82E+02	1.28E+01	9.31E+00	6.89E-01	1.29E+02	2.13E-01	1.53E-02	**1.22E-02**
	Std	1.43E+02	4.21E+00	1.80E+00	1.83E-01	3.52E+01	9.11E-02	4.02E-02	**3.61E-02**
	Min	1.43E+00	5.78E+00	6.59E+00	3.55E-01	5.86E+01	8.53E-02	1.13E-12	**8.91E-04**
	Time(s)	0.24	0.49	3.76	50.23	8.1	19.56	5.95	13.61
f_5	Mean	8.34E+01	2.02E+08	1.80E+08	1.15E+07	2.59E+07	1.84E-01	4.62E+01	**6.12E-02**
	Std	4.49E+02	1.18E+08	4.52E+07	7.23E+06	5.35E+07	5.09E-02	3.42E-01	**2.28E-01**
	Min	2.21E-07	5.01E+07	1.12E+08	2.14E+06	1.44E+03	9.23E-02	4.52E+01	**5.74E-10**
	Time(s)	0.83	1.17	5.05	50.75	17.98	18.56	6.67	13.73
f_6	Mean	1.29E+02	7.15E+02	4.58E+02	6.75E+02	2.69E+05	2.85E+02	4.63E+01	**4.34E+01**
	Std	8.25E+01	4.28E+02	1.60E+02	1.79E+02	2.31E+05	8.43E+01	4.06E-01	**2.37E+01**
	Min	3.68E+01	2.75E+02	2.09E+02	3.72E+02	2.47E+04	1.74E+02	4.55E+01	**3.38E+01**
	Time(s)	3.8	5.16	12.41	55.57	69.34	20.7	8.92	16.83
f_7	Mean	**1.52E+04**	1.53E+04	**1.52E+04**	1.74E+04	5.94E+05	1.75E+04	1.53E+04	1.59E+04
	Std	**3.10E+02**	3.06E+02	**3.02E+02**	2.33E+02	4.07E+05	2.13E+02	3.21E+02	3.68E+02
	Min	**1.42E+04**	1.46E+04	**1.46E+04**	1.68E+04	1.30E+05	1.67E+04	1.47E+04	1.50E+04
	Time(s)	6.04	9.25	19.09	62.09	117.24	22.88	11.61	19.72
f_8	Mean	3.77E+03	3.77E+03	3.77E+03	**3.76E+03**	7.70E+03	3.77E+03	3.77E+03	**3.76E+03**
	Std	1.74E+00	1.89E+00	1.91E+00	**6.93E-01**	8.15E+00	1.17E+00	1.89E+00	**1.69E+00**
	Min	3.77E+03	3.77E+03	3.77E+03	**3.76E+03**	7.70E+03	3.77E+03	3.77E+03	**3.76E+03**
	Time(s)	7.37	10.47	21.86	62.1	134.85	23.57	12.56	21.35
f_9	Mean	5.94E-01	5.79E+00	2.60E+00	4.32E+00	1.16E+02	6.25E+00	1.03E-02	**2.63E-03**
	Std	5.54E-01	2.89E+00	9.49E-01	1.18E+00	1.31E+02	2.57E+00	1.35E-02	**6.72E-03**
	Min	7.54E-03	8.72E-01	9.80E-01	2.50E+00	1.85E+01	2.78E+00	2.31E-34	**1.48E-04**
	Time(s)	0.23	0.51	3.79	50.26	8.07	18.61	5.95	13.56
f_{10}	Mean	−9.97E+02	−1.05E+04	−9.49E+03	−1.20E+04	**−6.21E+65**	−1.19E+04	−1.14E+04	−1.02E+04
	Std	1.19E+02	1.31E+03	1.28E+03	3.56E+02	**1.76E+66**	3.61E+02	4.02E+03	1.40E+03
	Min	−1.19E+03	−1.34E+04	−1.17E+04	−1.28E+04	**−7.82E+66**	−1.27E+04	−2.40E+04	−1.30E+04
	Time(s)	0.23	0.51	3.74	50.42	7.82	18.57	6.27	13.56
f_{11}	Mean	2.47E+02	9.38E+01	7.61E+01	1.07E+02	8.90E+02	3.19E+01	4.02E-02	**0.00E+00**
	Std	7.48E+01	1.67E+01	1.65E+01	1.07E+01	3.01E+02	9.15E+00	2.03E-01	**0.00E+00**
	Min	7.48E+01	5.24E+01	4.71E+01	8.68E+01	5.91E+02	1.61E+01	3.55E-15	**0.00E+00**
	Time(s)	0.28	0.66	4.02	49.9	9.53	18.37	6	13.35
f_{12}	Mean	−2.49E+03	−4.21E+03	−3.68E+03	−4.35E+03	−1.98E+03	**−4.43E+03**	−4.33E+03	−4.39E+03
	Std	4.78E+02	1.39E+02	3.10E+02	2.12E+01	2.82E+03	**7.08E+00**	5.97E+02	-4.39E+03
	Min	-3.21E+03	-4.39E+03	-4.08E+03	-4.39E+03	-7.36E+03	**-4.46E+03**	-4.50E+03	-4.39E+03
	Time(s)	9.4	15.14	29.93	70.59	173.73	25.12	6.21	23.46
f_{13}	Mean	3.23E+00	7.72E+00	4.43E+00	3.13E+00	2.07E+01	3.24E+00	1.98E+01	**2.18E+00**
	Std	8.65E-01	1.40E+00	5.13E-01	4.26E-01	6.79E-01	1.54E+00	4.82E-01	**6.61E-01**
	Min	1.87E+00	5.01E+00	3.57E+00	2.50E+00	1.98E+01	1.89E-01	1.77E+01	**1.07E-06**
	Time(s)	0.43	0.97	4.71	58.41	13.64	18.89	6.12	14.08
f_{14}	Mean	6.98E-03	2.83E+00	2.93E+00	1.27E+00	1.58E+02	5.76E-01	1.02E+00	**1.90E-03**
	Std	8.45E-03	1.15E+00	4.89E-01	1.29E-01	7.34E+01	2.34E-01	2.21E-01	**6.71E-03**
	Min	7.74E-10	1.23E+00	1.78E+00	1.10E+00	6.96E+01	2.37E-01	3.08E-01	**7.66E-15**
	Time(s)	2.63	4.31	10.92	62.59	51.54	19.81	8.21	16.17
f_{15}	Mean	5.97E-01	1.14E+01	3.26E+00	4.06E+00	2.38E+07	5.97E-01	**1.68E-02**	3.21E-01
	Std	8.78E-01	4.13E+00	1.15E+00	2.20E+00	2.54E+07	6.61E-01	**5.81E-03**	4.46E-01
	Min	9.00E-06	5.23E+00	1.46E+00	1.13E+00	2.03E+05	3.95E-03	**8.91E-03**	2.43E-13
	Time(s)	4.05	5.89	14	67.71	83.99	21.58	9.44	17.57
f_{16}	Mean	5.03E-02	1.34E+02	1.98E+01	4.32E+00	1.70E+03	1.53E+00	1.40E+00	**5.50E-03**
	Std	5.03E-02	6.68E+01	4.34E+00	1.83E+00	7.10E+02	1.72E+00	4.16E-01	**8.85E-03**
	Min	1.16E-07	2.28E+01	1.33E+01	1.72E+00	8.78E+02	1.60E-01	7.02E-01	**8.85E-03**
	Time(s)	9.73	14.35	29.92	87.34	184.47	25.22	14.48	23.25

Table 9. Friedman test of mean values on unimodal functions.

Average Rank	Algorithm	Ranking	20D		30D		50D	
			Algorithm	Ranking	Algorithm	Ranking	Algorithm	Ranking
1	LFIACL-PSO	1.53	LFIACL-PSO	1.92	LFIACL-PSO	1.33	LFIACL-PSO	1.33
2	XPSO	2.31	XPSO	2.42	XPSO	2.17	XPSO	2.33
3	PSO	3.50	PSO	2.83	PSO	4.00	PSO	3.67
4	CLPSO-LOT	4.72	CLPSO-LOT	5.67	CLPSO-LOT	4.33	CLPSO-LOT	4.17
5	DMSPSO	5.00	DMSPSO	4.58	DMSPSO	5.33	DMSPSO	5.08
6	KPSO	5.34	KPSO	4.67	KPSO	5.67	CLPSO	5.67
7	CLPSO	6.42	CLPSO	7.00	CLPSO	6.00	KPSO	6.25
8	TSLPSO	7.20	TSLPSO	6.92	TSLPSO	7.17	TSLPSO	7.50

Table 10. Friedman test of mean values on multimodal functions.

Average Rank	Algorithm	Ranking	20D		30D		50D	
			Algorithm	Ranking	Algorithm	Ranking	Algorithm	Ranking
1	LFIACL-PSO	2.23	LFIACL-PSO	2.15	LFIACL-PSO	2.40	LFIACL-PSO	2.15
2	CLPSO-LOT	3.87	CLPSO-LOT	4.15	CLPSO-LOT	3.50	CLPSO-LOT	3.95
3	XPSO	4.05	XPSO	4.24	XPSO	4.35	XPSO	3.55
5	CLPSO	4.45	CLPSO	4.90	CLPSO	4.30	CLPSO	4.15
4	PSO	4.33	PSO	4.45	PSO	4.25	PSO	4.30
6	DMSPSO	5.03	DMSPSO	5.10	DMSPSO	4.95	DMSPSO	5.05
7	KPSO	5.30	KPSO	4.80	KPSO	5.55	KPSO	5.55
8	TSLPSO	6.97	TSLPSO	6.90	TSLPSO	6.70	TSLPSO	7.30

domain CD. In that case, the hypothesis that the two algorithms perform equally is rejected with the corresponding confidence level.

Table 11. Friedman test of mean values on different dimensions.

	PSO	KPSO	DMSPSO	CLPSO	TSLPSO	CLPSO-LOT	XPSO	LFIACL-PSO
20D	3.64	4.74	4.84	5.95	6.91	4.91	3.33	2.04
30D	4.13	5.61	5.14	5.15	6.94	3.92	3.26	1.87
50D	3.99	5.90	5.07	4.91	7.40	4.06	2.94	1.74
Ave rank	3.92	5.42	5.02	5.34	7.08	4.30	3.18	1.77
Final rank	3	7	5	6	8	4	2	1

$$CD = q_\alpha \sqrt{\frac{K(K+1)}{6N}}, \tag{4.3}$$

where $\alpha = 0.05$, $q_\alpha = 3.031$, $K = 8$ denotes the number of algorithms, $N = 16$, indicating the data set.

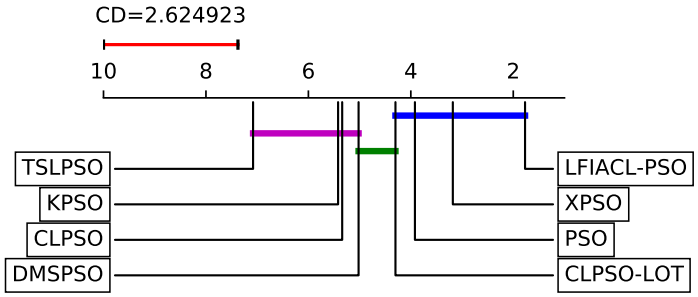


Figure 3. Critical difference with Friedman rankings.

As can be seen from Figure 3, the LFIACL-PSO significantly outperforms the TSLPSO, CLPSO, CLPSO-LOT, KPSO, and DMSPSO in terms of performance. The result suggests that LFIACL-PSO does not significantly differ with XPSO, PSO, and CLPSO-LOT, though it offers the best performance in terms of the Friedman ranking values.

4.5. Comparison of CPU time

Tables 6–8 records the CPU time consumed by each algorithm on 16 test functions ($f_1 - f_{16}$) in different dimensions. Table 12 shows the comprehensive ranking results of CPU time. The results show that the LFIACL-PSO consumes more CPU time compared to the other compared algorithms, with an average sequential value of 6.40, ranking 6th. Ranking 7th and 8th are the TSLPSO and CLPSO, whose average ordinal values are 6.46 and 6.70, respectively. There are two main reasons why the LFIACL-PSO algorithm is more time-consuming. Firstly, the learning paradigm of the particle changes from one to four. Secondly, adopt a comprehensive learning strategy as the self-learning part of particles to update the learning paradigm of particles and increase the time of particle search.

Table 12. Friedman test of CPU time on different dimensions.

	PSO	KPSO	DMSPSO	CLPSO	TSLPSO	CLPSO-LOT	XPSO	LFIACL-PSO
20D	2.22	2.91	1.00	5.88	6.44	6.19	3.94	7.44
30D	1.06	2.19	3.75	7.56	6.44	5.38	3.13	6.50
50D	1.00	2.06	3.81	7.56	6.50	6.38	3.44	5.25
Ave rank	1.43	2.39	2.85	6.70	6.46	5.98	3.50	6.40
Final rank	1	2	3	8	7	5	4	6

Figure 4 shows the combined performance graph of LFIACL-PSO and other comparison algorithms. It is a combined ranking graph based on the accuracy of the benchmark function and CPU time. It can be seen from Figure 4 that the closer the circle is to the lower-left corner in the figure, the better the performance of the algorithm. From Figure 4, we can see that LFIACL-PSO and XPSO have the better solution accuracy among all algorithms, showing similar performance. While LFIACL-PSO is more accurate than XPSO, it requires more CPU time. This result indicates that LFIACL-PSO has higher

solution accuracy and performs well in terms of algorithmic complexity. The PSO and KPSO consume less CPU time but have lower solution accuracy.

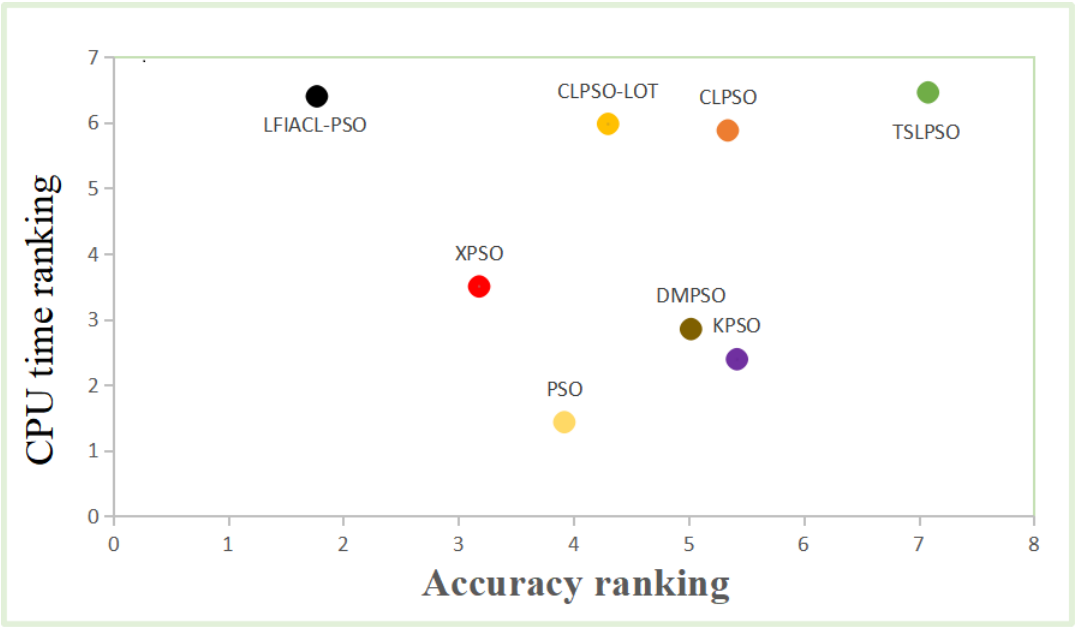


Figure 4. Comprehensive performance of all algorithms.

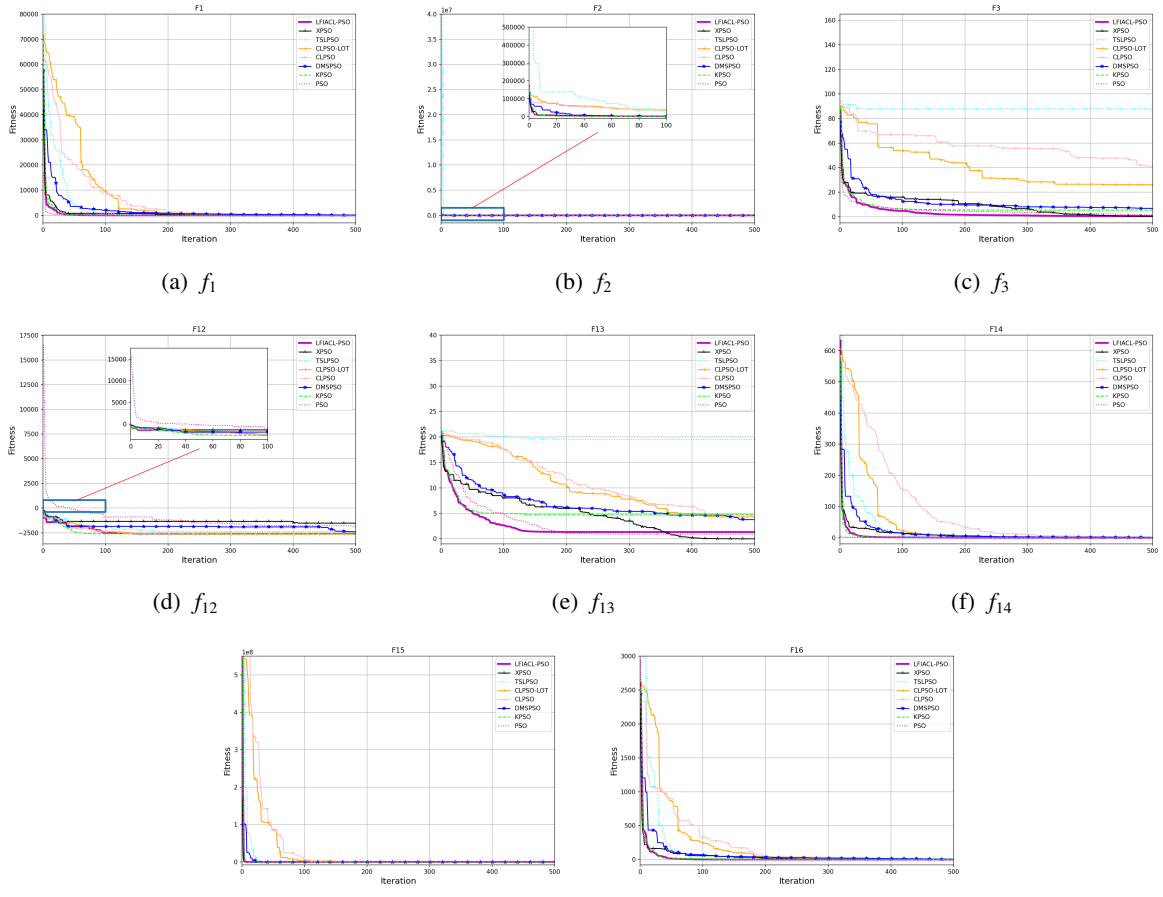
4.6. Comparison of convergence speed

To test the convergence of the LFIACL-PSO algorithm and the seven comparison algorithms, we selected eight benchmark functions to do experiments. These 8 benchmark functions include 3 unimodal functions ($f_1 - f_3$) and 5 multimodal functions ($f_{12} - f_{16}$). Population size: $PS = 40$, search space: $N = 30$ dimensions, maximum number of iterations: $Iter\_Max = 500$, and 30 independent runs of each function. Figure 5 shows the experimental results and zooms in on the dense area in the figure. From Figure 5, as the number of iterations increases, the LFIACL-PSO algorithm achieves the best convergence performance overall on the randomly selected benchmark functions $f_1 - f_3$, $f_{13} - f_{16}$. On the multimodal function f_{12}, it is slightly lower than the KPSO algorithm. In general, the LFIACL-PSO algorithm proposed in this paper shows good convergence speed and convergence accuracy on both unimodal functions and multimodal functions.

4.7. LFIACL-PSO performance for a real-world problem

In this part, we will use a widely used real-world optimization problem to verify the LFIACL-PSO performance. Dukic et al. [47] 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. It can be modeled as a minimum-maximum nonlinear optimization problem, defined as follows [48]:

$$Global_{x\in X} min f(x) = max\{\phi_1(x), ..., \phi_{2m}(x)\}, \tag{4.4}$$



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

(d) f_{12} (e) f_{13} (f) f_{14}

(g) f_{15} (h) f_{16}

Figure 5. The convergence curves under 30 dimensions for 3 unimodal functions and 5 multimodal functions.

$$X = \{(x_1, ..., x_n)x \in R^n | 0 \leq x_j \leq 2\pi, j = 1, ..., n\}, \tag{4.5}$$

where $m = 2n - 1$, and

$$\phi_{2i-1}(x) = \sum_{j=i}^{n} cos\left(\sum_{k=|2i-j-1|+1}^{j} x_k\right), i = 1, ..., n, \tag{4.6}$$

$$\phi_{2i}(x) = 0.5 + \sum_{j=i+1}^{n} cos\left(\sum_{k=|2i-j|+1}^{j} x_k\right), i = 1, ..., n-1, \tag{4.7}$$

$$\phi_{m+1}(x) = -\phi_i(x), i = 1, ..., m. \tag{4.8}$$

Table 13 shows in detail the performance of all the algorithms on the spread spectrum radar polyphase code design problem. It can be seen from Table 14 that LFIACL-PSO is superior to the other seven algorithms. In summary, the LFIACL-PSO algorithm has outstanding performance advantages in benchmark functions and a solid ability to solve practical engineering problems, which has considerable potential application value.

Table 13. Comparison on the spread spectrum radar poly phase code design problem.

		PSO	KPSO	DMSPSO	CLPSO	TSLPSO	CLPSO-LOT	XPSO	LFIACL-PSO
20D	Mean	1.69E+00	2.00E+00	3.73E+00	4.34E+00	4.21E-02	1.92E+00	2.02E-02	**1.01E-02**
	Std	1.37E+00	1.61E+00	1.47E+00	1.23E+00	8.07E-02	8.58E-01	1.84E-02	**8.04E-03**
	Min	4.12E-03	5.46E-04	6.58E-01	1.12E+00	4.44E-16	9.71E-04	9.77E-05	**8.20E-04**
30D	Mean	3.37+00	4.58E+00	7.29E+00	7.81E+00	2.00E-01	4.56E+00	1.11E-01	**9.35E-02**
	Std	2.12E+00	1.80E+00	1.59E+00	1.53E+00	4.98E-01	9.10E-01	1.66E-01	**8.16E-02**
	Min	3.78E-02	1.78E-01	4.13E+00	4.41E+00	1.89E-15	2.54E+00	1.88E-03	**2.06E-03**
50D	Mean	9.78E+00	9.30E+00	1.47E+01	1.38E+01	3.12E+00	1.01E+01	**6.53E-01**	9.65E-01
	Std	3.72E+00	3.82E+00	2.54E+00	2.70E+00	2.88E+00	1.68E+00	**8.49E-01**	7.89E-01
	Min	1.09E-01	7.93E-02	1.05E+01	5.58E+00	6.11E-16	7.10E+00	**2.73E-02**	2.12E-02

Table 14. Ranks achieved by the Friedman test of mean values.

	PSO	KPSO	DMSPSO	CLPSO	TSLPSO	CLPSO-LOT	XPSO	LFIACL-PSO
20D	4	6	7	8	3	5	2	1
30D	4	6	7	8	3	5	2	1
50D	5	4	8	7	3	6	1	2
Ave rank	4.33	5.33	7.33	7.67	3.00	5.33	1.67	1.33
Final Rank	4	5	6	7	3	5	2	1

5. Conclusions

Aiming at the disadvantage that particle swarm optimization easily falls into local optimum, we propose the LFIACL-PSO algorithm. The implementation of LFIACL-PSO is described in detail in the paper, and sensitivity analysis of the newly introduced parameters is performed through extensive experiments. In addition, the comparison of LFIACL-PSO with seven other variants of PSO algorithms on 16 benchmark functions and a real problem validates the benefits of LFIACL-PSO. The comparison results show that LFIACL-PSO has the following advantages. Firstly, the use of comprehensive learning strategies and Ring-type topology as part of the learning paradigm makes the particles more informative when updating iterations, thus maintaining the diversity of particles. Secondly, when the particle falls into the local optimum and cannot jump out, let the particle learn from its own worst position and use the Levy flight to obtain the learning step size to enhance the global exploration ability of the population. Finally, use the historical information of elite particles in the population to update the acceleration coefficients to meet the different requirements of different evolutionary stages.

Although LFIACL-PSO has good performance, there are still some problems in the LFIACL-PSO algorithm. For example, the LFIACL-PSO algorithm has high solution accuracy but consumes more CPU time, which needs further research in our future work. What's more, except the methods used in the paper, some of the most representative computational intelligence algorithms can be used to solve the problems, like monarch butterfly optimization (MBO), earthworm optimization algorithm (EWA), elephant herding optimization (EHO), moth search (MS) algorithm, Slime mould algorithm (SMA), and Harris hawks optimization (HHO) and so on.

Acknowledgments

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

The authors declare there is no conflict of interest.

References

1. J. Kennedy, R. Eberhart, Particle swarm optimization, in *Icnn95-international Conference on Neural Networks*, **4** (1995), 1942–1948. https://doi.org/10.1109/ICNN.1995.488968
2. S. M. Mirjalili, A. Lewis, The whale optimization algorithm, *Adv. Eng. Software*, **95** (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
3. X. S. Yang, Firefly algorithm, stochastic test functions and design optimisation, *Int. J. Bio-Inspired Comput.*, **2** (2010), 78–84. https://doi.org/10.1504/IJBIC.2010.032124
4. S. M. Mirjalili, The ant lion optimizer, *Adv. Eng. Software*, **83** (2015), 80–98. https://doi.org/10.1016/j.advengsoft.2015.01.010
5. R. Rajabioun, Cuckoo optimization algorithm, *Appl. Soft Comput.*, **11** (2011), 5508–5518. https://doi.org/0.1016/j.asoc.2011.05.008
6. D. E. Goldberg, J. H. Holland, Genetic algorithms and machine learning, *Mach. Learn.*, **3** (2005), 95–99. https://doi.org/10.1023/A:1022602019183
7. G. G. Wang, S. Deb, Z. H. Cui, Monarch butterfly optimization, *Neural Comput. Appl.*, **31** (2019), 1995–2014. https://doi.org/10.1007/s00521-015-1923-y
8. G. G. Wang, S. Deb, L. D. S. Coelho, Earthworm optimisation algorithm: a bio-inspired metaheuristic algorithm for global optimisation problems, *Int. J. Bio-Inspired Comput.*, **12** (2018), 1–22. https://doi.org/10.1504/IJBIC.2015.10004283
9. G. G. Wang, S. Deb, L. D. S. Coelho, Elephant herding optimization, in *2015 3rd International Symposium on Computational and Business Intelligence (ISCBI)*, (2015), 1–5. https://doi.org/10.1109/ISCBI.2015.8
10. G. G. Wang, Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems, *Memetic Comput.*, **10** (2018), 151–164. https://doi.org/10.1007/s12293-016-0212-3
11. R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, (1995), 39–43. https://doi.org/10.1109/MHS.1995.494215

12. W. Li, G. G. Wang, A. H. Gandomi, A survey of learning-based intelligent optimization algorithms, *Arch. Comput. Methods Eng.*, **28** (2021), 3781–3799. https://doi.org/10.1007/S11831-021-09562-1

13. Y. Feng, S. Deb, G. G. Wang, A. H. Gandomi, Monarch butterfly optimization: a comprehensive review, *Exp. Syst. Appl.*, **168** (2021), 114418. https://doi.org/10.1016/j.eswa.2020.114418

14. S. Zhou, Y. Han, L. Sha, S. Zhu, A multi-sample particle swarm optimization algorithm based on electric field force, *Math. Biosci. Eng.*, **18** (2021), 7464–7489. https://doi.org/10.3934/mbe.2021369

15. Y. Feng, G. G. Wang, S. Deb, M. Lu, X. J. Zhao, Solving 0–1 knapsack problem by a novel binary monarch butterfly optimization, *Neural Comput. Appl.*, **28** (2017), 1619–1634. https://doi.org/10.1007/s00521-015-2135-1

16. Y. Feng, G. G. Wang, J. Dong, L. Wang, Opposition-based learning monarch butterfly optimization with gaussian perturbation for large-scale 0-1 knapsack problem, *Comput. Electr. Eng.*, **67** (2018), 454–468. https://doi.org/10.1016/j.compeleceng.2017.12.014

17. F. Liu, Y. Sun, G. Wang, T. Wu, An artificial bee colony algorithm based on dynamic penalty and lévy flight for constrained optimization problems, *Arabian J. Sci. Eng.*, **43** (2018), 7189–7208. https://doi.org/10.1007/S13369-017-3049-2

18. L. Guo, G. G. Wang, A. H. Gandomi, A. H. Alavi, H. Duan, A new improved krill herd algorithm for global numerical optimization, *Neurocomputing*, **138** (2014), 392–402. https://doi.org/10.1016/j.neucom.2014.01.023

19. I. Hanafi, F. M. Cabrera, F. Dimane, J. T. Manzanares, Application of particle swarm optimization for optimizing the process parameters in turning of peek cf30 composites, *Proc. Technol.*, **22** (2016), 195–202. https://doi.org/10.1016/J.PROTCY.2016.01.044

20. J. Lu, J. Zhang, J. Sheng, Enhanced multi-swarm cooperative particle swarm optimizer, *Swarm Evol. Comput.*, **69** (2022), 100989. https://doi.org/10.1016/j.swevo.2021.100989

21. H. Zhang, M. Yuan, Y. Liang, L. Qi, A novel particle swarm optimization based on prey-predator relationship, *Appl. Soft Comput.*, **68** (2018), 202–218. https://doi.org/10.1016/j.asoc.2018.04.008

22. 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. https://doi.org/10.1109/TEVC.2005.857610

23. X. Xia, L. Gui, F. Yu, H. Wu, B. Wei, Y. Zhang, et al., Triple archives particle swarm optimization, *IEEE Trans. Cyber.*, **50** (2020), 4862–4875. https://doi.org/10.1109/TCYB.2019.2943928

24. N. Lynn, P. N. Suganthan, Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation, *Swarm Evol. Comput.*, **24** (2015), 11–24. https://doi.org/10.1016/j.swevo.2015.05.002

25. X. Zhang, W. Sun, M. Xue, A. Lin, Probability-optimal leader comprehensive learning particle swarm optimization with bayesian iteration, *Appl. Soft Comput.*, **103** (2021), 107132. https://doi.org/10.1016/j.asoc.2021.107132

26. Y. Shi, R. C. Eberhart, A modified particle swarm optimizer, in *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360)*, (1998), 69–73. https://doi.org/10.1109/ICEC.1998.699146
27. X. Xia, L. Gui, G. He, B. Wei, Y. Zhang, F. Yu, et al., An expanded particle swarm optimization based on multi-exemplar and forgetting ability, *Inf. Sci.*, **508** (2020), 105–120. https://doi.org/10.1016/j.ins.2019.08.065
28. G. G. Wang, A. H. Gandomi, X. S. Yang, A. H. Alavi, A novel improved accelerated particle swarm optimization algorithm for global numerical optimization, *Eng. Comput.*, **2014** (2014). https://doi.org/10.1108/EC-10-2012-0232
29. S. Mirjalili, G. G. Wang, L. D. S. Coelho, Binary optimization using hybrid particle swarm optimization and gravitational search algorithm, *Neural Comput. Appl.*, **25** (2014), 1423–1435. https://doi.org/10.1007/s00521-014-1629-6
30. O. Kahouli, H. Alsaif, Y. Bouteraa, N. B. Ali, M. Chaabene, Power system reconfiguration in distribution network for improving reliability using genetic algorithm and particle swarm optimization, *Appl. Sci.*, **11** (2021), 3092. https://doi.org/10.3390/APP11073092
31. A. Lin, W. Sun, H. Yu, G. Wu, H. Tang, Global genetic learning particle swarm optimization with diversity enhancement by ring topology, *Swarm Evol. Comput.*, **44** (2019), 571–583. https://doi.org/10.1016/j.swevo.2018.07.002
32. E. Naderi, M. Pourakbari-Kasmaei, M. Lehtonen, Transmission expansion planning integrated with wind farms: a review, comparative study, and a novel profound search approach, *Int. J. Electr. Power Energy Syst.*, **115** (2020), 05460. https://doi.org/10.1016/J.IJEPES.2019.105460
33. R. Jamous, H. Alrahhal, M. El-Darieby, A new ann-particle swarm optimization with center of gravity (ann-psocog) prediction model for the stock market under the effect of covid-19, *Sci. Prog.*, **2021** (2021), 6656150. https://doi.org/10.1155/2021/6656150
34. B. Mohammadi, Y. Guan, R. Moazenzadeh, M. J. S. Safari, Implementation of hybrid particle swarm optimization-differential evolution algorithms coupled with multi-layer perceptron for suspended sediment load estimation, *Catena*, **198** (2020), 105024. https://doi.org/10.1016/j.catena.2020.105024
35. M. Clerc, The swarm and the queen: towards a deterministic and adaptive particle swarm optimization, in *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, **3** (1999), 1951–1957. https://doi.org/10.1109/CEC.1999.785513
36. M. Asada, Modeling early vocal development through infantaregiver interaction: a review, *IEEE Trans. Cognit. Dev. Syst.*, **8** (2016), 128–138. https://doi.org/10.1109/TCDS.2016.2552493
37. A. Baeck, K. Maes, C. V. Meel, H. P. O. de Beeck, The transfer of object learning after training with multiple exemplars, *Front. Psychol.*, **7** (2016), 1386. https://doi.org/10.3389/fpsyg.2016.01386
38. K. E. Twomey, S. Ranson, J. S. Horst, That's more like it: Multiple exemplars facilitate word learning, *Infant Child Dev.*, **23** (2014), 105–122. https://doi.org/10.1002/ICD.1824

39. X. S. Yang, S. Deb, Cuckoo search via lévy flights, in *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*, (2009), 210–214. https://doi.org/10.1109/NABIC.2009.5393690
40. O. Aoun, M. Sarhani, A. E. Afia, Particle swarm optimisation with population size and acceleration coefficients adaptation using hidden markov model state classification, *Int. J. Metaheuristics*, **7** (2018), 1–29. https://doi.org/10.1504/IJMHEUR.2018.10012905
41. A. T. Kiani, M. F. Nadeem, A. N. Ahmed, I. Khan, H. I. Alkhammash, I. A. Sajjad, et al., An improved particle swarm optimization with chaotic inertia weight and acceleration coefficients for optimal extraction of pv models parameters, *Energies*, **14** (2021), 2980. https://doi.org/10.3390/EN14112980
42. Z. H. Zhan, J. Zhang, Y. Li, H. S. H. Chung, Adaptive particle swarm optimization, *IEEE Trans. Syst. Man Cybern. B*, **39** (2009), 1362–1381. https://doi.org/10.1109/TSMCB.2009.2015956
43. J. J. Liang, P. N. Suganthan, Dynamic multi-swarm particle swarm optimizer, in *Proceedings 2005 IEEE Swarm Intelligence Symposium*, (2005), 124–129. https://doi.org/10.1109/SIS.2005.1501611
44. G. Xu, Q. Cui, X. Shi, H. W. Ge, Z. H. Zhan, H. P. Lee, et al., Particle swarm optimization based on dimensional learning strategy, *Swarm Evol. Comput.*, **45** (2019), 33–51. https://doi.org/10.1016/J.SWEVO.2018.12.009
45. K. Zhang, Q. Huang, Y. Zhang, Enhancing comprehensive learning particle swarm optimization with local optima topology, *Inf. Sci.*, **471** (2019), 1–18. https://doi.org/10.1016/j.ins.2018.08.049
46. O.Olorunda, A. P. Engelbrecht, Measuring exploration/exploitation in particle swarms using swarm diversity, in *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, (2008), 1128–1134. https://doi.org/10.1109/CEC.2008.4630938
47. 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. https://doi.org/10.1109/49.56381
48. S. Gil-Lopez, J. D. Ser, S. Salcedo-Sanz, Á. M. Pérez-Bellido, J. M. Cabero, J. A. Portilla-Figueras, A hybrid harmony search algorithm for the spread spectrum radar polyphase codes design problem, *Expert Syst. Appl.*, **39** (2012), 11089–11093. https://doi.org/10.1016/j.eswa.2012.03.063

AIMS Press