Applied Soft Computing 121 (2022) 108731

Contents lists available at ScienceDirect

Applied Soft Computing

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

Pyramid particle swarm optimization with novel strategies of competition and cooperation

Taiyong Li [a,*], Jiayi Shi [a], Wu Deng [b], Zhenda Hu [c]

[a] Southwestern University of Finance and Economics, Chengdu 611130, China
[b] College of Electronic Information and Automation, Civil Aviation University of China, Tianjin 300300, China
[c] School of Information Management and Engineering, Shanghai University of Finance and Economics, Shanghai 200433, China

ARTICLE INFO

Article history:
Received 28 August 2021
Received in revised form 27 February 2022
Accepted 7 March 2022
Available online 18 March 2022

Keywords:
Particle swarm optimization (PSO)
Numerical optimization
Pyramid structure
Evolutionary computation
Competition and cooperation

ABSTRACT

Particle swarm optimization (PSO) has shown its advantages in various optimization problems. Topology and updating strategies are among its key concepts and have significant impacts on optimization ability. This paper proposes a pyramid PSO (PPSO) with novel competitive and cooperative strategies to update particles' information. PPSO builds a pyramid and assigns each particle to a specific layer according to its fitness. The particles at the same layer will make a pairwise comparison to determine the winners and the losers. The losers will cooperate with their corresponding winners, while the winners will cooperate with the particles at the upper layer and those at the top layer. Each particle in PPSO has its own learning behavior, having more than one exemplar rather than the only global best to learn from. The diversity of the swarm is enhanced and it positively affects the performance of PSO. Extensive experiments demonstrate that the PPSO has superior performance in terms of accuracy, Wilcoxon signed-rank test and convergence speed, yet achieves comparable running time in most cases, when compared with the canonical PSO and eight state-of-the-art PSO variants. Furthermore, we analyze the influence of parameters for the PPSO. All these illustrate that the PPSO is promising for numerical optimization.

1. Introduction

Evolutionary computation (EC) has constantly attracted concerns from researchers for optimization. It has many forms, such as genetic algorithm (GA) [1], differential evolution (DE) [2], particle swarm optimization (PSO) [3,4], artificial bee colony (ABC) [5], gray wolf optimization (GWO) [6], and so on. Many variants of these algorithms have emerged and they have been widely applied in many fields. For example, Vasile's group proposed structured-chromosome GA for stochastic satellite tracking [7]. They also studied adaptive multi-population inflationary DE algorithm and used it to solve constrained min–max problems [8,9]. Savvaris' group and Chung's group used several EC algorithms for trajectory optimization problems and achieved good results [10–13].

Among the EC algorithms, PSO has become more and more popular for its simplicity and high effectiveness in the last decades. It was inspired by the flocking behaviors of seeking food of birds, in which the position of a bird (particle) is updated by its own best historical position and the position of the global best bird [3,4]. In other words, a particle iteratively learns from its own experience and exchanges information with the best particle in the evolutionary procedure. When it meets some requirements, the evolution completes, and the best particle's position denotes the optimal solution. PSO has shown its power in various real-world optimization problems [14–18].

Despite its success in applications, the canonical PSO still has some limitations, which lead to premature convergence and lapsing into the local optimum, especially in some complex problems. To address this issue, researchers have improved the canonical PSO from several aspects, including parameter tuning [19–21], topological structure [22–24], learning strategy [25–27], combination with other EC algorithms [28,29], and so on. These PSO variants have improved the searching ability over the canonical PSO.

Parameter tuning mainly focuses on selecting the inertia weight, population size, accelerating coefficient, and others. For example, the initial PSO had no concept of the inertia weight. It directly added a change decided by the distance between the current particle and the global best particle to the current velocity. In other words, the inertia weight of the initial PSO was fixed to "1" [3]. Later, Shi and Eberhart used a time-varying inertia weight to balance the exploration and exploitation abilities of PSO [30]. The inertia weight is problem-dependent, and hence it is hard to use a general strategy for all problems. Therefore,

* Corresponding author.
E-mail address: litaiyong@gmail.com (T. Li).

https://doi.org/10.1016/j.asoc.2022.108731
1568-4946/

more research aimed at studying how to choose adaptive inertia weights for specific problems [31–34].

The topology of PSO determines each particle's neighbors, and hence it has an important impact on whom the current particles exchange information with during evolution [35]. Widely existing topologies include ring topology, network topology, chain topology, hierarchical topology, random topology, and others [36]. Scholars have found that the topology plays an essential role in improving PSO's performance [37].

The learning strategy concerns how a particle learns from other particles in PSO. Many PSO variants use a uniform learning strategy for all the particles in a swarm, ignoring their difference and decreasing the population diversity. Hence, they tend to result in a local optimal solution and premature convergence [38,39]. An ideal strategy is that different particles use different strategies to update their information according to their characteristics. To this end, a competitive strategy is necessary.

Each EC algorithm has its own strategies of evolution. The idea in one EC algorithm may be introduced into another and vice versa. The combination of PSO and some other EC algorithms has become a very popular way to improve PSO's performance in recent years [40,41]. Tang et al. presented a combination of PSO and modified DE. The experiments reveal that the proposed scheme can well balance PSO's global and local search capabilities [42]. Li et al. presented an approach to combine PSO and ABC for high-dimensional optimization problems [43]. Other combination forms, such as PSO and cuckoo search algorithm (CSO), PSO and GA, and PSO and firefly algorithm (FA) were also studied [44–46]. The studies reveal that all the combinations can improve PSO's performance.

The pyramid structure is a simple but popular hierarchical structure that splits objects into a different hierarchy according to their quality. This structure exists widely in society, but little research has focused on it in the field of PSO. With this structure, different learning strategies can be designed. Inspired by the analysis, this paper proposes a pyramid PSO (PPSO) algorithm along with novel strategies of competition and cooperation to improve the performance of PSO.

The proposed PPSO focuses on topology and learning strategies. It has the following attributes: (1) A pyramid is built according to the particles' fitness. The better a particle, the higher its hierarchy in the pyramid; (2) The particles in the same hierarchy will be randomly paired for competition, and each particle will be labeled as either a winner or a loser; and (3) The winners and the losers will use different cooperative strategies to complete their evolution. Specifically, the winners will update their positions based on their own historical experience, the particles at a higher layer and the global best particles at the top layer, while the losers will only learn from their own historical experience as well as the winners at the same layer. The first two attributes can be thought of as a typical competitive strategy, while the last one is a cooperative strategy.

The key contributions of the proposed PPSO are summarized as follows: (1) A pyramid structure is introduced into PSO. Each particle will be assigned a specific layer in the pyramid according to the competition results of fitness. (2) The particles at one layer continue to compete to decide the winners and the losers. (3) The winners and the losers use different cooperative strategies to update their information. Besides their own experiences, the losers will learn from the paired winners while the winners learn from the particles at a higher layer and the global best particles at the top layer. In this way, the winners have more chances to learn from better particles. The different cooperative strategies enhance the swarm's diversity. (4) Extensive experiments demonstrate that the proposed PPSO significantly outperforms the state-of-the-art compared PSO approaches.

The rest of this paper is structured as follows: Section 2 reviews related work of PSO. We describe the proposed PPSO in detail in Section 3. The experimental settings and results are analyzed and discussed in Section 4. Finally, we conclude the paper and provide recommendations for future work in Section 5.

2. Related work

In this section, we briefly review the canonical PSO, the topology, and competitive and cooperative strategies, followed by some discussions on the concepts.

2.1. Canonical PSO

PSO is a representative swarm intelligent optimization algorithm that was initially proposed to simulate birds' behaviors of seeking food by Eberhart and Kennedy in 1995 [3]. As pointed out by Tang et al., PSO has the advantages of simplicity, effectiveness and low computational cost, and it has obtained significant popularity from researchers in recent years [47]. More specifically, PSO consists of N birds, also known as particles, and the motion of the particles is the evolutionary process of the swarm to search for the optimal solution in the $d-$dimensional search space D. Each particle has its own velocity and position at any time, and the position denotes a solution to a specific problem. For the ith particle at time t, its velocity and position can be formulated as $V_i^t = (v_{i,1}^t, v_{i,2}^t, \ldots, v_{i,d}^t)$ and $X_i^t = (x_{i,1}^t, x_{i,2}^t, \ldots, x_{i,d}^t)$, respectively. During the evolutionary process, PSO updates the velocities and positions via the following rules:

$$\begin{cases} V_i^{t+1} = w^t \cdot V_i^t + c_1 \cdot r_1 \cdot (B_i^t - X_i^t) + c_2 \cdot r_2 \cdot (G^t - X_i^t) \\ X_i^{t+1} = X_i^t + V_i^{t+1} \end{cases}, \tag{1}$$

where w_t is called inertia weight which controls the velocity change, c_1 and c_2 are constants called acceleration factors, r_1 and r_2 are two random variables, B_i^t is the best historical position of the ith particle in the last t generations, and G^t denotes the global best position of all particles in the last t generations.

In the canonical PSO, w_t usually decreases linearly to balance the exploration and exploitation ability, as formulated by Eq. (2).

$$w_t = w_{max} + \frac{t}{T}(w_{max} - w_{min}), \tag{2}$$

where w_{max} and w_{min} denote the upper boundary and the lower boundary of the inertia weight, respectively, and T is the maximal numbers of evolutionary generations. In Eq. (1), the items $P_i^t - X_i^t$ and $G^t - X_i^t$ denote self-cognition and social-cognition, respectively. The canonical PSO can be thought of as having two layers. The higher layer has only one particle, i.e., the global best particle G^t, while the lower layer contains all the remaining particles. In such a structure, the particles at the lower layer cannot learn from each other but from the global best particle, hence limiting learning ability and losing diversity.

2.2. Topology of PSO

The topology is one of the key concepts in PSO, which defines the connections among particles in a swarm. To design better topologies, researchers have dedicated themselves to studying how the topologies can influence PSO's performance. The local best topology (*lbest*) and the global best topology (*gbest*) are two simple and mostly common topologies for PSO. In the former topology, each particle is associated with K particles in the neighborhood. Typically, PSO sets 2 for K to denote the two neighbors in the particle's both sides in a ring lattice. In the latter topology, each particle is associated with all the other particles in the

swarm [3,37]. Furthermore, pyramid, "small", star, and von Neumann were tested on several functions and measures in [3]. The authors found that the overall performance depended on function types, topologies, and measures, and none of the topologies is always superior to the others with all the functions and measures. A similar conclusion was drawn in [48], where four topologies, namely cycle, random graph, star, and wheel were evaluated on four functions with 30 dimensions. Wang et al. proposed a novel topology called multi-layer particle swarm optimization (MLPSO) which used multiple swarms to increase the diversity of searching space to improve the performance [49]. Motivated by the structure of mixed-level students, Yang et al. presented a level-based learning PSO for large-scale optimization [50]. With this topology, the presented PSO outperformed several state-of-the-art PSO variants in terms of computational efficiency and solution quality.

2.3. Competition and cooperation

The canonical PSO has a simple mechanism of competition and cooperation. In an iteration of the evolution, each particle competes with its corresponding historical best. If the former is better, then the latter will be replaced by the former. Another competition exists between each particle and the global best. The particle will also replace the global best if it has better fitness. When updating each particle's velocity in evolution, the particle cooperates with its historical best and the global best simultaneously, as shown in Eq. (1).

In [51], the authors introduced an Lotka–Volterra model to PSO and studied the intraspecific and interspecific strategy of cooperation and competition. They found that the strategy increased the diversity of PSO and achieved good performance. In [52], it was found that a cooperation mechanism associated with particles on hub and/or non-hub nodes of scale-free networks could improve diversity and flexibility. Cheng and Jin put forward a competitive swarm optimizer for large-scale optimization, in which a pairwise competition mechanism was introduced and the loser would learn from the corresponding winner. The extensive experiments demonstrated the algorithm achieved better performance than several compared state-of-the-art methods [53]. Zhang et al. classified the particles into different types via competition and then designed adaptive learning strategies to improve the diversity of PSO [26].

2.4. Discussions

Although there has existed some research on topologies of PSO, it lacks deep investigation to pyramid topologies. For example, the pyramid structure in [3] does not have the concept of "multiple layers". The MLPSO essentially adopts a typical tree structure, in which the particles at the same layer but in different swarms do not communicate with each other [49]. In real-world scenarios, there are many forms of "multiple layers", such as government officials, family trees, and academic titles. In such forms, some individuals (officials, family members, and researchers) appear at the same levels. They can be put together to compete and/or cooperate with each other and cooperate with the individuals in higher levels. The higher the level, the fewer the number of individuals. It is a typical pyramid, and it is possible to introduce such a structure into PSO to improve its performance. Motivated by this analysis, this paper proposes a pyramid PSO (PPSO) along with corresponding strategies of competition and cooperation.

3. Proposed algorithm

In this section, we first describe the fundamental concepts, i.e., how to build the pyramid and how to design the strategies of competition and cooperation, and then propose the PPSO in detail.

3.1. Pyramid building

Canonical PSO can be thought of as having two layers: the first layer with the global best particle and the second layer with all the remaining particles. In this way, the second layer particles can only learn from themselves and the only particle at the first layer. It lacks diversity and is prone to premature convergence and/or getting trapped in local optima.

Motivated by the phenomena of "multiple layers" in real-world scenarios, we use a pyramid topology to organize the particles. The pyramid operates following the shape of its name, with one or several leaders at the top, a small leadership team below, and layers of management or operations guiding down to the bottom teams. Specifically, the particles at the top layer can guide all the particles in the pyramid, while the particles at the other layers can guide the particles at the layers below only. This topology is capable of distributing the responsibility more evenly than the traditional topologies. At the same time, each particle serves only a couple of particles at the below layer. Their quality determines the layer of particles. The better a particle, the higher its layer in the pyramid.

Without loss of generality, this paper assumes the optimization problems are minimization problems and uses the fitness of a particle to decide its layer. Suppose the swarm has N particles $P = \{p_1, p_2, \ldots, p_N\}$ and the numbers of particles at all layers are $n_i (i = 1, 2, \ldots, L)$ where L is the number of layers and $N = \sum_{i=1}^{L} n_i$, we can sort the particles in ascending order according to their fitness ($F = \{f_{p_1}, f_{p_2}, \ldots, f_{p_N}\}$) and obtain the corresponding particles $P' = \{p'_1, p'_2, \ldots, p'_N\}$. Then the first n_1 particles in P' are assigned to the top layer in the pyramid, the next n_2 ones are assigned to the second layer, and so on. The process repeats until the last n_L particles in P' are placed at the bottom of the pyramid. Finally, the pyramid is completed. The building process can be described in Algorithm 1.

Algorithm 1 Pyramid building.

Input: N particles $P = \{p_1, p_2, \cdots, p_N\}$ and their fitness $F = \{f_{p_1}, f_{p_2}, \cdots, f_{p_N}\}$, the number of the layers of the pyramid L, and the number of the particles at each layer $n_i (i = 1, 2, \cdots, L)$

Output: Pyramid particles PP with L layers that each layer has $n_i (i = 1, 2, \cdots, L)$ particles

```
1: function PB(P, F, L, n_i)
2:     Sort P in ascending order according to fitness F and obtain the sorted particles P';
3:     idx1 ← 1;
4:     for i = 1 → L do
5:         idx2 ← idx1 + n_i - 1;
6:         PP_{i,1:n_i} ← P'_{idx1:idx2}; //assign the sorted idx1−th to idx2−th particles in P' to the i−th layer of the pyramid PP
7:         idx1 ← idx2 + 1;
8:     end for
9:     return PP;
10: end function
```

In Line 6 of Algorithm 1, $PP_{i,j}$ denotes the jth particle at the ith layer of the pyramid. We can see that the pyramid has the following attributes: (1) each particle is assigned to a specific layer; (2) the higher layer the particle at, the better the particle; the global best particle is located at the top layer while the worst particle is located at the bottom layer; (3) the particles at the same layer have close fitness.

3.2. Competitive strategy

Competition widely exists in evolutionary computation. Particularly, the canonical PSO has at least two types of competition: one for the local best and the other for the global best. When one particle is updated in the canonical PSO, it has to be compared with its historical best to determine whether the latter will be replaced or not. It is a typical type of competition. Then, if the particle is better than its historical best, it will further be compared with the global best to decide if the latter will be replaced or not. It is another type of competition. Despite the two competitive strategies, the form of competition is very simple because all the particles except for the global best only compete with their own historical best and the global best. In other words, any two particles excluding the global best do not compete with each other, and hence it lacks diversity. As a result, the canonical PSO tends to result in early maturity and converge to a local solution.

To cope with this issue, more competitive strategies should be introduced into PSO. In addition to the competition with the local historical best, the PPSO adopts two other competitive strategies. First, when building the pyramid, all the particles participate in sorting to decide their layers in the pyramid. Since the sorting involves all particles, we call it a global competitive strategy. Second, each particle is placed in a specific layer, and the particles at the same layer can be treated differently to improve the effectiveness. Following this idea, we introduce another competitive strategy for the particles at one layer. The particles at one layer except for the top layer are paired randomly to be divided into two groups: the winners and the losers. The winners can learn from the particles at higher layers, while the losers can only learn from the winners at the same layer. Since the competition involves the particles at the layer only, we call it a local competitive strategy, which can be described as Algorithm 2. Note that, for convenience, we assume that the number of particles at each layer is even.

Algorithm 2 Competition at one layer.

Input: n particles $P = \{p_1, p_2, \cdots, p_n\}$ at one layer along with their fitness $F = \{f_{p_1}, f_{p_2}, \cdots, f_{p_n}\}$
Output: Losers PL and Winners PW

```
1: function COMPETE(P, F)
2:     r ← randperm(n); //a random permutation of integers [1, n]
3:     for i = 1 → n/2 do
4:         if f_{r_i} > f_{r_{i+n/2}} then
5:             PL_i ← p_{r_i};
6:             PW_i ← p_{r_{i+n/2}};
7:         else
8:             PL_i ← p_{r_{i+n/2}};
9:             PW_i ← p_{r_i};
10:        end if
11:    end for
12:    return PL, PW;
13: end function
```

With the proposed competitive strategies, the global best is located at the top layer and all the particles may learn from different particles that are better than the learners instead of from the only global best in the canonical PSO. Therefore, the competitive strategies can enhance the diversity of PSO population.

3.3. Cooperative strategy

In the canonical PSO, each particle (learner) cooperates with its own historical best and the only global best in the evolution. As a result, any other particles have no chance to cooperate with the learner. An ideal cooperative strategy is that each learner has chances to cooperate with more particles, not only the global best but also any other particles better than the learner. In this way, the diversity of cooperation will be enhanced and hence it may lead to better performance of PSO.

Inspired by the above analysis, this paper proposes a novel strategy for cooperation that each particle has chances to cooperate with a couple of better particles besides its own best. Specifically, the strategy lies in two aspects. First, the losers at one layer cooperate with their own best and the winners at the same layer. Second, the winners except for those at the top layer cooperate with their own best, the particles at the upper layer, and the particles at the top layer.

Mathematically, the losers and the winners at the ith layer ($i = 2, 3, \ldots, L$) update their velocities and positions according to Eqs. (3) and (4), respectively.

$$\begin{cases} V_{PL_{i,j}}^{t+1} = r_1 \cdot V_{PL_{i,j}}^{t} + r_2 \cdot (B_{PL_{i,j}}^{t} - X_{PL_{i,j}}^{t}) + r_3 \cdot (X_{PW_{i,j}}^{t} - X_{PL_{i,j}}^{t}) \\ X_{PL_{i,j}}^{t+1} = X_{PL_{i,j}}^{t} + V_{PL_{i,j}}^{t+1} \end{cases}, \tag{3}$$

$$\begin{cases} V_{PW_{i,j}}^{t+1} = r_4 \cdot V_{PW_{i,j}}^{t} + r_5 \cdot (B_{PW_{i,j}}^{t} - X_{PW_{i,j}}^{t}) + r_6 \cdot (X_{P_{i-1,k}}^{t} - X_{PW_{i,j}}^{t}) \\ \qquad\qquad + \rho \cdot r_7 \cdot (X_{P_{1,m}}^{t} - X_{PW_{i,j}}^{t}) \\ X_{PW_{i,j}}^{t+1} = X_{PW_{i,j}}^{t} + V_{PW_{i,j}}^{t+1} \end{cases}, \tag{4}$$

where $r_i (i = 1, 2, \ldots, 7)$ are random real values that are uniformly distributed in the range of (0,1), ρ is a constant to control the impact of one particle at the top layer on the current particle; $P_{i,j}$, $PL_{i,j}$ and $PW_{i,j}$ denote the jth particle, loser and winner at the ith layer respectively; X_p^t, V_p^t and B_p^t denote the position, velocity and the historical best of particle p at generation t respectively; k and m are random positive integers not greater than n_{i-1} and n_1 respectively. For those particles at the top layer, the losers also use the updating strategy in Eq. (3) while the winners will directly move to the next generation. Therefore, the winners at the top layer (the elites in the population) will reserve until they are replaced by new elites. Note that, different from using an inertia weight (w_t in Eq. (1)) in the canonical PSO, the proposed cooperative strategies use a random real value instead. The losers and the winners use different strategies to update their velocities. Besides cooperation with their own historical best, the losers will learn from the paired winners while each winner cooperates with a random particle at the upper layer as well as a random particle at the top layer. Both the cooperated particles are better than the learner. Each particle has chances to cooperate with more better particles rather than the only global best particle in the canonical PSO. As such, the diversity will be enhanced in the proposed approach.

3.4. PPSO

The proposed PPSO has three major concepts: pyramid building, a competitive strategy that divides the particles at one layer into losers and winners, and a cooperative strategy that the losers and the winners use different rules to update their velocities.

With the three concepts, the proposed framework of the PPSO can be described as Algorithm 3. In the algorithm, Lines 2–6 initialize and evaluate the population. In each iteration of the while loop (Lines 7–24), the PPSO consists of four parts: (1) building the pyramid to assign a particle to a specific layer (Line 8); (2) competition and cooperation from the bottom layer (Layer L) to the second layer (Layer 2) (Lines 10–16); (3) competition at the top layer (Layer 1) and updating the losers' information (Lines

Algorithm 3 The framework of PPSO.

Input: Population size N, maximal times of fitness evaluation MFE, the number of the layers of the pyramid L, the number of the particles at each layer $n_i (i = 1, 2, \cdots, L)$, and the ρ in Eq. (4).
Output: optimal solution G and its fitness f_G

1: **function** PPSO(N, MFE, L, n_i)
2: $fe \leftarrow 0$;
3: Randomly initialize N particles as the population P;
4: Evaluate the N particles in P and obtain their fitness F;
5: Assign the best particle and its fitness to G and f_G, respectively;
6: $fe \leftarrow fe + N$;
7: **while** $fe \leq MFE$ **do**
8: $PP \leftarrow PB(P, F, L, n_i)$; //Build the pyramid
9: $P \leftarrow NULL$;
10: **for** $j = L \rightarrow 2$ *step* -1 **do**
11: $PL, PW \leftarrow Compete(PP_j, F_j)$;
12: Update $PL's$ velocities and positions by Eq. (3);
13: Update $PW's$ velocities and positions by Eq. (4);
14: Concatenate P and PL;
15: Concatenate P and PW;
16: **end for**
17: $PL, PW \leftarrow Compete(PP_1, F_1)$;
18: Update $PL's$ velocities and positions by Eq. (3);
19: Concatenate P and PL;
20: Concatenate P and PW;
21: Evaluate the particles in P and obtain their fitness F;
22: Assign the best particle and its fitness to G and f_G, respectively;
23: $fe \leftarrow fe + N$;
24: **end while**
25: **return** G, f_G;
26: **end function**

17–20); and (4) particle evaluation (Lines 21–22). The iteration will terminate when the total evaluation times exceed MFE.

Note that since the winners of Layer 1 are the best particles of the swarm so far and the global optimal solution is usually in their neighborhood, PPSO leaves them unchanged and transfers them directly to the next generation.

The PPSO framework can be illustrated in Fig. 1. In each generation, the particles are sorted in ascending order according to their fitness. In this figure, the better the particle, the darker its color. Next, each particle is assigned to a specific layer in a pyramid from top to bottom and left to right by its index in the sorted particles. In this way, the better a particle, the higher its layer in the pyramid. Then, the particles at each layer will be paired randomly and produce winners and losers. The winners and the corresponding losers will be placed on the left side and right side of the layer, respectively. Finally, the losers at each layer will learn from the corresponding winners at the same layer as well as the losers' historical best. In contrast, the winners except for those at Layer 1 will have chances to learn from more better particles: the particles at a higher layer, the best several particles in the swarm (those at Layer 1), and the winners' historical best. The winners at Layer 1 will be transferred to the next generation directly. In the figure, the blue particles indicate the red particles' historical best in the corresponding positions.

3.5. Dynamic analysis

In this section, we use the theory of linear and discrete-time dynamic system to analyze the dynamic behavior of particles [54,55]. In the PPSO, the winners and losers have different updating strategies. Since the losers' dynamic analysis is similar to but easier than the winners' and it also may refer to [53], here we mainly analyze the dynamic behavior of the winners.

A deterministic implementation of the winners in PPSO is considered for this analysis by setting the random values in Eq. (4) to their expected values, i.e.,

$$r_4 = r_5 = r_6 = r_7 = \frac{1}{2}. \tag{5}$$

Without loss of generality, we can reformulate (4) by considering a one-dimension deterministic case:

$$\begin{cases} V_i^{t+1} = \frac{1}{2}\cdot V_i^t + \frac{1}{2}\cdot(B_i^t - X_i^t) + \frac{1}{2}\cdot(X_k^t - X_i^t) \\ \qquad\qquad + \rho\cdot\frac{1}{2}\cdot(X_m^t - X_i^t) \\ X_i^{t+1} = X_i^t + V_i^{t+1} \end{cases}, \tag{6}$$

where B_i^t, X_k^t and X_m^t are the historical best of X_i, the position of a particle at X_i's upper layer, and the position of a particle at the top layer at generation t respectively. This equation can be simplified as Eq. (7) using the newly-defined notations: $p = \frac{2+\rho}{2}$ and $q = \frac{1}{2+\rho}\cdot B_i^t + \frac{1}{2+\rho}\cdot X_k^t + \frac{\rho}{2+\rho}\cdot X_m^t$.

$$\begin{cases} V_i^{t+1} = \frac{1}{2}\cdot V_i^t + p(q - X_i^t) \\ X_i^{t+1} = X_i^t + V_i^{t+1} \end{cases}. \tag{7}$$

This equation can be further written in compact matrix form as follows:

$$y^{t+1} = Ay^t + Bq, \tag{8}$$

where

$$y^t = \begin{bmatrix} V_i^t \\ X_i^t \end{bmatrix}, \quad A = \begin{bmatrix} \frac{1}{2} & -p \\ \frac{1}{2} & 1-p \end{bmatrix}, \quad B = \begin{bmatrix} p \\ p \end{bmatrix}, \tag{9}$$

where y^t is the particle's state made up of its velocity and position at time t, A is a state matrix in dynamical system theory which determines the dynamical behaviors of the particle, q is called external input used to control the particle to move towards a specified position and B is the input matrix to control external effect on the particle's dynamics.

An equilibrium point is a state maintained by the dynamic system without external excitation, i.e., $q = 0$. Note that it is not necessarily the global optimum. If there exists an equilibrium point y_* that hold the obvious condition $y_*^{t+1} = y_*^t$ for any t, it can be obtained from Eqs. (8) and (9):

$$y_* = \begin{bmatrix} 0 \\ q \end{bmatrix}, \tag{10}$$

which means that the velocity of the particle is zero and the corresponding particle finally stops at the same position, provided that q is constant [54,55].

A particle's convergence means it will eventually settle down at y_*. According to the dynamical system theory, the convergence property depends on the eigenvalues of A, i.e., the solutions of Eq. (11):

$$|A - \lambda E| = \begin{vmatrix} \frac{1}{2} - \lambda & -p \\ \frac{1}{2} & 1 - p - \lambda \end{vmatrix} = \lambda^2 + (\frac{3}{2} - p)\lambda + \frac{1}{2} = 0. \tag{11}$$

The eigenvalues of A are $\lambda_1 = \frac{3}{4} - \frac{p}{2} + \frac{\sqrt{\left(\frac{3}{2}-p\right)^2 - 2}}{2}$ and $\lambda_2 = \frac{3}{4} - \frac{p}{2} - \frac{\sqrt{\left(\frac{3}{2}-p\right)^2 - 2}}{2}$. The necessary and sufficient condition for the

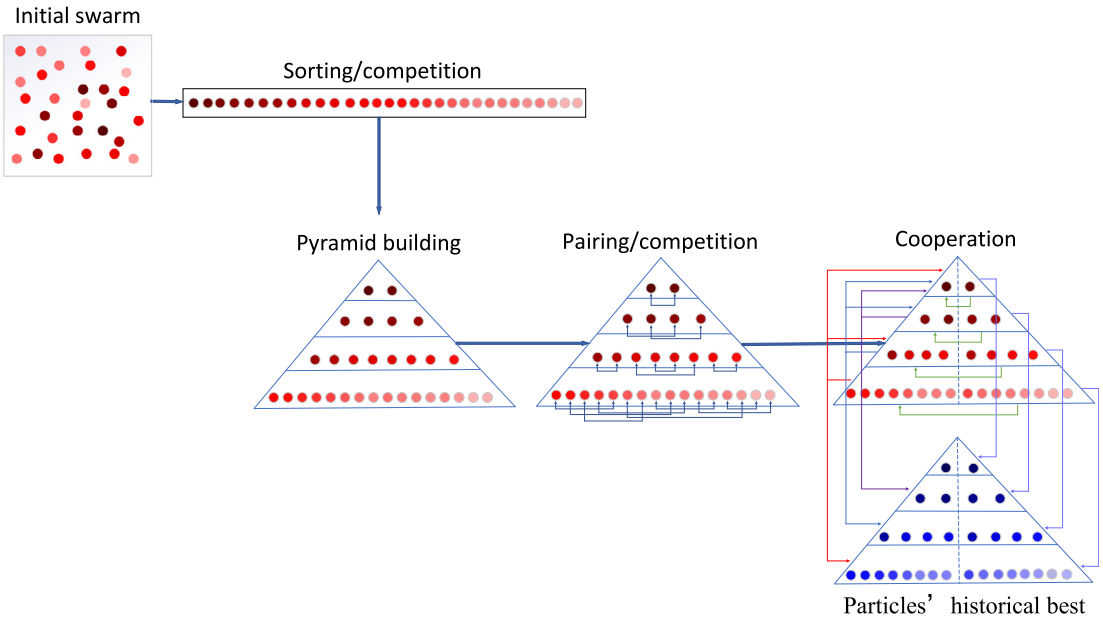

Fig. 1. The framework of PPSO.

particle's convergence is that $|\lambda_1| < 1$ and $|\lambda_2| < 1$, i.e., y_* is a stable attractor. It leads to the result:

$$p = \frac{2+\rho}{2} > 0. \tag{12}$$

Therefore, $\rho > -2$ is a sufficient condition for the convergence of the winners in the swarm. The analysis can be introduced to the losers in the PPSO. From the analysis in [53] and here, it can be found that the condition for the convergence of the losers always satisfies. In this paper, only nonnegative $\rho \geq 0$ will be used.

It should be pointed out that although the condition of particles' convergence in PPSO satisfies, it does not guarantee to converge to the global optimum.

3.6. Complexity analysis

Thanks to its fairly algorithmic simplicity, it is very straightforward to compute the time complexity of the proposed PPSO, which consists of four parts: fitness evaluation, pyramid building, pairing/competition and learning/cooperation. For practical applications, the time of fitness evaluation is problem-dependent [50, 54] and it is excluded in this analysis. And hence, we focus on the last three parts.

When building the pyramid, PPSO needs to sort the particles in the whole swarm first and then assign each particle to a specific layer, as shown in Algorithm 1. For the first operation, any sorting algorithm, such as quick sort, selection sort, merge sort, and etc., can be used. For the N particles in the swarm, the mean time complexity of sorting is $O(Nlog(N))$. The second operation only needs to scan the whole swarm once and then assign each particle to one of the L layers, and its complexity is N. Therefore, the total time complexity of pyramid building is $O(Nlog(N) + N)$. The pairing operation needs to scan the whole swarm only once, and hence its time complexity is $O(N)$. Suppose each particle's dimension is D, except for the winners at Layer 1 that will enter the next generation directly, the N particles will update their D dimensions in the cooperation. Therefore, the time complexity of learning operation is $O(ND)$. Overall, the time complexity of the PPSO is $O(Nlog(N) + N + ND)$, which is slightly higher than the canonical PSO's $O(ND)$.

As far as the space complexity is concerned, the PPSO needs to store all particles and their corresponding locally best positions, either of which takes $O(ND)$ space. Therefore, the whole space complexity of the PPSO is $O(2ND)$, which is the same as that of the canonical PSO.

In conclusion, although PPSO introduces a new topology and new learning strategies, it slightly increases the time complexity while still keeping the space complexity compared with the canonical PSO. Therefore, the proposed PPSO is still efficient.

3.7. Differences between PPSO and other PSO variants

In the PPSO, a structure of pyramid is introduced to organize the particles in swarm, and novel competitive and cooperative strategies are proposed to update particles' information. Two random particles at one layer are paired and divided into one loser and one winner. The losers and the winners use different rules to evolve. Any particles except for the worst one in the swarm may become exemplars of other particles. Specifically, all these attributes make the proposed PPSO very different from the existing PSO variants, as analyzed below.

(1) The topology of pyramid makes the particles in the swarm are not equally treated. The better a particle, the higher its layer. Any particles except for the worst particle in the swarm have chances to guide the updating process of other particles. Especially, the particles at Layer 1 have the most chances for such guidance. The topology makes the PPSO apparently different from the canonical PSO and many PSO variants [56–60], where only the global best has chances to guide other particles.

(2) The particles at one layer use different updating strategies. The particles are firstly divided into either losers or winners according to their fitness. For the losers at all layers, they will only learn from their own historical best and their opponents (the corresponding winners). However, the winners not at Layer 1 have more chances to learn from extensive particles: their own historical best, the particles at their upper layer, and those at Layer 1. The winners at Layer 1 will directly enter the next generation. Therefore, besides its own historical best, a loser only has an exemplar which is at the same layer while a winner not at Layer 1 has two exemplars: one from its upper layer and the other from

Layer 1. This is a big difference between PPSO and some PSO variants in which each particle has only one exemplar, i.e., the global best [61–64].

(3) As one of the parameters of PSO, the inertia weight plays an essential role in the evolution of PSO. Most existing PSO variants use it to balance the ability of exploration and exploitation [38,65–67]. However, it is problem-dependent and hard to set a general value for different problems. To avoid setting the inertia weight by users, the proposed PPSO uses a random value in the range of (0, 1) instead. This is a big difference between PPSO and some other PSO variants.

(4) Although there are some PSO variants with the concepts of multiple layers or novel strategies of competition and /or cooperation [23,37,49,50,68,69], they are different from the proposed PPSO. For example, the multi-layer PSO (MLPSO) proposed by Wang et al. generates a sub-swarm at a higher layer from several sub-swarms at the corresponding lower layer, which indicates a new sub-swarm is generated from a local view [49]. On the contrary, the particles at each layer of the PPSO are from the whole swarm. That is to say, the layers in the PPSO are generated from a global view. Yang et al. presented a level-based learning swarm (LLSO) for large-scale optimization [50]. It has two versions: the basic LLSO that each level has the same number of particles as well as the dynamic LLSO that each level has a different size of particles. There is no further competition among the particles at the same layer in both versions, and all the particles at the same layer use the same learning strategies. Simultaneously, all the particles only learn from the particles at the top level and the particles' historical best are discarded in the LLSO framework. Li et al. proposed an information sharing mechanism (ISM) and designed a competitive and cooperative (CC) operator for PSO, namely CCPSO-ISM, for global optimization [69]. The CCPSO-ISM conducts the CC operator on the dimension-level velocity. It has no concept of multi-layer and all the particles use a fixed updating strategy. Therefore, all the mentioned PSO variants are apparently different from the proposed PPSO.

With the above attributes of the proposed PPSO, it has the potential for optimizing parameters in many real-world applications [70–73].

4. Result analysis and discussion

In this section, the performance of PPSO is evaluated via extensive experiments. All the experiments are conducted on two very popular test suits, and canonical PSO and a couple of state-of-the-art PSO variants are used for comparison. We also discuss the influence of parameter settings in PPSO.

4.1. Benchmark functions and compared models

In order to validate the optimization performance of the proposed PPSO, we conduct extensive optimization experiments on two very popular suits of benchmark functions in the community of evolutionary computation, i.e., CEC2013 and CEC2017. CEC2013 test suit has 28 functions, among which $F1{\sim}F5$ are unimodal functions, $F6{\sim}F20$ are multimodal functions, and $F21{\sim}F28$ are composition functions. CEC2017 is an updated version of CEC2013 test suit that contains 30 functions that can be divided into four groups: unimodal functions ($F1{\sim}F3$), multimodal functions ($F4{\sim}F10$), hybrid functions ($F11{\sim}F20$) and composition functions ($F21{\sim}F30$). The details of CEC2013 and CEC2017 can be found in [74,75], respectively.

We compare the proposed PPSO with canonical PSO (PSO) [3] and eight state-of-the-art PSO variants, including unified PSO (UPSO) [76], weighted fully informed particle swarm (WFIPS) [77], fitness distance ratio PSO (FDRPSO) [78], comprehensive learning PSO (CLPSO) [79], cooperative PSO (CPSO) [80], multi-layer PSO (MLPSO) [49], a very recently proposed modified PSO using adaptive strategy (MPSO) [38], and CCPSO-ISM [69]. All the compared PSO algorithms have been demonstrated to be effective for numerical optimization problems and real-world problems.

4.2. Experimental settings

All the experiments are conducted with two types of dimensions, i.e., $d = 30$ or $d = 50$. To set up a fair comparison, we set the population size and maximum fitness evaluations to 64 and $10000 \times d$ respectively for all algorithms. For the proposed PPSO, a "$4 - 8 - 20 - 32$" structure is used for pyramid building for comparison, and ρ is set to 0.02 and 0.04 for $d = 30$ and $d = 50$, respectively. All the statistical results reported in this paper are based on 30 independent runs, unless otherwise specified. All the other parameters for the compared algorithms are set as those in the original papers.

We conduct all the experiments by Matlab R2017b on a 64-bit Windows 10 OS with an Intel Core i7-4790 @ 3.60 GHz CPU and 32 GB RAM. The source code of PPSO is available at https://github.com/ltyong/ppso.

4.3. Comparisons with state-of-the-art PSO variants

This subsection compares the PPSO with the competitive PSO variants from different perspectives, including accuracy, Wilcoxon signed-rank test, convergence speed and running time.

4.3.1. Accuracy

Each algorithm is evaluated on each test function for 30 independent runs. The results are based on the errors of the optimized values minus the target values. Therefore, the ideal optimal value for each benchmark function is 0. The mean values (Mean), standard deviation (Std), and the ranks (Rank) of the errors are reported for evaluating the accuracy. Mean reflects the optimization ability while Std reflects the stability of optimization algorithms. For each function, the minimum mean of the errors is shown in bold.

4.3.1.1. Accuracy with $d = 30$. We first report the accuracy on CEC2013 test suites with $d = 30$ in Table 1. It can be observed that the final rank by the PPSO is 1, which means that the PPSO performs the best in all the optimization models with CEC2013 and $d = 30$. Specifically, PPSO achieves the best results in 12 out of 28 cases, followed by CLPSO's 4 cases and MLPSO and CCPSO-ISM's 3 cases. The other algorithms perform poorly in getting the best results, each ranking first only once. Although FDRPSO and canonical PSO achieve the best mean only once, their final ranks are 2 and 3, respectively. It reveals that FDRPSO and canonical PSO perform well for simple low-dimensional optimization problems. It is worth pointing out that for two unimodal functions, $F1$ and $F5$, the Mean and Std values obtained by PPSO are both "0.0000e+00", indicating PPSO finds the exact optimal values in all the 30 independent runs. For the 15 multimodal functions ($F6{\sim}F20$), PPSO achieves the best values 8 times, which is significantly superior to the other models. The worst rank for multimodal functions by PPSO is 7 with $F14$. CPSO performs very poorly with the multimodal functions. More specifically, although CPSO achieves the best value with $F17$, it gets the worst values in 5 out of 15 functions. For the 8 composition functions ($F21{\sim}F28$), both CCPSO-ISM and PPSO achieve the best values 3 and 2 times respectively, and UPSO, FDRPSO and MLPSO achieve the best

Table 1
Results of CEC2013 with $d = 30$.

Function	Metrics	PSO	UPSO	FDRPSO	WFIPS	CPSO	CLPSO	MLPSO	MPSO	CCPSO-ISM	PPSO
F1	Mean	2.1979e−13	2.1979e−13	2.4253e−13	5.1432e−11	1.8741e−07	7.3517e−13	1.6674e−13	1.6127e−01	1.2172e−05	**0.0000e+00**
	Std	4.1513e−14	4.1513e−14	1.0227e−13	2.2935e−11	1.9116e−07	1.6551e−13	1.0227e−13	4.1883e−01	8.0098e−06	0.0000e+00
	Rank	3	3	5	7	8	6	2	10	9	**1**
F2	Mean	1.3135e+07	2.6712e+06	1.6156e+06	3.0937e+07	6.2146e+06	1.6884e+07	**3.7498e+05**	8.6732e+06	1.6981e+07	1.1778e+06
	Std	1.1545e+07	1.0446e+06	6.8138e+05	6.0579e+06	4.7605e+06	4.0193e+06	1.9465e+05	1.2453e+07	4.3999e+06	6.4602e+05
	Rank	7	4	3	10	5	8	**1**	6	9	2
F3	Mean	1.0472e+08	1.3235e+08	3.2115e+08	**3.3633e+07**	1.0163e+10	1.5258e+09	2.5817e+08	3.3497e+08	4.1012e+09	2.5575e+08
	Std	1.2370e+08	1.7613e+08	4.7650e+08	1.7523e+07	1.2509e+10	9.4444e+08	4.0593e+08	5.4108e+08	1.4318e+09	4.6793e+08
	Rank	2	3	6	**1**	10	8	5	7	9	4
F4	Mean	4.4533e+03	3.0211e+04	1.7901e+04	2.6263e+04	1.2911e+05	4.4307e+04	9.0550e+02	**2.3577e+02**	6.7024e+04	5.0682e+04
	Std	1.4613e+03	6.7441e+03	6.8246e+03	4.5936e+03	2.8600e+04	1.0004e+04	7.5015e+02	2.3538e+02	9.8948e+03	1.2018e+04
	Rank	3	6	4	5	10	7	2	**1**	9	8
F5	Mean	1.7811e−13	2.0085e−13	2.9938e−13	3.9381e−07	1.9027e−04	4.3162e−10	1.4021e−13	3.9940e+00	7.6583e−04	**0.0000e+00**
	Std	5.7299e−14	4.8906e−14	1.4157e−13	1.0080e−07	1.0944e−04	5.2143e−10	4.8906e−14	1.1233e+01	4.6349e−04	0.0000e+00
	Rank	3	4	5	7	8	6	2	10	9	**1**
F6	Mean	8.4710e+01	2.0549e+01	2.4244e+01	2.6410e+01	4.5915e+01	2.2208e+01	**7.9953e+00**	7.1035e+01	5.6160e+01	3.5814e+01
	Std	3.8345e+01	3.0294e+00	2.2797e+00	5.4466e−01	1.9278e+01	2.0592e+00	2.8306e+00	2.3838e+01	1.0650e+01	2.4646e+01
	Rank	10	2	4	5	7	3	**1**	9	8	6
F7	Mean	**3.8378e+01**	1.1621e+02	6.3346e+01	3.9714e+01	2.0051e+02	1.1808e+02	1.1521e+02	6.6634e+01	1.0211e+02	5.7248e+01
	Std	1.1802e+01	2.3741e+01	4.2978e+01	8.9447e+00	6.5432e+01	2.1546e+01	3.4648e+01	1.8814e+01	1.4312e+01	1.9304e+01
	Rank	**1**	8	4	2	10	9	7	5	6	3
F8	Mean	2.0938e+01	2.0932e+01	2.0984e+01	2.0932e+01	2.0933e+01	2.0929e+01	2.0930e+01	2.0934e+01	2.0939e+01	**2.0893e+01**
	Std	4.6590e−02	5.5431e−02	5.9727e−02	5.9918e−02	6.3513e−02	7.6282e−02	3.9631e−02	5.2614e−02	5.2564e−02	7.5934e−02
	Rank	8	4	10	5	6	2	3	7	9	**1**
F9	Mean	2.1241e+01	3.0457e+01	2.1203e+01	3.3873e+01	3.7687e+01	2.9922e+01	3.0175e+01	2.4823e+01	2.8813e+01	**1.7825e+01**
	Std	2.7879e+00	1.7659e+00	3.2770e+00	2.4522e+00	2.6993e+00	2.5072e+00	3.7087e+00	4.4466e+00	1.8746e+00	2.4455e+00
	Rank	3	8	2	9	10	6	7	4	5	**1**
F10	Mean	1.8011e−01	1.5479e−01	1.4027e−01	1.6510e+00	2.8805e+01	1.0022e+01	**1.3202e−01**	5.0036e−01	4.3492e+01	3.0127e−01
	Std	1.9167e−01	6.5525e−02	7.9384e−02	5.0096e−01	3.7786e+01	3.2820e+00	5.8787e−02	4.3787e−01	7.8597e+00	2.5975e−01
	Rank	4	3	2	7	9	8	**1**	6	10	5
F11	Mean	2.1756e+01	8.1221e+01	3.8140e+01	7.2899e+01	1.8107e−07	**4.5285e−13**	4.6498e+01	3.0946e+01	7.5101e+00	2.4973e+01
	Std	4.2992e+00	1.5133e+01	1.0534e+01	1.4010e+01	1.5905e−07	2.1084e−13	1.1862e+01	1.1645e+01	1.6100e+00	8.6516e+00
	Rank	4	10	7	9	2	**1**	8	6	3	5
F12	Mean	6.9768e+01	9.6258e+01	6.0692e+01	1.8091e+02	4.3733e+02	1.6989e+02	8.1885e+01	9.0920e+01	2.1471e+02	**3.1142e+01**
	Std	3.8594e+01	2.0281e+01	2.1208e+01	1.1293e+01	1.1039e+02	3.0364e+01	2.3464e+01	2.4426e+01	3.3090e+01	1.0494e+01
	Rank	3	6	2	8	10	7	4	5	9	**1**
F13	Mean	1.4051e+02	1.6241e+02	1.3433e+02	1.8093e+02	4.2327e+02	2.1402e+02	1.4995e+02	1.5474e+02	2.5820e+02	**8.7578e+01**
	Std	3.7099e+01	2.5657e+01	3.1884e+01	9.1233e+00	8.2790e+01	2.3127e+01	3.8611e+01	3.2433e+01	2.0963e+01	3.3250e+01
	Rank	3	6	2	7	10	8	4	5	9	**1**
F14	Mean	7.6920e+02	2.9400e+03	1.5811e+03	5.3647e+03	3.3690e−01	**1.9877e−01**	1.7681e+03	1.3717e+03	9.2230e+02	1.6186e+03
	Std	2.3232e+02	3.8438e+02	4.2770e+02	3.0707e+02	1.1431e+00	3.8931e−01	5.2674e+02	3.7635e+02	1.6464e+02	3.6706e+02
	Rank	3	9	6	10	2	**1**	8	5	4	7
F15	Mean	6.5730e+03	3.7393e+03	4.2155e+03	7.1506e+03	5.3197e+03	3.9997e+03	5.0333e+03	3.7495e+03	3.8834e+03	**2.5392e+03**
	Std	7.4980e+02	3.1697e+02	1.0090e+03	3.0957e+02	9.7655e+02	3.6143e+02	1.1905e+03	6.7781e+02	3.0449e+02	6.4130e+02
	Rank	9	2	6	10	8	5	7	3	4	**1**
F16	Mean	2.1137e+00	1.0069e+00	2.1880e+00	2.4197e+00	1.5835e+00	1.1787e+00	2.0001e+00	1.7451e+00	9.7784e−01	**3.5693e−01**
	Std	3.1521e−01	4.1878e−01	3.9751e−01	2.5895e−01	4.6086e−01	2.5961e−01	3.9639e−01	4.1684e−01	2.7135e−01	1.6199e−01
	Rank	8	3	9	10	5	4	7	6	2	**1**
F17	Mean	5.3066e+01	1.1625e+02	8.4241e+01	1.8106e+02	**3.0435e+01**	3.0438e+01	7.6616e+01	6.6033e+01	5.0349e+01	4.5796e+01
	Std	1.9741e+01	1.4056e+01	1.4931e+01	1.0416e+01	2.3508e−03	7.0763e−03	1.3475e+01	1.2351e+01	4.2610e+00	4.5658e+00
	Rank	5	9	8	10	**1**	2	7	6	4	3
F18	Mean	2.2061e+02	1.0229e+02	1.5818e+02	2.1034e+02	3.0701e+02	2.0979e+02	1.1843e+02	9.1398e+01	1.8825e+02	**4.7265e+01**
	Std	3.2790e+01	1.0635e+01	5.7811e+01	8.9625e+00	6.4522e+01	2.0551e+01	3.8082e+01	1.5785e+01	1.4790e+01	7.2691e+00
	Rank	9	3	5	8	10	7	4	2	6	**1**
F19	Mean	3.4311e+00	6.6826e+00	3.5919e+00	1.3047e+01	5.7129e−01	**4.3650e−01**	3.6338e+00	4.5698e+00	3.5707e+00	3.5130e+00
	Std	9.1486e−01	1.6026e+00	9.1433e−01	8.9435e−01	3.4345e−01	1.5934e−01	1.1178e+00	1.3128e+00	8.4364e−01	1.4969e+00
	Rank	3	9	6	10	2	**1**	7	8	5	4
F20	Mean	1.2164e+01	1.2271e+01	1.3264e+01	1.2523e+01	1.4155e+01	1.4453e+01	1.2191e+01	1.3095e+01	1.4684e+01	**1.1941e+01**
	Std	4.1294e−01	5.0489e−01	1.7900e+00	2.9680e−01	4.7943e−01	5.4058e−01	6.5574e−01	8.9147e−01	2.1004e−01	6.4445e−01
	Rank	2	4	7	5	8	9	3	6	10	**1**
F21	Mean	2.9435e+02	2.4812e+02	3.2349e+02	3.0559e+02	3.3452e+02	2.3028e+02	2.8914e+02	7.9262e+02	**2.2363e+02**	3.3263e+02
	Std	6.4556e+01	7.6531e+01	7.3737e+01	6.0350e+01	8.5155e+01	3.6738e+01	8.0704e+01	4.5719e+02	2.9127e+01	9.3718e+01
	Rank	5	3	7	6	9	2	4	10	**1**	8
F22	Mean	8.8267e+02	3.2452e+03	1.5806e+03	5.4822e+03	1.1530e+02	**3.9485e+01**	1.7946e+03	1.3115e+03	9.2364e+02	1.4856e+03
	Std	2.7018e+02	4.5147e+02	4.6480e+02	3.0036e+02	6.7819e+01	2.7044e+01	4.4484e+02	4.4334e+02	1.8958e+02	4.6702e+02
	Rank	3	9	7	10	2	**1**	8	5	4	6
F23	Mean	6.8906e+03	**4.2224e+03**	4.3294e+03	7.4971e+03	6.4382e+03	5.2588e+03	5.4386e+03	4.4676e+03	4.9289e+03	4.2898e+03
	Std	8.4290e+02	3.6117e+02	1.0369e+03	3.1697e+02	1.0800e+03	5.6132e+02	1.2511e+03	1.0138e+03	5.4544e+02	8.0288e+02
	Rank	9	**1**	3	10	8	6	7	4	5	2
F24	Mean	2.6228e+02	2.7703e+02	2.5878e+02	2.8989e+02	3.0517e+02	2.8251e+02	2.7954e+02	2.6126e+02	2.8533e+02	**2.4722e+02**
	Std	1.1926e+01	2.9406e+00	6.9455e+00	5.1425e+00	9.3688e+00	1.1309e+01	9.9186e+00	1.2373e+01	5.4826e+00	8.9538e+00
	Rank	4	5	2	9	10	7	6	3	8	**1**
F25	Mean	2.8330e+02	2.7851e+02	**2.5843e+02**	2.9625e+02	3.1127e+02	2.9532e+02	2.8435e+02	2.8285e+02	3.0695e+02	2.7120e+02
	Std	7.8807e+00	3.6925e+00	7.7281e+00	4.4621e+00	1.3033e+01	6.4904e+00	6.3102e+00	1.2589e+01	6.1981e+00	7.0210e+00
	Rank	5	3	**1**	8	10	7	6	4	9	2
F26	Mean	3.3235e+02	3.0029e+02	3.1549e+02	2.2791e+02	3.7293e+02	2.0732e+02	3.3350e+02	2.8618e+02	**2.0141e+02**	2.8261e+02
	Std	5.3087e+01	8.8800e+01	6.5507e+01	6.1584e+01	6.5758e+01	2.3596e+01	7.5571e+01	7.6486e+01	3.1393e−01	6.4320e+01
	Rank	8	6	7	3	10	2	9	5	**1**	4
F27	Mean	9.2339e+02	1.0825e+03	8.1008e+02	1.1803e+03	1.3113e+03	8.5819e+02	1.0870e+03	8.3836e+02	**5.0116e+02**	6.6499e+02
	Std	1.0899e+02	3.7287e+01	8.3862e+01	7.0704e+01	8.4762e+01	3.2385e+02	1.0947e+02	1.1255e+02	2.6176e+02	9.9048e+01
	Rank	6	7	3	9	10	5	8	4	**1**	2
F28	Mean	3.7998e+02	3.0000e+02	3.1975e+02	3.0006e+02	2.0117e+03	3.0029e+02	3.3824e+02	6.6320e+02	3.0323e+02	**3.0000e+02**
	Std	3.0465e+02	2.7647e−13	2.2971e+02	2.5993e−01	1.1389e+03	4.0234e−01	2.0947e+02	8.2606e+02	1.0321e+00	0.0000e+00
	Rank	8	2	6	3	10	4	7	9	5	**1**
Mean rank		5.04	5.07	4.96	7.25	7.50	5.07	5.18	5.75	6.18	2.96
Final rank		3	4	2	9	10	5	6	7	8	1

values once, while none of the other models achieves the best values. Therefore, CCPSO-ISM and PPSO outperform the other models in optimizing composition functions. Furthermore, note that PPSO achieves the best Mean and Std ("0.0000e+00") for $F28$, ranking the first among all the models and showing PPSO's stability.

As for CEC2017 with $d = 30$, we can find similar results from Table A.1 in Appendix A. PPSO achieves the best results 10 times in total, followed by CLPSO's 9 times. PSO, UPSO, WFIPS, MLPSO, MPSO and CCPSO-ISM share the remaining 10 best results, while FDRPSO and CPSO perform so poorly that they never obtain the best results. The mean rank of PPSO is 3.43, which is far below that of other algorithms, indicating PPSO performs the best when compared with the other algorithms. CPSO's mean rank is 8.73 that is close to the theoretical maximum 10, showing that CPSO performs worst among all the algorithms. For unimodal functions $F1 \sim F3$, each of PPSO, CLPSO, and MLPSO achieves the best results once. Note that although studies have shown that $F2$ is an unstable function, as confirmed in this table where the magnitude of the optimized results by the other algorithms is from 10^5 to 10^{19}, PPSO performs well with it and achieves the best results with a magnitude of 10^4. For the next multimodal functions $F4 \sim F10$, PPSO obtains the best results and the second best results for 3 times and 2 times, respectively, which is far better than the

other PSO models. For the next 10 hybrid functions ($F11{\sim}F20$), PPSO achieves the best results only once, underperforming CLPSO, MLPSO and MPSO, being on a par with PSO and WFIPS, and outperforming UPSO, FDRPSO, CPSO, and CCPSO-ISM. For the remaining 10 composition functions ($F21{\sim}F30$), PPSO achieves the best results 5 times, followed by CLPSO's 3 times and UPSO and CCPSO-ISM's once. The other algorithms fail to achieve the best results.

From the above analysis, we can conclude that PPSO performs the best with the 30-dimensional optimization problems among all the compared PSO algorithms. It might owe to the well-designed pyramid structure as well as the strategies of competition and cooperation.

Besides CEC2013 and CEC2017 test suites with $d = 30$, we also evaluate some other classical test functions in [26] with $d = 30$. The functions and the corresponding results are reported in Tables A.2 and A.3 in Appendix A, respectively. Interested readers can read them for more details.

4.3.1.2. Accuracy with $d = 50$. To show the performance of PPSO on problems with a larger dimension, we first report the experimental results on CEC2013 with $d = 50$ in Table 2.

For CEC2013, PPSO achieves the best results in 12 out of 28 cases, followed by CLPSO, FDRPSO and CCPSO-ISM, which all achieve the best results 3 times. UPSO, CPSO, and MLPSO perform quite moderately and each of them achieves the best results twice. MPSO only obtains the best result once, outperforming PSO and WFIPS that never achieve the best results. The mean rank of PPSO is 3.41, which is much lower than 4.04, the second best mean rank achieved by FDRPSO. The other mean ranks are all greater than 4.80, showing that they do not perform very well for the CEC2013 functions with 50 dimensions. Once again, for the unimodal functions $F1$ and $F5$, both the Mean and the Std values by PPSO are "0.0000e+00", indicating that PPSO is able to search the optimal values perfectly in each run. These two values by other algorithms except for MPSO are very close to but greater than 0, showing that they perform not as well as PPSO in terms of accuracy and stability. MPSO performs the worst because its mean values with $F1$ and $F5$ are 4.8642e+02 and 1.4699e+01, respectively. For the other 3 unimodal functions, MLPSO, FDRPSO and MPSO share the best results. PPSO also achieves the best results with 7 out of 15 multimodal functions and with 3 out of 8 composition functions. Among the algorithms, the mean rank values of CLPSO, MLPSO and MPSO fall within a narrow range of [5.32, 5.43], showing that they perform roughly equally. Although CPSO achieves the best results twice, i.e., with multimodal functions $F17$ and $F19$, its mean rank value is the highest. Therefore, CPSO performs worst averagely among all the algorithms.

As far as the results of CEC2017 with 50 dimensions are concerned, they further suggest the superiority of PPSO, as listed in Table A.4 in Appendix A. Specifically, the mean rank value of PPSO is 3.23, ranking first among all the PSO algorithms. PPSO obtains the best mean values 11 times, followed by CLPSO's 7 times. FDRPSO and MPSO achieve the best mean values 3 times, while WFIPS, MLPSO and CCPSO-ISM do only twice. PSO, UPSO and CPSO perform so poorly that none of them achieves the best mean value. For different types of benchmark functions, except for that PPSO fails to achieve the best results for the 3 unimodal functions, it outperforms the other algorithms in multimodal functions, hybrid functions and composition functions in terms of the times of achieving the best results. More specifically, PPSO achieves the best results for 4 out of 7 multimodal functions, 3 out of 10 hybrid functions, and 3 out of 10 composition functions. Although WFIPS achieves the best results twice, its mean rank is 7.80, which is identical to the CPSO's mean rank, showing that both WFIPS and CPSO obtain the worst accuracy among all the models.

From the above analysis, we can find that the proposed PPSO ranks first in terms of accuracy in all test function suites and dimensions, outperforming all the other PSO algorithms significantly. Furthermore, it does well in optimizing unimodal functions, multimodal functions, hybrid functions and composite functions.

4.3.2. Wilcoxon signed-rank test

To compare the performance from a statistical view, a well-known nonparametric hypothesis, Wilcoxon Signed-Rank Test (WSRT), is chosen to determine whether there is a statistical significance difference between two algorithms [81]. The null hypothesis of WSRT is that there is no difference between two independent samples.

Four indicators including $pvalue$, $R+$, $R-$ and *Winner* can be obtained from WSRT. p-value presents the p-value of the paired, two-sided test for the null hypothesis. When p-value is less than 0.05, it presents we can reject the null hypothesis at the 5% significance level. R+ indicates the sum of all the ranks of sign '+' (where mean error of the proposed algorithms is higher than that of the compared one) while R- indicates the sum of the ranks of sign '−' (where mean error of the proposed algorithms is lower than that of the compared one). In addition, *Winner* ('+', '=' or '−') shows whether the proposed algorithm is superior to the compared one: '=' indicates a failure to reject the null hypothesis and there is not the statistical significance difference between two algorithms; '+' indicates the null hypothesis is rejected and the proposed algorithm achieves better results at the 5% significance level; '−' indicates the null hypothesis is rejected and the proposed algorithm obtains worse results at the 5% significance level.

The WSRT results between PPSO and the compared algorithms on CEC2013 with 30 dimensions are reported in Table 3. Its last line summarizes the total times of '+/=/−' regarding the 28 test functions, which are also illustrated in Fig. B.1 in Appendix B. When PPSO is compared to the other algorithms, we can see that the results of many combinations of (p-value, $R-$, $R+$) are (1.734e−06, 0, 465). Such results indicate that PPSO completely outperforms the compared methods with the benchmark functions. Specifically, the results of '+/=/−' reveal that PPSO outperforms, ties and underperforms the compared models in 177, 34 and 41 out of 252 (28×9) cases, respectively. Therefore, the WSRT results strongly suggest that the best performance achieved by PPSO is statistically significant. We can conclude that PPSO is superior to the PSO and any other eight PSO variants on major benchmark functions from a statistical perspective.

4.3.3. Convergence speed

To further study the evolutionary processes of PPSO and the compared PSO algorithms, we draw the convergence curves of all the CEC2013 test functions with $d = 30$ to testify the convergence speed of PPSO. The convergence curves are categorized into 3 groups according to the characteristics of the corresponding test functions: unimodal functions, multimodal functions and composite functions, as shown in Figs. 2–4. Note that since some difference of the optimized values between two models is so narrow that the gap between them cannot be shown clearly, we use $log_{10}(Error)$ as the y axis.

For the first group's functions, the cyan lines with stars with $F1$ and $F5$ disappear after the first several fitness evaluation numbers. It is because PPSO finds the optimal results (zeros) and the value of $log_{10}(0)$ equals Inf which cannot be plotted in the figures. Therefore, the PPSO performs perfectly with these two functions. For $F2$ – $F4$, the cyan stars are located at the lowest

Table 2
Results of CEC2013 with $d = 50$.

Function	Metrics	PSO	UPSO	FDRPSO	WFIPS	CPSO	CLPSO	MLPSO	MPSO	CCPSO-ISM	PPSO
F1	Mean	3.4106e−13	3.9411e−13	3.2590e−13	4.9444e−08	4.4675e−07	6.0633e−13	2.5011e−13	4.8642e+02	3.4870e−07	**0.0000e+00**
	Std	1.1563e−13	1.0227e−13	1.1460e−13	1.4983e−08	2.9703e−07	1.2430e−13	6.9378e−14	1.6592e+03	1.6712e−07	0.0000e+00
	Rank	4	5	3	7	9	6	2	10	8	**1**
F2	Mean	3.5043e+07	4.4346e+06	2.2988e+06	1.0539e+08	6.3316e+06	3.3056e+07	**7.4208e+05**	3.1839e+07	3.6476e+07	1.9708e+06
	Std	1.6993e+07	1.2414e+06	6.8830e+05	1.6515e+07	2.4147e+06	9.5232e+06	2.5277e+05	2.3048e+07	6.5863e+06	6.3349e+05
	Rank	8	4	3	10	5	7	**1**	6	9	2
F3	Mean	1.5976e+09	5.3133e+08	**3.1818e+08**	7.8114e+08	1.5456e+10	8.1969e+09	6.7426e+08	4.7513e+08	1.0045e+10	1.4198e+09
	Std	1.7457e+09	5.7705e+08	3.4644e+08	5.5549e+08	1.7567e+10	3.2445e+09	1.4205e+09	4.1938e+08	3.0964e+09	1.5515e+09
	Rank	7	3	**1**	5	10	8	4	2	9	6
F4	Mean	1.1627e+04	3.3009e+04	4.4692e+03	3.8679e+04	2.0526e+05	3.5586e+04	6.6110e+02	**5.5819e+02**	8.9460e+04	7.1036e+04
	Std	2.5150e+03	6.3002e+03	1.9739e+03	5.1645e+03	3.8146e+04	7.7734e+03	1.0556e+03	3.2946e+02	7.6785e+03	9.0636e+03
	Rank	4	5	3	7	10	6	2	**1**	9	8
F5	Mean	5.2675e−13	3.7896e−13	3.9032e−13	1.0028e−04	2.1472e−04	1.7532e−09	2.8801e−13	1.4699e+01	3.1933e−04	**0.0000e+00**
	Std	1.3512e−13	6.2149e−14	1.2555e−13	2.7140e−05	1.3041e−04	3.9072e−09	7.7471e−14	1.8019e+01	1.2405e−04	0.0000e+00
	Rank	5	3	4	7	8	6	2	10	9	**1**
F6	Mean	4.9536e+01	4.3354e+01	4.5022e+01	4.7093e+01	5.3643e+01	4.6631e+01	**2.9416e+01**	1.1081e+02	4.6586e+01	5.3817e+01
	Std	1.2815e+01	3.5724e+00	9.3088e+00	6.5347e−01	1.9819e+01	1.0303e+00	1.0852e+01	3.2344e+01	9.9694e−01	1.7687e+01
	Rank	7	2	3	6	8	5	**1**	10	4	9
F7	Mean	7.6378e+01	1.3001e+02	**5.5174e+01**	7.2814e+01	3.0868e+02	1.3515e+02	1.3467e+02	7.0684e+01	1.2361e+02	6.4999e+01
	Std	1.9528e+01	1.7699e+01	1.9104e+01	9.4094e+00	2.3860e+02	9.7278e+00	3.8611e+01	1.0613e+01	1.3064e+01	1.1751e+01
	Rank	5	7	**1**	4	10	9	8	3	6	2
F8	Mean	2.1134e+01	**2.1102e+01**	2.1132e+01	2.1127e+01	2.1133e+01	2.1123e+01	2.1108e+01	2.1136e+01	2.1124e+01	2.1126e+01
	Std	2.9984e−02	4.3372e−02	3.2925e−02	3.6043e−02	3.5643e−02	3.5321e−02	3.9735e−02	4.0986e−02	3.8559e−02	6.8177e−02
	Rank	9	**1**	7	6	8	3	2	10	4	5
F9	Mean	4.3714e+01	6.0460e+01	3.5087e+01	6.8555e+01	6.9410e+01	5.6904e+01	6.4090e+01	5.0694e+01	5.3827e+01	**3.2495e+01**
	Std	4.0148e+00	2.4570e+00	5.3084e+00	2.4696e+00	4.1642e+00	2.3084e+00	4.8229e+00	6.6967e+00	3.0005e+00	2.7882e+00
	Rank	3	7	2	9	10	6	8	4	5	**1**
F10	Mean	2.4250e+00	**1.2639e−01**	1.6312e−01	1.2401e+01	3.2462e+01	4.0118e+01	1.4618e−01	1.4573e+00	4.5371e+01	8.4207e−01
	Std	9.9546e−01	5.2097e−02	1.1226e−01	3.7233e+00	4.0471e+01	9.4829e+00	6.5350e−02	3.5180e−01	7.3236e+00	6.0327e−01
	Rank	6	**1**	3	7	8	9	2	5	10	4
F11	Mean	5.0730e+01	1.9630e+02	8.1918e+01	2.1993e+02	5.1875e−07	**3.9980e−13**	1.4082e+02	6.8241e+01	2.5878e+01	7.1240e+01
	Std	9.7336e+00	3.2295e+01	1.5234e+01	1.6995e+01	3.3269e−07	6.6349e−13	3.7757e+01	2.5841e+01	3.5689e+00	1.5395e+01
	Rank	4	9	7	10	2	**1**	8	5	3	6
F12	Mean	1.9967e+02	2.2667e+02	1.2255e+02	3.7876e+02	8.7578e+02	3.8933e+02	2.1113e+02	1.8859e+02	4.3382e+02	**8.1188e+01**
	Std	8.5280e+01	4.2869e+01	2.5403e+01	2.2724e+01	2.4834e+02	4.9977e+01	7.2169e+01	5.0781e+01	5.5657e+01	1.6768e+01
	Rank	4	6	2	7	10	8	5	3	9	**1**
F13	Mean	3.6406e+02	3.5379e+02	2.5494e+02	3.8157e+02	9.9765e+02	4.3709e+02	3.4388e+02	3.1595e+02	4.6767e+02	**2.1813e+02**
	Std	7.6211e+01	4.6560e+01	4.9240e+01	1.4537e+01	1.9956e+02	3.1915e+01	7.2603e+01	4.9218e+01	4.0931e+01	4.0966e+01
	Rank	6	5	2	7	10	8	4	3	9	**1**
F14	Mean	1.7725e+03	5.7456e+03	2.2592e+03	1.0908e+04	5.2871e+00	**5.0702e−01**	4.1838e+03	2.5838e+03	1.9806e+03	3.0156e+03
	Std	3.8778e+02	6.1084e+02	4.7520e+02	4.6654e+02	2.1799e+01	6.6646e−01	6.9894e+02	8.3046e+02	4.3227e+02	5.5068e+02
	Rank	3	9	5	10	2	**1**	8	6	4	7
F15	Mean	1.3497e+04	7.5119e+03	8.7822e+03	1.3919e+04	9.2796e+03	8.3451e+03	1.1150e+04	7.5721e+03	8.1471e+03	**5.2047e+03**
	Std	5.0319e+02	4.4913e+02	2.3846e+03	2.7548e+02	1.0447e+03	7.3760e+02	2.1941e+03	7.2575e+02	7.5966e+02	5.8728e+02
	Rank	9	2	6	10	7	5	8	3	4	**1**
F16	Mean	3.1448e+00	1.6036e+00	3.1160e+00	3.3489e+00	2.2232e+00	1.8047e+00	2.8346e+00	2.4520e+00	1.7868e+00	**3.5132e−01**
	Std	3.7215e−01	4.4059e−01	3.8002e−01	3.5642e−01	6.2147e−01	4.3128e−01	4.0163e−01	5.4754e−01	3.5804e−01	1.3959e−01
	Rank	9	2	8	10	5	4	7	6	3	**1**
F17	Mean	1.2788e+02	2.4674e+02	1.5657e+02	3.7960e+02	**5.0786e+01**	5.0793e+01	1.7703e+02	1.4271e+02	9.9695e+01	1.0172e+02
	Std	3.2623e+01	3.5994e+01	2.5953e+01	1.6106e+01	5.8283e−04	1.1970e−02	3.5146e+01	2.0559e+01	1.1323e+01	1.3789e+01
	Rank	5	9	7	10	**1**	2	8	6	3	4
F18	Mean	4.8559e+02	2.1240e+02	3.0396e+02	4.2529e+02	6.5236e+02	4.3302e+02	2.9693e+02	2.2751e+02	3.5505e+02	**1.0202e+02**
	Std	4.2607e+01	2.5054e+01	1.1485e+02	1.2390e+01	1.3866e+02	2.2793e+01	9.5061e+01	5.1397e+01	2.9729e+01	1.7092e+01
	Rank	9	2	5	7	10	8	4	3	6	**1**
F19	Mean	9.4606e+00	1.9272e+01	7.5224e+00	3.0061e+01	**7.4790e−01**	1.1735e+00	1.0956e+01	8.9059e+00	6.6544e+00	1.2725e+01
	Std	2.3156e+00	4.4625e+00	1.5848e+00	1.5561e+00	3.0711e−01	3.5118e−01	3.3833e+00	3.4533e+00	1.4962e+00	3.1086e+00
	Rank	6	9	4	10	**1**	2	7	5	3	8
F20	Mean	2.1882e+01	2.2126e+01	2.1734e+01	2.2272e+01	2.4369e+01	2.3464e+01	2.1685e+01	2.2756e+01	2.4501e+01	**2.0886e+01**
	Std	4.0681e−01	8.8129e−01	1.5167e+00	3.1622e−01	3.8208e−01	5.6006e−01	8.3565e−01	7.6077e−01	9.2297e−02	1.2266e+00
	Rank	4	5	3	6	9	8	2	7	10	**1**
F21	Mean	8.9017e+02	5.0220e+02	8.9233e+02	3.2890e+02	9.4210e+02	5.0503e+02	8.7109e+02	2.2132e+03	**2.1358e+02**	7.3126e+02
	Std	3.6652e+02	3.8720e+02	3.3837e+02	2.1158e+02	3.1724e+02	2.5854e+02	3.6116e+02	6.9010e+02	2.9246e+01	3.7365e+02
	Rank	7	3	8	2	9	4	6	10	**1**	5
F22	Mean	1.7761e+03	6.7848e+03	2.7469e+03	1.1435e+04	7.3631e+01	**2.9050e+01**	4.2004e+03	2.7405e+03	2.2381e+03	3.6698e+03
	Std	4.4159e+02	5.4752e+02	6.6239e+02	4.3981e+02	8.3419e+01	3.6458e+01	8.0391e+02	6.9874e+02	4.7134e+02	8.6031e+02
	Rank	3	9	6	10	2	**1**	8	5	4	7
F23	Mean	1.3501e+04	8.2965e+03	8.4279e+03	1.4160e+04	1.2220e+04	1.0651e+04	1.1733e+04	9.3620e+03	9.9169e+03	**7.6911e+03**
	Std	7.7362e+02	6.4270e+02	1.3185e+03	3.3990e+02	1.3810e+03	8.3762e+02	2.0406e+03	1.1514e+03	7.3070e+02	1.2839e+03
	Rank	9	2	3	10	8	6	7	4	5	**1**
F24	Mean	3.2074e+02	3.5031e+02	3.1053e+02	3.7881e+02	3.9424e+02	3.5998e+02	3.6190e+02	3.1946e+02	3.5823e+02	**2.9382e+02**
	Std	1.2189e+01	4.3993e+00	1.1024e+01	4.9205e+00	1.3065e+01	1.0740e+01	1.0838e+01	1.4504e+01	9.0651e+00	1.1906e+01
	Rank	4	5	2	9	10	7	8	3	6	**1**
F25	Mean	3.6298e+02	3.4927e+02	**3.1782e+02**	3.8340e+02	3.9763e+02	3.7316e+02	3.6235e+02	3.6881e+02	4.1102e+02	3.3797e+02
	Std	1.5186e+01	5.5224e+00	1.5117e+01	3.3534e+00	1.2992e+01	1.1847e+01	1.1085e+01	1.7182e+01	7.5981e+00	9.3566e+00
	Rank	5	3	**1**	8	9	7	4	6	10	2
F26	Mean	4.1430e+02	4.5046e+02	3.8555e+02	3.0782e+02	4.8059e+02	2.0500e+02	4.5827e+02	3.7085e+02	**2.0333e+02**	3.7157e+02
	Std	9.8365e+00	5.4919e+00	5.1972e+01	1.1487e+02	1.0375e+01	3.6875e+00	1.2106e+01	8.6559e+01	6.4838e−01	4.8051e+01
	Rank	7	8	6	3	10	2	9	4	**1**	5
F27	Mean	1.4769e+03	1.8046e+03	1.3144e+03	2.0838e+03	2.1694e+03	1.8307e+03	1.9425e+03	1.4771e+03	1.3156e+03	**1.2128e+03**
	Std	1.2767e+02	5.7501e+01	1.2726e+02	4.3378e+01	1.0367e+02	1.7795e+02	9.9375e+01	1.4642e+02	7.0958e+02	1.1574e+02
	Rank	4	6	2	9	10	7	8	5	3	**1**
F28	Mean	1.8278e+03	6.2491e+02	9.3044e+02	4.0017e+02	2.9296e+03	4.0019e+02	1.2644e+03	1.0586e+03	**4.0003e+02**	8.3345e+02
	Std	1.6610e+03	8.5598e+02	1.2066e+03	6.1937e−01	2.4477e+03	5.0621e−01	1.4586e+03	1.2559e+03	7.5781e−03	1.1242e+03
	Rank	9	4	6	2	10	3	8	7	**1**	5
Mean rank		5.89	4.86	4.04	7.43	7.54	5.32	5.39	5.43	5.64	3.46
Final rank		8	3	2	9	10	4	5	6	7	1

positions of the y axis and they reach competitive results in the first several fitness evaluation times, indicating PPSO is fast for optimization. For the multimodal functions in Fig. 3, PPSO significantly outperforms the other algorithms with $F8$, $F9$, $F12$, $F13$, $F15$, $F16$, $F18$ and $F20$. Among them, PPSO converges rapidly with $F9$, $F12$, $F13$, $F15$, $F16$ and $F18$. As for $F7$, $F10$, $F17$ and $F19$, PPSO achieves competitive results at earlier stages when compared to other algorithms. PPSO performs moderately with the remaining three multimodal functions: $F6$, $F11$ and $F14$. As far as the third group's composite functions are concerned, PPSO is superior to other algorithms with $F24$ and $F28$ in terms of accuracy and convergence speed. It also achieves the second best results with $F23$ and $F27$. In the other cases, PPSO achieves comparable results.

From the above analysis, we can see that PPSO can converge with an ideal speed for unimodal functions, multimodal functions and composite functions. In most cases, it has the fastest convergence speed amongst all the algorithms. The possible reason is that, in PPSO, different particles have different updating strategies, and once some particles get trapped into a local optimum, the strategies will timely adjust the particles' search directions.

4.3.4. Running time

The time complexity and space complexity of the PPSO are $O(Nlog(N) + N + ND)$ and $O(2ND)$ respectively, as analyzed in

Table 3
WSRT results of CEC2013 with $d = 30$.

Function	PPSO vs. PSO				PPSO vs. UPSO				PPSO vs. FDRPSO			
	p-value	R+	R−	Winner	p-value	R+	R−	Winner	p-value	R+	R−	Winner
F1	7.238e−08	0	435	+	7.238e−08	0	435	+	5.356e−07	0	406	+
F2	1.734e−06	0	465	+	1.734e−06	0	465	+	2.849e−02	126	339	+
F3	8.730e−03	360	105	−	1.650e−01	300	165	=	7.343e−01	216	249	=
F4	1.734e−06	465	0	−	3.515e−06	458	7	−	1.734e−06	465	0	−
F5	7.864e−07	0	465	+	3.998e−07	0	465	+	8.770e−07	0	465	+
F6	5.307e−05	36	429	+	3.001e−02	338	127	−	2.369e−01	290	175	=
F7	3.317e−04	407	58	−	2.127e−06	2	463	+	7.343e−01	216	249	=
F8	2.304e−02	122	343	+	3.683e−02	131	334	+	1.251e−04	46	419	+
F9	5.307e−05	36	429	+	1.734e−06	0	465	+	5.287e−04	64	401	+
F10	1.852e−02	347	118	−	3.609e−03	374	91	−	1.833e−03	384	81	−
F11	1.020e−01	312	153	=	1.734e−06	0	465	+	3.724e−05	32	433	+
F12	2.603e−06	4	461	+	1.734e−06	0	465	+	2.879e−06	5	460	+
F13	6.892e−05	39	426	+	3.515e−06	7	458	+	1.359e−04	47	418	+
F14	1.734e−06	465	0	−	1.734e−06	0	465	+	8.774e−01	240	225	=
F15	1.734e−06	0	465	+	2.879e−06	5	460	+	2.127e−06	2	463	+
F16	1.734e−06	0	465	+	2.127e−06	2	463	+	1.734e−06	0	465	+
F17	1.064e−01	154	311	=	1.734e−06	0	465	+	1.734e−06	0	465	+
F18	1.734e−06	0	465	+	1.734e−06	0	465	+	1.734e−06	0	465	+
F19	4.528e−01	196	269	=	8.466e−06	16	449	+	2.989e−01	182	283	=
F20	6.268e−02	142	323	=	3.501e−02	130	335	+	1.287e−03	76	389	+
F21	4.875e−01	266	199	=	1.008e−01	312	153	=	7.552e−01	207	258	=
F22	2.597e−05	437	28	−	1.734e−06	0	465	+	3.389e−01	186	279	=
F23	2.127e−06	2	463	+	7.343e−01	249	216	=	8.612e−01	241	224	=
F24	6.320e−05	38	427	+	1.734e−06	0	465	+	1.973e−05	25	440	+
F25	1.238e−05	20	445	+	1.251e−04	46	419	+	8.466e−06	449	16	−
F26	9.627e−04	72	393	+	5.984e−02	141	324	=	6.424e−03	100	365	+
F27	2.127e−06	2	463	+	1.734e−06	0	465	+	6.984e−06	14	451	+
F28	1.977e−07	0	465	+	3.414e−07	0	351	+	3.258e−03	78	328	+
+/=/−				17/5/6				21/4/3				17/8/3
Function	PPSO vs. WFIPS				PPSO vs. CPSO				PPSO vs. CLPSO			
	p-value	R+	R−	Winner	p-value	R+	R−	Winner	p-value	R+	R−	Winner
F1	1.733e−06	0	465	+	1.734e−06	0	465	+	7.864e−07	0	465	+
F2	1.734e−06	0	465	+	1.921e−06	1	464	+	1.734e−06	0	465	+
F3	2.370e−05	438	27	−	1.734e−06	0	465	+	2.353e−06	3	462	+
F4	1.921e−06	464	1	−	1.734e−06	0	465	+	1.319e−02	353	112	−
F5	1.734e−06	0	465	+	1.734e−06	0	465	+	1.734e−06	0	465	+
F6	3.493e−01	278	187	=	8.590e−02	149	316	=	1.714e−01	299	166	=
F7	3.317e−04	407	58	−	1.734e−06	0	465	+	1.734e−06	0	465	+
F8	4.950e−02	137	328	+	1.108e−02	109	356	+	3.683e−02	131	334	+
F9	1.734e−06	0	465	+	1.734e−06	0	465	+	1.734e−06	0	465	+
F10	1.734e−06	0	465	+	9.271e−03	106	359	+	1.734e−06	0	465	+
F11	1.921e−06	1	464	+	1.734e−06	465	0	−	1.734e−06	465	0	−
F12	1.734e−06	0	465	+	1.734e−06	0	465	+	1.734e−06	0	465	+
F13	2.127e−06	2	463	+	1.734e−06	0	465	+	1.734e−06	0	465	+
F14	1.734e−06	0	465	+	1.734e−06	465	0	−	1.734e−06	465	0	−
F15	1.734e−06	0	465	+	1.734e−06	0	465	+	1.734e−06	0	465	+
F16	1.734e−06	0	465	+	1.921e−06	1	464	+	1.734e−06	0	465	+
F17	1.734e−06	0	465	+	1.734e−06	465	0	−	1.734e−06	465	0	−
F18	1.734e−06	0	465	+	1.734e−06	0	465	+	1.734e−06	0	465	+
F19	1.734e−06	0	465	+	1.734e−06	465	0	−	1.734e−06	465	0	−
F20	4.534e−04	62	403	+	1.734e−06	0	465	+	1.921e−06	1	464	+
F21	2.989e−01	283	182	=	1.714e−01	166	299	=	1.477e−04	417	48	−
F22	1.734e−06	0	465	+	1.734e−06	465	0	−	1.734e−06	465	0	−
F23	1.734e−06	0	465	+	2.353e−06	3	462	+	3.724e−05	32	433	+
F24	1.734e−06	0	465	+	1.734e−06	0	465	+	2.353e−06	3	462	+
F25	1.734e−06	0	465	+	1.734e−06	0	465	+	1.734e−06	0	465	+
F26	2.703e−02	340	125	−	3.065e−04	57	408	+	6.639e−04	398	67	−
F27	1.734e−06	0	465	+	1.734e−06	0	465	+	7.271e−03	102	363	+
F28	1.734e−06	0	465	+	1.734e−06	0	465	+	1.734e−06	0	465	+
+/=/−				22/2/4				21/2/5				19/1/8
Function	PPSO vs. MLPSO				PPSO vs. MPSO				PPSO vs. CCPSO-ISM			
	p-value	R+	R−	Winner	p-value	R+	R−	Winner	p-value	R+	R−	Winner
F1	2.727e−06	0	253	+	1.734e−06	0	465	+	1.734e−06	0	465	+
F2	1.127e−05	446	19	−	6.639e−04	67	398	+	1.734e−06	0	465	+
F3	6.733e−01	253	212	=	5.716e−01	205	260	=	1.921e−06	1	464	+
F4	1.734e−06	465	0	−	1.734e−06	465	0	−	2.597e−05	28	437	+
F5	3.998e−07	0	465	+	1.734e−06	0	465	+	1.734e−06	0	465	+
F6	1.921e−06	464	1	−	1.057e−04	44	421	+	1.036e−03	73	392	+
F7	2.353e−06	3	462	+	2.564e−02	124	341	+	1.921e−06	1	464	+
F8	4.492e−02	135	330	+	1.480e−02	114	351	+	6.836e−03	101	364	+
F9	1.734e−06	0	465	+	2.879e−06	5	460	+	1.734e−06	0	465	+
F10	6.156e−04	399	66	−	5.710e−02	140	325	=	1.734e−06	0	465	+
F11	2.603e−06	4	461	+	8.221e−02	148	317	=	1.734e−06	465	0	−
F12	2.127e−06	2	463	+	1.734e−06	0	465	+	1.734e−06	0	465	+
F13	1.127e−05	19	446	+	3.515e−06	7	458	+	1.734e−06	0	465	+
F14	1.254e−01	158	307	=	2.431e−02	342	123	−	1.921e−06	464	1	−
F15	1.921e−06	1	464	+	4.729e−06	10	455	+	2.353e−06	3	462	+
F16	1.734e−06	0	465	+	1.734e−06	0	465	+	1.921e−06	1	464	+
F17	1.734e−06	0	465	+	2.127e−06	2	463	+	2.412e−04	54	411	+
F18	1.734e−06	0	465	+	1.921e−06	1	464	+	1.734e−06	0	465	+
F19	2.452e−01	176	289	=	1.593e−03	79	386	+	1.915e−01	169	296	=
F20	2.452e−01	176	289	=	1.639e−05	23	442	+	1.734e−06	0	465	+
F21	9.801e−02	232	119	=	1.799e−05	24	441	+	7.514e−05	425	40	−
F22	6.836e−03	101	364	+	1.986e−01	295	170	=	3.112e−05	435	30	−
F23	7.157e−04	68	397	+	4.528e−01	196	269	=	3.589e−04	59	406	+
F24	1.734e−06	0	465	+	9.711e−05	43	422	+	1.734e−06	0	465	+
F25	4.286e−06	9	456	+	1.477e−04	48	417	+	1.734e−06	0	465	+
F26	2.957e−03	88	377	+	9.368e−02	151	314	=	6.156e−04	399	66	−
F27	1.734e−06	0	465	+	1.639e−05	23	442	+	6.836e−03	364	101	−
F28	4.789e−05	0	171	+	1.734e−06	0	465	+	1.734e−06	0	465	+
+/=/−				19/5/4				20/6/2				21/1/6

Section 3.6. To evaluate the efficiency of the PPSO, we report the running time of PPSO and the compared PSO algorithms on CEC2013 with $d = 30$ for 30 independent runs in Table 4.

It can be seen that PSO achieves the least running time in 22 out of 28 cases, significantly outperforming the other PSO algorithms. The reason is that the PSO has only basic operations that learn from particle's own best and the only global best. PSO is followed by UPSO, CPSO and WFIPS. The latter three add one or more simple operations to PSO, so they require a little more running time than PSO. Our PPSO ranks fifth and is slightly better

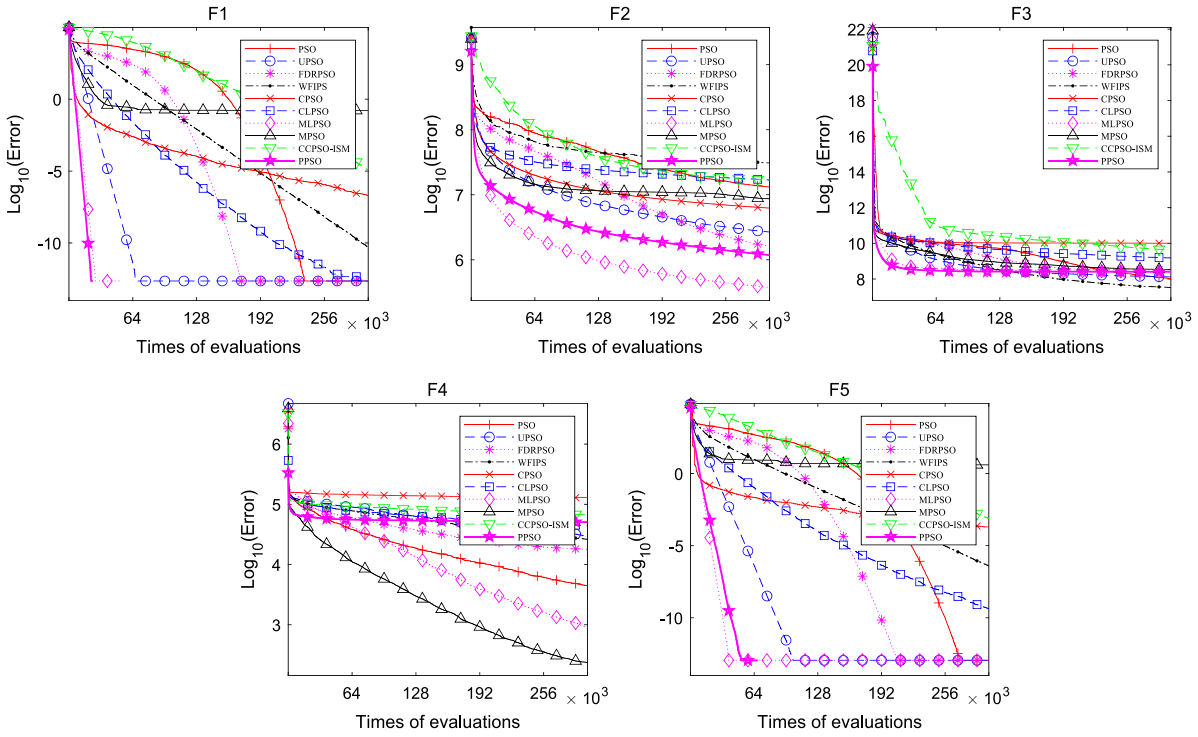

Fig. 2. Convergence speed of unimodal functions $F1$ – $F5$ of CEC2013 with $d = 30$.

than FDRPSO. Although the overall time performance of FDRPSO is average among all the methods, it achieves the best running time in 6 cases, especially with 4 composite functions ($F24$ — —$F27$). As for the rest four PSO variants (CLPSO, MLPSO, MPSO, and CCPSO-ISM), they clearly underperform the PPSO in terms of running time. An extreme instance is MPSO, whose running time is more than 10 times longer than PPSO on some test functions.

In conclusion, although the running time of PPSO is inferior to that of PSO and some early proposed PSO variants (UPSO, WFIPS and CPSO), it has obvious advantages over some recently proposed PSO variants (MLPSO, MPSO and CCPSO-ISM). Considering the accuracy, Wilcoxon signed-rank test results, and convergence together, the PPSO is promising for numeric optimization.

4.4. Influence of parameter settings

In this subsection, we will study the influence of pyramid structure as well as the parameter ρ in Eq. (4).

4.4.1. Pyramid structure

To test the influence of pyramid structure in PPSO, we fix all the parameters except for adjusting the pyramid structure to run experiments on CEC2013 with $d = 30$ for 30 independent times. We use three types of population sizes (64, 48, and 32) and eleven types of topology structures, i.e., "8−8−8−8−8−8−8−8" (PS1), "16 − 16 − 16 − 16" (PS2), "32 − 32" (PS3), "2 − 4 − 8 − 16 − 34" (PS4), "8 − 16 − 40" (PS5), "10 − 20 − 34" (PS6), "4 − 8 − 20 − 32" (PS7), "4 − 8 − 16 − 20" (PS8), "6 − 12 − 30" (PS9), "2 − 4 − 8 − 18" (PS10), and "4 − 8 − 20" (PS11) to run the PPSO. The first seven pyramid structures (PS1–PS7) are associated with the population size of 64, while PS8–PS9 and PS10–PS11 are associated with the population sizes of 48 and 32, respectively. Note that the first three structures are special because each of them has the identical size of each layer. We call this structure "stairs structure". The experimental results are listed in Table 5.

From this table, we can see that the accuracy of the PPSO becomes worse and worse with the decrease of population size. The best results with PS8–PS12 are worse than the worst results with PS1–PS7. Therefore, the following findings only focus on the population size of 64, i.e., the structure of PS1–PS7: (1) In general, PPSO performs well for the benchmark functions. For example, all the structures are able to find the optimal solutions of $F1$ and $F5$, while the other models fail to find them, as shown in the aforementioned Table 1; (2) The first two stairs structures (PS1 and PS2) perform the worst among all the seven structures. Although PS3 slightly improves the accuracy and its results are better than PS4, it does not perform as well as the last three pyramid structure (PS5–PS7), indicating that the pyramid structure is advantageous over stairs structure; (3) PS5 achieves the best results 11 times, followed by PS6's 8 times. Both PS5 and PS6 have a three-layer structure, outperforming the other structures; (4) PS3, PS4 and PS7 perform moderately among all the structures.

With these findings, we can conclude that the pyramid structure is generally superior to the stairs structure and a three-layer pyramid structure is suitable for 30-dimensional benchmark functions of CEC2013.

4.4.2. Controlling parameter ρ

In the proposed PPSO, the particles at one layer are divided into losers and winners. The winners have chances to learn from the several global best particles at the top layer. The parameter ρ in Eq. (4) is used to control the weight of learning from the particles at the top layer. In theory, a smaller ρ will make convergence speed slow while a larger one will result in premature. How to choose an appropriate ρ that can balance the convergence speed and the convergence accuracy is an important issue for PPSO.

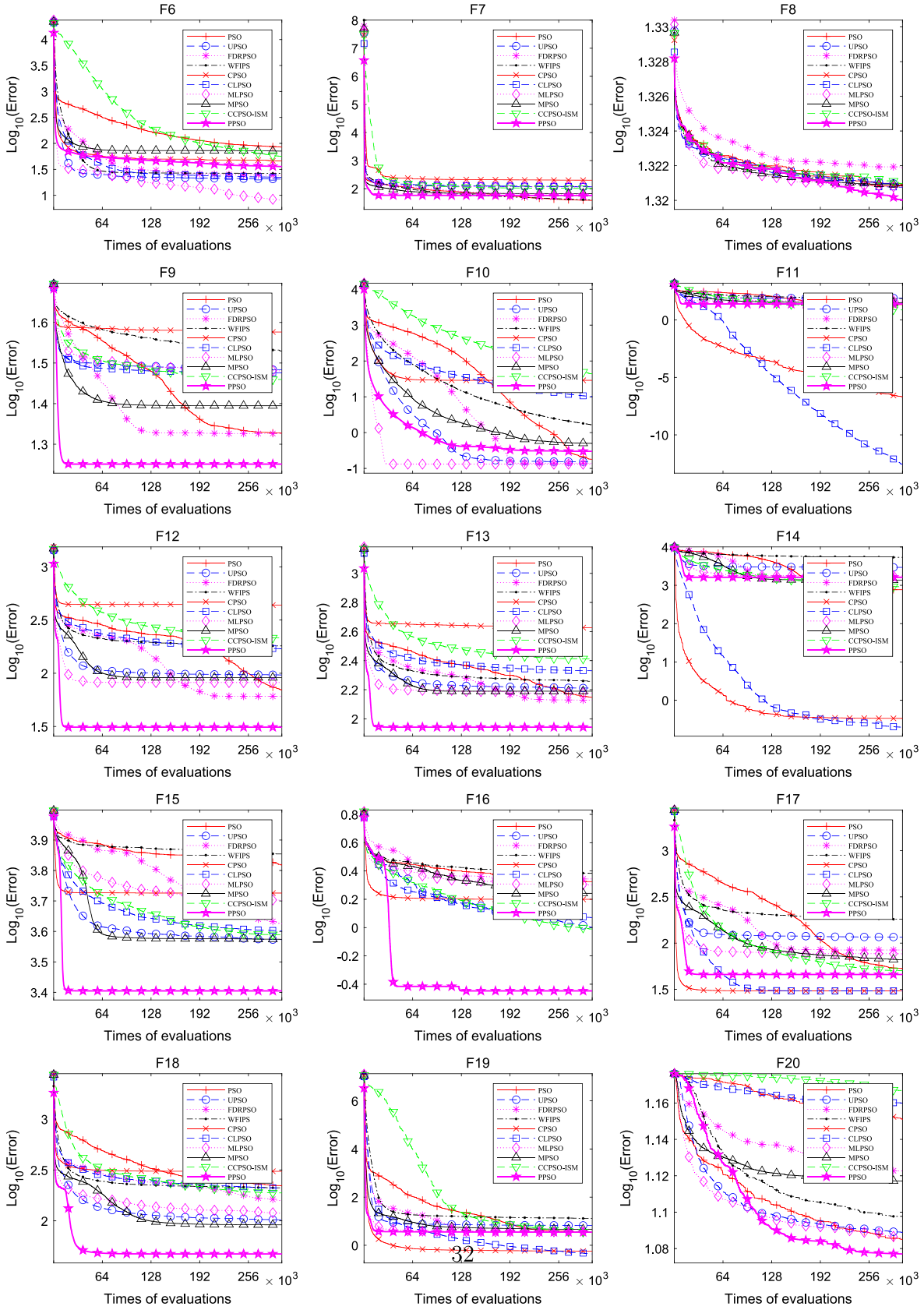

Fig. 3. Convergence speed of multimodal functions $F6$ – $F20$ of CEC2013 with $d = 30$.

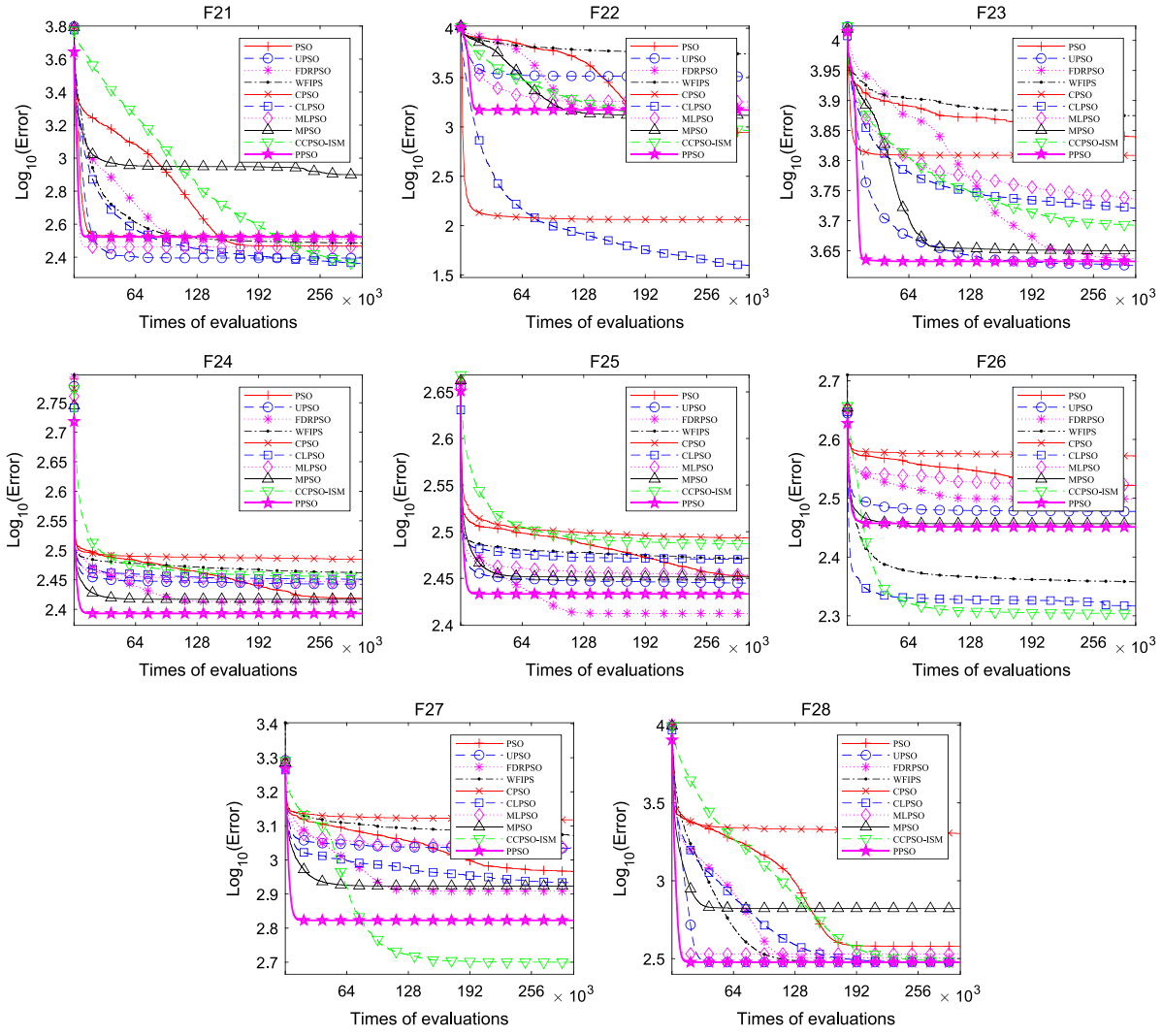

Fig. 4. Convergence speed of composite functions $F21$ – $F28$ of CEC2013 with $d = 30$.

To study the influence of ρ, we fix the other parameters and independently run the PPSO on CEC2013 benchmark functions with $d = 30$ for 30 times with a varying ρ in the set of $\{0, 0.01, 0.02, 0.04, 0.08, 0.15, 0.2, 0.4, 0.6, 0.8, 1, 1.4946, 2\}$. Among them, 1.4946 and 2 are very popular acceleration factors for common PSO variants. Note that $\rho = 0$ means that the winners will not learn from the particles at the top layer. The experimental results are listed in Table 6. Once again, the PPSO models with all ρs succeed in searching the optimal values of $F1$ and $F5$. Each model except for $\rho = 0$ is ranked first at least 3 times and the interval of mean rank of all models is $[5.43, 7.04]$ which is relatively narrow, showing that the performance of different models is relatively stable. It also reveals that the cooperation strategy of learning from the particles at the top layer for winners helps to find optimal solutions. The best ρ is 0.4 that achieves the best results in 6 out of 28 cases and obtains the minimal mean rank value of 5.43, whose distance to the second best mean rank, i.e., 5.89 by $\rho = 0.04$, is much farther than the distances between any other two neighboring mean ranks. For 1.4946 and 2, their final ranks are 12 and 10, respectively, indicating that they do not work well for PPSO. The experimental results suggest the selection of ρ near 0.4.

4.5. Summarization

The PPSO first builds a pyramid via competition among all the particles. Then the particles at one layer are grouped into winners and losers by the second round competition. Finally, different cooperation strategies are designed to update the information of the winners and the losers.

From the above analysis and comparisons, we can summarize the PPSO as the following: (1) The pyramid structure and the competitive and cooperative strategies are the key attributes of PPSO, which are significantly different from existing PSO variants; (2) The final ranks of PPSO are always 1 with different test suits and different dimensions, indicating the PPSO is constantly superior to the canonical PSO and the eight state-of-the-art PSO variants. This is confirmed by Wilcoxon signed-rank test from a statistical perspective; (3) The PPSO has good convergence speed. In many cases, it converges to a best solution at an earlier stage than the compared PSO algorithms; (4) The running time of PSO is inferior to the canonical PSO and some early-proposed PSO variants but is advantageous over some newly-proposed ones; (5) The pyramid structure and ρ are two important parameters for

Table 4
Running time of CEC2013 with $d = 30$ (in seconds).

Function	Metrics	PSO	UPSO	FDRPSO	WFIPS	CPSO	CLPSO	MLPSO	MPSO	CCPSO-ISM	PPSO
F1	Mean	**1.4339**	1.6731	6.4929	2.0251	1.8115	5.3578	3.8332	66.0056	11.0001	3.9368
	Std	0.0260	0.0248	0.1713	0.0287	0.0638	0.1208	0.1299	0.3085	0.1163	0.0870
	Rank	**1**	2	8	4	3	7	5	10	9	6
F2	Mean	**1.9099**	2.1384	6.6536	2.4947	2.3832	19.8978	4.5694	66.7734	12.7385	4.4800
	Std	0.0271	0.0273	0.1588	0.0457	0.0441	5.0121	0.1222	0.7149	0.2623	0.0630
	Rank	**1**	2	7	4	3	9	6	10	8	5
F3	Mean	**1.9236**	2.1985	6.6802	2.4865	2.4400	13.3907	4.5756	66.8280	12.9159	4.5584
	Std	0.0232	0.0420	0.2115	0.0404	0.0812	2.3015	0.1567	0.5046	0.4643	0.0995
	Rank	**1**	2	7	4	3	9	6	10	8	5
F4	Mean	**1.6470**	1.8842	6.5206	2.2337	2.0576	14.9837	4.2036	66.3600	12.3713	4.1935
	Std	0.0120	0.0354	0.1742	0.0288	0.0693	3.7585	0.6615	0.3475	0.2185	0.0958
	Rank	**1**	2	7	4	3	9	6	10	8	5
F5	Mean	**1.7157**	1.9569	6.5606	2.3057	2.1904	5.9683	4.3443	66.4919	11.6513	4.2966
	Std	0.0291	0.0381	0.1798	0.0209	0.0605	0.2541	0.4333	0.3695	0.1676	0.0982
	Rank	**1**	2	8	4	3	7	6	10	9	5
F6	Mean	**1.6024**	1.8400	6.5659	2.1695	1.9969	8.2008	4.4107	66.2764	11.6254	4.1446
	Std	0.0122	0.0158	0.1933	0.0183	0.0529	0.7554	0.2062	0.3392	0.0766	0.1288
	Rank	**1**	2	7	4	3	8	6	10	9	5
F7	Mean	**2.8824**	3.0452	6.8505	3.4277	3.6844	18.7790	5.8595	67.8319	16.9305	5.5601
	Std	0.0188	0.0298	0.1815	0.0480	0.0898	3.0214	0.3747	0.2670	0.1723	0.1027
	Rank	**1**	2	7	3	4	9	6	10	8	5
F8	Mean	**2.4100**	2.6327	6.7708	3.0011	2.9975	17.7062	7.7276	67.8495	14.7475	5.0250
	Std	0.0412	0.0411	0.1985	0.0472	0.0903	3.6491	14.5505	1.1720	0.2297	0.0846
	Rank	**1**	2	6	4	3	9	7	10	8	5
F9	Mean	16.5570	16.7258	**11.3995**	17.1511	20.3981	37.7469	24.6463	82.2465	79.2521	21.7908
	Std	0.4057	0.0812	0.5727	0.1052	0.5421	5.1983	0.7838	0.3030	0.3899	0.2957
	Rank	2	3	**1**	4	5	8	7	10	9	6
F10	Mean	**1.9578**	2.1114	6.6552	2.5450	2.4777	7.6257	4.4468	66.8061	12.9393	4.4629
	Std	0.0216	0.0360	0.1971	0.0480	0.0556	0.7935	0.1305	0.4417	0.1848	0.0592
	Rank	**1**	2	7	4	3	8	5	10	9	6
F11	Mean	**1.9840**	2.1966	6.6886	2.6207	2.4655	6.1245	4.5629	67.1982	13.5913	4.5702
	Std	0.0330	0.0318	0.1930	0.0700	0.0617	0.2043	0.1850	0.9776	0.1545	0.0972
	Rank	**1**	2	8	4	3	7	5	10	9	6
F12	Mean	**2.4824**	2.6468	6.8146	3.0417	3.0095	10.0244	5.1459	67.4735	15.2381	5.0363
	Std	0.0596	0.0431	0.1561	0.0335	0.0900	1.1307	0.1501	0.2762	0.2111	0.1279
	Rank	**1**	2	7	4	3	8	6	10	9	5
F13	Mean	**2.5294**	2.6972	6.7959	3.1090	3.0205	9.4276	5.1084	67.6980	15.2419	5.0799
	Std	0.0520	0.0353	0.1919	0.0419	0.0751	0.9129	0.1652	0.4225	0.1689	0.1239
	Rank	**1**	2	7	4	3	8	6	10	9	5
F14	Mean	**2.2045**	2.4333	6.7538	2.8369	2.6409	7.9935	4.8700	67.1671	13.9778	4.8539
	Std	0.0212	0.0159	0.2120	0.0436	0.0782	0.7012	0.1355	0.4456	0.1975	0.0815
	Rank	**1**	2	7	4	3	8	6	10	9	5
F15	Mean	**2.4069**	2.6140	6.7237	3.0082	2.9904	23.6945	5.1424	67.4353	14.6359	5.0362
	Std	0.0252	0.0442	0.2161	0.0522	0.0687	13.4074	0.1598	0.4070	0.2513	0.1046
	Rank	**1**	2	7	4	3	9	6	10	8	5
F16	Mean	12.2580	12.4681	**9.9910**	12.8449	15.0368	29.2376	18.3534	77.9197	34.1328	16.8421
	Std	0.0922	0.0757	0.3894	0.0869	0.2661	4.8367	0.5255	1.1086	0.1972	0.2124
	Rank	2	3	**1**	4	5	8	7	10	9	6
F17	Mean	**1.7271**	1.9542	6.5648	2.3318	2.1614	5.5836	4.2290	66.7294	12.8631	4.2817
	Std	0.0212	0.0408	0.1984	0.0207	0.0740	0.0931	0.1187	0.3543	0.2663	0.0650
	Rank	**1**	2	8	4	3	7	5	10	9	6
F18	Mean	**2.0332**	2.2543	6.6376	2.6266	2.5308	5.9129	4.6403	67.1454	13.9385	4.5838
	Std	0.0122	0.0245	0.1761	0.0495	0.0706	0.1281	0.1710	0.3402	0.1507	0.0861
	Rank	**1**	2	8	4	3	7	6	10	9	5
F19	Mean	**1.7183**	1.9474	6.5957	2.3259	2.1035	6.1599	4.3987	66.4241	12.4112	4.2184
	Std	0.0153	0.0203	0.1815	0.0643	0.0661	0.2763	0.1371	0.3072	0.1711	0.0666
	Rank	**1**	2	8	4	3	7	6	10	9	5
F20	Mean	**2.1056**	2.3685	6.5772	2.7319	2.9094	7.1016	4.6789	66.9908	14.0999	4.7365
	Std	0.0365	0.0436	0.2061	0.0648	0.1071	0.5026	0.1575	0.2991	0.2703	0.0950
	Rank	**1**	2	7	3	4	8	5	10	9	6
F21	Mean	**4.3835**	4.5892	7.6216	4.9898	5.5342	8.3353	7.7893	69.6009	15.9934	7.3638
	Std	0.0411	0.0477	0.2043	0.0502	0.1026	0.2004	0.3037	0.2320	0.2028	0.1348
	Rank	**1**	2	6	3	4	8	7	10	9	5
F22	Mean	**4.6711**	4.9108	7.5493	5.3113	5.7286	13.7849	8.3761	69.7275	19.0483	7.8338
	Std	0.0445	0.0641	0.2648	0.0463	0.1387	1.8739	0.2472	0.2004	0.2648	0.0897
	Rank	**1**	2	5	3	4	8	7	10	9	6
F23	Mean	**5.2325**	5.4388	7.6584	5.8372	6.4496	22.5468	9.2305	70.3601	20.6697	8.3045
	Std	0.0277	0.0748	0.2366	0.0498	0.1543	3.9295	0.2878	0.2184	0.1802	0.1228
	Rank	**1**	2	5	3	4	9	7	10	8	6
F24	Mean	19.1682	19.6109	**10.0685**	19.9233	23.5920	47.3225	28.3689	85.4023	86.3035	24.9450
	Std	0.1028	0.0886	0.8866	0.1219	0.4266	7.2501	0.8800	0.7131	0.3252	0.3327
	Rank	2	3	**1**	4	5	8	7	9	10	6
F25	Mean	19.2185	19.5962	**9.3707**	19.9689	23.6245	76.8134	29.0164	85.3594	86.3416	24.9502
	Std	0.0632	0.0865	0.7139	0.1026	0.4436	32.9934	1.9700	0.5862	0.3694	0.3654
	Rank	2	3	**1**	4	5	8	7	9	10	6
F26	Mean	20.9416	21.2567	**12.6079**	21.6362	25.4471	44.6176	30.8272	87.4216	89.7587	26.8601
	Std	0.0988	0.0817	0.7704	0.0933	0.4412	7.7163	2.3622	0.3899	0.4622	0.3854
	Rank	2	3	**1**	4	5	8	7	9	10	6
F27	Mean	20.5214	20.8957	**11.7867**	21.2598	24.9035	51.6714	30.6657	86.8093	88.3743	26.2590
	Std	0.1651	0.1438	0.7134	0.1149	0.4543	14.1547	2.8308	0.2594	0.3163	0.3619
	Rank	2	3	**1**	4	5	8	7	9	10	6
F28	Mean	**6.4514**	6.4619	8.3559	7.0059	8.0909	11.4915	11.1228	71.9775	23.6993	9.6132
	Std	0.0896	0.1043	0.2158	0.0790	0.1915	0.5450	3.8795	0.3791	0.2990	0.1627
	Rank	**1**	2	5	3	4	8	7	10	9	6
Mean rank		1.21	2.21	5.64	3.79	3.64	8.04	6.21	9.86	8.89	5.50
Final rank		1	2	6	4	3	8	7	10	9	5

the PPSO, and PS5 and 0.4 are two appropriate values for these two parameters on CEC2013 test suite with $d = 30$.

5. Conclusion

In this paper, a PSO variant called PPSO is presented, in which the concepts of pyramid and novel strategies of competition and cooperation are proposed and introduced into PSO. In PPSO, the particles are placed to different layers via competition and then further grouped into winners or losers. The winners and the losers use different strategies to update their information. To verify the effect of PPSO, extensive experiments are conducted on CEC2013 and CECE2017 test suites with 30 dimensions and 50 dimensions. PPSO achieves the best accuracy, the best Wilcoxon signed-rank test results, the best convergence speed, and comparable running time among the compared PSO algorithms.

The two parameters of PPSO have great impacts on optimization. One limitation of the current PPSO is that the

Table 5
Results of different pyramid structure on CEC2013 with $d = 30$.

Function	Metrics	Population size and pyramid structure										
		64							48		32	
		PS1	PS2	PS3	PS4	PS5	PS6	PS7	PS8	PS9	PS10	PS11
F1	Mean	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	1.4323e−02	3.7951e−01
	Std	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	4.3064e−02	1.0289e+00
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	10	11
F2	Mean	2.1963e+06	1.5405e+06	1.3967e+06	**1.1397e+06**	1.2965e+06	1.3569e+06	1.1778e+06	3.7080e+06	4.5622e+06	2.1515e+07	2.5931e+07
	Std	6.5793e+05	6.3418e+05	5.4700e+05	4.5798e+05	6.5816e+05	5.4610e+05	6.4602e+05	2.3131e+06	1.8525e+06	9.6257e+06	1.0814e+07
	Rank	7	6	5	**1**	3	4	2	8	9	10	11
F3	Mean	**1.6851e+08**	2.1765e+08	2.3816e+08	1.8040e+08	3.2184e+08	2.8798e+08	2.5575e+08	1.2896e+09	2.3686e+09	5.4441e+09	1.0061e+10
	Std	2.9694e+08	2.5399e+08	2.1995e+08	2.0412e+08	4.3681e+08	3.3471e+08	4.6793e+08	2.0669e+09	3.7676e+09	2.5661e+09	5.6643e+09
	Rank	**1**	3	4	2	7	6	5	8	9	10	11
F4	Mean	5.0900e+04	4.7480e+04	4.9647e+04	5.2067e+04	4.9628e+04	**4.7396e+04**	5.0682e+04	5.4207e+04	5.8988e+04	6.4804e+04	7.5975e+04
	Std	9.2979e+03	8.6355e+03	1.3233e+04	1.2711e+04	7.6222e+03	9.6827e+03	1.2018e+04	1.0412e+04	1.0629e+04	1.1894e+04	1.5906e+04
	Rank	6	2	4	7	3	**1**	5	8	9	10	11
F5	Mean	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	1.8948e−13	**0.0000e+00**	8.7586e+00	1.4484e+01
	Std	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	1.0378e−12	0.0000e+00	2.7066e+01	1.9107e+01
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	9	**1**	10	11
F6	Mean	3.9912e+01	**3.3781e+01**	4.2148e+01	3.6594e+01	3.9833e+01	3.7815e+01	3.5814e+01	4.4385e+01	5.7481e+01	8.9963e+01	9.7331e+01
	Std	2.1109e+01	2.3200e+01	2.7794e+01	2.3502e+01	2.5638e+01	2.6864e+01	2.4646e+01	2.5790e+01	2.4959e+01	2.4763e+01	3.4807e+01
	Rank	6	**1**	7	3	5	4	2	8	9	10	11
F7	Mean	7.2904e+01	5.7781e+01	5.9385e+01	5.9656e+01	6.1662e+01	**5.5120e+01**	5.7248e+01	7.7601e+01	7.2103e+01	9.9325e+01	9.8687e+01
	Std	2.5038e+01	1.8695e+01	1.7061e+01	1.5343e+01	1.7534e+01	1.5155e+01	1.9304e+01	2.1926e+01	1.6859e+01	3.0779e+01	3.6861e+01
	Rank	8	3	4	5	6	**1**	2	9	7	11	10
F8	Mean	2.0953e+01	2.0943e+01	2.0958e+01	2.0918e+01	2.0914e+01	2.0913e+01	**2.0893e+01**	2.0947e+01	2.0924e+01	2.0960e+01	2.0935e+01
	Std	5.6161e−02	5.0274e−02	6.2150e−02	7.2778e−02	5.8029e−02	5.8779e−02	7.5934e−02	5.2503e−02	5.7958e−02	8.5902e−02	6.4941e−02
	Rank	9	7	10	4	3	2	**1**	8	5	11	6
F9	Mean	1.8599e+01	1.8401e+01	1.7054e+01	1.7822e+01	1.6800e+01	**1.6372e+01**	1.7825e+01	1.9552e+01	1.7736e+01	2.2717e+01	2.1793e+01
	Std	2.2996e+00	2.3617e+00	2.7313e+00	2.3800e+00	2.6185e+00	2.1845e+00	2.4455e+00	2.7634e+00	2.8613e+00	2.2809e+00	2.7733e+00
	Rank	8	7	3	5	2	**1**	6	9	4	11	10
F10	Mean	2.6757e−01	3.8952e−01	3.9202e−01	2.9048e−01	2.6648e−01	**2.4855e−01**	3.0127e−01	1.9474e+00	3.8201e+00	4.1233e+01	6.3499e+01
	Std	2.0032e−01	2.3695e−01	3.3220e−01	1.7788e−01	2.0607e−01	1.5472e−01	2.5975e−01	1.0394e+00	4.4137e+00	2.5904e+01	3.9698e+01
	Rank	3	6	7	4	2	**1**	5	8	9	10	11
F11	Mean	3.1739e+01	2.7991e+01	2.6764e+01	2.8356e+01	**2.4808e+01**	2.5272e+01	2.4973e+01	3.3961e+01	3.3165e+01	6.6322e+01	7.3572e+01
	Std	7.8053e+00	7.9542e+00	8.3176e+00	7.5401e+00	8.5107e+00	7.1346e+00	8.6516e+00	8.9020e+00	1.1975e+01	1.6299e+01	2.3295e+01
	Rank	7	5	4	6	**1**	3	2	9	8	10	11
F12	Mean	4.1987e+01	3.5354e+01	3.3331e+01	3.2203e+01	**2.7461e+01**	3.1408e+01	3.1142e+01	4.0994e+01	3.9900e+01	7.3146e+01	6.9459e+01
	Std	1.0188e+01	8.6214e+00	8.3480e+00	1.1228e+01	7.8151e+00	9.3689e+00	1.0494e+01	1.1608e+01	1.2393e+01	2.2203e+01	2.1422e+01
	Rank	9	6	5	4	**1**	3	2	8	7	11	10
F13	Mean	1.0819e+02	8.6874e+01	9.1799e+01	8.7038e+01	**8.3809e+01**	8.4994e+01	8.7578e+01	1.2601e+02	1.0363e+02	1.7032e+02	1.5389e+02
	Std	2.3196e+01	2.6856e+01	2.5337e+01	2.3112e+01	3.0157e+01	2.9144e+01	3.3250e+01	2.8563e+01	3.1060e+01	3.5391e+01	4.1436e+01
	Rank	8	3	6	4	**1**	2	5	9	7	11	10
F14	Mean	1.9842e+03	1.5745e+03	1.3636e+03	1.7331e+03	**1.3611e+03**	1.4301e+03	1.6186e+03	1.8513e+03	1.7419e+03	2.4708e+03	2.4132e+03
	Std	5.3484e+02	4.2718e+02	3.7624e+02	4.0651e+02	3.2565e+02	2.9991e+02	3.6706e+02	5.2575e+02	4.4073e+02	5.8486e+02	6.8877e+02
	Rank	9	4	2	6	**1**	3	5	8	7	11	10
F15	Mean	2.8391e+03	2.5774e+03	2.2424e+03	2.5895e+03	**2.0349e+03**	2.3740e+03	2.5392e+03	2.7478e+03	2.6477e+03	3.4344e+03	3.4034e+03
	Std	6.0253e+02	5.0744e+02	5.3914e+02	4.5109e+02	5.6098e+02	5.9857e+02	6.4130e+02	5.3018e+02	6.0306e+02	6.3532e+02	6.5521e+02
	Rank	9	5	2	6	**1**	3	4	8	7	11	10
F16	Mean	3.9373e−01	3.7702e−01	2.8592e−01	3.1862e−01	2.5973e−01	**2.5447e−01**	3.5693e−01	3.3145e−01	3.7059e−01	5.3287e−01	4.7007e−01
	Std	1.6088e−01	1.8786e−01	1.2442e−01	1.5924e−01	1.2399e−01	1.1370e−01	1.6199e−01	1.1524e−01	2.1821e−01	2.3702e−01	1.6904e−01
	Rank	9	8	3	4	2	**1**	6	5	7	11	10
F17	Mean	5.4507e+01	4.8649e+01	4.8996e+01	4.7113e+01	4.6280e+01	4.6079e+01	**4.5796e+01**	5.3813e+01	5.7032e+01	8.7913e+01	8.6824e+01
	Std	7.9588e+00	6.0838e+00	5.3728e+00	5.0157e+00	4.7663e+00	4.7540e+00	4.5658e+00	8.6351e+00	6.6205e+00	1.7906e+01	1.9573e+01
	Rank	8	5	6	4	3	2	**1**	7	9	11	10
F18	Mean	5.7572e+01	5.0363e+01	4.8706e+01	4.7603e+01	**4.6022e+01**	4.8334e+01	4.7265e+01	5.8209e+01	5.7295e+01	1.0324e+02	1.1152e+02
	Std	8.2224e+00	4.6894e+00	5.2754e+00	3.5206e+00	5.4559e+00	4.9710e+00	7.2691e+00	9.9641e+00	9.2527e+00	1.9020e+01	2.1558e+01
	Rank	8	6	5	3	**1**	4	2	9	7	10	11
F19	Mean	5.0105e+00	4.0878e+00	3.6066e+00	3.6940e+00	3.5887e+00	**3.3802e+00**	3.5130e+00	5.3801e+00	5.5589e+00	1.6091e+01	2.2543e+01
	Std	2.3310e+00	1.0656e+00	1.3355e+00	9.1992e−01	1.1392e+00	1.1429e+00	1.4969e+00	2.7647e+00	2.6192e+00	1.0526e+01	1.7281e+01
	Rank	7	6	4	5	3	**1**	2	8	9	10	11
F20	Mean	1.2197e+01	1.0965e+01	**1.0597e+01**	1.3058e+01	1.0985e+01	1.0970e+01	1.1941e+01	1.2181e+01	1.1710e+01	1.3244e+01	1.2442e+01
	Std	4.9660e−01	7.5994e−01	7.6455e−01	5.3402e−01	8.0206e−01	7.8074e−01	6.4445e−01	7.9526e−01	9.4630e−01	8.6063e−01	8.2773e−01
	Rank	8	2	**1**	10	4	3	6	7	5	11	9
F21	Mean	3.3306e+02	3.2161e+02	3.2161e+02	3.1495e+02	**2.9538e+02**	3.1059e+02	3.3263e+02	3.2973e+02	3.1349e+02	3.4910e+02	3.8994e+02
	Std	7.9361e+01	8.3217e+01	8.3217e+01	8.8695e+01	9.1536e+01	6.9396e+01	9.3718e+01	8.2823e+01	8.2790e+01	1.1044e+02	1.7651e+02
	Rank	9	6	5	4	**1**	2	8	7	3	10	11
F22	Mean	2.0388e+03	1.5726e+03	**1.2607e+03**	1.8474e+03	1.3267e+03	1.2918e+03	1.4856e+03	2.2316e+03	1.7815e+03	3.0808e+03	2.8542e+03
	Std	5.8191e+02	4.6064e+02	4.4176e+02	7.3728e+02	6.6023e+02	4.0815e+02	4.6702e+02	6.0304e+02	6.8009e+02	1.0074e+03	9.8789e+02
	Rank	8	5	**1**	7	3	2	4	9	6	11	10
F23	Mean	4.8311e+03	3.8756e+03	**3.0018e+03**	4.6236e+03	3.6654e+03	4.0032e+03	4.2898e+03	4.6130e+03	3.9587e+03	5.1932e+03	4.6149e+03
	Std	8.6272e+02	9.8522e+02	7.0730e+02	8.7095e+02	9.1862e+02	1.0824e+03	8.0288e+02	8.6572e+02	9.2200e+02	1.0540e+03	1.0236e+03
	Rank	10	3	**1**	9	2	5	6	7	4	11	8
F24	Mean	2.4965e+02	2.4442e+02	2.4099e+02	2.4767e+02	**2.4053e+02**	2.4487e+02	2.4722e+02	2.5463e+02	2.5270e+02	2.6732e+02	2.6350e+02
	Std	9.6740e+00	9.9529e+00	9.3327e+00	1.3371e+01	9.6865e+00	1.3312e+01	8.9538e+00	1.0786e+01	8.8147e+00	1.1092e+01	9.4553e+00
	Rank	7	3	2	6	**1**	4	5	9	8	11	10
F25	Mean	2.7424e+02	2.7358e+02	2.7122e+02	**2.6993e+02**	2.7250e+02	2.7147e+02	2.7120e+02	2.7737e+02	2.7808e+02	2.8261e+02	2.8603e+02
	Std	7.1245e+00	8.1189e+00	5.6714e+00	7.8965e+00	7.3115e+00	6.2987e+00	7.0210e+00	8.4893e+00	8.0072e+00	8.0547e+00	6.6371e+00
	Rank	7	6	3	**1**	5	4	2	8	9	10	11
F26	Mean	2.7343e+02	2.7935e+02	2.8585e+02	**2.4675e+02**	2.7204e+02	2.9098e+02	2.8261e+02	2.6315e+02	2.8314e+02	2.9643e+02	3.2390e+02
	Std	7.0466e+01	6.3208e+01	6.2625e+01	6.2602e+01	6.4271e+01	5.6865e+01	6.4320e+01	6.8887e+01	6.9085e+01	7.4078e+01	5.6755e+01
	Rank	4	5	8	**1**	3	9	6	2	7	10	11
F27	Mean	7.0563e+02	6.8745e+02	6.7398e+02	**6.2203e+02**	6.6828e+02	6.5198e+02	6.6499e+02	7.1409e+02	7.3244e+02	8.4353e+02	8.6072e+02
	Std	7.0190e+01	9.8341e+01	1.0669e+02	1.0153e+02	7.7744e+01	1.2075e+02	9.9048e+01	1.1463e+02	1.0961e+02	1.0230e+02	7.7199e+01
	Rank	7	6	5	**1**	4	2	3	8	9	10	11
F28	Mean	2.9601e+02	3.7831e+02	**2.9333e+02**	3.7478e+02	**2.9333e+02**	2.9511e+02	3.0000e+02	4.2954e+02	4.3586e+02	1.1924e+03	1.3740e+03
	Std	3.4139e+01	4.2892e+02	3.6515e+01	4.0956e+02	3.6515e+01	3.8116e+01	0.0000e+00	4.2780e+02	4.3783e+02	3.6934e+02	5.2586e+02
	Rank	4	7	**1**	6	**1**	3	5	8	9	10	11
Mean rank		6.71	4.57	3.93	4.29	2.54	2.79	3.71	7.57	6.82	10.46	10.29
Final rank		7	6	4	5	1	2	3	9	8	11	10

parameters are user-defined and not self-regulated. It is a challenging task to set fixed parameter values for all optimization problems. Therefore, intelligent adaptive parameter adjustment mechanisms are deserved in the future. Furthermore, we will apply the PPSO to real-world problems and extend it for multi-objective optimization.

CRediT authorship contribution statement

Taiyong Li: Methodology, Software, Supervision, Writing – original draft, Writing – review & editing. **Jiayi Shi:** Software, Visualization. **Wu Deng:** Investigation, Writing – review & editing. **Zhenda Hu:** Validation.

Table 6
Results of different ρ on CEC2013 with $d = 30$.

Function	Metrics	$\rho = 0$	$\rho = 0.01$	$\rho = 0.02$	$\rho = 0.04$	$\rho = 0.08$	$\rho = 0.15$	$\rho = 0.2$	$\rho = 0.4$	$\rho = 0.6$	$\rho = 0.8$	$\rho = 1$	$\rho = 1.4946$	$\rho = 2$
F1	Mean	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**
	Std	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F2	Mean	1.2240e+06	1.2920e+06	1.2306e+06	1.0637e+06	1.1039e+06	1.0963e+06	1.1852e+06	**9.8430e+05**	1.2606e+06	1.2604e+06	1.1895e+06	1.1975e+06	1.1403e+06
	Std	5.4750e+05	6.4569e+05	4.8754e+05	4.1833e+05	3.9125e+05	6.1257e+05	5.5633e+05	3.3123e+05	5.2201e+05	5.0911e+05	5.1176e+05	4.6819e+05	4.5642e+05
	Rank	9	13	10	2	4	3	6	**1**	12	11	7	8	5
F3	Mean	1.6617e+08	2.3058e+08	2.7446e+08	3.6888e+08	**1.3898e+08**	2.2655e+08	2.7994e+08	3.0556e+08	2.3936e+08	1.7107e+08	2.1489e+08	2.3842e+08	2.3066e+08
	Std	2.2319e+08	3.2152e+08	5.1396e+08	5.2819e+08	1.6955e+08	2.6508e+08	3.1670e+08	4.6516e+08	4.1874e+08	1.7583e+08	2.9916e+08	2.2692e+08	2.2726e+08
	Rank	2	6	10	13	**1**	5	11	12	9	3	4	8	7
F4	Mean	4.7338e+04	4.9671e+04	**4.6241e+04**	5.0374e+04	4.8167e+04	5.0128e+04	5.2368e+04	5.0040e+04	5.2305e+04	5.0286e+04	5.0787e+04	5.1582e+04	5.2116e+04
	Std	7.8081e+03	8.6613e+03	9.6708e+03	1.0723e+04	1.2743e+04	9.8389e+03	1.0531e+04	9.8556e+03	9.0185e+03	1.2788e+04	9.1575e+03	8.9107e+03	1.1856e+04
	Rank	2	4	**1**	8	3	6	13	5	12	7	9	10	11
F5	Mean	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**	**0.0000e+00**
	Std	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00	0.0000e+00
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
F6	Mean	3.8151e+01	3.6829e+01	3.6129e+01	3.6749e+01	3.6142e+01	3.7845e+01	3.5599e+01	3.6162e+01	3.6079e+01	**3.2601e+01**	3.4591e+01	3.4090e+01	3.9389e+01
	Std	2.4915e+01	2.5725e+01	2.6002e+01	2.6533e+01	2.8803e+01	2.5303e+01	2.5647e+01	2.8318e+01	2.5747e+01	2.5282e+01	2.6933e+01	2.6571e+01	2.5663e+01
	Rank	12	10	6	9	7	11	4	8	5	**1**	3	2	13
F7	Mean	5.8006e+01	5.9289e+01	6.2563e+01	5.4310e+01	5.9460e+01	6.2334e+01	5.7170e+01	5.8576e+01	5.8977e+01	**5.4279e+01**	5.9869e+01	6.1939e+01	5.9250e+01
	Std	2.2699e+01	2.0625e+01	2.0025e+01	1.3571e+01	1.7968e+01	2.2610e+01	2.3139e+01	1.8781e+01	1.9100e+01	1.3308e+01	1.6731e+01	1.8231e+01	1.9810e+01
	Rank	4	8	13	2	9	12	3	5	6	**1**	10	11	7
F8	Mean	2.0924e+01	2.0916e+01	2.0900e+01	2.0888e+01	2.0908e+01	**2.0884e+01**	2.0903e+01	2.0899e+01	2.0901e+01	2.0898e+01	2.0915e+01	2.0910e+01	2.0899e+01
	Std	7.8347e−02	7.6965e−02	7.0555e−02	6.5343e−02	8.5471e−02	8.2821e−02	7.0079e−02	9.1240e−02	7.7550e−02	5.8327e−02	5.8086e−02	5.5640e−02	8.1913e−02
	Rank	13	12	6	2	9	**1**	8	4	7	3	11	10	5
F9	Mean	1.6541e+01	1.6540e+01	**1.6475e+01**	1.8210e+01	1.7017e+01	1.7538e+01	1.6655e+01	1.6688e+01	1.7281e+01	1.7329e+01	1.7117e+01	1.7602e+01	1.7202e+01
	Std	2.7428e+00	2.8095e+00	2.0212e+00	2.6882e+00	2.7853e+00	2.1164e+00	3.0015e+00	2.7444e+00	2.5235e+00	2.5378e+00	2.8408e+00	2.9534e+00	2.2912e+00
	Rank	3	2	**1**	13	6	11	4	5	9	10	7	12	8
F10	Mean	2.6948e−01	2.7119e−01	2.7377e−01	2.4867e−01	2.6962e−01	2.8625e−01	2.4720e−01	**2.2635e−01**	2.7938e−01	3.2759e−01	2.7932e−01	2.8960e−01	3.3745e−01
	Std	1.9009e−01	1.6644e−01	1.7515e−01	1.6729e−01	1.6644e−01	2.0382e−01	1.3488e−01	1.4596e−01	2.2108e−01	2.6484e−01	1.7839e−01	2.0205e−01	2.9829e−01
	Rank	4	6	7	3	5	10	2	**1**	9	12	8	11	13
F11	Mean	2.7063e+01	2.6366e+01	2.5869e+01	2.3017e+01	2.6565e+01	2.4410e+01	2.4575e+01	**2.1657e+01**	2.4642e+01	2.4609e+01	2.3282e+01	2.5869e+01	2.5272e+01
	Std	8.1395e+00	7.4444e+00	7.6580e+00	5.6931e+00	9.7175e+00	6.7670e+00	5.8083e+00	7.1446e+00	8.5217e+00	8.0616e+00	7.0963e+00	8.7911e+00	1.0352e+01
	Rank	13	11	10	2	12	4	5	**1**	7	6	3	9	8
F12	Mean	2.7627e+01	3.0313e+01	3.2203e+01	**2.5405e+01**	2.8754e+01	3.1374e+01	2.8986e+01	2.7991e+01	3.0976e+01	2.8787e+01	2.9285e+01	3.1573e+01	2.9550e+01
	Std	7.2959e+00	8.2946e+00	6.5764e+00	7.3799e+00	8.9082e+00	1.0219e+01	1.1454e+01	8.8866e+00	8.3887e+00	8.6539e+00	8.1532e+00	1.1054e+01	1.1662e+01
	Rank	2	9	13	**1**	4	11	6	3	10	5	7	12	8
F13	Mean	8.8302e+01	8.8597e+01	**8.0622e+01**	8.4987e+01	8.5043e+01	9.8113e+01	9.3934e+01	8.8818e+01	9.0572e+01	9.2383e+01	9.2274e+01	8.6604e+01	8.6962e+01
	Std	2.8342e+01	2.8217e+01	2.3879e+01	2.7916e+01	2.8744e+01	2.7878e+01	2.7881e+01	2.5919e+01	3.1236e+01	2.6656e+01	2.5164e+01	2.4651e+01	3.5488e+01
	Rank	6	7	**1**	2	3	13	12	8	9	11	10	4	5
F14	Mean	1.5398e+03	1.6046e+03	1.7628e+03	1.6141e+03	1.6175e+03	1.4998e+03	1.6404e+03	1.7217e+03	1.7604e+03	1.5808e+03	1.6022e+03	1.5649e+03	**1.4803e+03**
	Std	3.8988e+02	5.3852e+02	3.6734e+02	4.3002e+02	4.1346e+02	3.8093e+02	4.9752e+02	5.0506e+02	4.7418e+02	4.7064e+02	5.5461e+02	3.7461e+02	4.2482e+02
	Rank	3	7	13	8	9	2	10	11	12	5	6	4	**1**
F15	Mean	2.5094e+03	2.6115e+03	2.5710e+03	2.6414e+03	**2.3969e+03**	2.4762e+03	2.5627e+03	2.6242e+03	2.5281e+03	2.6428e+03	2.4915e+03	2.6302e+03	2.7414e+03
	Std	5.4591e+02	5.5483e+02	6.0991e+02	4.5522e+02	5.4524e+02	5.6538e+02	5.2859e+02	6.5700e+02	6.1456e+02	5.4869e+02	5.9822e+02	5.8984e+02	5.8309e+02
	Rank	4	8	7	11	**1**	2	6	9	5	12	3	10	13
F16	Mean	2.9644e−01	3.2210e−01	3.2167e−01	2.8956e−01	3.2094e−01	3.0232e−01	3.1525e−01	3.0368e−01	3.0626e−01	3.6470e−01	2.8674e−01	3.2569e−01	**2.7966e−01**
	Std	9.8044e−02	1.8192e−01	1.7596e−01	1.2842e−01	1.3236e−01	1.3928e−01	1.2427e−01	1.6701e−01	1.9707e−01	2.3996e−01	1.3747e−01	1.6760e−01	1.0309e−01
	Rank	4	11	10	3	9	5	8	6	7	13	2	12	**1**
F17	Mean	4.7958e+01	4.6106e+01	4.6482e+01	4.6573e+01	4.5281e+01	4.7207e+01	4.5846e+01	**4.5193e+01**	4.5500e+01	4.7018e+01	4.6007e+01	4.6601e+01	4.6857e+01
	Std	6.4211e+00	4.4566e+00	4.7575e+00	4.4632e+00	4.1415e+00	5.6637e+00	4.4002e+00	5.3396e+00	3.9363e+00	5.8715e+00	5.6731e+00	4.8217e+00	6.8618e+00
	Rank	13	6	7	8	2	12	4	**1**	3	11	5	9	10
F18	Mean	4.7285e+01	**4.6990e+01**	4.7368e+01	4.9022e+01	4.8542e+01	4.8815e+01	4.8631e+01	4.8207e+01	4.8492e+01	4.9492e+01	4.8309e+01	4.7254e+01	4.7920e+01
	Std	4.8509e+00	5.1966e+00	7.0561e+00	4.3680e+00	4.8463e+00	6.2036e+00	6.5813e+00	5.2885e+00	5.1669e+00	6.4385e+00	5.2449e+00	4.4333e+00	5.9987e+00
	Rank	3	**1**	4	12	9	11	10	6	8	13	7	2	5
F19	Mean	3.4635e+00	3.4895e+00	3.2450e+00	3.5419e+00	3.5574e+00	3.4257e+00	3.8825e+00	3.6398e+00	3.5575e+00	3.6418e+00	3.7746e+00	**3.1791e+00**	3.5751e+00
	Std	1.0793e+00	8.0588e−01	7.9661e−01	9.3665e−01	7.5840e−01	9.5712e−01	1.2983e+00	8.7576e−01	1.2637e+00	8.1161e−01	1.2840e+00	8.0839e−01	9.3888e−01
	Rank	4	5	2	6	7	3	13	10	8	11	12	**1**	9
F20	Mean	1.1889e+01	1.1860e+01	1.1682e+01	1.1820e+01	1.1724e+01	1.1910e+01	**1.1603e+01**	1.2032e+01	1.1976e+01	1.2167e+01	1.1877e+01	1.1831e+01	1.2052e+01
	Std	9.1400e−01	8.8111e−01	9.6913e−01	8.3323e−01	9.3160e−01	9.0543e−01	8.7801e−01	9.0859e−01	7.6105e−01	7.1775e−01	7.7692e−01	9.1649e−01	7.2619e−01
	Rank	8	6	2	4	3	9	**1**	11	10	13	7	5	12
F21	Mean	3.0914e+02	3.0392e+02	3.2306e+02	3.2118e+02	3.0538e+02	3.1161e+02	3.1016e+02	2.9957e+02	3.1495e+02	3.2451e+02	**2.9102e+02**	3.3930e+02	2.9624e+02
	Std	6.1564e+01	7.4871e+01	8.8957e+01	9.6952e+01	8.1529e+01	9.1122e+01	8.9276e+01	4.9551e+01	8.8695e+01	9.4325e+01	6.6798e+01	8.7162e+01	5.2780e+01
	Rank	6	4	11	10	5	8	7	3	9	12	**1**	13	2
F22	Mean	1.5186e+03	1.6148e+03	1.5065e+03	1.3981e+03	1.5010e+03	1.4621e+03	1.7810e+03	1.3848e+03	1.4317e+03	1.6618e+03	1.4094e+03	1.6172e+03	**1.3602e+03**
	Std	4.1748e+02	4.3787e+02	5.3573e+02	5.5649e+02	5.1234e+02	5.9000e+02	9.4103e+02	4.8983e+02	5.1782e+02	5.7266e+02	4.0106e+02	5.2037e+02	3.5156e+02
	Rank	9	10	8	3	7	6	13	2	5	12	4	11	**1**
F23	Mean	4.0651e+03	4.2820e+03	4.4281e+03	**3.8179e+03**	4.3751e+03	4.8238e+03	4.5388e+03	4.3694e+03	4.2144e+03	4.1270e+03	4.3899e+03	4.1961e+03	4.4024e+03
	Std	9.0406e+02	9.9924e+02	1.0374e+03	9.1084e+02	9.9514e+02	8.8132e+02	1.0764e+03	9.3508e+02	9.3370e+02	9.1399e+02	9.0794e+02	1.0644e+03	9.5428e+02
	Rank	2	6	11	**1**	8	13	12	7	5	3	9	4	10
F24	Mean	2.4553e+02	**2.4483e+02**	2.4837e+02	2.4788e+02	2.4659e+02	2.4609e+02	2.4891e+02	2.4651e+02	2.4634e+02	2.4817e+02	2.4506e+02	2.4741e+02	2.4539e+02
	Std	1.1478e+01	1.1872e+01	9.7562e+00	1.1719e+01	1.1301e+01	1.0802e+01	1.1068e+01	1.1324e+01	1.0409e+01	1.1266e+01	1.0073e+01	1.1639e+01	1.0984e+01
	Rank	4	**1**	12	10	8	5	13	7	6	11	2	9	3
F25	Mean	2.7237e+02	2.7262e+02	2.7215e+02	2.7189e+02	2.7283e+02	**2.7001e+02**	2.7218e+02	2.7157e+02	2.7209e+02	2.7199e+02	2.7531e+02	2.7308e+02	2.7213e+02
	Std	7.2135e+00	7.0704e+00	6.2479e+00	8.3710e+00	7.5678e+00	7.4695e+00	5.8932e+00	8.4358e+00	7.1091e+00	8.3509e+00	8.6157e+00	7.9238e+00	7.7262e+00
	Rank	9	10	7	3	11	**1**	8	2	5	4	13	12	6
F26	Mean	2.7762e+02	2.8746e+02	2.5918e+02	2.7983e+02	2.6461e+02	**2.4339e+02**	2.5441e+02	2.7765e+02	2.9453e+02	2.6393e+02	2.8442e+02	2.5443e+02	2.8226e+02
	Std	6.4822e+01	6.3292e+01	6.4363e+01	6.2930e+01	6.5995e+01	6.2348e+01	6.3962e+01	5.8803e+01	5.9855e+01	6.5181e+01	6.1556e+01	6.3652e+01	6.4113e+01
	Rank	7	12	4	9	6	**1**	2	8	13	5	11	3	10
F27	Mean	6.7786e+02	6.8794e+02	6.7412e+02	6.7388e+02	6.6539e+02	7.0103e+02	6.6532e+02	6.8232e+02	**6.5018e+02**	6.5315e+02	6.7799e+02	6.6071e+02	6.8837e+02
	Std	8.7626e+01	7.9871e+01	9.2001e+01	9.5802e+01	1.0755e+02	1.0806e+02	8.4838e+01	1.0643e+02	1.0591e+02	7.9127e+01	8.8231e+01	1.0600e+02	9.5368e+01
	Rank	8	11	7	6	5	13	4	10	**1**	2	9	3	12
F28	Mean	4.8524e+02	**2.9333e+02**	3.0000e+02	3.7656e+02	3.5506e+02	3.0000e+02	**2.9333e+02**	3.0000e+02	3.0000e+02	3.0000e+02	3.1584e+02	2.9414e+02	3.4208e+02
	Std	4.9690e+02	3.6515e+01	0.0000e+00	4.1936e+02	2.3022e+02	0.0000e+00	3.6515e+01	0.0000e+00	0.0000e+00	0.0000e+00	1.2975e+02	6.6743e+01	2.3047e+02
	Rank	13	**1**	4	12	11	4	**1**	4	4	4	9	3	10
Mean rank		6.04	6.75	6.75	5.89	5.93	6.57	6.71	5.43	7.07	7.25	6.46	7.14	7.04
Final rank		4	8	9	2	3	6	7	1	11	13	5	12	10

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This study was supported by the Ministry of Education of Humanities and Social Science Project, China (No. 19YJAZH047), the National Natural Science Foundation of China (Grant no. 61771087) and the Financial Intelligence and Financial Engineering Key Laboratory of Sichuan Province, China. The authors thank Prof. Suganthan (http://www.ntu.edu.sg/home/epnsugan/) for providing source codes of some compared PSO variants.

Appendix A

See Tables A.1–A.4.

Appendix B

See Fig. B.1.

Table A.1
Results of CEC2017 with $d = 30$.

Function	Metrics	PSO	UPSO	FDRPSO	WFIPS	CPSO	CLPSO	MLPSO	MPSO	CCPSO-ISM	PPSO
F1	Mean	4.4804e+03	1.6307e+03	6.3636e+03	1.1335e+04	4.5178e+03	**4.6020e+01**	7.4884e+03	9.1634e+07	8.3633e+04	2.4415e+03
	Std	4.9933e+03	2.4804e+03	9.4981e+03	9.9991e+03	4.9551e+03	1.0284e+02	1.0911e+04	3.7839e+08	2.8053e+04	2.5473e+03
	Rank	4	2	6	8	5	**1**	7	10	9	3
F2	Mean	2.7766e+15	1.6491e+05	1.0547e+07	4.1939e+19	8.2312e+14	1.1624e+13	5.6623e+05	3.0804e+17	2.1896e+17	**6.3148e+04**
	Std	1.1701e+16	6.3870e+05	5.7168e+07	1.3175e+20	2.6187e+15	2.6529e+13	3.0934e+06	1.5921e+18	5.9973e+17	2.2678e+05
	Rank	7	2	4	10	6	5	3	9	8	**1**
F3	Mean	2.1509e+03	9.2936e+03	1.6541e+03	2.8033e+04	1.2901e+05	4.0824e+04	**1.1764e−04**	2.4942e+00	5.8807e+04	1.6450e+04
	Std	9.5357e+02	3.7701e+03	2.0414e+03	6.3729e+03	4.3585e+04	1.1321e+04	6.4431e−04	4.8327e+00	9.7738e+03	5.1384e+03
	Rank	4	5	3	7	10	8	**1**	2	9	6
F4	Mean	1.3038e+02	2.7568e+01	4.5310e+01	2.6331e+01	8.1668e+01	4.1598e+01	**4.3594e+00**	7.7306e+01	8.6795e+01	6.9524e+01
	Std	2.2678e+01	1.8523e+01	2.7694e+01	7.8365e−01	2.1472e+01	2.2041e+01	2.1047e+00	3.0202e+01	1.7079e+01	2.7712e+01
	Rank	10	3	5	2	8	4	**1**	7	9	6
F5	Mean	6.5601e+01	8.4448e+01	6.7856e+01	1.5699e+02	1.1482e+02	5.3469e+01	6.7425e+01	6.1038e+01	1.3056e+02	**2.2121e+01**
	Std	1.5258e+01	1.7171e+01	1.9352e+01	9.3456e+00	2.9976e+01	8.9110e+00	1.8562e+01	1.5959e+01	1.9290e+01	5.8524e+00
	Rank	4	7	6	10	8	2	5	3	9	**1**
F6	Mean	8.4315e−02	1.7160e+00	5.4211e−01	1.1334e−04	9.1975e−05	**4.6207e−06**	8.6636e−02	2.7920e−02	5.1945e−01	7.3544e−02
	Std	2.6714e−01	2.0814e+00	6.3898e−01	2.8730e−05	4.0153e−05	2.6254e−06	1.3558e−01	5.9664e−02	4.9257e−01	1.3618e−01
	Rank	6	10	9	3	2	**1**	7	4	8	5
F7	Mean	1.0732e+02	1.1636e+02	9.8635e+01	1.9644e+02	1.3248e+02	8.1135e+01	1.0137e+02	9.4343e+01	1.0511e+02	**4.7098e+01**
	Std	2.5273e+01	1.7236e+01	1.6658e+01	1.1208e+01	2.4621e+01	7.6409e+00	2.3331e+01	1.7172e+01	1.1442e+01	5.7677e+00
	Rank	7	8	4	10	9	2	5	3	6	**1**
F8	Mean	6.9747e+01	8.3071e+01	6.2019e+01	1.5571e+02	1.3614e+02	6.1100e+01	7.5318e+01	5.8252e+01	1.3353e+02	**2.0463e+01**
	Std	1.5547e+01	1.7261e+01	1.5092e+01	1.2527e+01	3.6044e+01	8.3236e+00	2.1697e+01	1.6282e+01	1.8373e+01	5.3592e+00
	Rank	5	7	4	10	9	3	6	2	8	**1**
F9	Mean	1.6519e+02	3.9315e+02	4.5127e+01	**5.6808e−07**	2.0785e+03	2.5876e+02	5.1987e+01	9.0703e+01	5.2676e+03	5.1421e+00
	Std	3.7928e+02	3.6879e+02	4.0572e+01	4.4887e−07	1.4324e+03	1.4942e+02	7.7232e+01	8.9563e+01	1.2581e+03	4.8107e+00
	Rank	6	8	3	**1**	9	7	4	5	10	2
F10	Mean	2.9921e+03	3.2138e+03	3.3613e+03	6.7143e+03	3.1527e+03	**2.0477e+03**	3.9382e+03	2.9921e+03	2.7825e+03	2.1833e+03
	Std	9.0004e+02	4.0781e+02	5.7082e+02	2.6993e+02	5.8225e+02	4.2572e+02	7.6866e+02	6.5112e+02	3.2599e+02	5.2601e+02
	Rank	4	7	8	10	6	**1**	9	5	3	2
F11	Mean	9.0889e+01	5.3599e+01	1.1145e+02	5.5051e+01	1.1407e+02	1.9155e+02	**3.1030e+01**	9.3487e+01	2.0314e+02	9.7482e+01
	Std	4.1803e+01	2.1504e+01	3.7344e+01	1.0087e+01	5.2829e+01	1.2172e+02	1.3023e+01	3.6153e+01	5.8566e+01	4.4176e+01
	Rank	4	2	7	3	8	9	**1**	5	10	6
F12	Mean	6.0606e+05	2.1465e+05	1.2231e+05	1.3881e+06	4.3260e+06	5.4018e+05	**7.6154e+04**	1.0242e+06	7.9541e+05	1.9661e+05
	Std	1.1253e+06	2.3993e+05	1.0139e+05	6.7383e+05	5.4820e+06	3.6242e+05	6.2158e+04	1.1374e+06	2.9030e+05	1.5176e+05
	Rank	6	4	2	9	10	5	**1**	8	7	3
F13	Mean	1.3879e+04	1.5038e+04	2.0690e+04	1.9865e+04	1.6511e+05	**2.9615e+02**	1.8642e+04	1.0324e+04	4.5379e+03	1.3433e+04
	Std	1.4089e+04	1.9071e+04	2.5570e+04	1.9226e+04	8.0756e+05	2.3750e+02	1.9755e+04	9.9501e+03	2.5548e+03	1.2047e+04
	Rank	5	6	9	8	10	**1**	7	3	2	4
F14	Mean	1.5559e+04	9.4379e+03	2.2927e+04	8.1435e+03	6.8990e+05	8.8714e+04	6.9460e+03	**3.4477e+02**	3.6678e+04	2.4385e+04
	Std	1.3395e+04	1.0249e+04	2.5519e+04	6.0674e+03	6.3473e+05	1.0218e+05	5.6012e+03	1.5077e+02	2.0896e+04	1.7354e+04
	Rank	5	4	6	3	10	9	2	**1**	8	7
F15	Mean	1.1872e+04	2.3116e+03	7.3091e+03	5.6193e+03	1.5040e+04	**9.8982e+01**	1.2283e+04	1.1525e+03	3.7611e+02	1.7639e+03
	Std	9.0625e+03	2.2121e+03	1.2491e+04	4.6798e+03	1.3965e+04	1.0648e+02	1.7949e+04	1.3731e+03	1.4226e+02	2.4549e+03
	Rank	8	5	7	6	10	**1**	9	3	2	4
F16	Mean	6.4211e+02	6.9837e+02	8.8195e+02	1.1015e+03	1.1993e+03	7.3868e+02	6.6499e+02	8.2167e+02	6.4997e+02	**6.3836e+02**
	Std	1.7043e+02	1.9361e+02	2.3546e+02	1.7638e+02	2.8797e+02	1.6363e+02	2.9242e+02	2.6148e+02	1.6707e+02	2.2442e+02
	Rank	2	5	8	9	10	6	4	7	3	**1**
F17	Mean	2.2925e+02	3.0246e+02	2.8881e+02	**1.7590e+02**	7.2317e+02	2.1232e+02	2.8202e+02	2.6566e+02	2.0826e+02	2.3816e+02
	Std	1.5845e+02	1.0639e+02	1.3089e+02	5.3864e+01	2.2615e+02	9.8178e+01	1.3801e+02	1.5176e+02	8.3623e+01	1.4329e+02
	Rank	4	9	8	**1**	10	3	7	6	2	5
F18	Mean	3.2325e+05	1.9178e+05	1.2764e+05	5.3361e+05	2.3106e+06	1.7956e+05	1.3624e+05	**3.6986e+04**	1.3193e+05	1.7901e+05
	Std	2.3004e+05	1.1905e+05	8.1763e+04	2.9152e+05	2.6747e+06	1.1184e+05	1.8836e+05	2.9410e+04	4.4230e+04	1.4477e+05
	Rank	8	7	2	9	10	6	4	**1**	3	5
F19	Mean	2.2236e+04	1.9028e+03	7.4407e+03	3.5575e+03	1.3484e+04	**7.2325e+01**	1.3149e+04	5.4199e+03	1.1466e+02	5.4609e+03
	Std	1.7461e+04	2.1119e+03	1.1625e+04	2.9284e+03	1.2787e+04	6.4755e+01	1.3847e+04	5.2374e+03	4.5014e+01	5.0231e+03
	Rank	10	3	7	4	9	**1**	8	5	2	6
F20	Mean	**1.6480e+02**	3.5363e+02	3.0948e+02	1.9534e+02	6.0894e+02	2.7379e+02	2.9899e+02	2.7049e+02	3.2942e+02	2.1081e+02
	Std	8.7134e+01	1.0432e+02	1.3882e+02	7.4489e+01	2.0766e+02	1.1409e+02	1.5121e+02	1.2189e+02	9.2832e+01	9.5067e+01
	Rank	**1**	9	7	2	10	5	6	4	8	3
F21	Mean	2.6892e+02	2.8451e+02	2.6379e+02	3.5647e+02	3.4309e+02	2.5759e+02	2.6350e+02	2.6397e+02	2.9776e+02	**2.2508e+02**
	Std	1.8998e+01	1.7349e+01	1.9591e+01	1.1789e+01	3.8436e+01	2.4032e+01	1.6579e+01	2.1225e+01	8.6019e+01	5.8490e+00
	Rank	6	7	4	10	9	2	3	5	8	**1**
F22	Mean	2.2746e+03	1.4217e+03	1.7410e+03	1.2415e+03	3.4715e+03	1.0706e+03	3.3933e+03	1.2523e+02	4.9819e+02	**1.0000e+02**
	Std	1.5772e+03	1.6710e+03	1.8335e+03	2.5620e+03	1.0463e+03	1.0771e+03	1.5727e+03	3.7505e+01	1.0338e+03	1.7244e−13
	Rank	8	6	7	5	10	4	9	2	3	**1**
F23	Mean	3.9990e+02	4.5982e+02	4.4687e+02	5.1565e+02	4.7597e+02	4.2198e+02	4.4105e+02	4.3516e+02	4.4486e+02	**3.8974e+02**
	Std	1.7086e+01	2.2002e+01	2.0127e+01	1.6062e+01	2.6362e+01	1.7579e+01	2.6285e+01	3.5193e+01	4.4784e+01	1.2790e+01
	Rank	2	8	7	10	9	3	5	4	6	**1**
F24	Mean	4.9733e+02	5.2838e+02	5.2859e+02	6.0482e+02	6.8734e+02	6.0356e+02	5.0303e+02	5.0011e+02	4.5312e+02	**4.5191e+02**
	Std	4.7435e+01	2.0874e+01	2.2411e+01	9.0216e+00	7.9136e+01	4.9364e+01	2.1640e+01	2.9930e+01	2.0113e+02	1.0492e+01
	Rank	3	6	7	9	10	8	5	4	2	**1**
F25	Mean	4.0849e+02	**3.7689e+02**	3.7938e+02	3.7850e+02	3.9640e+02	3.7888e+02	3.7824e+02	4.5248e+02	3.9497e+02	3.9319e+02
	Std	2.6048e+01	1.8033e+00	1.0091e+00	2.5646e−01	1.9183e+01	7.4290e−01	1.5193e+00	3.4198e+01	2.8401e+00	1.2941e+01
	Rank	9	**1**	5	3	8	4	2	10	7	6
F26	Mean	1.7394e+03	1.8789e+03	1.7090e+03	2.4558e+03	2.7619e+03	1.2629e+03	1.6648e+03	1.6117e+03	**6.0921e+02**	1.0966e+03
	Std	2.3247e+02	2.9722e+02	6.8933e+02	5.4282e+02	4.1363e+02	7.8133e+02	4.3306e+02	1.0988e+03	3.1580e+02	4.7310e+02
	Rank	7	8	6	9	10	3	5	4	**1**	2
F27	Mean	5.3031e+02	5.0001e+02	4.6746e+02	5.0001e+02	5.4743e+02	**4.5557e+02**	5.0001e+02	5.4085e+02	5.2061e+02	5.3008e+02
	Std	1.4198e+01	1.2535e−04	1.8516e+01	9.7843e−05	1.6093e+01	5.1902e+00	2.2264e−04	1.6278e+01	5.3025e+00	1.1579e+01
	Rank	8	4	2	5	10	**1**	3	9	6	7
F28	Mean	4.3435e+02	4.9851e+02	4.0697e+02	4.9872e+02	5.0293e+02	4.4350e+02	4.9888e+02	5.3226e+02	4.7259e+02	**3.9578e+02**
	Std	4.5019e+01	3.8783e+00	4.9557e+01	5.3529e+00	1.5658e+02	2.0437e+01	3.4447e+00	4.4129e+01	1.2536e+01	3.8656e+01
	Rank	3	6	2	7	9	4	8	10	5	**1**
F29	Mean	6.2339e+02	6.8103e+02	6.8843e+02	7.5144e+02	1.0210e+03	**5.3929e+02**	5.9606e+02	6.8543e+02	6.8861e+02	6.3025e+02
	Std	1.0887e+02	1.6646e+02	2.0167e+02	1.3298e+02	1.9224e+02	1.1211e+02	1.6089e+02	1.4998e+02	9.2687e+01	1.2510e+02
	Rank	3	5	7	9	10	**1**	2	6	8	4
F30	Mean	1.5100e+04	1.6428e+03	3.1944e+03	2.6151e+03	1.2693e+04	**4.1583e+02**	2.7936e+03	3.7323e+03	2.1831e+04	4.5147e+03
	Std	1.9162e+04	1.4603e+03	2.6025e+03	2.5264e+03	1.1851e+04	1.9396e+02	2.4368e+03	1.2781e+03	8.9238e+03	1.4346e+03
	Rank	9	2	5	3	8	**1**	4	6	10	7
Mean rank		5.60	5.53	5.57	6.50	8.73	3.70	4.77	5.10	6.07	3.43
Final rank		7	5	6	9	10	2	3	4	8	1

Table A.2
Nine benchmark functions for comparison.

Group	Name	Test function	D	Search range	F_{min}
Unimodal	Schwefel's P2.22	$F1(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	30	$[-10, 10]^D$	0
	Quadric	$F2(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	30	$[-100, 100]^D$	0
	Rosenbrock	$F3(x) = \sum_{i=1}^{D-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right]$	30	$[-10, 10]^D$	0
Multimodal	Ackley	$F4(x) = -20 \exp\left(-0.2 \sqrt{1/D \sum_{i=1}^{D} x_i^2} \right) - \exp\left(1/D \sum_{i=1}^{D} \cos 2\pi x_i \right) + 20 + e$	30	$[-32, 32]^D$	0
	Griewank	$F5(x) = 1/4000 \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(x_i / \sqrt{i} \right) + 1$	30	$[-600, 600]^D$	0
	Noncontinuous Rastrigin	$F6(x) = \sum_{i=1}^{D} \left[y_i^2 - 10 \cos(2\pi y_i) + 10 \right]$	30	$[-5.12, 5.12]^D$	0
Shifted and rotated	Shifted Schwefel's P1.2	$F7(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} z_j \right)^2 - 450, z = x - o$	30	$[-100, 100]^D$	−450
	Shifted rotated Rastrigin	$F8(x) = \sum_{i=1}^{D} \left[z_i^2 - 10 \cos(2\pi z_i) + 10 \right] - 330, z = (x - o) \times M$	30	$[-5, 5]^D$	−330
	Shifted rotated Weierstrass	$F9(x) = \sum_{i=1}^{D} \left(\sum_{k=0}^{kmax} \left[a^k \cos\left(2\pi b^k (z_i + 0.5) \right) \right] \right) - n \sum_{k=0}^{kmax} \left[a^k \cos\left(2\pi b^k \cdot 0.5 \right) \right] + 90, a = 0.5, b = 3, kmax = 20, z = (x - o) \times M$	30	$[-0.5, 0.5]^D$	90

Table A.3
Results of nine benchmark functions with $d = 30$.

Function	Metrics	PSO	UPSO	FDRPSO	WFIPS	CPSO	CLPSO	MLPSO	MPSO	CCPSO-ISM	PPSO
F1	Mean	1.9081e−22	4.8763e−43	1.3537e−21	2.9360e−07	1.2371e−04	1.0199e−08	3.9494e−105	**4.7638e−173**	3.4297e−22	2.1051e−104
	Std	3.8408e−22	3.8447e−43	7.0890e−21	6.3761e−08	4.3266e−05	5.6003e−09	1.0252e−104	0.0000e+00	1.8374e−21	1.0983e−103
	Rank	5	4	7	9	10	8	2	**1**	6	3
F2	Mean	1.8317e−31	2.3348e−71	1.7325e−51	5.5027e−10	4.0368e−06	7.7704e−13	3.4860e−193	**2.4430e−319**	3.4665e−50	2.4247e−212
	Std	9.3604e−31	2.4878e−71	5.0068e−51	2.1380e−10	3.1285e−06	6.6748e−13	0.0000e+00	0.0000e+00	1.7131e−49	0.0000e+00
	Rank	7	4	5	9	10	8	3	**1**	6	2
F3	Mean	3.5513e+01	1.4517e+01	2.4880e+01	2.4846e+01	3.7660e+00	**3.5770e−01**	1.3292e+00	2.0032e+01	1.9812e+01	2.2098e+01
	Std	2.7630e+01	7.4381e+00	2.2199e+01	4.5717e−01	1.2762e+01	3.0783e−01	1.9113e+00	8.8360e−01	2.1749e+01	1.4730e+00
	Rank	10	4	9	8	3	**1**	2	6	5	7
F4	Mean	1.0599e−14	7.9936e−15	1.2375e−14	1.3811e−06	1.3994e−04	6.5216e−08	7.4015e−15	4.1128e−12	1.1783e−14	**4.4409e−15**
	Std	3.3553e−15	0.0000e+00	3.9234e−15	2.9607e−07	8.4340e−05	2.4832e−08	1.3467e−15	2.2508e−11	4.4680e−15	1.6047e−30
	Rank	4	3	6	9	10	8	2	7	5	**1**
F5	Mean	1.1067e−02	2.4653e−04	1.4191e−02	1.4616e−06	2.5684e−02	1.6046e−11	5.1722e−03	**0.0000e+00**	2.0789e−02	8.2165e−04
	Std	1.5125e−02	1.3503e−03	1.3427e−02	3.2045e−06	3.3289e−02	1.2111e−11	6.4749e−03	0.0000e+00	2.3313e−02	2.5347e−03
	Rank	7	4	8	3	10	2	6	**1**	9	5
F6	Mean	1.8787e+01	5.8769e+01	1.9433e+01	6.5767e+01	8.0391e−08	**5.0804e−14**	2.5367e+01	3.0172e+00	1.9200e+01	1.4400e+01
	Std	8.8071e+00	1.4748e+01	7.5324e+00	7.7010e+00	8.3214e−08	3.7736e−14	6.7235e+00	3.9445e+00	6.7333e+00	4.3518e+00
	Rank	5	9	7	10	2	**1**	8	3	6	4
F7	Mean	4.4695e+00	2.1995e−02	5.5349e−02	4.3912e+02	1.1838e+03	3.9844e+03	**4.6801e−13**	2.0346e+00	4.7642e−02	1.0057e+01
	Std	2.7734e+00	1.2412e−02	2.5048e−01	1.2278e+02	3.3901e+03	6.9422e+02	1.6056e−13	2.1478e+00	1.0663e−01	1.6162e+01
	Rank	6	2	4	8	9	10	**1**	5	3	7
F8	Mean	9.8191e+01	9.0056e+01	6.7757e+01	1.7892e+02	3.9663e+02	1.6687e+02	7.2731e+01	8.5086e+01	6.4280e+01	**3.2203e+01**
	Std	5.3744e+01	1.8119e+01	1.7267e+01	1.2644e+01	1.0720e+02	3.6042e+01	2.0198e+01	2.0565e+01	1.7642e+01	9.2582e+00
	Rank	7	6	3	9	10	8	4	5	2	**1**
F9	Mean	2.3968e+01	2.9955e+01	2.0217e+01	3.8585e+01	3.5920e+01	2.7267e+01	3.1995e+01	2.2248e+01	1.9619e+01	**1.3640e+01**
	Std	3.7728e+00	1.7511e+00	3.8299e+00	1.2161e+00	3.8743e+00	2.2882e+00	3.7497e+00	3.5783e+00	4.0199e+00	2.2421e+00
	Rank	5	7	3	10	9	6	8	4	2	**1**
Mean rank		6.22	4.78	5.78	8.33	8.11	5.78	4.00	3.67	4.89	3.44
Final rank		8	4	6	10	9	7	3	2	5	1

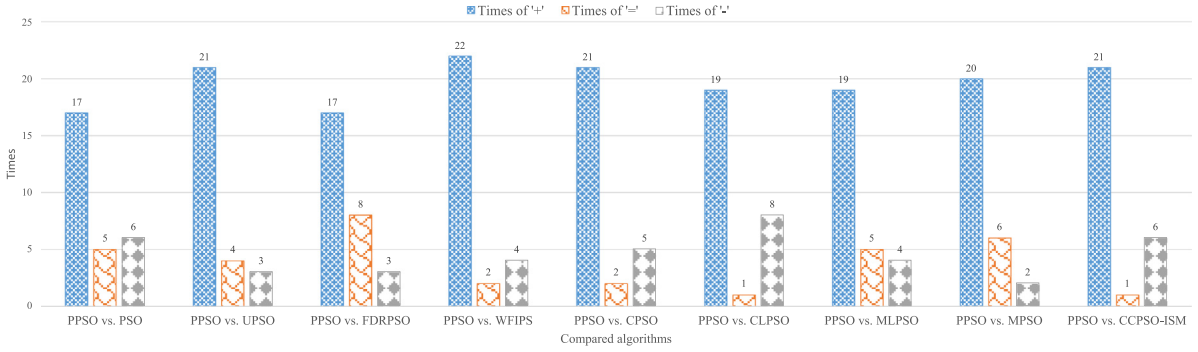

Fig. B.1. Times of winner by Wilcoxon signed-rank test.

Table A.4
Results of CEC2017 with $d = 50$.

Function	Metrics	PSO	UPSO	FDRPSO	WFIPS	CPSO	CLPSO	MLPSO	MPSO	CCPSO-ISM	PPSO
F1	Mean	1.0020e+04	5.0656e+03	9.8870e+03	1.5294e+05	7.1182e+03	**1.4028e+02**	8.5196e+03	1.7616e+09	5.7239e+04	3.1753e+03
	Std	9.1441e+03	7.5073e+03	2.0668e+04	1.9552e+05	8.4740e+03	1.5561e+02	1.0644e+04	1.6624e+09	2.8993e+04	3.2960e+03
	Rank	7	3	6	9	4	**1**	5	10	8	2
F2	Mean	2.9422e+41	4.9300e+14	**1.0938e+14**	5.3647e+45	7.9941e+35	1.4741e+27	8.3208e+16	1.9432e+39	6.3488e+33	1.9386e+14
	Std	1.5909e+42	2.4423e+15	5.5529e+14	2.0086e+46	2.7610e+36	6.6604e+27	4.3773e+17	1.0620e+40	1.6564e+34	6.2393e+14
	Rank	9	3	**1**	10	7	5	4	8	6	2
F3	Mean	2.0919e+04	7.0238e+04	4.2773e+03	9.0488e+04	2.6563e+05	5.7466e+04	5.8239e+02	**3.8580e+02**	1.4061e+05	6.3220e+04
	Std	5.1275e+03	1.3608e+04	3.2710e+03	1.2110e+04	6.6808e+04	1.3154e+04	8.6999e+02	4.7639e+02	1.5470e+04	1.4762e+04
	Rank	4	7	3	8	10	5	2	**1**	9	6
F4	Mean	2.6480e+02	4.1191e+01	4.7480e+01	4.6619e+01	1.2434e+02	7.6542e+01	**2.7118e+01**	5.0885e+02	1.1036e+02	1.0881e+02
	Std	5.0444e+01	2.1680e+01	2.1442e+01	5.6381e−01	4.8546e+01	2.4727e+01	2.1553e+01	2.2393e+02	2.3891e+01	4.6667e+01
	Rank	9	2	4	3	8	5	**1**	10	7	6
F5	Mean	1.3710e+02	2.0497e+02	1.2503e+02	3.4766e+02	2.3733e+02	1.2785e+02	1.5239e+02	1.5509e+02	2.9654e+02	**5.8043e+01**
	Std	2.8914e+01	2.9017e+01	2.8598e+01	1.4219e+01	5.9572e+01	1.5881e+01	4.0054e+01	3.3652e+01	2.7541e+01	1.0321e+01
	Rank	4	7	2	10	8	3	5	6	9	**1**
F6	Mean	9.0175e−01	2.2054e+01	1.3357e+00	2.0316e−03	1.1902e−04	**8.0599e−07**	1.3383e+00	2.7016e+00	1.4339e+00	5.7503e−01
	Std	8.2882e−01	6.4836e+00	1.2862e+00	4.4933e−04	2.3803e−04	4.8252e−07	1.8189e+00	2.1927e+00	1.1730e+00	3.9284e−01
	Rank	5	10	6	3	2	**1**	7	9	8	4
F7	Mean	2.7384e+02	2.6137e+02	2.0500e+02	4.1662e+02	2.4096e+02	1.7766e+02	2.2616e+02	2.2811e+02	2.2625e+02	**1.0210e+02**
	Std	5.1197e+01	3.5041e+01	3.6161e+01	1.6866e+01	3.6283e+01	1.2207e+01	4.3751e+01	4.0877e+01	3.0879e+01	9.6237e+00
	Rank	9	8	3	10	7	2	4	6	5	**1**
F8	Mean	1.3508e+02	1.9618e+02	1.1963e+02	3.4481e+02	2.4426e+02	1.3169e+02	1.5674e+02	1.3724e+02	2.7772e+02	**6.0095e+01**
	Std	2.3300e+01	2.8872e+01	2.1260e+01	1.9776e+01	4.7477e+01	1.4520e+01	3.8315e+01	3.1166e+01	3.2209e+01	1.1973e+01
	Rank	4	7	2	10	8	3	6	5	9	**1**
F9	Mean	5.4510e+02	3.7955e+03	2.6904e+02	**2.4574e−02**	5.7905e+03	2.0398e+03	1.1410e+03	8.9897e+02	1.9814e+04	2.4879e+02
	Std	2.9072e+02	9.4617e+02	1.9231e+02	2.6753e−02	2.8216e+03	7.4471e+02	1.1798e+03	6.4356e+02	3.4808e+03	2.0521e+02
	Rank	4	8	3	**1**	9	7	6	5	10	2
F10	Mean	5.8598e+03	6.2757e+03	5.7083e+03	1.3073e+04	5.1382e+03	4.1121e+03	6.9525e+03	5.7506e+03	5.2533e+03	**3.8783e+03**
	Std	1.0412e+03	4.0519e+02	8.2820e+02	4.2281e+02	8.3167e+02	5.4798e+02	9.5464e+02	8.7832e+02	4.9921e+02	6.7550e+02
	Rank	7	8	5	10	3	2	9	6	4	**1**
F11	Mean	2.2410e+02	1.3979e+02	1.8796e+02	1.7480e+02	9.8497e+02	2.4928e+02	**7.2088e+01**	2.4844e+02	2.7880e+02	1.5279e+02
	Std	5.7456e+01	4.6968e+01	7.4555e+01	1.2031e+01	1.4191e+03	1.2128e+02	2.9956e+01	9.1746e+01	5.0516e+01	3.6883e+01
	Rank	6	2	5	4	10	8	**1**	7	9	3
F12	Mean	3.6445e+06	9.7639e+05	**5.7688e+05**	1.3337e+07	5.2962e+06	3.8327e+06	5.9448e+05	4.4426e+07	5.1897e+06	9.7407e+05
	Std	2.9556e+06	6.8332e+05	4.4413e+05	4.6698e+06	5.5208e+06	1.2929e+06	3.7226e+05	5.3862e+07	2.0734e+06	4.8493e+05
	Rank	5	4	**1**	9	8	6	2	10	7	3
F13	Mean	1.0060e+04	9.6195e+03	7.4725e+03	2.8952e+04	9.3413e+03	**4.9793e+02**	9.1514e+03	5.3642e+05	5.8486e+03	4.7314e+03
	Std	8.9208e+03	1.1764e+04	8.2069e+03	3.0950e+04	1.0055e+04	2.0204e+02	1.0316e+04	1.9917e+06	4.1091e+03	4.6288e+03
	Rank	8	7	4	9	6	**1**	5	10	3	2
F14	Mean	7.5687e+04	5.9572e+04	6.8731e+04	1.7359e+05	2.5056e+06	5.1382e+05	2.2981e+04	**4.1505e+03**	3.9633e+05	5.5149e+04
	Std	6.7957e+04	4.6209e+04	7.5110e+04	1.0106e+05	2.4655e+06	3.4147e+05	2.2658e+04	3.1542e+03	1.8015e+05	3.0937e+04
	Rank	6	4	5	7	10	9	2	**1**	8	3
F15	Mean	9.1115e+03	9.1556e+03	8.8220e+03	1.2918e+04	9.4387e+03	**4.0187e+02**	1.1215e+04	8.4599e+03	1.0692e+03	5.8253e+03
	Std	6.9880e+03	1.0093e+04	8.6532e+03	9.3342e+03	8.7450e+03	4.6016e+02	1.1458e+04	6.2236e+03	4.8226e+02	5.0162e+03
	Rank	6	7	5	10	8	**1**	9	4	2	3
F16	Mean	1.3457e+03	1.5474e+03	1.1833e+03	2.6826e+03	2.1445e+03	1.1405e+03	1.4247e+03	1.4071e+03	1.2700e+03	**1.0473e+03**
	Std	4.0520e+02	2.6523e+02	4.0937e+02	2.8997e+02	5.0058e+02	2.1196e+02	4.8935e+02	4.9115e+02	1.5304e+02	3.0805e+02
	Rank	5	8	3	10	9	2	7	6	4	**1**
F17	Mean	1.0839e+03	1.3763e+03	1.0631e+03	1.7740e+03	1.6654e+03	8.5904e+02	1.1953e+03	1.0939e+03	8.9644e+02	**8.4994e+02**
	Std	2.8885e+02	2.5388e+02	3.3246e+02	2.0978e+02	3.0949e+02	1.9783e+02	3.4689e+02	3.2576e+02	1.6614e+02	2.5493e+02
	Rank	5	8	4	10	9	2	7	6	3	**1**
F18	Mean	1.9185e+06	3.0332e+05	2.0219e+05	2.2795e+06	3.6224e+06	7.1147e+05	1.1354e+05	**8.0782e+04**	6.4430e+05	3.2957e+05
	Std	1.7347e+06	1.5548e+05	1.0447e+05	9.5099e+05	3.1144e+06	4.2275e+05	6.9835e+04	4.3508e+04	3.1494e+05	1.5942e+05
	Rank	8	4	3	9	10	7	2	**1**	6	5
F19	Mean	1.1946e+04	5.4245e+03	1.3204e+04	1.9514e+04	1.5155e+04	7.4239e+02	1.7037e+04	1.5446e+04	**3.3717e+02**	1.6391e+04
	Std	1.2196e+04	4.9076e+03	1.1642e+04	8.7609e+03	1.2317e+04	1.1790e+03	1.6067e+04	6.6316e+03	4.2719e+02	8.2612e+03
	Rank	4	3	5	10	6	2	9	7	**1**	8
F20	Mean	7.0831e+02	1.0306e+03	7.9419e+02	1.3396e+03	1.3137e+03	6.6291e+02	8.8623e+02	7.1575e+02	7.7307e+02	**5.4097e+02**
	Std	2.3965e+02	1.7750e+02	3.2726e+02	2.4509e+02	3.8162e+02	1.7657e+02	3.2393e+02	3.1460e+02	1.6534e+02	2.0094e+02
	Rank	3	8	6	10	9	2	7	4	5	**1**
F21	Mean	3.3371e+02	3.9435e+02	3.2717e+02	5.4673e+02	4.4428e+02	3.3714e+02	3.7774e+02	3.4592e+02	4.9444e+02	**2.5813e+02**
	Std	2.4433e+01	2.5927e+01	2.9533e+01	1.3133e+01	4.5548e+01	2.5753e+01	4.0249e+01	2.9393e+01	3.0966e+01	1.1351e+01
	Rank	3	7	2	10	8	4	6	5	9	**1**
F22	Mean	6.5283e+03	6.4699e+03	5.2064e+03	1.2643e+04	6.1353e+03	4.5895e+03	6.9687e+03	4.1798e+03	6.0189e+03	**2.4762e+03**
	Std	1.7580e+03	4.6069e+02	2.4721e+03	2.1724e+03	7.2091e+02	1.1464e+03	9.7972e+02	3.3002e+03	4.0850e+02	2.4900e+03
	Rank	8	7	4	10	6	3	9	2	5	**1**
F23	Mean	5.7263e+02	6.4283e+02	5.8126e+02	7.9151e+02	7.0260e+02	5.9428e+02	6.0506e+02	6.1872e+02	7.2145e+02	**5.2458e+02**
	Std	2.8842e+01	5.3818e+01	3.3410e+01	1.4625e+01	5.1552e+01	2.5427e+01	5.0957e+01	6.6935e+01	6.4393e+01	2.5867e+01
	Rank	2	7	3	10	8	4	5	6	9	**1**
F24	Mean	7.1753e+02	7.2564e+02	6.7577e+02	8.7220e+02	1.1097e+03	8.9747e+02	6.9197e+02	6.8058e+02	9.9276e+02	**5.8747e+02**
	Std	1.0714e+02	3.3791e+01	4.5395e+01	1.1923e+01	1.2587e+02	5.9582e+01	3.9841e+01	7.9884e+01	5.8817e+01	2.6628e+01
	Rank	5	6	2	7	10	8	4	3	9	**1**
F25	Mean	6.1063e+02	4.3965e+02	4.4801e+02	**4.3126e+02**	5.5081e+02	4.4758e+02	4.5436e+02	9.1960e+02	5.5635e+02	5.6557e+02
	Std	3.6142e+01	1.5431e+01	2.4230e+01	7.3119e−02	3.5725e+01	1.7280e+01	2.8321e+01	2.1021e+02	1.5079e+01	2.9391e+01
	Rank	9	2	4	**1**	6	3	5	10	7	8
F26	Mean	2.6000e+03	3.1299e+03	2.2656e+03	4.8422e+03	4.3935e+03	2.7222e+03	2.7034e+03	3.4938e+03	**2.1568e+03**	2.2188e+03
	Std	2.9782e+02	5.6182e+02	9.2811e+02	2.2682e+02	1.0523e+03	5.9554e+02	6.0305e+02	2.0693e+03	1.3455e+03	2.6704e+02
	Rank	4	7	3	10	9	6	5	8	**1**	2
F27	Mean	6.7971e+02	5.0001e+02	4.7981e+02	5.0001e+02	9.5276e+02	**4.6637e+02**	5.0001e+02	7.7877e+02	6.5960e+02	7.6097e+02
	Std	4.4424e+01	1.5441e−04	2.1205e+01	9.7916e−05	1.3137e+02	1.0201e+01	3.1410e−04	1.0617e+02	2.7640e+01	8.1057e+01
	Rank	7	4	2	5	10	**1**	3	9	6	8
F28	Mean	5.3540e+02	5.0001e+02	**4.4663e+02**	4.9858e+02	6.0518e+02	4.6961e+02	4.9833e+02	1.2659e+03	5.4296e+02	5.1115e+02
	Std	3.4577e+01	1.3079e−04	8.1590e+00	7.8438e+00	2.8346e+02	1.3835e+01	6.4709e+00	2.1668e+02	1.3570e+01	2.2567e+01
	Rank	7	5	**1**	4	9	2	3	10	8	6
F29	Mean	8.1982e+02	1.5806e+03	9.5122e+02	1.6899e+03	1.6337e+03	**8.1357e+02**	1.0833e+03	1.1139e+03	1.0884e+03	1.1680e+03
	Std	2.5776e+02	2.2616e+02	2.9257e+02	2.1482e+02	4.4473e+02	1.7159e+02	2.9229e+02	3.6211e+02	1.2237e+02	2.4565e+02
	Rank	2	8	3	10	9	**1**	4	6	5	7
F30	Mean	2.6302e+06	5.8402e+03	2.0599e+03	3.4313e+05	1.4902e+06	**6.2310e+02**	4.2902e+03	3.1869e+06	1.0510e+06	9.2954e+05
	Std	9.0650e+05	7.2659e+03	1.5738e+03	3.1111e+05	1.5501e+06	2.8377e+02	4.1313e+03	3.8020e+06	1.0761e+05	1.3366e+05
	Rank	9	4	2	5	8	**1**	3	10	7	6
Mean rank		5.80	5.83	3.40	7.80	7.80	3.57	4.90	6.37	6.30	3.23
Final rank		5	6	2	9	9	3	4	8	7	1

References

[1] D. Whitley, A genetic algorithm tutorial, Stat. Comput. 4 (2) (1994) 65–85.

[2] S. Das, P.N. Suganthan, Differential evolution: A survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (1) (2010) 4–31.

[3] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95-International Conference on Neural Networks, Vol. 4, IEEE, 1995, pp. 1942–1948.

[4] Y. Shi, R.C. Eberhart, Empirical study of particle swarm optimization, in: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99, Vol. 3, IEEE, 1999, pp. 1945–1950.

[5] D. Karaboga, B. Basturk, On the performance of artificial bee colony (abc) algorithm, Appl. Soft Comput. 8 (1) (2008) 687–697.

[6] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, Adv. Eng. Softw. 69 (2014) 46–61.

[7] L. Gentile, C. Greco, E. Minisci, T. Bartz-Beielstein, M. Vasile, Stochastic satellite tracking with constrained budget via structured-chromosome genetic algorithms, Opt. Eng. (2021) 1–34.

[8] M. Di Carlo, M. Vasile, E. Minisci, Adaptive multi-population inflationary differential evolution, Soft Comput. 24 (5) (2020) 3861–3891.
[9] G. Filippi, M. Vasile, Global solution of constrained min–max problems with inflationary differential evolution, Opt. Eng. (2021) 1–47.
[10] R. Chai, A. Savvaris, A. Tsourdos, Y. Xia, S. Chai, Solving multiobjective constrained trajectory optimization problem by an extended evolutionary algorithm, IEEE Trans. Cybern. 50 (4) (2018) 1630–1643.
[11] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, Y. Xia, Solving constrained trajectory planning problems using biased particle swarm optimization, IEEE Trans. Aerosp. Electron. Syst. (2021).
[12] B. Aubin, B. Conway, S.-J. Chung, Finding optimal relative orbit transfer trajectories with the particle swarm algorithm and primer vector theory, in: 21st AAS/AIAA Space Flight Mechanics Meeting, 2011, pp. 853–871.
[13] Y.K. Nakka, A. Liu, G. Shi, A. Anandkumar, Y. Yue, S.-J. Chung, Chance-constrained trajectory optimization for safe exploration and learning of nonlinear systems, IEEE Robot. Autom. Lett. 6 (2) (2020) 389–396.
[14] S.-M. Chen, X.-Y. Zou, G.C. Gunawan, Fuzzy time series forecasting based on proportions of intervals and particle swarm optimization techniques, Inform. Sci. 500 (2019) 127–139, http://dx.doi.org/10.1016/j.ins.2019.05.047.
[15] W. Deng, R. Yao, H. Zhao, X. Yang, G. Li, A novel intelligent diagnosis method using optimal LS-SVM with improved PSO algorithm, Soft Comput. 23 (7, SI) (2019) 2445–2462, http://dx.doi.org/10.1007/s00500-017-2940-9.
[16] M.A. Hossain, H.R. Pota, S. Squartini, F. Zaman, J.M. Guerrero, Energy scheduling of community microgrid with battery cost using particle swarm optimisation, Appl. Energy 254 (NOV 15 2019) http://dx.doi.org/10.1016/j.apenergy.2019.113723.
[17] T. Li, M. Zhou, C. Guo, M. Luo, J. Wu, F. Pan, Q. Tao, T. He, Forecasting crude oil price using EEMD and RVM with adaptive PSO-based kernels, Energies 9 (12) (2016) 1014.
[18] K.-J. Wang, B. Makond, K.-H. Chen, K.-M. Wang, A hybrid classifier combining SMOTE with PSO to estimate 5-year survivability of breast cancer patients, Appl. Soft Comput. 20 (2014) 15–24, http://dx.doi.org/10.1016/j.asoc.2013.09.014.
[19] A. Chatterjee, P. Siarry, Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization, Comput. Oper. Res. 33 (3) (2006) 859–871, http://dx.doi.org/10.1016/j.cor.2004.08.012.
[20] K. Chen, F. Zhou, A. Liu, Chaotic dynamic weight particle swarm optimization for numerical function optimization, Knowl.-Based Syst. 139 (2018) 23–40, http://dx.doi.org/10.1016/j.knosys.2017.10.011.
[21] W.-C. Lin, Y. Yin, S.-R. Cheng, T.C.E. Cheng, C.-H. Wu, C.-C. Wu, Particle swarm optimization and opposite-based particle swarm optimization for two-agent multi-facility customer order scheduling with ready times, Appl. Soft Comput. 52 (2017) 877–884, http://dx.doi.org/10.1016/j.asoc.2016.09.038.
[22] 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, Inform. Sci. 508 (2020) 105–120, http://dx.doi.org/10.1016/j.ins.2019.08.065.
[23] X. Xia, L. Gui, Z.-H. Zhan, A multi-swarm particle swarm optimization algorithm based on dynamical topology and purposeful detecting, Appl. Soft Comput. 67 (2018) 126–140, http://dx.doi.org/10.1016/j.asoc.2018.02.042.
[24] J. Zou, Q. Deng, J. Zheng, S. Yang, A close neighbor mobility method using particle swarm optimizer for solving multimodal optimization problems, Inform. Sci. 519 (2020) 332–347, http://dx.doi.org/10.1016/j.ins.2020.01.049.
[25] K. Zhang, Q. Huang, Y. Zhang, Enhancing comprehensive learning particle swarm optimization with local optima topology, Inform. Sci. 471 (2019) 1–18, http://dx.doi.org/10.1016/j.ins.2018.08.049.
[26] Y. Zhang, X. Liu, F. Bao, J. Chi, C. Zhang, P. Liu, Particle swarm optimization with adaptive learning strategy, Knowl.-Based Syst. 196 (2020) http://dx.doi.org/10.1016/j.knosys.2020.105789.
[27] W. Li, X. Meng, Y. Huang, Z.-H. Fu, Multipopulation cooperative particle swarm optimization with a mixed mutation strategy, Inform. Sci. 529 (2020) 179–196.
[28] M.G. Epitropakis, V.P. Plagianakos, M.N. Vrahatis, Evolving cognitive and social experience in particle swarm optimization through differential evolution: A hybrid approach, Inform. Sci. 216 (2012) 50–92, http://dx.doi.org/10.1016/j.ins.2012.05.017.
[29] S. Goudarzi, W.H. Hassan, M.H. Anisi, A. Soleymani, M. Sookhak, M.K. Khan, A.-H.A. Hashim, M. Zareei, ABC-PSO For vertical handover in heterogeneous wireless networks, Neurocomputing 256 (2017) 63–81, http://dx.doi.org/10.1016/j.neucom.2016.08.136.
[30] Y. Shi, R.C. Eberhart, Parameter selection in particle swarm optimization, in: International Conference on Evolutionary Programming, Springer, 1998, pp. 591–600.
[31] A. Nickabadi, M.M. Ebadzadeh, R. Safabakhsh, A novel particle swarm optimization algorithm with adaptive inertia weight, Appl. Soft Comput. 11 (4) (2011) 3658–3670, http://dx.doi.org/10.1016/J.Asoc.2011.01.037.
[32] P. Chauhan, K. Deep, M. Pant, Novel inertia weight strategies for particle swarm optimization, Memet. Comput. 5 (3) (2013) 229–251, http://dx.doi.org/10.1007/s12293-013-0111-9.
[33] A.A. Nagra, F. Han, Q.H. Ling, An improved hybrid self-inertia weight adaptive particle swarm optimization algorithm with local search, Eng. Optim. 51 (7) (2019) 1115–1132, http://dx.doi.org/10.1080/0305215X.2018.1525709.
[34] M. Taherkhani, R. Safabakhsh, A novel stability-based adaptive inertia weight for particle swarm optimization, Appl. Soft Comput. 38 (2016) 281–295, http://dx.doi.org/10.1016/j.asoc.2015.10.004.
[35] M. Li, D. Lin, J. Kou, A hybrid niching PSO enhanced with recombination-replacement crowding strategy for multimodal function optimization, Appl. Soft Comput. 12 (3) (2012) 975–987, http://dx.doi.org/10.1016/j.asoc.2011.11.032.
[36] G. Das, P.K. Pattnaik, S.K. Padhy, Artificial neural network trained by particle swarm optimization for non-linear channel equalization, Expert Syst. Appl. 41 (7) (2014) 3491–3496, http://dx.doi.org/10.1016/j.eswa.2013.10.053.
[37] Q. Liu, W. Wei, H. Yuan, Z.-H. Zhan, Y. Li, Topology selection for particle swarm optimization, Inform. Sci. 363 (2016) 154–173, http://dx.doi.org/10.1016/j.ins.2016.04.050.
[38] H. Liu, X.-W. Zhang, L.-P. Tu, A modified particle swarm optimization using adaptive strategy, Expert Syst. Appl. 152 (2020) http://dx.doi.org/10.1016/j.eswa.2020.113353.
[39] X.-F. Liu, Y.-R. Zhou, X. Yu, Cooperative particle swarm optimization with reference-point-based prediction strategy for dynamic multiobjective optimization, Appl. Soft Comput. 87 (2020) http://dx.doi.org/10.1016/j.asoc.2019.105988.
[40] H. Liu, Z. Cai, Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, Appl. Soft Comput. 10 (2) (2010) 629–640, http://dx.doi.org/10.1016/j.asoc.2009.08.031.
[41] R. Hemalatha, R. Prakash, C. Sivapragash, Analysis on energy consumption in smart grid WSN using path operator calculus centrality based HSA-PSO algorithm, Soft Comput. 24 (14) (2020) 10771–10783, http://dx.doi.org/10.1007/s00500-019-04580-5.
[42] B. Tang, K. Xiang, M. Pang, An integrated particle swarm optimization approach hybridizing a new self-adaptive particle swarm optimization with a modified differential evolution, Neural Comput. Appl. 32 (9, SI) (2020) 4849–4883, http://dx.doi.org/10.1007/s00521-018-3878-2.
[43] Z. Li, W. Wang, Y. Yan, Z. Li, PS-ABC: A Hybrid algorithm based on particle swarm and artificial bee colony for high-dimensional optimization problems, Expert Syst. Appl. 42 (22) (2015) 8881–8895, http://dx.doi.org/10.1016/j.eswa.2015.07.043.
[44] J. Ding, Q. Wang, Q. Zhang, Q. Ye, Y. Ma, A hybrid particle swarm optimization-cuckoo search algorithm and its engineering applications, Math. Probl. Eng. 2019 (2019) http://dx.doi.org/10.1155/2019/5213759.
[45] B. Pitchaimanickam, G. Murugaboopathi, A hybrid firefly algorithm with particle swarm optimization for energy efficient optimal cluster head selection in wireless sensor networks, Neural Comput. Appl. 32 (12, SI) (2020) 7709–7723, http://dx.doi.org/10.1007/s00521-019-04441-0.
[46] Y.-J. Gong, J.-J. Li, Y. Zhou, Y. Li, H.S.-H. Chung, Y.-H. Shi, J. Zhang, Genetic learning particle swarm optimization, IEEE Trans. Cybern. 46 (10) (2016) 2277–2290, http://dx.doi.org/10.1109/TCYB.2015.2475174.
[47] J. Tang, G. Liu, Q. Pan, A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends, IEEE/CAA J. Autom. Sin. 8 (10) (2021) 1627–1643.
[48] J. Kennedy, Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance, in: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99, Vol. 3, IEEE, 1999, pp. 1931–1938.
[49] L. Wang, B. Yang, Y. Chen, Improving particle swarm optimization using multi-layer searching strategy, Inform. Sci. 274 (2014) 70–94, http://dx.doi.org/10.1016/j.ins.2014.02.143.
[50] Q. Yang, W.-N. Chen, J. Da Deng, Y. Li, T. Gu, J. Zhang, A level-based learning swarm optimizer for large-scale optimization, IEEE Trans. Evol. Comput. 22 (4) (2018) 578–594, http://dx.doi.org/10.1109/TEVC.2017.2743016.
[51] X. Wu, M. Sun, X. Chen, J. Wang, B. Guo, Empirical study of particle swarm optimization inspired by Lotka–Volterra model in ecology, Soft Comput. 23 (14) (2019) 5571–5582, http://dx.doi.org/10.1007/s00500-018-3215-9.
[52] D. Wu, N. Jiang, W. Du, K. Tang, X. Cao, Particle swarm optimization with moving particles on scale-free networks, IEEE Trans. Netw. Sci. Eng. 7 (1) (2020) 497–506, http://dx.doi.org/10.1109/TNSE.2018.2854884.
[53] R. Cheng, Y. Jin, A competitive swarm optimizer for large scale optimization, IEEE Trans. Cybern. 45 (2) (2015) 191–204, http://dx.doi.org/10.1109/TCYB.2014.2322602.
[54] R. Cheng, Y. Jin, A social learning particle swarm optimization algorithm for scalable optimization, Inform. Sci. 291 (2015) 43–60, http://dx.doi.org/10.1016/j.ins.2014.08.039.
[55] I.C. Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Inform. Process. Lett. 85 (6) (2003) 317–325.

[56] Y. Zhao, W. Zu, H. Zeng, A modified particle swarm optimization via particle visual modeling analysis, Comput. Math. Appl. 57 (11–12) (2009) 2022–2029, http://dx.doi.org/10.1016/j.camwa.2008.10.007.
[57] K.H. Tsai, T.I. Wang, T.C. Hsieh, T.K. Chiu, M.C. Lee, Dynamic computerized testlet-based test generation system by discrete PSO with partial course ontology, Expert Syst. Appl. 37 (1) (2010) 774–786, http://dx.doi.org/10.1016/j.eswa.2009.05.090.
[58] D. Zhou, X. Gao, G. Liu, C. Mei, D. Jiang, Y. Liu, Randomization in particle swarm optimization for global search ability, Expert Syst. Appl. 38 (12) (2011) 15356–15364, http://dx.doi.org/10.1016/j.eswa.2011.06.029.
[59] T. Wu, L. Xie, X. Chen, A.H. Ashrafzadeh, S. Zhang, A novel quantum-behaved particle swarm optimization algorithm, CMC-Comput. Mater. Contin. 63 (2) (2020) 873–890, http://dx.doi.org/10.32604/cmc.2020.07478.
[60] J. Ding, J. Liu, K.R. Chowdhury, W. Zhang, Q. Hu, J. Lei, A particle swarm optimization using local stochastic search and enhancing diversity for continuous optimization, Neurocomputing 137 (SI) (2014) 261–267, http://dx.doi.org/10.1016/j.neucom.2013.03.075.
[61] Z. Xinchao, A perturbed particle swarm algorithm for numerical optimization, Appl. Soft Comput. 10 (1) (2010) 119–124, http://dx.doi.org/10.1016/j.asoc.2009.06.010.
[62] G. Xu, An adaptive parameter tuning of particle swarm optimization algorithm, Appl. Math. Comput. 219 (9) (2013) 4560–4569, http://dx.doi.org/10.1016/j.amc.2012.10.067.
[63] C. Wang, Y. Liu, Y. Chen, Y. Wei, Self-adapting hybrid strategy particle swarm optimization algorithm, Soft Comput. 20 (12, SI) (2016) 4933–4963, http://dx.doi.org/10.1007/s00500-015-1784-4.
[64] Q. Wang, S. Chen, X. Luo, An adaptive latent factor model via particle swarm optimization, Neurocomputing 369 (2019) 176–184, http://dx.doi.org/10.1016/j.neucom.2019.08.052.
[65] R.P. Parouha, Nonconvex/nonsmooth economic load dispatch using modified time-varying particle swarm optimization, Comput. Intell. 35 (4) (2019) 717–744, http://dx.doi.org/10.1111/coin.12210.
[66] J. Wu, C. Song, C. Fan, A. Hawbani, L. Zhao, X. Sun, DENPSO: A Distance evolution nonlinear PSO algorithm for energy-efficient path planning in 3D UASNs, IEEE Access 7 (2019) 105514–105530, http://dx.doi.org/10.1109/ACCESS.2019.2932148.
[67] G. Singh, A. Singh, A hybrid algorithm using particle swarm optimization for solving transportation problem, Neural Comput. Appl. 32 (15) (2020) 11699–11716, http://dx.doi.org/10.1007/s00521-019-04656-1.
[68] X. Xia, Y. Tang, B. Wei, L. Gui, Dynamic multi-swarm particle swarm optimization based on elite learning, IEEE Access 7 (2019) 184849–184865, http://dx.doi.org/10.1109/ACCESS.2019.2960890.
[69] Y. Li, Z.-H. Zhan, S. Lin, J. Zhang, X. Luo, Competitive and cooperative particle swarm optimization with information sharing mechanism for global optimization problems, Inform. Sci. 293 (2015) 370–382.
[70] T. Li, M. Zhou, ECG classificaton using wavelet packet entropy and random forests, Entropy 18 (8) (2016) 285.
[71] T.N. Huynh, D.T. Do, J. Lee, Q-learning-based parameter control in differential evolution for structural optimization, Appl. Soft Comput. 107 (2021) 107464, http://dx.doi.org/10.1016/j.asoc.2021.107464.
[72] T. Li, J. Shi, D. Zhang, Color image encryption based on joint permutation and diffusion, J. Electron. Imaging 30 (1) (2021) 013008, http://dx.doi.org/10.1117/1.JEI.30.1.013008.
[73] T. Li, Z. Qian, W. Deng, D. Zhang, H. Lu, S. Wang, Forecasting crude oil prices based on variational mode decomposition and random sparse Bayesian learning, Appl. Soft Comput. 113 (2021) 108032, http://dx.doi.org/10.1016/j.asoc.2021.108032.
[74] X. Li, A. Engelbrecht, M.G. Epitropakis, Benchmark Functions for Cec'2013 Special Session and Competition on Niching Methods for Multimodal Function Optimization, Tech. Rep., RMIT University, Evolutionary Computation and Machine Learning Group, Australia, 2013.
[75] G. Wu, R. Mallipeddi, P.N. Suganthan, Problem Definitions and Evaluation Criteria for the Cec 2017 Competition on Constrained Real-Parameter Optimization, Technical Report, National University of Defense Technology, Changsha, Hunan, PR China and Kyungpook National University, Daegu, South Korea and Nanyang Technological University, Singapore, 2017.
[76] K.E. Parsopoulos, UPSO: A Unified particle swarm optimization scheme, Lect. Ser. Comput. Comput. Sci. 1 (2004) 868–873.
[77] R. Mendes, J. Kennedy, J. Neves, The fully informed particle swarm: simpler, maybe better, IEEE Trans. Evol. Comput. 8 (3) (2004) 204–210.
[78] T. Peram, K. Veeramachaneni, C.K. Mohan, Fitness-distance-ratio based particle swarm optimization, in: Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03, IEEE, 2003, pp. 174–181.
[79] 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 (3) (2006) 281–295, http://dx.doi.org/10.1109/TEVC.2005.857610.
[80] F. vandenBergh, A. Engelbrecht, A cooperative approach to particle swarm optimization, IEEE Trans. Evol. Comput. 8 (3) (2004) 225–239, http://dx.doi.org/10.1109/tevc.2004.826069.
[81] F. Wilcoxon, Individual comparisons by ranking methods, in: Breakthroughs in Statistics, Springer, 1992, pp. 196–202.