Applied Soft Computing 132 (2023) 109852

Contents lists available at ScienceDirect

Applied Soft Computing

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

A particle swarm optimizer with dynamic balance of convergence and diversity for large-scale optimization

Dongyang Li [a], Lei Wang [b], Weian Guo [a,c,*], Maoqing Zhang [b], Bo Hu [b], Qidi Wu [b]

[a] Sino-German College of Applied Sciences, Tongji University, Shanghai, 200092, China
[b] Department of Electronics and Information Engineering, Tongji University, Shanghai, 201804, China
[c] Shanghai Institute of Intelligent Science and Technology, Tongji University, Shanghai, 200092, China

ARTICLE INFO

Article history:
Received 6 February 2021
Received in revised form 13 October 2022
Accepted 14 November 2022
Available online 5 December 2022

Keywords:
Particle swarm optimization
Convergence and diversity
Large-scale optimization
Multi-swarm mechanism
Centralized electric vehicles charging

ABSTRACT

Particle swarm optimization is found ineffective in large-scale optimization. The main reason is that particle swarlarge-scalem optimization cannot effectively balance convergence and diversity. This paper proposes a particle swarm optimizer with a dynamic balance of convergence and diversity (PSO-DBCD). In the proposed algorithm, a competitive multi-swarm mechanism is put forward, based on which a convergence-guiding learning strategy is proposed for the management of convergence pressure. Furthermore, an entropy-based local diversity measurement is proposed to measure the local diversity of particles. Afterwards, a diversity-guiding learning strategy is proposed based on the local diversity information to further improve the diversity preservation ability of the algorithm. Theoretical analyses are presented to investigate the characteristics of PSO-DBCD. Comprehensive experiments are conducted based on the benchmarks posted on CEC 2013 and several state-of-the-art algorithms to test the performance and scalability of the proposed algorithm. The PSO-DBCD exhibits evident advantages over the compared algorithms in the optimization results with respect to the statistical test results. The proposed strategies are demonstrated to be effective in managing the convergence speed and the swarm diversity. Lastly, a case study of centralized electric vehicle charging optimization shows that PSO-DBCD can reduce the cost of charging for people who use electric vehicles.

1. Introduction

Canonical particle swarm optimization (cPSO) has received a lot of attention in recent years as a type of meta-heuristic algorithm because of its simplicity and efficiency [1–4]. Despite significant advancements over the last two decades [5–9], cPSO and its variants continue to face the challenge of large-scale optimization problems (LSOPs), which can be expressed by

$$minf(x), \quad x = [x^1, x^2, \ldots, x^D], \tag{1}$$

where D is the dimensionality of the optimization problem. The reasons are that, first, the increasing dimensionality causes an exponential growth of the number of local optima; second, the size of the areas around the local optima expands dramatically with the increase of the dimensionality; and finally, only a few parts of the decision space can be searched by the optimizers with limited computation resources [10,11].

Focusing on improving cPSO for LSOPs, the current efforts can be mainly unfolded into two categories. The first category is to propose new learning or updating strategies to balance the convergence and diversity of cPSO. For example, using multi-swarm techniques to improve the algorithm in diversity preservation [10,12]; using a competition mechanism to diversify the exemplars to improve cPSO's diversity preservation ability [13,14]; hybridizing cPSO with other techniques [15,16]; and adjusting the parameters [17,18]. The second category adopts a cooperatively coevolutionary (CC) framework to decompose the whole decision vector into different sub-segments to transform LSOPs into a set of low-dimensional optimization problems, such as the random grouping based techniques [19–21], differential grouping based methods [22,23], and [24,25].

Despite of the success of the existing researches, these methods suffer from the following problems. For the methods in the first category, the balance of convergence and diversity remains a challenging task. The main reason is that (i) Both the social learning component and the cognitive learning component in cPSO contribute to the convergence [26], resulting in difficulties in tuning the parameters for balancing convergence and diversity. (ii) The convergence pressure and diversity cannot be adjusted effectively. For instance, competitive swarm optimizer (CSO) cannot control the difference between the updated particles and the exemplars (the convergence pressure), whilst it is difficult for

* Corresponding author at: Department of Electronics and Information Engineering, Tongji University, Shanghai, 201804, China.
E-mail address: guoweian@163.com (W. Guo).

https://doi.org/10.1016/j.asoc.2022.109852
1568-4946/

CSO to enhance the swarm diversity as the mean position of the swarm is shared by all the updated particles. More particles with high quality cannot be updated if the level-based learning swarm optimizer (DLLSO) [10] tries to strengthen its convergence pressure, which is adverse to convergence. Furthermore, the distribution of the particles is not considered by DLLSO, which means particles might learn from the individuals in crowded areas for diversity preservation. (iii) Hybridizing cPSO with other algorithms commonly results in additional parameters [14]. For the methods in the second category, their performance is highly influenced by variable grouping techniques, and both solution evaluation and variable grouping in such methods will cause heavy computational cost [10].

Therefore, it is still challenging for cPSO and its variants to solve LSOPs. Particularly, improving the learning or updating strategies for cPSO is necessary to help further balance the convergence and diversity. To address this issue, this paper focuses on designing novel learning strategies to explicitly manage the convergence pressure and diversity. By this means, one can balance these two factors with specific parameters. Following this idea, we propose a novel cPSO variant which adopts two components to explicitly adjust the convergence pressure and diversity. The main contributions of this paper are listed as follows:

1. To balance convergence and diversity, a dynamic and competitive multi-swarm mechanism (DCMM) is designed, where a multi-swarm mechanism is adopted to dynamically adjust the convergence pressure while a competition mechanism is employed for diversity preservation. A convergence-guiding learning strategy (CGLS) is built based on DCMM, which is able to dynamically adjust the swarm convergence pressure with the sub-swarm size. Furthermore, the competitive mechanism used in DCMM ensures that the proposed CGLS does not significantly reduce swarm diversity.
2. To explicitly adjust the diversity, this paper proposes an entropy-based local diversity measurement (ELD) to measure the local diversity information for particles, based on which a diversity-guiding learning strategy (DGLS) is designed for further adjusting the swarm diversity. ELD is independent of the attributes of problems and can provide individuals with diversity information without introducing additional parameters.

With the proposed learning strategies, a novel cPSO variant of particle swarm optimizer with a dynamic balance of convergence and diversity (PSO-DBCD) has been developed in this paper. PSO-DBCD can manage convergence pressure and diversity simultaneously and explicitly, which is advantageous for balancing convergence and diversity. Furthermore, PSO-DBCD is simple in design and implementation.

The rest of this paper is organized as follows. Section 2 provides a comprehensive review of the main stream of meta-heuristics in solving LSOPs, with a focus on cPSO-based methods. Section 3 provides a detailed description of PSO-DBCD. Section 4 theoretically analyzes the searching behavior, the convergence stability, and the computational complexity of PSO-DBCD. Section 5 experimentally tests the performance of the proposed algorithm. In Section 6, PSO-DBCD is applied to solve an engineering problem of large-scale electric vehicles' charging dispatch problem. Finally, we conclude this paper and provide the future work directions in Section 7.

2. Related work

2.1. A brief of particle swarm optimization

In cPSO, every particle has two attributes of position and velocity [27,28], which are iteratively updated according to

$$v_i^d(t+1) = \omega v_i^d(t) + c_1 r_1 (pbest_i^d(t) - p_i^d(t)) + c_2 r_2 (gbest^d(t) - p_i^d(t)) \quad (2)$$

$$p_i^d(t+1) = p_i^d(t) + v_i^d(t+1), \quad (3)$$

where $v_i^d(t)$ and $p_i^d(t)$ are the dth dimension of the ith particle's velocity and position at generation t, respectively; $gbest(t)$ is the best position searched by the whole swarm at generation t; $pbest_i(t)$ denotes the best position located by the ith particle so far; ω is the inertia weight; c_1 and c_2 are termed as acceleration coefficients; r_1 and r_2 are randomly generated within (0, 1). This enables cPSO to traverse the decision space and find promising solutions by updating the swarm iteratively.

As seen in (2), the greedy strategy of cPSO updates particles with *pbest* and *gbest*. Therefore, particles are usually trapped by local optima while solving large-scale problems [29]. To date, the two most extensively used approaches for solving LSOPs are presenting new updating strategies and embedding algorithms into the CC framework [10].

2.2. New learning or updating strategies

A huge amount of effort has been put into proposing new learning or updating strategies to balance the convergence and diversity of meta-heuristics.

Zhao et al. suggest DMS-PSO to encourage the exchange of information between particles by dynamically changing the sub-swarm size, while a local search method is adopted for convergence [12]. Montes et al. propose a tuning-in-the-loop methodology for redesigning the parameters in IPSOLS [17,18]. Cheng et al. come up with FBE, which uses a dual-swarm learning technique and a pairwise competition mechanism to improve the algorithm's ability to keep the diversity [13]. Later, they propose CSO, which divides the swarm into the winner group and loser group based on the competition mechanism. The winners are then directly kept to the next generation, while the losers learn from the corresponding winners and the mean position of the swarm for diversity enhancement [14]. Cheng et al. come up with SLPSO, in which each dimension learns from a randomly chosen better particle to make the exemplars even more diverse [30]. Saban et al. propose PCLPSO, where a multi-swarm strategy and distributed computing technique are adopted to enhance the search ability of comprehensive learning particle swarm optimization [31]. Lynn et al. propose HCLPSO, where a dual-swarm strategy is adopted and the example selection procedures in the two sub-swarms are different for convergence and diversity preservation, respectively [32]. Yang et al. propose DLLSO, which presents a dynamic level-based learning strategy to balance the exploration and exploitation for the algorithm [10]. Salih et al. propose MPSO, where a novel multi-swarm strategy is developed [33]. Kong et al. propose AMCPSO, which introduces a multi-swarm competition learning strategy to balance convergence and diversity [34]. Yang et al. propose DEGLSO, where an asynchronous and an adaptive communication strategy are designed based on a request response mechanism [35]. Li et al. propose a multi-swarm-based method to balance the exploration and exploitation of cPSO [36].

Furthermore, hybridizing cPSO with other techniques is also a hot topic. Jia et al. for example, propose CGPSO, which combines Chaotic local search and Gaussian optimization with cPSO for

diversity preservation and convergence, respectively [37]. Tang et al. propose SMQPSO, which hybridizes a kind of memetic algorithm with cPSO for exploration [38]. Tao et al. propose SA-PSO, where the solution acceptance rule in simulated annealing algorithm is adopted to improve the diversity preservation ability of the algorithm [39]. Ali et al. propose HPSOGA, where genetic operators are adopted to improve cPSO in diversity preservation [16]. Wang et al. propose a parameter adjustment strategy based on locality-sensitive hashing and logistic regression [40]. Li et al. propose an adaptive hybrid-model learning strategy to optimize the mixed-variables for convolutional neural networks [41]. Jian et al. propose a region encoding scheme to extend the solution representation from a single point to a region. This can help the algorithm improve the convergence of social learning particle swarm optimization [42].

Except for cPSO, a lot of focus has been paid to enhancing other kinds of algorithms. Ros et al. propose sep-CMA-ES, where a diagonal covariance matrix is introduced to help reduce the calculation burden of CMA-ES [43]. Loshchilov et al. propose LM-CMA-ES, where the covariance matrix is reproduced by m selected direction vectors to reduce the time and memory complexity [44]. Molina et al. propose Ma-Sw-Chains, where several optimization techniques are adopted for exploration and exploitation, respectively [45]. Latorre et al. propose MOS, where multiple kinds of optimizers are dispatched to solve LSOPs [46]. Brest et al. propose jDElscop, where different mutation operators are combined for diversity preservation and a population reduction mechanism is designed for convergence [47]. Hadi et al. propose LSADE-SPA, where several optimization methods are integrated together to solve LSOPs [48,49]. Molina et al. put forward SHADE-ILS, where a re-start strategy and two large-scale local search methods are employed to enhance the algorithm's performance [50]. Liu et al. propose a distributed differential evolution algorithm, where a resource-aware strategy is designed to efficiently and adaptively dispatch the computing resources [51].

However, the current approaches still struggle to balance convergence and diversity. The main reason is the conflict between convergence and diversity, which makes these two factors difficult to be explicitly and effectively managed [10,26].

2.3. Adopting CC framework

These approaches try to solve LSOPs in low-dimensional decision space, which helps to address the challenges brought on by increasing dimensionality.

Van et al. propose CCPSO $-$ S_K and CCPSO $-$ S_H, which is the first attempt to incorporate cPSO into the CC framework [19]. The former divides the decision vector into D/K sub-segments which are independently and simultaneously optimized by cPSO, while the latter is designed to optimize the swarm by CCPSO $-$ S_K and cPSO, iteratively. Li et al. propose CCPSO, where a random variable grouping strategy and an adaptive weighting scheme are combined with CCPSO $-$ S_K [20]. Furthermore, Li et al. propose CCPSO2, Gaussian mutation and Cauchy mutation are introduced in CCPSO for diversity preservation and convergence, respectively [21]. Tang et al. propose AM-CCPSO, where a multi-context vector strategy is designed for exploration and a Gaussian sampling method is adopted for convergence [24]. Peng et al. propose MMO-CC, where a multi-objective mechanism is proposed for the selection of context vectors [25].

The variable grouping technique is crucial to the methods within the CC framework. Chen et al. propose variable interaction learning, which iteratively detects the dependency between variables in the optimization process [52]. Mahdavi et al. propose a meta-modeling decomposition to detect the dependency between variables [53]. Omidvar et al. propose DG, where the interactions between variables are detected based on the fitness variation [22]. Sun et al. propose XDG, where the indirect interactions among variables can be identified [54]. Mei et al. propose GDG, which suggests a computational error to solve the sensitivity problem of DG [55]. Omidvar et al. propose DG2, where a systematic selection strategy is proposed to save computational cost [56]. Yang et al. propose a recursive differential grouping method to accelerate the variable grouping process [25]. Li et al. propose dual differential grouping to decompose multiplicatively separable functions [57].

Nevertheless, the current variable grouping methods face the accuracy difficulty, and both the variable grouping and the solution evaluation incur significant calculation costs [10,25].

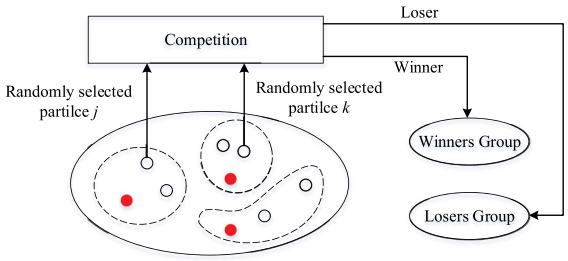


Fig. 1. A sketch of the random competition mechanism, where the swarm is divided into three sub-swarms, and the best particles in each sub-swarm are highlighted in red.

3. Particle swarm optimizer with dynamic balance of convergence and diversity

3.1. Proposed convergence operator

For convergence, a convergence-guiding learning strategy (CGLS) with dynamic and competitive multi-swarm mechanism (DCMM) is proposed. The key idea is to manage the convergence pressure with dynamic multi-swarm techniques. The details are presented as follows.

3.1.1. Dynamic and competitive multi-swarm mechanism

First, the whole swarm will be randomly divided into several sub-swarms, where the best particle in each sub-swarm will be directly kept to the next generation, while other particles in each sub-swarm will learn from the corresponding best particle.

Second, as shown in Fig. 1, the fitness-based competition [14] will be carried out at each generation to determine the winners and losers. The winners will be directly passed on to the next generation.

One can observe two features in DCMM: first, the sub-swarm size can be changed to adjust the mean fitness difference between the updated particles and the exemplars (convergence pressure) in each sub-swarm. For better understanding, Fig. 2 shows a sketch of two swarm grouping scenarios. The swarms in Figs. 2(a) and 2(b) are divided into three and two sub-swarms, respectively. The best particles in each sub-swarms are marked by red. Obviously, the mean distance between the best particles and the remaining individuals in each sub-swarm in Fig. 2(a) is smaller than that in Fig. 2(b), indicating lower convergence pressure. Second, at each generation, half of the particles can be preserved, ensuring that the swarm diversity is not heavily influenced.

Considering the huge decision space in LSOPs, the algorithm is expected to explore the decision space at an early stage and focus on exploitation in the latter stage. Therefore, the sub-swarm size should be dynamically increased during the optimization process

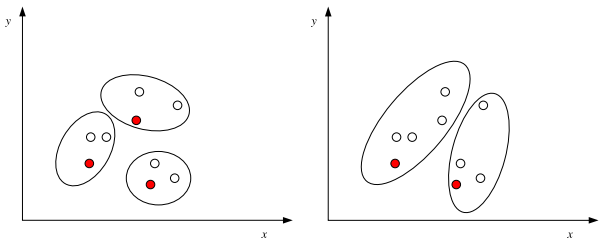


Fig. 2. Different swarm grouping scenarios.

to balance the convergence pressure and diversity. In this paper, a set of sub-swarm sizes $S_{sw} = \{s_1, s_2, \ldots, s_m\}$ is needed. The whole optimization process, afterwards, is equally divided into m stages and the sub-swarm size is dynamically adjusted from small to large. The pseudo-code of DCMM is shown in Algorithm 1, where P_{uc} and P_{ec} are two sets to store the particles to be updated and the corresponding convergence exemplars, respectively.

Algorithm 1: Pseudo-code of DCMM

Input: Swarm $P(t)$, fitness value vector *fitness*, sub-swarm size s, swarm size N
Output: The particle set to be updated: P_{uc} and the particle set should be preserved: P_{ec} at tth generation

$P_{uc} \leftarrow \emptyset$;
$P_{ec} \leftarrow \emptyset$;
Equally divide $P(t)$ into $n_{sw} = N/s$ sub-swarms;
for $i = 1$ *to* n_{sw} **do**
  for $j = 1$ *to* s **do**
    if *the* jth *particle is not the best individual in the* ith *sum-swarm* **then**
      $P_{uc} \leftarrow P_{uc} \bigcup$ the jth particle;
      $P_{ec} \leftarrow P_{ec} \bigcup$ the best particle in the ith swarm;
    end
  end
end
$P_l \leftarrow$ the losers group with respect to the randomly pairwise competition;
for $i = 1$ *to* $|P_{uc}|$ **do**
  if *the* ith *particle in* $P_{uc} \notin P_l$ **then**
    Remove the ith particle from P_{uc};
    Remove the ith particle from P_{ec};
  end
end

3.1.2. Convergence-guiding learning strategy

With the proposed DCMM, a convergence-guiding learning strategy (CGLS) is presented as follows. First, DCMM is adopted to determine P_{uc} and P_{ec}. Then for convergence, CGLS updates the velocity of the ith particle in P_{uc} at generation t according to

$$v_i^d(t+1) = r_1(p_{i,ec}^d(t) - p_i^d(t)), \tag{4}$$

where v_i^d and p_i^d are the dth dimension of the ith particle's velocity and position, respectively; t is the index of generation; r_1 is a random number generated within $(0, 1)$; $p_{i,ec}^d$ is the dth dimension of the corresponding convergence exemplar in P_{ec}.

3.2. Proposed diversity operator

For further diversity preservation, this paper proposes a diversity-guiding learning strategy (DGLS) with a novel entropy-based local diversity measurement (ELD). The main idea is to quantify the local diversity information and direct particles from crowded areas to sparse areas.

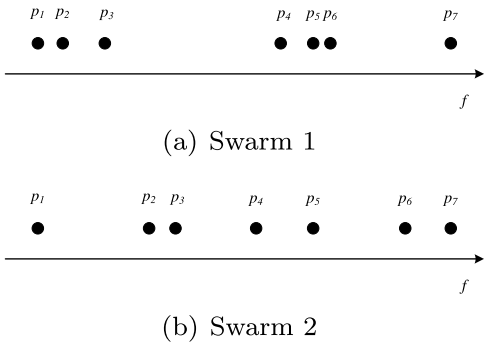


Fig. 3. Two potential fitness distribution sketches.

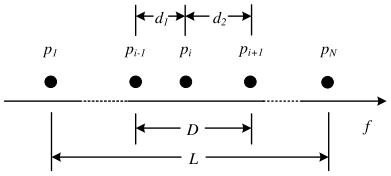


Fig. 4. The calculation of ELD.

3.2.1. Entropy-based local diversity measurement

The suggested ELD is based on phenotypic diversity [58] in order to reduce computational costs. In Fig. 3, two potential fitness distribution sketches are displayed, with each swarm consisting of seven particles. Obviously, swarm 1's diversity is lower than swarm 2's. In order to increase the diversity of Swarm 1, one should guide particles moving to the sparse areas, such as the area between p_3 and p_4. To this end, evaluating the local diversity information is needed to determine whether a particle is able to guide others moving to sparse areas. Consequently, ELD is proposed by taking the size and distribution of the local neighborhood spaces of a particle into consideration.

First, while evaluating local diversity, the neighborhood space size around a particle is crucial. For instance, in Fig. 3(a), p_4 is located in a sparser area than p_2. Second, it is important to consider the distribution of the neighborhood spaces around particles. For instance, as illustrated in Fig. 3(b), the neighborhood spaces around p_2 and p_4 are nearly identical. However, since the neighborhood space of p_2 is more congested than that of p_4, moving particles to p_4 is more reasonable than moving particles to p_2.

Consequently, to provide a clear measurement for local diversity, an entropy-based local diversity (ELD) of the ith particle is evaluated by (5) to (8). Fig. 4 shows the calculation of d_1, d_2, D, and L, where N is the swarm size.

$$NS(i) = D/L \tag{5}$$

$$ND(i) = -\frac{d_1}{D} ln(\frac{d_2}{D}) - \frac{d_2}{D} ln(\frac{d_1}{D}) \tag{6}$$

$$ELD(i) = Norm(NS(i))Norm(ND(i)) \tag{7}$$

$$Norm(\vec{v}(i)) = \frac{\vec{v}(i) - min(\vec{v})}{max(\vec{v}) - min(\vec{v})}. \tag{8}$$

It is clear that (7) is an evaluation of the size and distribution of the ith particle's neighborhood space. As a result, if the $ELD(i)$ is larger than $ELD(j)$, the ith particle is located in a more sparse area in comparison to the jth particle. Because they only have one neighbor, the best and worst particles' ELD is set to zero. The optimization will not be impacted by ignoring these two border particles because the swarm size of N is typically quite large. Furthermore, $NS(i)$ will be set to 0 if $L = 0$, whereas $ND(i)$ will be set to 0 if $d_1 \vee d_2 \vee D = 0$.

3.2.2. Diversity-guiding learning strategy

In this paper, the particles are guided towards sparse areas for diversity preservation based on ELD. First, sort the swarm in ascending order according to fitness and calculate the *ELD*. Second, a random weighting strategy (RWS) is adopted to choose the particles to be updated and the diversity exemplars: the ith particle learns from the jth particle if $ELD(i) \leq r \cdot ELD(j)$ or the ith particle is the worst individual in the corresponding sub-swarm obtained with DCMM, where $r \in (0, 1)$ and the jth particle is randomly selected from the individuals with larger *ELD* than the ith particle, otherwise, the ith particle will be kept to the next generation without any operation. The pseudo-code of RWS is shown in Algorithm 2, where P_{ud} and P_{ed} are the particles to be updated and the corresponding diversity exemplars, respectively.

Algorithm 2: Pseudo-code of RWS

Input: Swarm $P(t)$, fitness value vector *fitness*, swarm size N
Output: The particle set to be updated of P_{ud} and the particle set should be preserved of P_{ed} at tth generation

$P_{uc} \leftarrow \emptyset$;
$P_{ec} \leftarrow \emptyset$;
Compute the *ELD* according to (5) to (8);
for $i = 1$ *to* N **do**
  Randomly selected the jth particle from those with larger *ELD* than the ith particle;
  kth sub-swarm $\leftarrow$ the sub-swarm that the ith particle in;
  if $ELD(i) <= r \cdot ELD(j)$ *or the* ith *particle is the worst individual in the* kth *sub-swarm* **then**
    $P_{ud} \leftarrow P_{ud} \bigcup$ the ith particle;
    $P_{ed} \leftarrow P_{ed} \bigcup$ the jth swarm;
  end
end

Afterwards, DGLS updates the velocity of the ith particle in P_{ud} for diversity preservation according to

$$v_i^d(t+1) = \phi r_2(p_{i,ed}^d(t) - p_i^d(t)), \tag{9}$$

where v_i^d and p_i^d are the dth dimension of the ith particle's velocity and position, respectively; t is the generation number; ϕ is predefined by users to adjust the learning rate for diversity preservation; r_2 is randomly generated within $(0, 1)$; $p_{i,ed}^d$ is the dth dimension of the corresponding diversity exemplar stored in P_{ed} for the ith particle in P_{ud}.

3.3. PSO-DBCD

With the proposed CGLS and DGLS, a novel particle updating strategy is proposed as shown in (10) and (11), leading to PSO-DBCD. In (10) and (11), v_i^d and p_i^d are the dth dimension of the ith particle's velocity and position, respectively; t is the generation number; w, r_1 and r_2 are randomly generated within $(0, 1)$; $p_{i,ec}$ and $p_{i,ed}$ are the ith particle's convergence exemplars and diversity exemplars, respectively. By this means, PSO-DBCD is able to dynamically and explicitly manage the convergence pressure and diversity with two independent operators. The pseudo-code of PSO-DBCD is shown in Algorithm 3 and the flowchart of PSO-DBCD is shown in Fig. 5.

$$v_i^d(t+1) = \omega v_i^d(t) + r_1(p_{i,ec}^d(t) - p_i^d(t)) + \phi r_2(p_{i,ed}^d(t) - p_i^d(t)) \tag{10}$$

$$p_i^d(t+1) = p_i^d(t) + v_i^d(t+1), \tag{11}$$

Algorithm 3: Pseudo-code of PSO-DBCD

Input: Swarm size N, set S, parameter ϕ
Output: *gbest*: the best solution

Initialize a swarm P;
fitness $\leftarrow$ Compute the fitness vector;
$t = 1$;
while *terminal condition is not met* **do**
  Sort the P_t according to the fitness in ascending order;
  Select the sub-swarm size s from S_{sw};
  $P_{ec}, P_{uc} \leftarrow$ Run Algorithm 1;
  $P_{ed}, P_{ud} \leftarrow$ Run Algorithm 2;
  for $i = 1$ *to* N **do**
    if *the* ith *particle* $\in P_{uc} \bigcap P_{ud}$ **then**
      Update the ith particle according to (10) and (11);
      Check the feasibility of the ith particle;;
    else
      Pass the ith particle to the next generation without doing anything;
    end
  end
  fitness $\leftarrow$ Compute the fitness vector;
  gbest $\leftarrow$ The current best particle;
  $t = t + 1$;
end

4. Theoretical analysis of PSO-DBCD

4.1. Search behavior analysis

To examine its exploitation and exploration behavior, we contrast PSO-DBCD with cPSO and DLLSO. The velocity update formulas of cPSO and DLLSO are shown in (2) and (12), respectively.

$$v_i^d(t+1) = \omega v_i^d(t) + r_1(p_{i,rl1}^d(t) - p_i^d(t)) + \phi r_2(p_{i,rl2}^d(t) - p_i^d(t)), \tag{12}$$

where ω, r_1 and r_2 are randomly generated within $(0, 1)$; $p_{i,rl1}$ and $p_{i,rl2}$ are the exploitation and exploration exemplar, respectively. The selection of $p_{i,rl1}$ and $p_{i,rl2}$ is introduced as follows. First, assume that DLLSO divides the whole swarm into M levels according to the particles' quality; second, in order to select exemplars for the ith particle, two levels of $rl1$ and $rl2$ are randomly selected, which are prior than the level that the ith particle in. Furthermore, $rl1$ level is prior than $rl2$ level.

4.1.1. Exploration

To investigate the exploration ability of PSO-DBCD, we rewrite (10) as follows.

$$v_i^d(t+1) \leftarrow \omega v_i^d(t) + \theta_1(p_1^d(t) - p_i^d(t)) \tag{13}$$

$$\theta_1 = r_1 + \phi r_2 \tag{14}$$

$$p_1^d(t) = \frac{r_1}{r_1 + \phi r_2} p_{i,ec}^d(t) + \frac{\phi r_2}{r_1 + \phi r_2} p_{i,ed}^d(t). \tag{15}$$

Similarly, the update strategies of cPSO (2) and DLLSO (12) are rewritten as (16) and (19), respectively.

$$v_i^d(t+1) \leftarrow \omega v_i^d(t) + \theta_2(p_2^d(t) - p_i^d(t)) \tag{16}$$

$$\theta_2 = c_1 r_1 + c_2 r_2 \tag{17}$$

$$p_2^d(t) = \frac{c_1 r_1}{c_1 r_1 + c_2 r_2} pbest_i^d(t) + \frac{c_2 r_2}{c_1 r_1 + c_2 r_2} gbest^d(t) \tag{18}$$

$$v_i^d(t+1) \leftarrow \omega v_i^d(t) + \theta_3(p_3^d(t) - p_i^d(t)) \tag{19}$$

$$\theta_3 = r_1 + \phi r_2 \tag{20}$$

$$p_3^d(t) = \frac{r_1}{r_1 + \phi r_2} p_{i,rl1}^d(t) + \frac{\phi r_2}{r_1 + \phi r_2} p_{i,rl2}^d(t). \tag{21}$$

The main source of diversity is given by the difference between $p_i (i = 1, 2, 3)$ and the particle to be updated, according to (13), (16), and (19). In comparison to cPSO, PSO-DBCD can offer

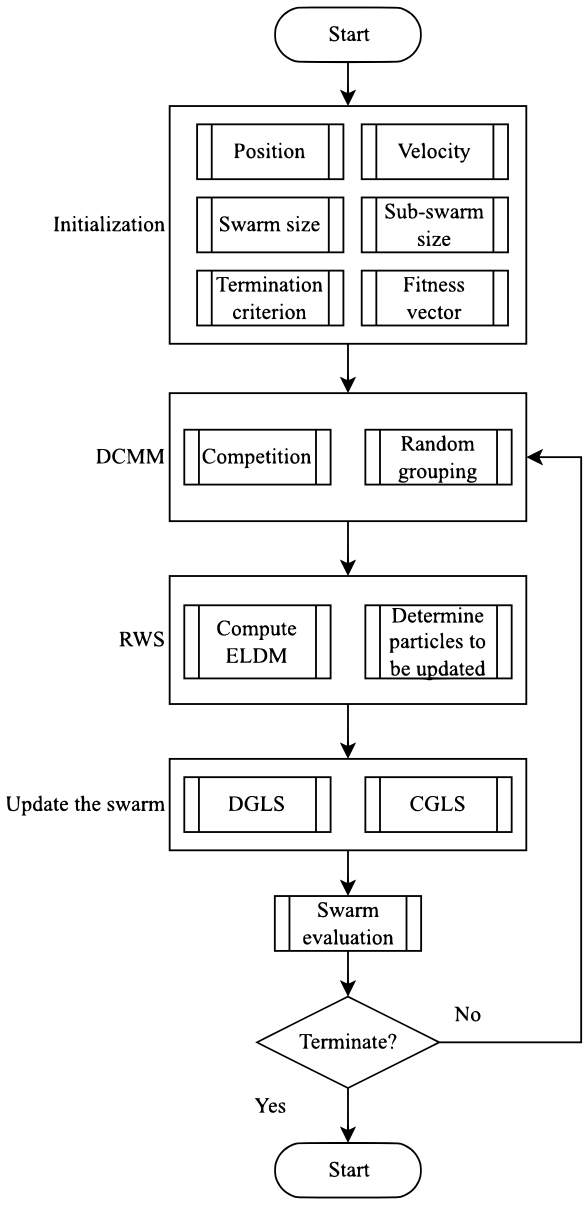


Fig. 5. The flowchart of PSO-DBCD.

higher diversity on p_i because *pbest* and *gbest* may not vary over a long period of time. On the other hand, PSO-DBCD also has advantages over DLLSO. The two reasons are that (i) in PSO-DBCD, all particles, aside from the best and worst individuals, can be chosen as the second part of p_1, whereas in DLLSO, the particles in the first and last levels cannot be chosen as the second part of p_3; (ii) the second part of p_1 in PSO-DBCD can direct particles moving to sparse areas.

In conclusion, it is clear that PSO-DBCD performs better at enhancing exemplars' diversity than cPSO and DLLSO, suggesting a stronger capacity for exploration.

4.1.2. Exploitation

Exploitation is important to meta-heuristics, which can be indicated by the difference between the convergence exemplars and the updated particles [10].

Assume that p_{ec} is the convergence exemplar for the ith particle p_i. We have

$$f(p_{ec}) \leq f(p_i). \tag{22}$$

Then we have

$$\begin{aligned} \Delta F_{PSO-DBCD} &= |f(p_{i,ec}) - f(p_i)| \\ &= |f(\bar{p}_1) - f(p_i)|, \end{aligned} \tag{23}$$

where $\Delta F_{PSO-DBCD}$ is the fitness difference between the ith particle and its convergence exemplar; $\bar{p}_1$ is the expectation of p_1 with $\phi = 0$ defined in (15). Similarly, for cPSO and DLLSO, we have

$$\begin{aligned} \Delta F_{cPSO} &= |f(gbest) - f(p_i)| \\ &= |f(\bar{p}_2) - f(p_i)| \end{aligned} \tag{24}$$

$$\begin{aligned} \Delta F_{DLLSO} &= |f(p_{i,rl1}) - f(p_i)| \\ &= |f(\bar{p}_3) - f(p_i)|, \end{aligned} \tag{25}$$

where ΔF_{cPSO} and ΔF_{DLLSO} are the fitness differences between the ith particle and the convergence exemplar in cPSO and DLLSO, respectively; *gbest* and $p_{i,rl1}$ are defined in (2) and (12); $\bar{p}_2$ is the expectation of p_2 defined in (18); $\bar{p}_3$ is the expectation of p_3 with $\phi = 0$ defined in (21).

According to the characteristics of p_1, p_2 and p_3:

$$f(\bar{p}_2) \leq f(\bar{p}_3) \leq f(\bar{p}_1). \tag{26}$$

Then we have

$$\Delta F_{PSO-DBCD} \leq \Delta F_{DLLSO} \leq \Delta F_{cPSO}. \tag{27}$$

Based on (27), compared with cPSO and DLLSO, one can find that PSO-DBCD shows better exploitation ability to refine solutions within a smaller gap between two positions.

Additionally, PSO-DBCD should outperform DLLSO in terms of convergence in the latter stage: with the increase of the sub-swarm size, (i) the convergence pressure in PSO-DBCD will be leveraged; (ii) more promising individuals in PSO-DBCD can be updated, which is advantageous for convergence. On the contrary, if DLLSO tries to leverage its convergence pressure, more promising individuals cannot be updated.

4.2. Convergence analysis

For the convergence stability analysis of PSO-DBCD, as the convergence and diversity exemplars are randomly selected, the method proposed by [59] is adopted to investigate the convergence stability of $E(p(t))$. Here, $E(p(t))$ denotes the expectation of an arbitrary particle in PSO-DBCD at generation t. The details of the convergence proof are presented as follows.

Since the position of particles is independently updated for each dimension, the 1−D space convergence stability analysis is presented as follows [59]. For simplicity, the updating of a particle at generation t can be rewritten as

$$v(t+1) = \omega(p(t) - p(t-1)) + \phi r_1(e_1 - p(t)) + r_2(e_2 - p(t)) \tag{28}$$

$$p(t+1) = p(t) + v(t+1). \tag{29}$$

Based on (28) and (29), $p(t+1)$ can be represented by (30), where $l = 1 + \omega - \phi r_1 - r_2$.

$$p(t+1) = lp(t) - \omega p(t-1) + \phi r_1 e_1 + r_2 e_2. \tag{30}$$

Then the expectation of $p(t+1)$ is shown in (31), where μ_ω, μ_{r_1}, μ_{r_2}, μ_{e_1} and μ_{e_2} are the expectation values of the corresponding variables.

$$E(p(t+1)) = E(l)E(p(t)) - \mu_\omega E(p(t-1)) + \phi\mu_{r_1}\mu_{e_1} + \mu_{r_2}\mu_{e_2}. \tag{31}$$

Based on (31), a recurrence relation can be obtained as shown in (32).

$$\begin{bmatrix} E(p(t+1)) \\ E(p(t)) \end{bmatrix} = \begin{bmatrix} E(l) & -\mu_\omega \\ 1 & 0 \end{bmatrix} \begin{bmatrix} E(p(t)) \\ E(p(t-1)) \end{bmatrix} + \begin{bmatrix} \phi\mu_{r_1}\mu_{e_1} + \mu_{r_2}\mu_{e_2} \\ 0 \end{bmatrix}. \tag{32}$$

For simplicity, we define M as

$$M = \begin{bmatrix} E(l) & -\mu_\omega \\ 1 & 0 \end{bmatrix}. \tag{33}$$

According to [59], the necessary and sufficient condition to ensure the convergence of $E(p(t))$ is that the magnitude of the eigenvalues of M, $|\lambda_1, \lambda_2| = |(E(l) \pm \sqrt{E(l)^2 - 4\mu_\omega})/2|$, should be smaller than 1. Since ω, r_1, and r_2 are randomly generated within (0, 1) in the proposed algorithm, μ_ω, μ_{r_1} and μ_{r_2} are 1/2. Consequently, the necessary and sufficient condition for convergence of $E(p(t))$ of an arbitrary particle are

$$\left|\frac{2 - \phi \pm \sqrt{\phi^2 - 4\phi - 4}}{4}\right| < 1. \tag{34}$$

The analysis of (34) is presented as follows.

1. If the eigenvalues of (34) are complex roots.

Then

$$\phi^2 - 4\phi - 4 < 0. \tag{35}$$

Then

$$2 - 2\sqrt{2} < \phi < 2 + 2\sqrt{2}. \tag{36}$$

Given that

$$|2 - \phi \pm \sqrt{\phi^2 - 4\phi - 4}| = 2\sqrt{2}. \tag{37}$$

Therefore, for $\forall\phi$,

$$\left|\frac{2 - \phi \pm \sqrt{\phi^2 - 4\phi - 4}}{4}\right| < 1. \tag{38}$$

With considering (36), the feasible domain of ϕ under this scenario is shown in (39).

$$2 - 2\sqrt{2} < \phi < 2 + 2\sqrt{2} \tag{39}$$

2. If the eigenvalues of (34) are real roots.

Then

$$\phi^2 - 4\phi - 4 \geqslant 0. \tag{40}$$

Then

$$\phi \geqslant 2 + 2\sqrt{2} \quad or \quad \phi \leqslant 2 - 2\sqrt{2}. \tag{41}$$

Then for $\left|\frac{2-\phi\pm\sqrt{\phi^2-4\phi-4}}{4}\right| < 1$, (42) should be ensured.

$$-4 < 2 - \phi \pm \sqrt{\phi^2 - 4\phi - 4} < 4. \tag{42}$$

For (42), the following four situations should be considered.

Then

$$-1 < \phi < 5. \tag{43}$$

Consequently, the feasible domain of ϕ that guarantees the convergence stability of $E(p(t))$ can be derived from (39)∪[(43)∩(41)], which is

$$-1 < \phi < 5. \tag{44}$$

In conclusion, only $-1 < \phi < 5$ can ensure the convergence stability of PSO-DBCD. However, it should be noted that the above proof does not necessarily ensure PSO-DBCD converges to the global optimum.

4.3. The computational complexity analysis

Investigating the computational complexity of PSO-DBCD takes into account the calculation cost every generation and the necessary memory cost [10]. It is expected that cPSO and PSO-DBCD employ the same swarm size N as the quantity of fitness evaluations (FEs) is commonly adopted as the deciding factor.

First, the extra computational cost per generation introduced by PSO-DBCD is caused by the additional operations of Algorithms 1 and 2. More specifically, Algorithm 1 brings $O(6N - \frac{2N}{s_i})$ additional computational cost (s_i is the sub-swarm size at generation t): $O(N)$ is costed for grouping swarm; $O(2N)$ is costed by lines 4 to 11; $O(N)$ is costed by the random competition in line 12 and $O(2(N - \frac{N}{s_i}))$ is costed by lines 13 to 18. In Algorithm 2, $O(N)$ is needed to compute the EDLM and $O(3N)$ is costed by lines 4 to 10, leading to $O(4N)$. Since only $|P_{uc} \cap P_{ud}|$ particles will be updated per generation, the additional computational cost at each generation in PSO-DBCD is $O(9N - \frac{2N}{s_i} + |P_{uc} \cap P_{ud}|)$ which is linearly related to N. Second, since the *gbest* and *pbest* are not needed in PSO-DBCD, PSO-DBCD outperforms cPSO in memory cost.

5. Experimental study

In this section, comprehensive experiments are carried out to test PSO-DBCD. First, PSO-DBCD is compared to seven peer large-scale optimization algorithms with the CEC 2013 large-scale benchmarks [60]; second, experiments are conducted to validate the suggested convergence and diversity learning strategies; third, a parameter analysis is presented; and finally, the CEC 2013 low-dimensional benchmarks, which are numbered from F_{16} to F_{43} in the experiments, are adopted to test the scalability of PSO-DBCD. All the experiments are conducted on a PC with an Intel Core i7-9700k CPU and a Microsoft Windows 10 Enterprise 64-bit operating system. The algorithms are written and run in Matlab 2018a. Each algorithm is applied to each benchmark 30 times.

For saving computational resources, the experiments for validating the convergence and diversity learning strategies and the parameter analysis are conducted based on six functions including F_1, F_3, F_6, F_{11}, F_{14}, F_{15}, which cover the unimodal functions (F_1, F_{11}, F_{14}, F_{15}), the multimodal functions (F_3, F_6), the fully separable functions (F_1, F_3), the partial separable functions (F_6, F_{11}), an overlapping function (F_{14}) and a nonseparable function (F_{15}).

5.1. Empirical comparisons

5.1.1. Compared algorithms

In the numerical comparisons, CSO [14], SLPSO [30], DLLSO [10], APSO-DEE [36], MA-SW-CHAINS [45], DECC-DG2 [56], MMO-CC [25] are selected as the comparison algorithms, where CSO, SLPSO, DLLSO and DEGLSO are cPSO's variants; MA-SW-CHAINS is the winner of the LSOPs competition in CEC 2013; DECC-DG2 and MMO-CC are two popular CC framework based methods.

5.1.2. Experimental settings

The maximum FEs is set to $3E+06$ as suggested by [10,14,25] in order to provide a fair comparison. For the parameter settings, the default settings in the corresponding references are adopted for the seven peer algorithms; for PSO-DBCD, the swarm size is set to 1000 while ϕ is set to 0.2 according to the parameter analysis in Section 5.5; the sub-swarm size set S_{sw} is set to {2, 4, 8, 10, 20, 25, 40, 50}. For the statistical test, suppose that $\mathcal{F}_k(\mathcal{A}_j) = \{f_k^1(\mathcal{A}_j), f_k^2(\mathcal{A}_j), \ldots, f_k^n(\mathcal{A}_j)\}$ denotes the results of n independent runs with algorithm $\mathcal{A}_j$ on the kth benchmark ($n = 30$ in the experiments; $k \in [1, 15]$ for large-scale optimization

Table 1
The rules for selecting the statistical test methods.

Cases	Test method
$\mathcal{P}\{\mathcal{F}_k(PSO\text{-}DBCD)\} \sim \mathcal{N}(\mu, \sigma)$ and $\mathcal{P}\{\mathcal{F}_k(\mathcal{A}_j)\} \sim \mathcal{N}(\mu, \sigma)$	t-test
Otherwise	Wilcoxon rank sum test

and $k \in [16, 43]$ for low-dimensional optimization); $\mathcal{P}(\cdot)$ is used to evaluate the distribution and $N(\mu, \sigma)$ is normal distribution. The statistical test between PSO-DBCD and the jth peer algorithm is performed using the t-test if both $\mathcal{P}\{PSO\text{-}DBCD\}$ and $\mathcal{P}\{\mathcal{A}_j\}$ follow normal distribution as shown by Table 1. Otherwise, the Wilcoxon rank sum test is adopted. Furthermore, the Friedman ranking and Quade ranking are also employed in the comparisons. The significance level in all statistical tests is set to 0.05.

5.1.3. Results

Table 3 and Fig. 6 show the results of 30 independent runs of the algorithms. In the table, the best results of the average performance are highlighted by a gray background; the p-values will be marked in bold font if PSO-DBCD performs significantly better than the corresponding algorithms; w/l/t at the bottom of the table indicates how many times PSO-DBCD wins/loses/ties in the competitions.

As shown in Fig. 6, PSO-DBCD shows competitive performance in comparison to the peer algorithms. From the results in Table 3 PSO-DBCD wins 9 times on the average optimization results out of the 15 functions and ranks in second on F_1, F_4 and F_5. With a deep insight, PSO-DBCD outperforms the peer algorithms by 4 times out of the 8 multi-modal functions of F_2, F_3, F_5, F_6, F_7, F_9, F_{10}, F_{12}; PSO-DBCD wins 5 times on the 7 unimodal functions; for the competition on the 9 partially separable benchmarks of F_4 to F_{12}, PSO-DBCD wins 6 times; PSO-DBCD wins on all the three overlapping functions of F_{12} to F_{14}; PSO-DBCD also outperforms the peer algorithms on the fully non-separable function of F_{15}.

The Shapiro–Wilk test is used to first determine normality in preparation for the statistical test. The results are shown in Table 2, where p-value >0.05 indicates that the corresponding results follow the normal distribution. The statistical analysis is then carried out according to the rules in Table 1. The results are shown in Table 3, where t-tests are marked by "*" and Wilcoxon rank sum tests are not marked. According to the results, one can find that PSO-DBCD outperforms CSO, SLPSO, DLLSO, and AP-SODEE on 12, 14, 10, and 11 benchmark functions, respectively; PSO-DBCD outperforms DECC-DG2, MMO-CC, and MASWCHAINS for 13, 13, and 11 times, respectively. Furthermore, the Friedman ranking and the Quade ranking analysis results also demonstrate that the suggested algorithm performs competitively.

Furthermore, the proposed method is compared with two promising hybrid algorithms of jDEsps [61] and MOS2013 [46] in terms of average results. Out of the 15 benchmarks, PSO-DBCD outperforms jDEsps by 8 times, while it only outperforms MOS2013 by 3 times. However, it is worth noting that PSO-DBCD is of better usability in comparison to MOS2013 for the following reasons: (i) The proposed algorithm inherits the updating structure of cPSO, leading to low computational cost and easy implementation. However, MOS2013 incorporates different algorithms, resulting in higher computational cost; (ii) In PSO-DBCD, only two parameters need to be adjusted, i.e., the population size and the parameter for controlling the diversity, and the functions of these parameters are explicit. However, except for the population size, MOS2013 needs to adjust the parameters in each adopted algorithm; and (iii) MOS2013 is ineffective in partially separable multimodal problems [62].

Fig. 7 illustrates the convergence curves of the compared algorithms. The results show that PSO-DBCD performs competitively, especially in the middle and latter optimization stages. This can be explained by that PSO-DBCD prioritizes exploration in the early stage and concentrates on exploitation in the latter stage with the increase of the sub-swarm size.

The experimental findings and analyses shown above demonstrate the competitive performance of PSO-DBCD on LSOPs. The main reason is that, thanks to the proposed convergence pressure and diversity adjustments, PSO-DBCD is able to dynamically balance the convergence and the diversity during the optimization process.

5.2. The impacts of s on convergence

To show the effects of s on the convergence, Fig. 8 exhibits the convergence curves of PSO-DBCD obtained with different settings of $s = \{2, 10, 20\}$, where the swarm size N and ϕ are set to 1000 and 0.2, respectively. The maximum FEs is set to 3E+06.

With the exception of the results on F_3, the results show that the convergence speed of PSO-DBCD increases as the growing of s. This can be explained by that the dynamic sub-swarm size modification allows for an effective adjustment of the convergence pressure in PSO-DBCD during the optimization process: as s increases, (i) CGLS tends to choose more promising individuals as the convergence exemplars, raising convergence pressure; (ii) the convergence exemplars can be exploited by more particles.

5.3. The impacts of ϕ on swarm diversity

In this section, experiments are conducted to investigate the impacts of ϕ on diversity preservation, where ϕ varies from 0.3 to 0.6 with step of 0.1 and the swarm size N is set 1000. The diversity is computed according to (45) and (46), where p_i^d is the dth dimension of the ith particle, $\bar{p}^d$ is the expectation of the dth dimension of all the particles; $Std(P)$ is the swarm diversity and N is the swarm size. The results are shown in Fig. 9. Obviously, larger ϕ results in higher swarm diversity, indicating the effectiveness of DGLS in diversity preservation. Consequently, one can adjust ϕ to manage the swarm diversity to balance convergence and diversity.

$$Std(P) = \frac{1}{N}\sum_{i=1}^{N}\sqrt{\sum_{d=1}^{D}(p_i^d - \bar{p}^d)^2} \tag{45}$$

$$\bar{p}^d = \frac{1}{N}\sum_{i=1}^{N} x_i^d. \tag{46}$$

5.4. A comparison between DLLSO and PSO-DBCD

Comparisons between DLLSO and PSO-DBCD on convergence and diversity are provided as a complement to the theoretical analysis of searching behavior. The results on F_{11} (Partial separable function), F_{13} (Overlapping function) and F_{15} (Nonseparable function) are selected as examples, which are shown in Fig. 10.

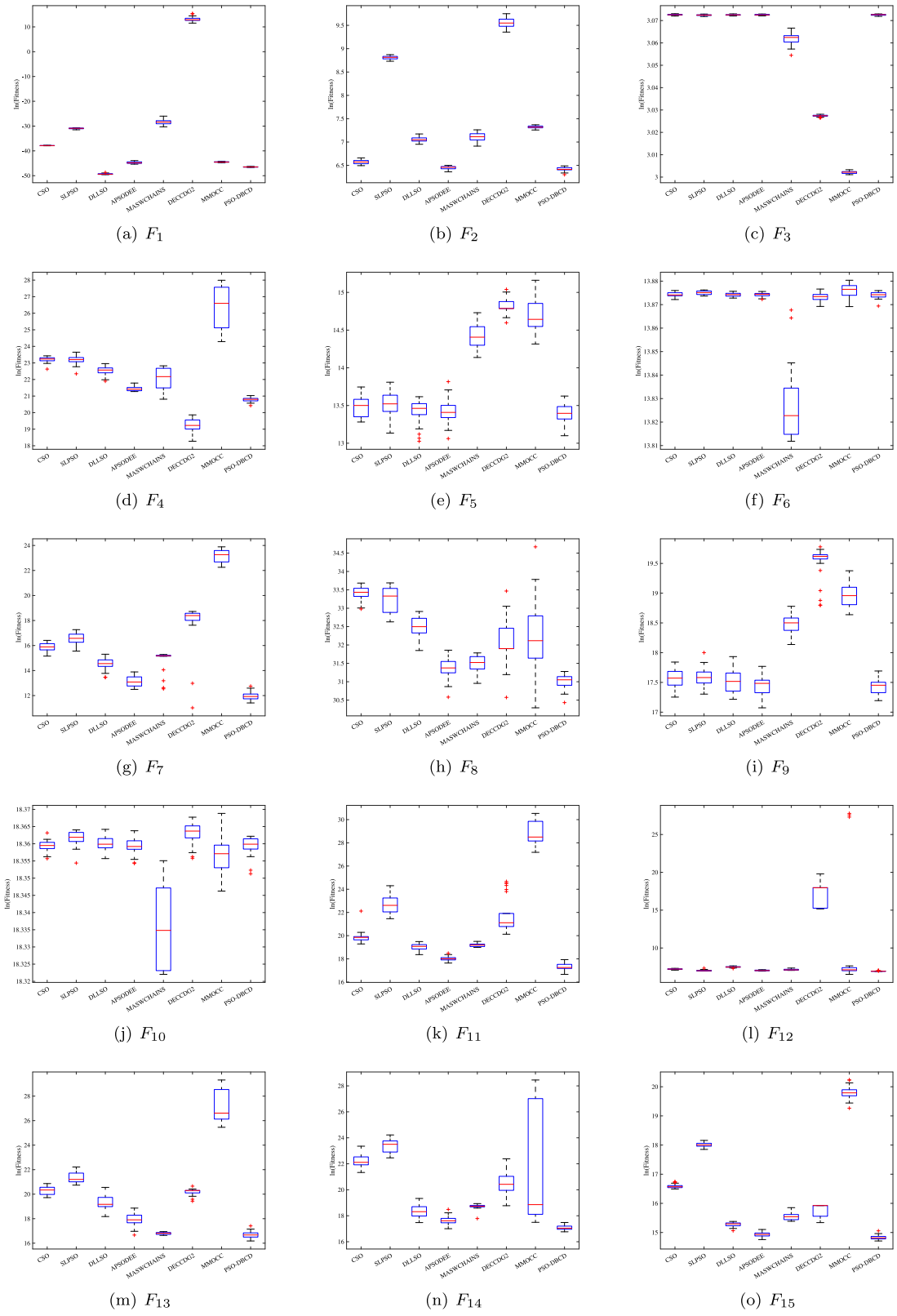
One can observe that (i) PSO-DBCD outperforms DLLSO on diversity in the early optimization stage on F_{11} and F_{15}; (ii) PSO-DBCD outperforms DLLSO on convergence in all cases, especially in the later stage. These results show that (i) PSO-DBCD has a better capacity for diversity preservation than DLLSO, especially in the early optimization stage. This is consistent with the analysis in Section 4.1.1; (ii) PSO-DBCD outperforms DLLSO in convergence, which is consistent with the discussions in Section 4.1.2. In conclusion, PSO-DBCD shows competitive performance in balancing convergence and diversity with the proposed learning strategies.

Table 2
p-values of the Shapiro–Wilk test results on CEC 2013 large-scale benchmarks.

Function	PSO-DBCD	CSO	SLPSO	DLLSO	MASWCHAINs	DECCDG2	MMOCC	APSODEE
F_1	3.34E−01*	4.68E−02	6.74E−01*	2.96E−09	1.23E−09	2.73E−07	8.41E−01*	3.83E−03
F_2	1.52E−01*	4.36E−01*	6.20E−01*	1.61E−02	8.18E−01*	7.77E−01*	9.04E−01*	2.06E−01*
F_3	1.52E−01*	3.55E−02	5.86E−01*	1.76E−01*	8.38E−02*	1.05E−01*	8.30E−01*	8.60E−01*
F_4	9.80E−01*	8.68E−02*	6.71E−01*	6.34E−01*	2.90E−06	6.12E−02*	8.40E−04	4.96E−03
F_5	9.22E−01*	5.27E−01*	9.28E−01*	1.33E−01*	4.47E−01*	2.20E−02	4.35E−02	9.53E−02*
F_6	2.08E−02	2.99E−01*	1.34E−01*	4.56E−03	5.25E−06	5.02E−03	5.26E−02	1.54E−01*
F_7	3.71E−03	3.60E−01*	1.67E−01*	5.54E−03	3.67E−12	1.10E−02	3.19E−01*	2.91E−02
F_8	2.13E−01*	3.93E−03	7.47E−02*	3.59E−01*	3.22E−02*	3.90E−05	7.54E−08	6.10E−01*
F_9	8.02E−01*	1.39E−01*	2.06E−01*	2.37E−02	9.66E−02*	2.02E−05	2.96E−01*	8.53E−01*
F_{10}	7.11E−04	3.90E−01*	3.32E−03	2.10E−03	4.09E−05	8.20E−02*	8.30E−01*	2.74E−01*
F_{11}	9.86E−02*	3.86E−15	5.50E−05	5.15E−01*	1.31E−02*	3.46E−07	2.52E−04	2.16E−01*
F_{12}	2.36E−04	9.65E−01*	5.08E−05	8.14E−01*	7.95E−02*	1.42E−07	4.53E−09	2.85E−01*
F_{13}	2.71E−02	4.93E−06	5.35E−04	1.99E−04	9.49E−02*	2.54E−03	1.42E−04	2.59E−01*
F_{14}	3.00E−01*	7.18E−06	6.77E−02*	1.98E−07	1.61E−03	1.14E−05	1.37E−05	2.43E−03
F_{15}	1.38E−02	1.16E−02	7.78E−01*	4.23E−01*	2.14E−03	3.61E−05	1.78E−01*	6.82E−01*

Table 3
The experimental results of 1000-dimensional IEEE CEC 2013 benchmark functions with fitness evaluations of 3e6. The best average result on each benchmark is highlighted by a gray background.

Function	Quality	CSO	SLPSO	DLLSO	MASWCHAINS	DECCDG2	MMOCC	APSODEE	CLSO
F_1	Mean	3.68E−17	3.70E−14	4.32E−22	8.49E−13	8.65E+05	4.82E−20	4.56E−20	6.52E−21
	Std	3.89E−19	1.44E−15	3.88E−23	2.18E−13	2.27E+05	1.30E−21	3.56E−21	1.58E−22
	p-value	**1.42E−09**	**2.79E−29***	**1.42E−09**	**1.42E−09**	**1.41E−09**	**1.35E−33***	**1.41E−09**	–
F_2	Mean	7.07E+02	6.70E+03	1.15E+03	1.22E+03	1.41E+04	1.51E+03	6.31E+02	6.14E+02
	Std	7.17E+00	4.98E+01	1.31E+01	2.28E+01	3.02E+02	8.43E+00	5.66E+00	5.75E+00
	p-value	**1.34E−15***	**5.57E−61***	**1.42E−09**	**2.26E−29***	**2.77E−40***	**2.59E−54***	**2.53E−02***	–
F_3	Mean	2.16E+01	2.16E+01	2.16E+01	2.14E+01	2.06E+01	2.01E+01	2.16E+01	2.16E+01
	Std	1.39E−03	1.14E−03	1.23E−03	1.12E−02	1.69E−03	2.36E−03	6.87E−04	1.24E−03
	p-value	8.46E−01	1.52E−01*	5.47E−01*	6.03E−25*	1.82E−88*	1.71E−92*	7.65E−01*	–
F_4	Mean	1.14E+10	1.20E+10	5.99E+09	4.58E+09	2.51E+08	5.15E+11	2.11E+09	1.08E+09
	Std	2.65E+08	5.54E+08	2.98E+08	4.91E+08	1.89E+07	9.71E+10	5.14E+07	2.67E+07
	p-value	**1.78E−33***	**3.06E−24***	**3.69E−23***	**1.04E−08**	6.40E−29*	**1.42E−09**	**4.41E−02**	–
F_5	Mean	7.38E+05	7.58E+05	7.30E+05	1.87E+06	2.74E+06	2.42E+06	6.48E+05	6.65E+05
	Std	2.64E+04	2.14E+04	1.98E+04	6.13E+04	5.66E+04	1.14E+05	2.33E+04	1.75E+04
	p-value	**1.53E−02***	**1.75E−03***	6.23E−01*	**1.42E−23***	**1.16E−09**	**1.42E−09**	4.23E−01*	–
F_6	Mean	1.06E+06	1.06E+06	1.06E+06	1.01E+06	1.06E+06	1.06E+06	1.06E+06	1.06E+06
	Std	1.91E+02	1.64E+02	2.31E+02	3.06E+03	4.50E+02	6.41E+02	1.91E+02	2.99E+02
	p-value	7.42E−01	**2.63E−03**	8.46E−01	1.42E−09	**2.65E−02**	**2.63E−03**	**2.35E−02**	–
F_7	Mean	8.16E+06	1.73E+07	1.74E+06	3.45E+06	8.93E+07	1.28E+10	5.49E+05	1.67E+05
	Std	4.77E+05	1.49E+06	1.67E+05	2.53E+05	7.16E+06	1.07E+09	3.99E+04	1.24E+04
	p-value	**1.42E−09**	**1.42E−09**	**1.42E−09**	**2.29E−09**	**2.45E−08**	**1.42E−09**	**4.14E−03**	–
F_8	Mean	3.15E+14	2.89E+14	1.17E+14	4.85E+13	1.01E+14	1.54E+14	4.22E+13	2.94E+13
	Std	1.11E+13	1.75E+13	7.87E+12	2.03E+12	1.31E+13	4.45E+13	2.70E+12	1.07E+12
	p-value	**1.42E−09**	**3.45E−19***	**1.25E−20***	**1.29E−10***	**2.69E−08**	**6.96E−05**	**6.66E−07***	–
F_9	Mean	4.42E+07	4.44E+07	4.32E+07	1.07E+08	3.08E+08	1.76E+08	3.80E+07	3.75E+07
	Std	1.56E+06	1.47E+06	1.28E+06	3.36E+06	1.39E+07	7.03E+06	1.35E+06	9.53E+05
	p-value	**2.19E−03***	**3.11E−04***	5.98E−02	**1.73E−24***	**1.40E−09**	**3.96E−24***	7.52E−01*	–
F_{10}	Mean	9.40E+07	9.43E+07	9.40E+07	9.18E+07	9.44E+07	9.38E+07	9.40E+07	9.40E+07
	Std	4.23E+04	3.99E+04	4.63E+04	2.12E+05	5.82E+04	1.02E+05	2.40E+04	5.08E+04
	p-value	4.04E−01	**1.19E−03**	6.14E−01	2.90E−09	**6.42E−05**	2.70E−02	**1.22E−03**	–
F_{11}	Mean	3.56E+08	9.98E+09	1.82E+08	2.19E+08	9.93E+09	5.66E+12	6.85E+07	3.39E+07
	Std	1.47E+07	1.82E+09	9.42E+06	5.96E+06	3.26E+09	1.09E+12	2.80E+06	1.96E+06
	p-value	**1.42E−09**	**1.42E−09**	**3.50E−19***	**1.42E−09**	**1.42E−09**	**1.42E−09**	**3.22E−13***	–
F_{12}	Mean	1.38E+03	1.13E+03	1.78E+03	1.25E+03	5.81E+07	1.14E+11	1.12E+03	1.03E+03
	Std	2.40E+01	2.12E+01	3.06E+01	2.11E+01	1.53E+07	6.32E+10	1.24E+02	7.40E+00
	p-value	**1.42E−09**	**1.49E−06**	**1.42E−09**	**1.42E−09**	**1.05E−09**	**1.06E−05**	1.42E−01	–
F_{13}	Mean	8.05E+08	2.05E+09	2.89E+08	1.98E+07	6.03E+08	1.32E+12	6.65E+07	1.85E+07
	Std	6.56E+07	2.13E+08	2.41E+07	3.64E+05	2.69E+07	2.88E+11	6.11E+06	1.12E+06
	p-value	**1.42E−09**	**1.42E−09**	**1.42E−09**	5.47E−02	**1.09E−09**	**1.42E−09**	**1.51E−09**	–
F_{14}	Mean	6.95E+09	1.60E+10	9.21E+07	1.36E+08	1.11E+09	4.12E+11	4.92E+07	2.64E+07
	Std	9.23E+08	1.62E+09	1.39E+07	4.22E+06	2.10E+08	1.21E+11	3.74E+06	9.16E+05
	p-value	**1.42E−09**	**7.07E−13***	**1.60E−09**	**1.42E−09**	**1.42E−09**	**1.42E−09**	**3.42E−06**	–
F_{15}	Mean	1.65E+07	6.68E+07	4.25E+06	5.71E+06	7.11E+06	4.05E+08	3.13E+06	2.75E+06
	Std	2.20E+05	1.01E+06	5.52E+04	1.51E+05	2.70E+05	1.91E+07	5.01E+04	4.50E+04
	p-value	**1.42E−09**	**1.42E−09**	**1.42E−09**	**1.42E−09**	**9.30E−10**	**1.42E−09**	**1.41E−09**	–
w/l/t		12/0/3	14/0/1	10/0/5	11/3/1	13/2/0	13/0/2	11/0/4	–
Friedman ranking		5.23	6.13	3.90	3.87	5.80	6.27	2.77	2.03
Quade ranking		5.59	6.30	3.91	3.74	5.48	7.05	2.43	1.49



(a) F_1 (b) F_2 (c) F_3

(d) F_4 (e) F_5 (f) F_6

(g) F_7 (h) F_8 (i) F_9

(j) F_{10} (k) F_{11} (l) F_{12}

(m) F_{13} (n) F_{14} (o) F_{15}

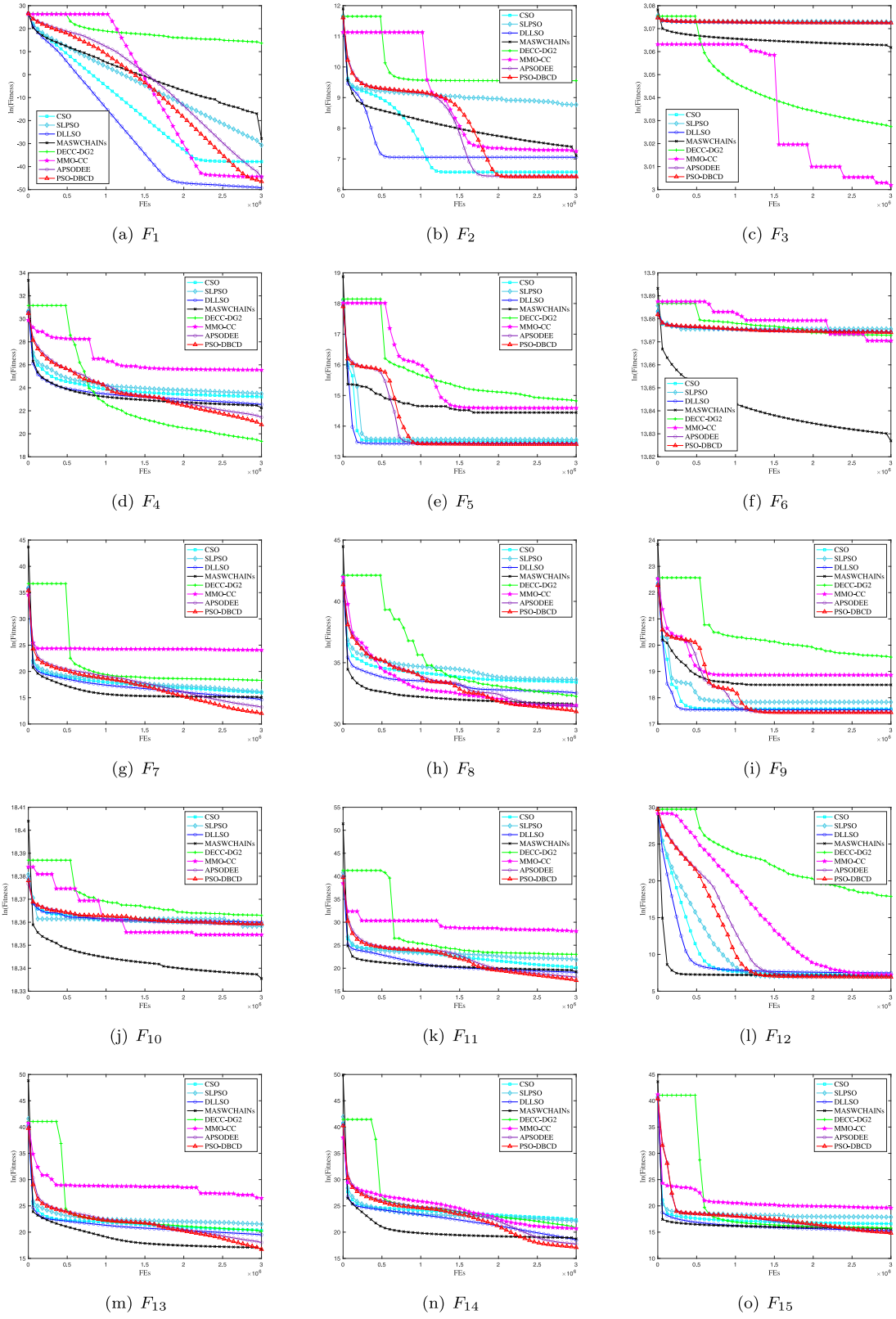
Fig. 6. The experimental results of 1000-dimensional IEEE CEC 2013 benchmark functions with fitness evaluations of 3e6. Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.

5.5. Parameter sensitivity analyses

PSO-DBCD only has two parameters of ϕ and N as the sub-swarm size is dynamically adjusted. Consequently, the parameter analyses on ϕ and N are presented as follows.

5.5.1. Parameter ϕ

The performance of PSO-DBCD with different settings of ϕ on the six chosen functions is shown in Fig. 11, where N is set to 1000. As illustrated in Fig. 11, the performance of PSO-DBCD is typically not ϕ-sensitive. For F_1, the convergence should



(a) F_1 (b) F_2 (c) F_3

(d) F_4 (e) F_5 (f) F_6

(g) F_7 (h) F_8 (i) F_9

(j) F_{10} (k) F_{11} (l) F_{12}

(m) F_{13} (n) F_{14} (o) F_{15}

Fig. 7. Convergence profiles of different algorithms obtained on the CEC 2013 benchmarks with 1000 dimensions. Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.

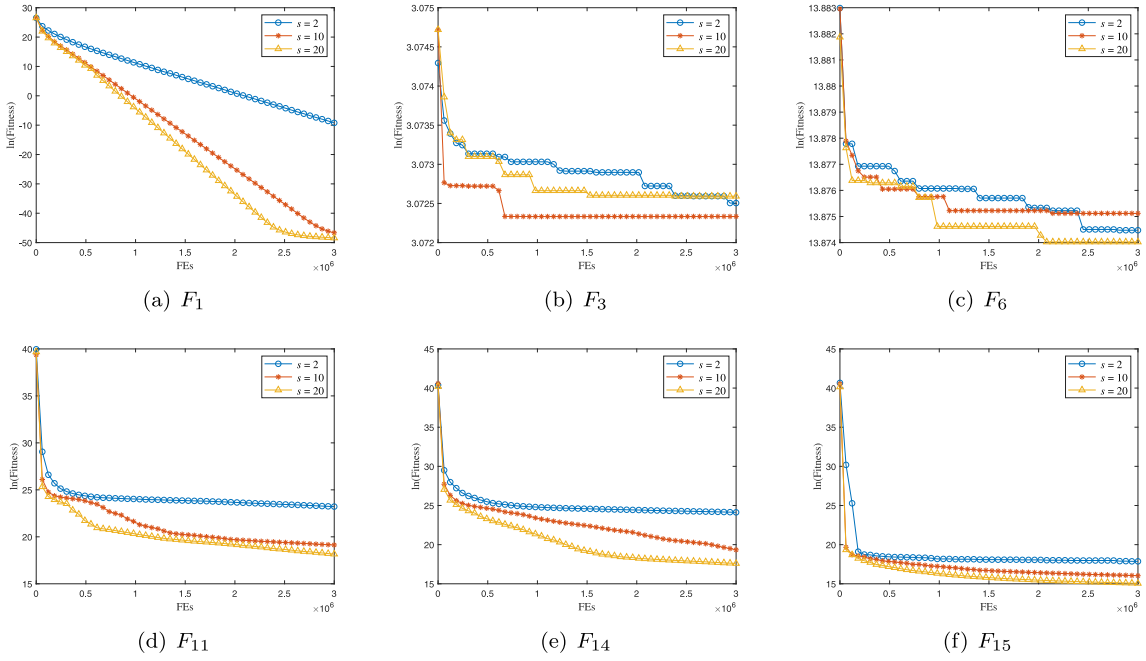


Fig. 8. Convergence comparisons on the selected six functions with respect to different settings of s. Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.

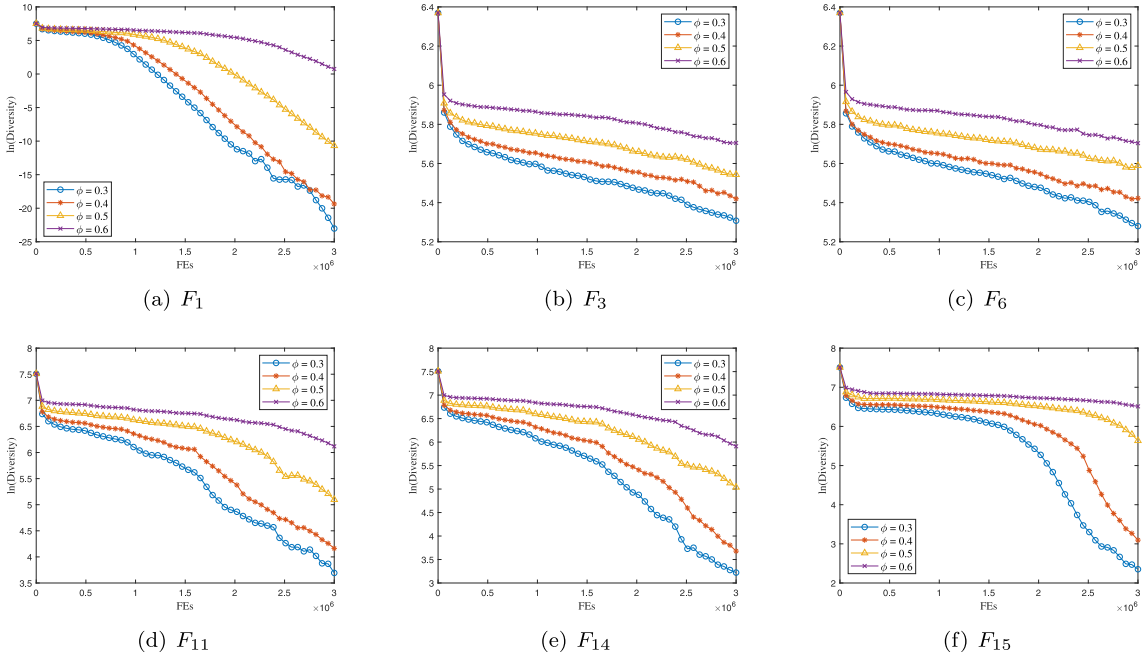
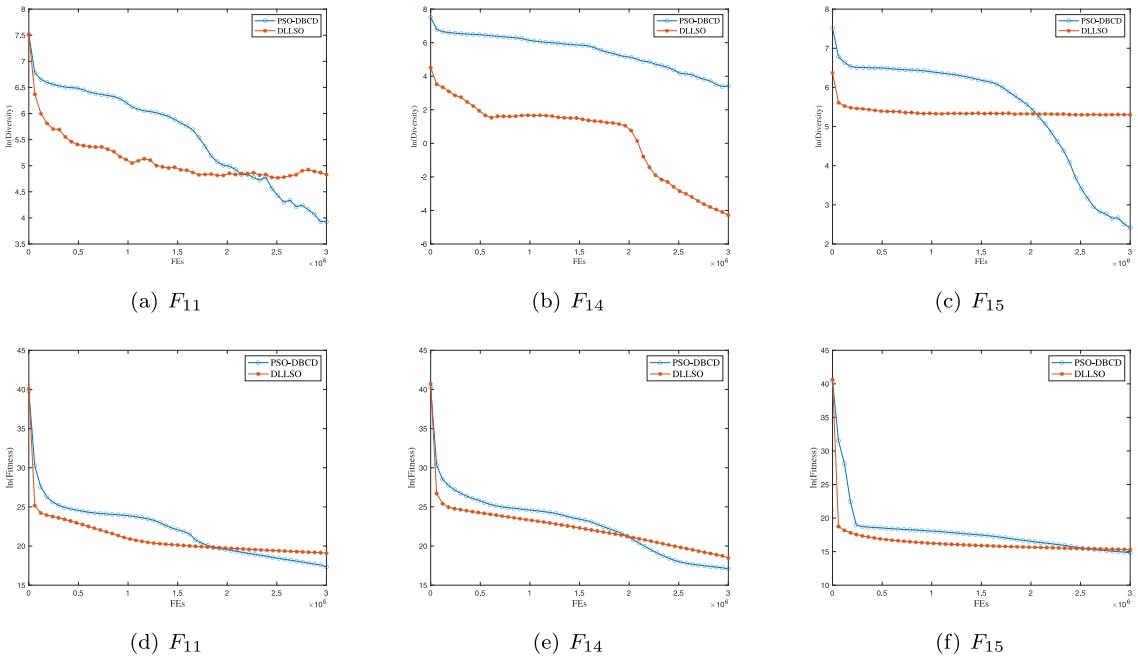


Fig. 9. Swarm diversity comparisons on the selected six functions with FEs of $3E+06$. Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.

be more important as F_1 is a simple convex function, thus, a small ϕ is expected. For the other five complex benchmarks, ϕ of $\{0.2, 0.3, 0.4\}$ performs competitive. Furthermore, the Friedman ranking test is executed and the results are shown in Table 4, which indicates the competitiveness of PSO-DBCD with ϕ of $\{0.2, 0.3, 0.4\}$. Based on these findings, $\phi = 0.2$ is suggested in this paper.

5.5.2. Parameter N

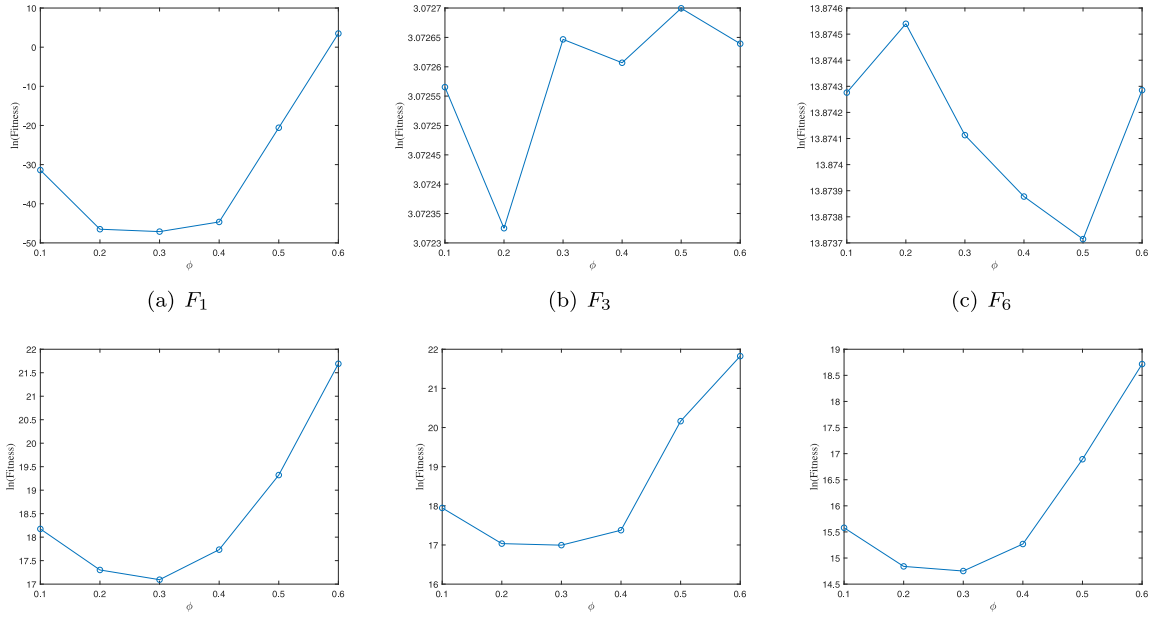
Fig. 12 shows the performance of PSO-DBCD on the six selected functions with different settings of N, where ϕ is set to 0.2. As shown in Fig. 12, except for the results on F_1, the performance of PSO-DBCD is not sensitive to N and the results obtained with $N = \{400, 600, 800, 1000\}$ is competitive. The reason is that large swarm size leads to high swarm diversity, which is not suitable for the simple convex function F_1. Furthermore, the Friedman ranking test is executed and the results are shown in Table 5, which indicates the competitiveness of the settings of $N = \{400, 600, 800, 1000\}$. In general, diversity preservation is crucial to algorithms on LSOPs. Therefore, $N = 1000$ is suggested in this paper.



(a) F_{11} (b) F_{14} (c) F_{15}

(d) F_{11} (e) F_{14} (f) F_{15}

Fig. 10. A comparison between DLLSO and PSO-DBCD with respect to diversity and convergence. Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.



(a) F_1 (b) F_3 (c) F_6

(d) F_{11} (e) F_{14} (f) F_{15}

Fig. 11. Average optimization results obtained with different settings of ϕ. Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.

Table 4
The Friedman ranking test results on ϕ.

ϕ	Friedman ranking
0.1	3.67
0.2	2.50
0.3	2.00
0.4	2.83
0.5	4.50
0.6	5.50

Table 5
The Friedman ranking test results on N.

N	Friedman ranking
400	2.67
600	1.67
800	3.17
1000	3.33
1200	4.17

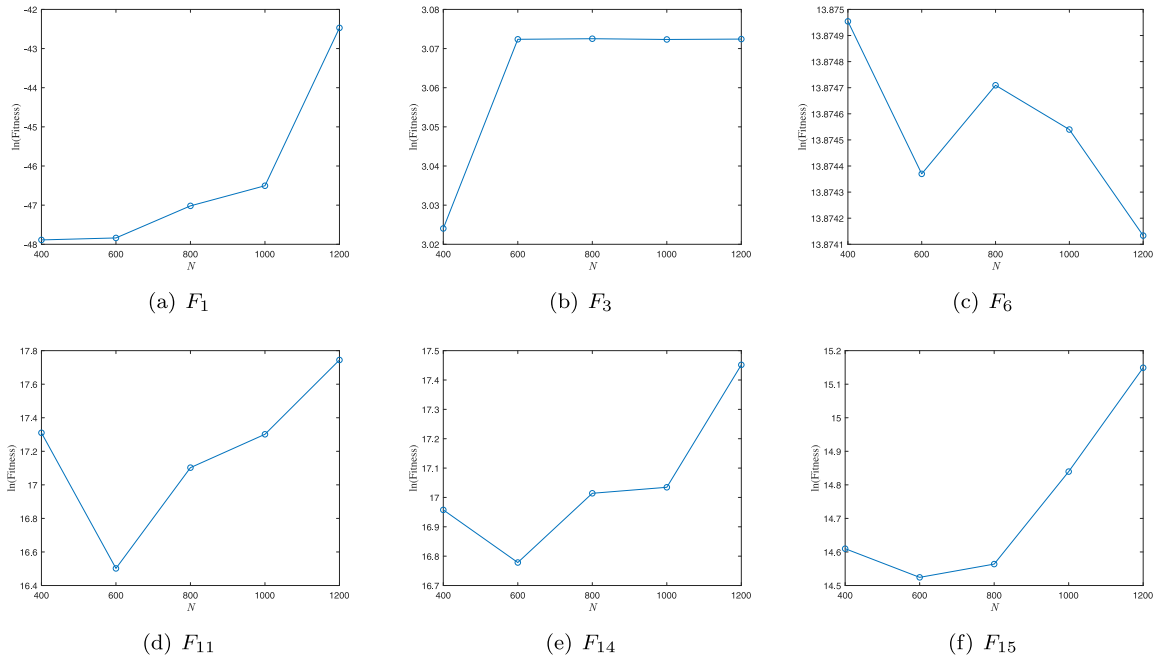


Fig. 12. Average optimization results obtained with different settings of N. Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.

5.5.3. The relationship between ϕ and N

To further investigate the characteristics of PSO-DBCD with respect to ϕ and N, experiments are conducted to show PSO-DBCD's performance with different combinations of these two parameters. The results obtained by 30 independent runs are shown in Table 6, which leads to the following findings. First, PSO-DBCD performs competitively with $\phi = \{0.2, 0.3, 0.4\}$. Second, overall, large ϕ should be adopted if the swarm size is relatively small, *e.g.* 400. This can be explained by that a small swarm size leads to poor swarm diversity. A large ϕ thus should be adopted to enhance PSO-DBCD's diversity preservation ability. In summary, $\phi = \{0.2, 0.3, 0.4\}$ is suggested in this paper.

5.6. Performance of PSO-DBCD on low-dimensional optimization

This section conducts experiments to test PSO-DBCD's scalability. In the experiments, the following algorithms are chosen for comparison: HPSO-TVAC [63], CLPSO [5], ALCPSO [7], LIPS [64], CSO [14], and DLLSO [10]. CEC 2013's 28 low-dimensional benchmarks are adopted [65], which are numbered from F_{16} to F_{43}. The maximum *FEs* is set to $6E+05$ and the dimensionality is 50.

For the parameter settings, the swarm size N of CLSO is set to 200; S_{sw} is set to $\{2, 4, 8, 10, 20, 25, 40, 50\}$; ϕ is set to 0 to further simplify the algorithm's structure. For the other six peer algorithms, the default parameter settings suggested in the corresponding references are adopted.

Table 8 and Fig. 13 show the comparison results. One can see that PSO-DBCD wins for times out of the 28 benchmarks in the average result competition. For the statistical analysis, according to the rules in Table 1 and the normality test results in Table 7, the statistical comparisons between PSO-DBCD and the peer algorithms are presented in Table 7. From the paired statistical test, PSO-DBCD significantly outperforms HPSO-TVAC, CLPSO, ALCPSO, LIPS, CSO, and DLLSO for 23, 22, 24, 24, 18, and 24 times, respectively. The Friedman test and the Quade test also show the competitiveness of the proposed algorithm.

Fig. 14 shows the corresponding convergence curves of PSO-DBCD and the compared algorithms, where the convergence curves of PSO-DBCD and CSO are incomplete on F_{16} as these two algorithms quickly reach 0 on F_{16}. The results demonstrate that PSO-DBCD is promising in convergence in comparison to other algorithms.

In summary, the above experimental results suggest that the proposed algorithm is competitive in low-dimensional optimization with respect to both convergence speed and final optimization results.

6. Centralized electric vehicles charging strategy optimization

In this section, the proposed algorithm is applied to the problem of charging strategy optimization for centralized electric vehicles [66].

6.1. Optimization problem

Centralized charging of electric vehicles (EVs) under a battery swapping scenario is promising for EVs' large-scale utilization. The model proposed by [66] assumes that 10 charging stations are available to 50 000 EVs' batteries that are divided into 50 groups to be charged with respect to a time-varying electricity price. The goal is to select the charging station and charging time for each battery group to minimize a weighed loss function formulated as

$$f_{loss} = min(P_{los} + \sigma V_{dev} + \gamma Cost) \tag{47}$$

$$P_{los} = \sum_{t=1}^{TS} \sum_{l=1}^{NL} |I_{lt}|^2 R_l \tag{48}$$

$$V_{dev} = \sum_{t=1}^{TS} \sum_{bt=1}^{NB} |V_{bt} - 1.0| \text{ (p.u)} \tag{49}$$

$$Cost = \sum_{t=1}^{TS} P_t \delta \varphi_t, \tag{50}$$

where P_{los} is the active power loss; V_{dev} is the load bus voltage deviation from 1.0 p.u; $Cost$ is the total electricity cost; σ and γ are adopted to weight the importance of the corresponding sub-optimization targets. According to the settings in [66], σ and γ

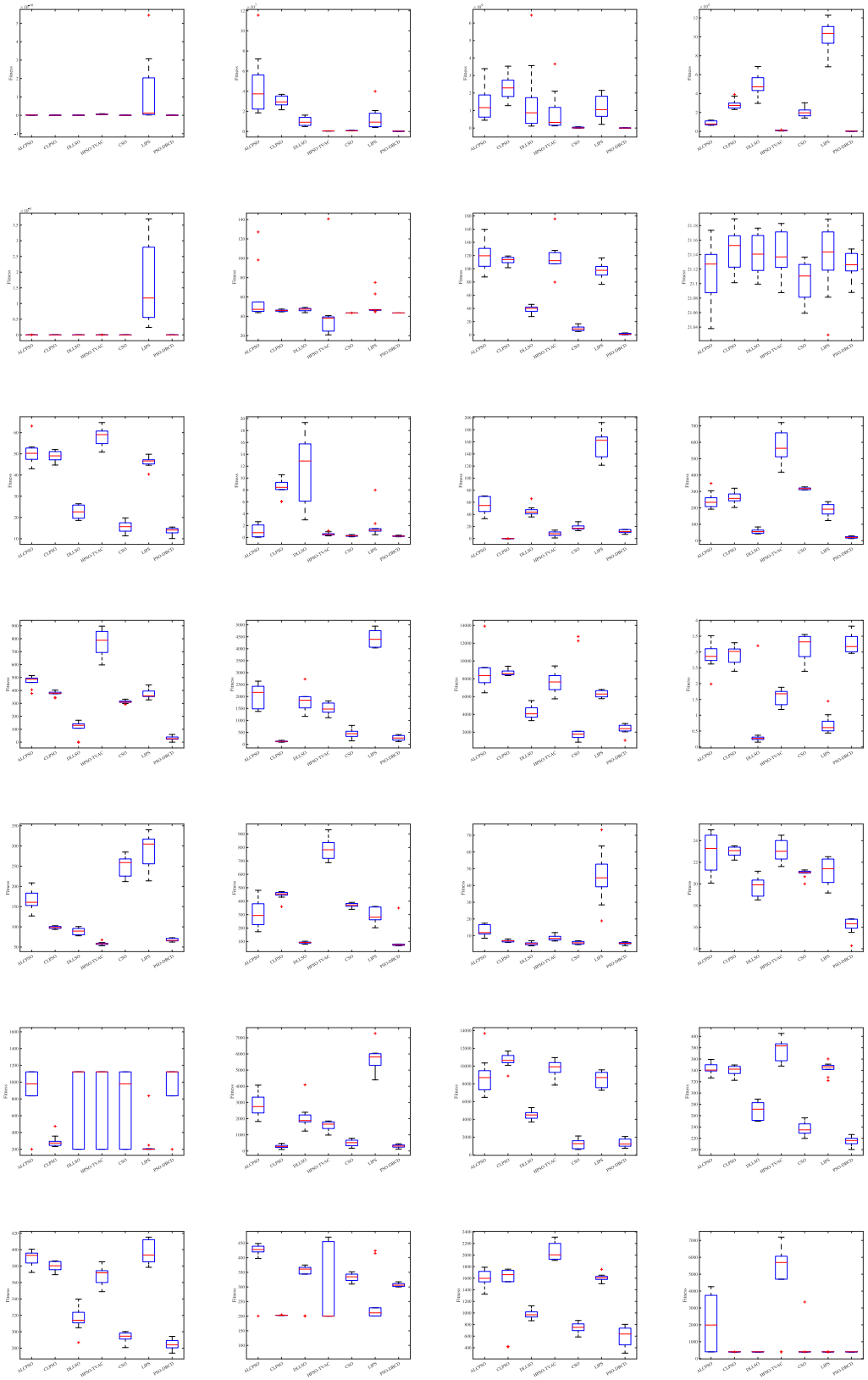
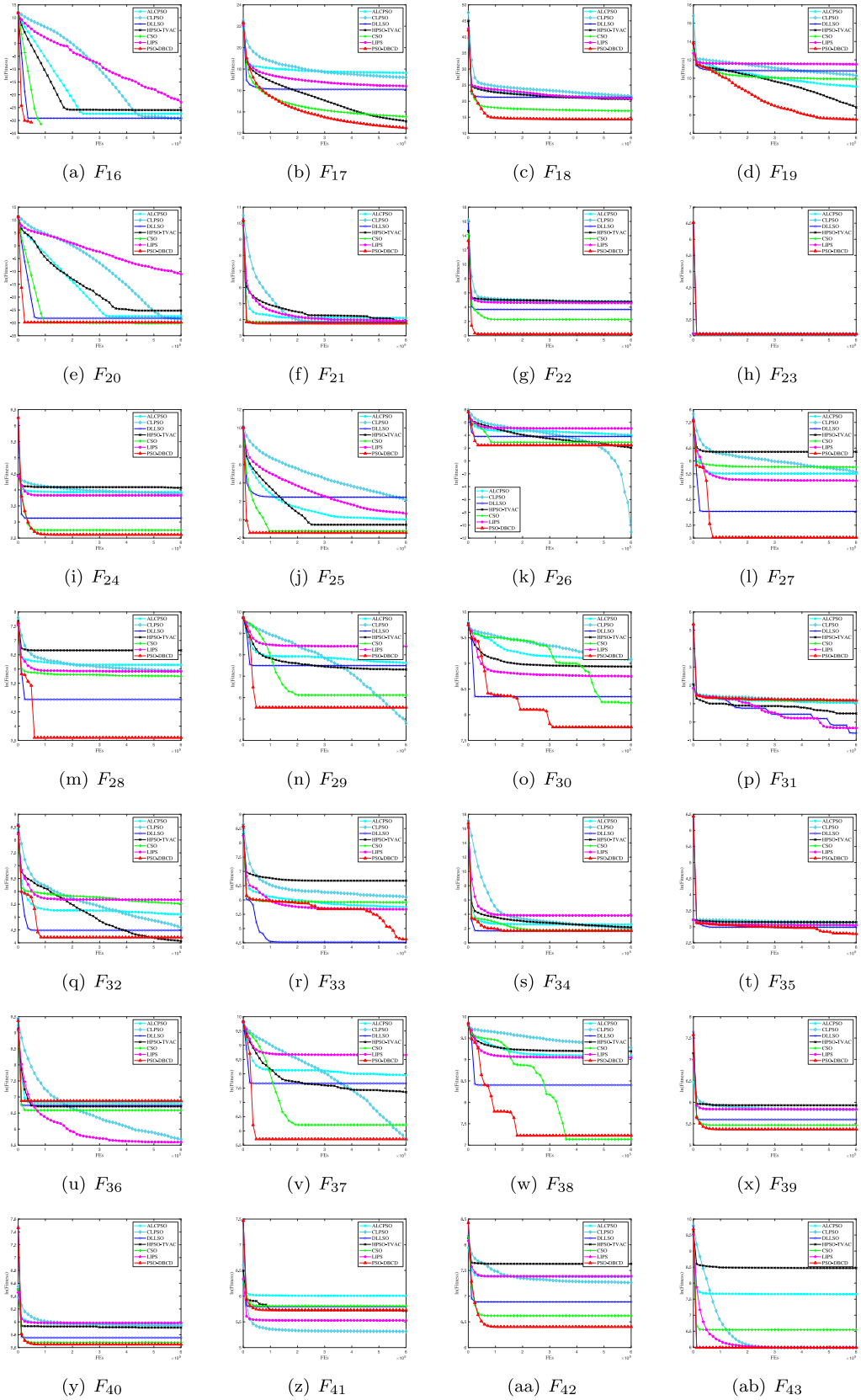


Fig. 13. The experimental results of low-dimensional IEEE CEC 2013 benchmark functions with fitness evaluations of 6E+05.



(a) F_{16} (b) F_{17} (c) F_{18} (d) F_{19}

(e) F_{20} (f) F_{21} (g) F_{22} (h) F_{23}

(i) F_{24} (j) F_{25} (k) F_{26} (l) F_{27}

(m) F_{28} (n) F_{29} (o) F_{30} (p) F_{31}

(q) F_{32} (r) F_{33} (s) F_{34} (t) F_{35}

(u) F_{36} (v) F_{37} (w) F_{38} (x) F_{39}

(y) F_{40} (z) F_{41} (aa) F_{42} (ab) F_{43}

Fig. 14. Convergence profiles of different algorithms obtained on the CEC 2013 low-dimensional benchmarks with 50 dimensions. Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.

Table 6
The experimental results obtained by PSO-DBCD, where N is selected in {400, 600, 800, 1000, 1200} and ϕ varies from 0.1 to 0.5. The best results of each combination of N and ϕ are marked in bold font.

N	Func.	ϕ				
		0.1	0.2	0.3	0.4	0.5
400	F_1	1.5412E−20	1.5938E−21	**5.5525E−22**	9.8354E−22	9.4105E−20
	F_3	2.0746E+01	**2.0574E+01**	2.0715E+01	2.0740E+01	2.1343E+01
	F_6	1.0611E+06	1.0612E+06	1.0608E+06	**1.0607E+06**	1.0611E+06
	F_{11}	1.7511E+07	3.2944E+07	3.3996E+07	**1.3427E+07**	2.8394E+07
	F_{14}	1.9159E+07	2.3152E+07	2.6106E+07	**1.8568E+07**	2.5493E+07
	F_{15}	2.4612E+06	2.2127E+06	2.1365E+06	**2.0673E+06**	3.1291E+06
600	F_1	1.2002E−19	1.6760E−21	**4.7751E−22**	1.6822E−21	1.7962E−18
	F_3	2.1595E+01	**2.1593E+01**	2.1596E+01	2.1598E+01	2.1594E+01
	F_6	**1.0601E+06**	1.0606E+06	1.0603E+06	1.0610E+06	1.0604E+06
	F_{11}	2.4808E+07	1.4679E+07	1.8073E+07	**1.1888E+07**	6.1323E+07
	F_{14}	2.2674E+07	1.9352E+07	1.8073E+07	**1.6930E+07**	4.9836E+077
	F_{15}	2.7846E+06	2.0315E+06	**1.9005E+06**	2.3154E+06	4.6333E+06
800	F_1	1.3289E−18	3.8041E−21	**1.1983E−21**	5.5801E−21	1.4154E−13
	F_3	2.1597E+01	2.1596E+01	2.1595E+01	2.1597E+01	**2.1594E+01**
	F_6	**1.0603E+06**	1.0610E+06	1.0604E+06	1.0606E+06	1.0608E+06
	F_11	4.3898E+07	2.6765E+07	**1.8454E+07**	2.6420E+07	1.4920E+08
	F_14	3.3900E+07	2.4499E+07	**1.8359E+07**	2.3202E+07	2.2778E+08
	F_15	4.1716E+06	**2.1135E+06**	2.1484E+06	3.1044E+06	8.8412E+06
1000	F_1	2.3026E−14	6.3593E−21	**3.4496E−21**	4.0761E−20	1.1575E−09
	F_3	2.1597E+01	**2.1592E+01**	2.1599E+01	2.1598E+01	2.1600E+01
	F_6	1.0605E+06	1.0608E+06	1.0604E+06	1.0601E+06	**1.0599E+06**
	F_{11}	7.8218E+07	3.2650E+07	**2.6545E+07**	5.0318E+07	2.4617E+08
	F_{14}	6.2233E+07	2.5005E+07	**2.4036E+07**	3.5228E+07	5.7089E+08
	F_{15}	5.8375E+06	2.7843E+06	**2.5480E+06**	4.2755E+06	2.1695E+07
1200	F_1	7.3584E−11	3.5909E−19	**1.4340E−20**	4.3511E−16	1.1643E−06
	F_3	**2.1592E+01**	2.1594E+01	2.1599E+01	2.1599E+01	2.1598E+01
	F_6	1.0608E+06	1.0604E+06	**1.0599E+06**	1.0600E+06	1.0603E+06
	F_{11}	1.2077E+08	5.0846E+07	**3.1209E+07**	8.0966E+07	5.1045E+08
	F_{14}	1.4646E+08	3.7945E+07	**3.0620E+07**	5.1838E+07	1.1178E+09
	F_{15}	9.6506E+06	3.7947E+06	**3.2108E+06**	6.0368E+06	5.3941E+07

Table 7
p-values of the Shapiro–Wilk test results on CEC 2013 low-dimensional benchmarks.

Function	PSO-DBCD	HPSO-TVAC	CLPSO	ALCPSO	LIPS	CSO	DLLSO
F_{16}	0.00E+00	8.86E−01*	0.00E+00	6.69E−03	4.46E−04	0.00E+00	0.00E+00
F_{17}	6.29E−01*	9.26E−01*	6.68E−01*	6.70E−02*	2.07E−02	2.19E−01*	2.02E−01*
F_{18}	4.32E−03	2.17E−02	1.98E−03	8.90E−01*	5.59E−02*	4.91E−01*	1.52E−02
F_{19}	7.81E−01*	7.43E−01*	8.17E−02*	4.22E−02	7.10E−01*	3.01E−01*	7.44E−01*
F_{20}	0.00E+00	2.81E−06	0.00E+00	2.85E−01*	8.28E−02*	4.71E−05	5.52E−02*
F_{21}	0.00E+00	2.03E−05	8.44E−01*	1.27E−04	6.71E−05	0.00E+00	6.74E−01*
F_{22}	5.02E−02*	3.93E−02	2.51E−01*	9.91E−01*	9.34E−01*	1.73E−01*	5.35E−01*
F_{23}	4.81E−01*	6.62E−01*	8.63E−01*	4.92E−01*	1.84E−01*	4.60E−01*	7.73E−01*
F_{24}	2.06E−01*	8.80E−01*	8.38E−01*	2.26E−01*	2.08E−01*	7.96E−01*	1.83E−01*
F_{25}	5.47E−01*	1.45E−01*	3.55E−01*	4.39E−02	1.67E−05	8.26E−01*	5.17E−01*
F_{26}	3.83E−01*	9.59E−01*	6.17E−04	1.24E−01*	6.37E−01*	1.49E−01*	6.41E−02*
F_{27}	8.01E−01*	9.10E−01*	9.88E−01*	2.26E−01*	7.73E−01*	4.48E−01*	3.21E−01*
F_{28}	4.21E−01*	2.46E−01*	1.43E−01*	3.61E−02	4.30E−02	5.60E−01*	1.20E−02
F_{29}	4.43E−01*	2.80E−01*	7.62E−01*	1.77E−01*	2.24E−01*	9.88E−01*	4.50E−01*
F_{30}	1.38E−01*	9.56E−01*	1.05E−01*	1.13E−02	4.99E−01*	3.52E−05	4.91E−01*
F_{31}	3.22E−02	9.96E−02*	4.65E−01*	6.30E−01*	3.98E−02	1.53E−01*	5.52E−07
F_{32}	3.60E−01*	2.67E−01*	6.52E−01*	6.71E−01*	3.79E−01*	4.94E−01*	4.80E−01*
F_{33}	3.79E−07	2.59E−01*	9.05E−04	4.57E−01*	3.30E−01*	6.26E−01*	8.34E−01*
F_{34}	8.33E−01*	4.00E−01*	5.78E−01*	2.36E−01*	9.81E−01*	4.88E−01*	9.63E−01*
F_{35}	2.47E−02	6.02E−01*	5.34E−01*	4.02E−01*	1.74E−01*	2.14E−03	6.97E−01*
F_{36}	4.48E−05	3.51E−04	7.29E−03	1.23E−03	2.99E−07	8.94E−04	4.73E−05
F_{37}	9.09E−01*	1.97E−01*	6.36E−01*	9.02E−01*	3.15E−01*	8.26E−01*	5.54E−03
F_{38}	3.07E−01*	4.65E−01*	4.44E−01*	1.80E−01*	7.84E−02*	5.95E−01*	8.33E−01*
F_{39}	9.29E−01*	1.95E−01*	2.42E−01*	4.71E−01*	2.68E−01*	6.03E−01*	1.69E−01*
F_{40}	9.97E−01*	7.18E−01*	3.35E−01*	1.02E−01*	2.66E−01*	4.83E−02	8.42E−01*
F_{41}	1.90E−01*	3.41E−04	1.06E−01*	1.08E−05	5.93E−05	8.22E−01*	1.06E−04
F_{42}	5.94E−01*	6.26E−02*	7.41E−05	8.46E−01*	3.93E−01*	9.19E−01*	4.19E−01*
F_{43}	0.00E+00	5.69E−03	2.32E−07	1.23E−03	1.65E−07	1.00E−07	0.00E+00

are set to 40 and 1000, respectively. TS is the number of time slots that a whole day is divided into; NL is the number of lines in the power system; I_{lt} is the current in the lth line at the tth time slot; R_l is the resistance of the lth line; NB is the number of buses in the power system; V_{bt} is the node voltage (p.u) of the btth bus at the tth time slot; P_t is the power consumption of EVs at the tth time slot; φ_t is the electricity price at the tth time slot; δ denotes the time span of a time slot.

The constraints of the model are

$$P_t \le P_t^{lim}, t \in NT = \{1, 2, \ldots, TS\} \tag{51}$$

$$P_t^{lim} = P_{max} - P_t^{load}, \tag{52}$$

Table 8
The experimental results of 50-dimensional IEEE CEC 2013 benchmark functions with fitness evaluations of 6E+05. The best average result on each benchmark is highlighted by a gray background.

Function	Quality	HPSO-TVAC	CLPSO	ALCPSO	LIPS	CSO	DLLSO	PSO-DBCD
F_{16}	Mean	5.39E−12	2.27E−13	1.30E−12	1.11E−10	0.00E+00	2.27E−13	0.00E+00
	Std	5.72E−13	0.00E+00	7.91E−14	5.56E−11	0.00E+00	0.00E+00	0.00E+00
	p-value	**6.39E−05**	**1.59E−05**	**5.11E−05**	**6.39E−05**	1.00E+00	**1.59E−05**	–
F_{17}	Mean	4.94E+05	3.01E+07	4.61E+07	1.33E+07	7.61E+05	9.82E+06	2.66E+05
	Std	2.55E+04	1.53E+06	9.05E+06	3.30E+06	6.11E+04	1.30E+06	1.74E+04
	p-value	**3.61E−18***	**2.14E−39***	**5.55E−12***	**1.83E−04**	**3.37E−19***	**4.47E−18***	–
F_{18}	Mean	8.99E+08	2.34E+09	1.49E+09	1.12E+09	2.32E+07	1.59E+09	1.74E+06
	Std	3.46E+08	2.12E+08	3.27E+08	2.03E+08	7.96E+06	6.00E+08	5.77E+05
	p-value	**1.83E−04**	**1.83E−04**	**1.83E−04**	**1.83E−04**	**9.11E−03**	**1.83E−04**	–
F_{19}	Mean	9.16E+02	2.87E+04	8.73E+03	1.01E+05	2.00E+04	4.90E+04	2.46E+02
	Std	1.00E+02	1.64E+03	6.44E+02	5.01E+03	1.40E+03	3.30E+03	2.78E+01
	p-value	**1.00E−15***	**1.31E−36***	**1.83E−04**	**4.62E−40***	**8.36E−32***	**6.20E−33***	–
F_{20}	Mean	1.07E−11	3.41E−13	1.19E−12	1.51E−05	7.96E−14	5.57E−13	1.14E−13
	Std	4.76E−12	0.00E+00	8.24E−14	3.72E−06	1.65E−14	5.44E−14	0.00E+00
	p-value	**6.39E−05**	**1.59E−05**	**6.11E−05**	**6.39E−05**	7.67E−02	**5.51E−05**	–
F_{21}	Mean	4.39E+01	4.59E+01	6.02E+01	5.08E+01	4.34E+01	4.71E+01	4.34E+01
	Std	1.05E+01	2.61E−01	8.60E+00	3.02E+00	3.57E−12	5.06E−01	4.77E−14
	p-value	**2.68E−03**	**1.69E−04**	**1.69E−04**	**1.69E−04**	4.70E−01	**1.69E−04**	–
F_{22}	Mean	1.17E+02	1.12E+02	1.21E+02	9.71E+01	9.71E+00	3.92E+01	1.26E+00
	Std	7.27E+00	1.87E+00	6.34E+00	3.20E+00	1.25E+00	1.66E+00	2.94E−01
	p-value	**1.83E−04**	**1.33E−66***	**1.28E−38***	**8.62E−50***	**3.60E−16***	**8.24E−43***	–
F_{23}	Mean	2.11E+01	2.11E+01	2.11E+01	2.11E+01	2.11E+01	2.11E+01	2.11E+01
	Std	8.93E−03	8.51E−03	1.17E−02	1.49E−02	7.57E−03	7.59E−03	5.87E−03
	p-value	**9.05E−03***	**8.41E−04***	2.52E−01*	2.54E−01*	1.01E−03*	**5.33E−03***	–
F_{24}	Mean	5.82E+01	4.89E+01	5.07E+01	4.61E+01	1.56E+01	2.26E+01	1.35E+01
	Std	1.35E+00	7.06E−01	1.62E+00	7.45E−01	8.64E−01	9.05E−01	5.24E−01
	p-value	**1.15E−50***	**3.32E−57***	**3.82E−42***	**2.73E−54***	**1.16E−03***	**2.22E−21***	–
F_{25}	Mean	5.79E−01	8.40E+00	1.04E+00	1.93E+00	2.90E−01	1.14E+01	2.41E−01
	Std	8.58E−02	4.34E−01	3.10E−01	6.54E−01	3.40E−02	1.67E+00	2.64E−02
	p-value	**2.85E−08***	**1.68E−38***	5.71E−01	**1.83E−04**	5.61E−02*	**2.16E−16***	–
F_{26}	Mean	7.96E+00	1.57E−05	5.50E+01	1.57E+02	1.80E+01	4.56E+01	1.16E+01
	Std	1.16E+00	6.41E−06	4.42E+00	7.19E+00	1.29E+00	2.49E+00	8.33E−01
	p-value	4.88E−05*	1.82E−04	**1.82E−23***	**3.58E−40***	**2.42E−09***	**7.84E−30***	–
F_{27}	Mean	5.75E+02	2.61E+02	2.47E+02	1.88E+02	3.17E+02	5.66E+01	2.06E+01
	Std	2.94E+01	1.04E+01	1.48E+01	1.16E+01	1.80E+00	3.68E+00	1.61E+00
	p-value	**1.42E−38***	**3.04E−43***	**1.34E−33***	**4.57E−32***	**3.84E−85***	**5.64E−22***	–
F_{28}	Mean	7.76E+02	3.78E+02	4.67E+02	3.72E+02	3.11E+02	1.10E+02	2.80E+01
	Std	3.12E+01	6.14E+00	1.30E+01	1.25E+01	3.30E+00	1.85E+01	5.65E+00
	p-value	**4.73E−44***	**3.37E−58***	**1.77E−04**	**1.77E−04**	**5.77E−59***	**1.67E−02**	–
F_{29}	Mean	1.50E+03	1.21E+02	2.03E+03	4.43E+03	4.52E+02	1.80E+03	2.55E+02
	Std	7.36E+01	6.52E+00	1.41E+02	1.04E+02	5.55E+01	1.30E+02	3.08E+01
	p-value	**3.86E−34***	1.23E−09*	**9.38E−29***	**4.65E−56***	**1.90E−06***	**2.90E−27***	–
F_{30}	Mean	7.61E+03	8.70E+03	8.75E+03	6.30E+03	3.76E+03	4.23E+03	2.34E+03
	Std	3.38E+02	9.95E+01	6.06E+02	1.08E+02	1.39E+03	2.18E+02	1.64E+02
	p-value	**1.19E−31***	**2.30E−52***	**1.83E−04**	**3.60E−40***	1.04E−01	**4.87E−17***	–
F_{31}	Mean	1.59E+00	2.91E+00	2.89E+00	7.28E−01	3.17E+00	5.48E−01	3.26E+00
	Std	6.94E−02	8.70E−02	1.26E−01	9.21E−02	1.13E−01	2.80E−01	9.84E−02
	p-value	1.83E−04	7.57E−02	4.52E−02	1.83E−04	9.70E−01	1.01E−03	–
F_{32}	Mean	5.84E+01	9.83E+01	1.66E+02	2.91E+02	2.51E+02	8.88E+01	6.77E+01
	Std	1.23E+00	9.11E−01	7.55E+00	1.28E+01	7.38E+00	2.38E+00	1.14E+00
	p-value	2.10E−13*	**3.54E−41***	**9.55E−30***	**9.21E−37***	**5.36E−45***	**9.51E−20***	–
F_{33}	Mean	7.92E+02	4.46E+02	3.16E+02	2.91E+02	3.71E+02	9.18E+01	1.02E+02
	Std	2.58E+01	9.79E+00	3.12E+01	1.72E+01	4.89E+00	2.13E+00	2.61E+01
	p-value	**1.83E−04**	**1.83E−04**	**1.31E−03**	**1.31E−03**	**2.46E−04**	4.59E−03	–
F_{34}	Mean	8.73E+00	6.89E+00	1.29E+01	4.55E+01	5.83E+00	5.32E+00	5.45E+00
	Std	5.02E−01	2.01E−01	9.37E−01	4.74E+00	2.42E−01	2.65E−01	2.04E−01
	p-value	**1.04E−14***	**6.72E−12***	**2.86E−19***	**8.27E−21***	**4.53E−02***	5.16E−01*	–
F_{35}	Mean	2.32E+01	2.30E+01	2.29E+01	2.12E+01	2.10E+01	1.98E+01	1.61E+01
	Std	2.99E−01	1.31E−01	5.10E−01	3.58E−01	1.12E−01	2.61E−01	2.33E−01
	p-value	**1.82E−04**	**1.82E−04**	**1.82E−04**	**1.81E−04**	**1.82E−04**	**1.81E−04**	–

(*continued on next page*)

Table 8 (*continued*).

F_{36}	Mean	8.17E+02	2.93E+02	9.16E+02	2.69E+02	7.25E+02	8.46E+02	9.73E+02
	Std	1.30E+02	2.19E+01	8.66E+01	6.00E+01	1.38E+02	1.34E+02	8.89E+01
	p-value	2.39E−01	2.70E−03	3.42E−01	4.40E−03	6.12E−01	2.02E−01	–
F_{37}	Mean	1.56E+03	2.89E+02	2.83E+03	5.75E+03	4.98E+02	2.13E+03	3.03E+02
	Std	8.09E+01	3.37E+01	1.98E+02	2.18E+02	6.16E+01	2.28E+02	3.05E+01
	p-value	**1.30E−32***	6.17E−01*	**2.63E−29***	**3.53E−45***	**9.63E−06***	**1.83E−04**	–
F_{38}	Mean	9.81E+03	1.07E+04	8.84E+03	8.46E+03	1.26E+03	4.46E+03	1.37E+03
	Std	2.69E+02	2.42E+02	6.37E+02	2.74E+02	1.50E+02	1.57E+02	1.45E+02
	p-value	**7.56E−48***	**3.43E−52***	**3.93E−27***	**2.65E−43***	3.49E−01*	**2.20E−32***	–
F_{39}	Mean	3.77E+02	3.40E+02	3.42E+02	3.43E+02	2.37E+02	2.69E+02	2.15E+02
	Std	5.36E+00	2.67E+00	3.13E+00	3.39E+00	3.65E+00	4.34E+00	2.40E+00
	p-value	**9.45E−48***	**1.59E−53***	**1.16E−51***	**1.35E−50***	**6.31E−12***	**4.08E−26***	–
F_{40}	Mean	3.69E+02	3.80E+02	3.89E+02	3.97E+02	2.93E+02	3.15E+02	2.84E+02
	Std	3.46E+00	1.79E+00	2.96E+00	4.03E+00	2.12E+00	4.23E+00	1.78E+00
	p-value	**4.64E−42***	**1.18E−55***	**3.98E−50***	**6.10E−46***	**1.13E−02**	**1.54E−16***	–
F_{41}	Mean	3.03E+02	2.03E+02	4.06E+02	2.52E+02	3.33E+02	3.29E+02	3.07E+02
	Std	3.99E+01	2.10E−01	2.21E+01	2.67E+01	3.96E+00	2.05E+01	1.98E+00
	p-value	4.73E−01	8.64E−64*	**2.83E−03**	2.57E−02	**3.76E−14***	**2.57E−02**	–
F_{42}	Mean	2.05E+03	1.43E+03	1.60E+03	1.61E+03	7.49E+02	9.81E+02	6.04E+02
	Std	4.54E+01	1.61E+02	4.24E+01	1.93E+01	2.69E+01	2.48E+01	4.96E+01
	p-value	**6.86E−42***	**1.73E−02**	**1.42E−33***	**8.55E−39***	**5.04E−05***	**1.00E−16***	–
F_{43}	Mean	4.77E+03	4.00E+02	2.11E+03	4.00E+02	6.95E+02	4.00E+02	4.00E+02
	Std	7.21E+02	1.82E−03	5.44E+02	6.28E−03	2.80E+02	5.39E−14	0.00E+00
	p-value	**6.39E−05**	**6.39E−05**	**6.20E−05**	**6.39E−05**	1.00E+00	**6.67E−04**	–
w/l/t		23/3/2	22/4/2	24/1/3	24/3/1	18/1/9	24/2/2	–
Friedman ranking		4.71	4.21	5.43	4.88	3.14	3.57	2.05
Quade ranking		4.60	4.32	5.52	4.78	3.18	3.74	1.86

where P_t is the total power consumption for charging the EVs at the tth time slot; P_t^{lim} is the maximum permissible power consumption for charging EVs at the tth time slot; P_{max} is the maximum residential load demands with EVs being charged; P_t^{load} is the total residential power consumption at the tth time slot without EVs being charged. Furthermore, all EVs' batteries should be fully charged once they start charging.

Each battery group has two attributes of charging priority and charging station. The former determines a sequence of charging slots; the latter means which station is used to charge the corresponding battery group. As a result, the dimensionality of this optimization problem is 100. Note that all the used data, including the battery information and charging rate et al. are cited from real-world data resources. Readers are referred to [66] for details.

6.2. Simulation

In this section, PSO-DBCD and PSO-GA+ (The suggested method in [66]) are applied to solve (47).

For a fair comparison, the parameters in the model, such as the initial charging state and the charging rate et al. are adopted with the default settings suggested in [66]. The simulation is conducted based on the IEEE 30-bus system. All of the results listed below are the average of the results from 30 independent runs.

For the algorithm settings, the infeasible solution in PSO-DBCD with respect to (51) will be preserved, which is the same as the settings in [66]. However, the infeasible solutions will not be selected as the convergence exemplar. Furthermore, the maximum *FEs* is set to 12 000, since the charging strategy should be obtained within a limited time [66]. The swarm size N, ϕ in PSO-DBCD are set to 100 and 0.3, respectively. Note that such parameter settings are not obtained by comprehensive experiments, this case study is only adopted to test the practicability of PSO-DBCD. For PSO-GA+, the parameter settings suggested in [66] are adopted.

The average results of 30 independent runs are shown in Table 9. One can observe that PSO-DBCD outperforms PSO-GA+ on f_{loss} and the other 11 metrics. To be specific, although PSO-DBCD slightly performs worse than PSO-GA+ on P_{los} and V_{dev} with respect to several metrics, PSO-DBCD outperforms PSO-GA+ on *Cost* and f_{loss}. Note that *Cost* is the most important factor in f_{los}, since the main purpose of [66] is to minimize the charging cost without heavily causing adverse impact to the power system. Although the above results only show small differences between the two algorithms, this study can be considered effective to test the practicability of PSO-DBCD due to the scalability of the EVs' utilization.

Table 9
Simulation results.

Metric (Units)	Quality	PSO-DBCD	PSO-GA+
P_{los} (MW)	Optimum	1.40731E+03	**1.39997E+03**
	Worst	1.52126E+03	**1.49184E+03**
	Median	**1.42152E+03**	1.46508E+03
	Mean	**1.44542E+03**	1.44849E+03
	Std	7.31150E+00	**6.23135E+00**
V_{dev} (p.u)	Optimum	1.50990E+03	**1.48920E+03**
	Worst	1.58117E+03	**1.55702E+03**
	Median	**1.52603E+03**	1.55278E+03
	Mean	1.54021E+03	**1.53046E+03**
	Std	5.65934E+00	**4.93380E+00**
Cost ($)	Optimum	**1.30574E+05**	1.30604E+05
	Worst	**1.30773E+05**	1.30792E+05
	Median	1.30680E+05	**1.30630E+05**
	Mean	**1.30668E+05**	1.30687E+05
	Std	**1.15282E+01**	1.39881E+01
f_{loss}	Optimum	**1.33563E+05**	1.33566E+05
	Worst	**1.33765E+05**	1.33814E+05
	Median	**1.33628E+05**	1.33648E+05
	Mean	**1.33653E+05**	1.33666E+05
	Std	**1.53038E+01**	1.74403E+01

Fig. 15 presents the convergence curves of these two algorithms, PSO-DBCD obviously outperforms PSO-GA+ on convergence in the middle and latter optimization stages. PSO-DBCD still has good convergence in the final optimization stage, which indicates the potential of PSO-DBCD. In summary, the results demonstrate that PSO-DBCD can be used to solve real-world optimization problems.

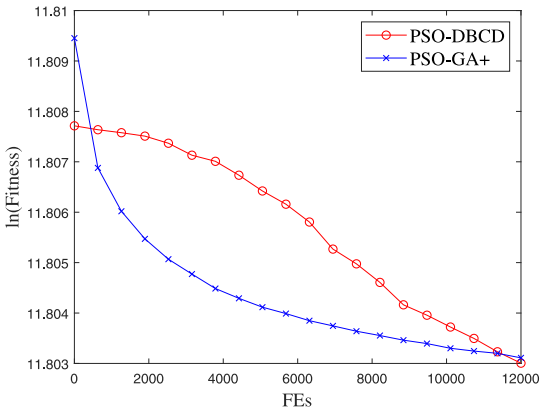


Fig. 15. Convergence curves of PSO-DBCD and PSO-GA+. Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.

7. Conclusions and future work

In this paper, we have proposed a novel cPSO variant for LSOPs that manages the convergence pressure and diversity by using different components. Consequently, the proposed algorithm is capable of balancing convergence and diversity by adjusting specific parameters. Furthermore, the theoretical analysis proves that the proposed algorithm is still computationally efficient in comparison to cPSO and shows promising characteristics of diversity preservation and convergence. The experimental results show that (i) the proposed algorithm is competitive in solving LSOPs in comparison to several state-of-the-art EAs; (ii) the proposed strategies, i.e., the convergence learning strategy and the diversity learning strategy, can effectively manage the convergence pressure and diversity, respectively; (iii) the proposed algorithm shows competitive performance on low-dimensional optimization in a simplified form; (iv) the real-world case study validates the practicability of the proposed algorithm.

However, the proposed algorithm should be further improved. In our future work, we plan to conduct the following studies: (i) investigate the combinations of the proposed diversity indicator with other kinds of meta-heuristics; (ii) adapt the proposed diversity indicator to multi-objective optimization; (iii) propose adaptive adjustments for the parameters in the diversity learning strategy; (iv) hybridize the proposed algorithm with other optimization techniques; and (v) improve the proposed algorithm for discrete optimization.

CRediT authorship contribution statement

Dongyang Li: Methodology, Data analysis, Writing – original draft, Software, Validation. **Lei Wang:** Supervision, Investigation. **Weian Guo:** Conceptualization, Methodology, Supervision, Validation, Funding. **Maoqing Zhang:** Software, Validation. **Bo Hu:** Software, Validation. **Qidi Wu:** Supervision.

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.

Data availability

The authors do not have permission to share data.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant Number 62273263, 72171172 and 71771176; Natural Science Foundation of Shanghai, China under Grant Number 19ZR1479000, 20692191200; Shanghai Municipal Science and Technology Major Project (2022-5-YB-09); Shanghai Municipal Science and Technology Major Project (2021SHZDZ-X0100) and the Fundamental Research Funds for the Central Universities; Education research and reform project of Tongji University: Research on talent training program and model of Sino-German international factory.

References

[1] Q.-b. Zhang, P. Wang, Z.-h. Chen, An improved particle filter for mobile robot localization based on particle swarm optimization, Expert Syst. Appl. 135 (2019) 181–193.
[2] R.L. Patibandla, B.T. Rao, P.S. Krishna, V.R. Maddumala, Medical data clustering using particle swarm optimization method, J. Crit. Rev. 7 (6) (2020) 363–367.
[3] K. Gholami, E. Dehnavi, A modified particle swarm optimization algorithm for scheduling renewable generation in a micro-grid under load uncertainty, Appl. Soft Comput. 78 (2019) 496–514.
[4] M. Gholamghasemi, E. Akbari, M.B. Asadpoor, M. Ghasemi, A new solution to the non-convex economic load dispatch problems using phasor particle swarm optimization, Appl. Soft Comput. 79 (2019) 111–124.
[5] 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.
[6] Z.-H. Zhan, J. Zhang, Y. Li, H.S.-H. Chung, Adaptive particle swarm optimization, IEEE Trans. Syst. Man Cybern. B 39 (6) (2009) 1362–1381.
[7] W.-N. Chen, J. Zhang, Y. Lin, N. Chen, Z.-H. Zhan, H.S.-H. Chung, Y. Li, Y.-H. Shi, Particle swarm optimization with an aging leader and challengers, IEEE Trans. Evol. Comput. 17 (2) (2012) 241–258.
[8] Y. Chen, L. Li, J. Xiao, Y. Yang, J. Liang, T. Li, Particle swarm optimizer with crossover operation, Eng. Appl. Artif. Intell. 70 (2018) 159–169.
[9] A. Lin, W. Sun, H. Yu, G. Wu, H. Tang, Adaptive comprehensive learning particle swarm optimization with cooperative archive, Appl. Soft Comput. 77 (2019) 533–546.
[10] 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) (2017) 578–594.
[11] F. Caraffini, F. Neri, G. Iacca, Large scale problems in practice: the effect of dimensionality on the interaction among variables, in: European Conference on the Applications of Evolutionary Computation, Springer, 2017, pp. 636–652.
[12] S.-Z. Zhao, J.J. Liang, P.N. Suganthan, M.F. Tasgetiren, Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization, in: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 3845–3852.
[13] R. Cheng, C. Sun, Y. Jin, A multi-swarm evolutionary framework based on a feedback mechanism, in: 2013 IEEE Congress on Evolutionary Computation, IEEE, 2013, pp. 718–724.
[14] R. Cheng, Y. Jin, A competitive swarm optimizer for large scale optimization, IEEE Trans. Cybern. 45 (2) (2014) 191–204.
[15] H. Wang, Z. Wu, S. Rahnamayan, Y. Liu, M. Ventresca, Enhancing particle swarm optimization using generalized opposition-based learning, Inform. Sci. 181 (20) (2011) 4699–4714.
[16] A.F. Ali, M.A. Tawhid, A hybrid particle swarm optimization and genetic algorithm with population partitioning for large scale optimization problems, Ain Shams Eng. J. 8 (2) (2017) 191–206.
[17] M.A. Montes de Oca, D. Aydın, T. Stützle, An incremental particle swarm for large-scale continuous optimization problems: an example of tuning-in-the-loop (re) design of optimization algorithms, Soft Comput. 15 (11) (2011) 2233–2255.
[18] M.A.M. De Oca, T. Stutzle, K. Van den Enden, M. Dorigo, Incremental social learning in particle swarms, IEEE Trans. Syst. Man Cybern. B 41 (2) (2010) 368–384.
[19] F. Van den Bergh, A.P. Engelbrecht, A cooperative approach to particle swarm optimization, IEEE Trans. Evol. Comput. 8 (3) (2004) 225–239.
[20] X. Li, X. Yao, Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms, in: 2009 IEEE Congress on Evolutionary Computation, IEEE, 2009, pp. 1546–1553.
[21] X. Li, X. Yao, Cooperatively coevolving particle swarms for large scale optimization, IEEE Trans. Evol. Comput. 16 (2) (2011) 210–224.

[22] M.N. Omidvar, X. Li, Y. Mei, X. Yao, Cooperative co-evolution with differential grouping for large scale optimization, IEEE Trans. Evol. Comput. 18 (3) (2013) 378–393.
[23] M. Yang, A. Zhou, C. Li, X. Yao, An efficient recursive differential grouping for large-scale continuous problems, IEEE Trans. Evol. Comput. 25 (1) (2020) 159–171.
[24] R.-L. Tang, Z. Wu, Y.-J. Fang, Adaptive multi-context cooperatively coevolving particle swarm optimization for large-scale problems, Soft Comput. 21 (16) (2017) 4735–4754.
[25] X. Peng, Y. Jin, H. Wang, Multimodal optimization enhanced cooperative coevolution for large-scale optimization, IEEE Trans. Cybern. 49 (9) (2018) 3507–3520.
[26] Y. Shi, R. Eberhart, A modified particle swarm optimizer, in: 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360), IEEE, 1998, pp. 69–73.
[27] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95-International Conference on Neural Networks, Vol. 4, IEEE, 1995, pp. 1942–1948.
[28] R. Eberhart, J. Kennedy, Particle swarm optimization, in: Proceedings of the IEEE International Conference on Neural Networks, Vol. 4, Citeseer, 1995, pp. 1942–1948.
[29] Q. Yang, K.-X. Zhang, X.-D. Gao, D.-D. Xu, Z.-Y. Lu, S.-W. Jeon, J. Zhang, A dimension group-based comprehensive elite learning swarm optimizer for large-scale optimization, Mathematics 10 (7) (2022) 1072.
[30] R. Cheng, Y. Jin, A social learning particle swarm optimization algorithm for scalable optimization, Inform. Sci. 291 (2015) 43–60.
[31] S. Gulcu, H. Kodaz, A novel parallel multi-swarm algorithm based on comprehensive learning particle swarm optimization, Eng. Appl. Artif. Intell. 45 (2015) 33–45.
[32] N. Lynn, P.N. Suganthan, Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation, Swarm Evol. Comput. 24 (2015) 11–24.
[33] S.Q. Salih, A. Alsewari, Solving large-scale problems using multi-swarm particle swarm approach, Int. J. Eng. Technol. 7 (3) (2018) 1725–1729.
[34] F. Kong, J. Jiang, Y. Huang, An adaptive multi-swarm competition particle swarm optimizer for large-scale optimization, Mathematics 7 (6) (2019) 521–534.
[35] Q. Yang, W.-N. Chen, T. Gu, H. Zhang, H. Yuan, S. Kwong, J. Zhang, A distributed swarm optimizer with adaptive communication for large-scale optimization, IEEE Trans. Cybern. 50 (7) (2019) 3393–3408.
[36] D. Li, W. Guo, A. Lerch, Y. Li, L. Wang, Q. Wu, An adaptive particle swarm optimizer with decoupled exploration and exploitation for large scale optimization, Swarm Evol. Comput. 60 (2021) 100789.
[37] D. Jia, G. Zheng, B. Qu, M.K. Khan, A hybrid particle swarm optimization algorithm for high-dimensional problems, Comput. Ind. Eng. 61 (4) (2011) 1117–1122.
[38] D. Tang, Y. Cai, J. Zhao, Y. Xue, A quantum-behaved particle swarm optimization with memetic algorithm and memory for continuous non-linear large scale problems, Inform. Sci. 289 (2014) 162–189.
[39] M. Tao, S. Huang, Y. Li, M. Yan, Y. Zhou, SA-PSO based optimizing reader deployment in large-scale RFID systems, J. Netw. Comput. Appl. 52 (2015) 90–100.
[40] Z.-J. Wang, Z.-H. Zhan, S. Kwong, H. Jin, J. Zhang, Adaptive granularity learning distributed particle swarm optimization for large-scale optimization, IEEE Trans. Cybern. 51 (3) (2020) 1175–1188.
[41] J.-Y. Li, Z.-H. Zhan, J. Xu, S. Kwong, J. Zhang, Surrogate-assisted hybrid-model estimation of distribution algorithm for mixed-variable hyperparameters optimization in convolutional neural networks, IEEE Trans. Neural Netw. Learn. Syst. (2021) Early access.
[42] J.-R. Jian, Z.-G. Chen, Z.-H. Zhan, J. Zhang, Region encoding helps evolutionary computation evolve faster: A new solution encoding scheme in particle swarm for large-scale optimization, IEEE Trans. Evol. Comput. 25 (4) (2021) 779–793.
[43] R. Ros, N. Hansen, A simple modification in CMA-ES achieving linear time and space complexity, in: International Conference on Parallel Problem Solving from Nature, Springer, 2008, pp. 296–305.
[44] I. Loshchilov, A computationally efficient limited memory CMA-ES for large scale optimization, in: Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation, 2014, pp. 397–404.
[45] D. Molina, M. Lozano, F. Herrera, MA-SW-Chains: Memetic algorithm based on local search chains for large scale continuous global optimization, in: IEEE Congress on Evolutionary Computation, IEEE, 2010, pp. 1–8.
[46] A. LaTorre, S. Muelas, J.-M. Pena, Large scale global optimization: Experimental results with MOS-based hybrid algorithms, in: 2013 IEEE Congress on Evolutionary Computation, IEEE, 2013, pp. 2742–2749.
[47] J. Brest, M.S. Maucec, Self-adaptive differential evolution algorithm using population size reduction and three strategies, Soft Comput. 15 (11) (2011) 2157–2174.
[48] A.A. Hadi, A.W. Mohamed, K.M. Jambi, LSHADE-SPA memetic framework for solving large-scale optimization problems, Complex Intell. Syst. 5 (1) (2019) 25–40.
[49] D. Molina, A.R. Nesterenko, A. LaTorre, Comparing large-scale global optimization competition winners in a real-world problem, in: 2019 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2019, pp. 359–365.
[50] D. Molina, A. LaTorre, F. Herrera, SHADE with iterative local search for large-scale global optimization, in: 2018 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2018, pp. 1–8.
[51] X.-F. Liu, Z.-H. Zhan, J. Zhang, Resource-aware distributed differential evolution for training expensive neural-network-based controller in power electronic circuit, IEEE Trans. Neural Netw. Learn. Syst. (2021) Early access.
[52] W. Chen, T. Weise, Z. Yang, K. Tang, Large-scale global optimization using cooperative coevolution with variable interaction learning, in: International Conference on Parallel Problem Solving from Nature, Springer, 2010, pp. 300–309.
[53] S. Mahdavi, M.E. Shiri, S. Rahnamayan, Cooperative co-evolution with a new decomposition method for large-scale optimization, in: 2014 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2014, pp. 1285–1292.
[54] Y. Sun, M. Kirley, S.K. Halgamuge, Extended differential grouping for large scale global optimization with direct and indirect variable interactions, in: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, 2015, pp. 313–320.
[55] Y. Mei, M.N. Omidvar, X. Li, X. Yao, A competitive divide-and-conquer algorithm for unconstrained large-scale black-box optimization, ACM Trans. Math. Softw. 42 (2) (2016) 1–24.
[56] M.N. Omidvar, M. Yang, Y. Mei, X. Li, X. Yao, DG2: A faster and more accurate differential grouping for large-scale black-box optimization, IEEE Trans. Evol. Comput. 21 (6) (2017) 929–942.
[57] J.-Y. Li, Z.-H. Zhan, K.C. Tan, J. Zhang, Dual differential grouping: A more general decomposition method for large-scale optimization, IEEE Trans. Cybern. (2022) Early access.
[58] Y. Shi, R.C. Eberhart, Population diversity of particle swarms, in: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 1063–1067.
[59] M.R. Bonyadi, Z. Michalewicz, Stability analysis of the particle swarm optimization without stagnation assumption, IEEE Trans. Evol. Comput. 20 (5) (2015) 814–819.
[60] X. Li, K. Tang, M.N. Omidvar, Z. Yang, K. Qin, H. China, Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization, Gene 7 (33) (2013) 8–31.
[61] J. Brest, A. Zamuda, I. Fister, M.S. Maučec, et al., Self-adaptive differential evolution algorithm with a small and varying population size, in: 2012 IEEE Congress on Evolutionary Computation, IEEE, 2012, pp. 1–8.
[62] Q. Yang, W.N. Chen, T. Gu, H. Jin, J. Zhang, An adaptive stochastic dominant learning swarm optimizer for high-dimensional optimization, IEEE Trans. Cybern. PP (99) (2020) 1–17.
[63] A. Ratnaweera, S.K. Halgamuge, H.C. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, IEEE Trans. Evol. Comput. 8 (3) (2004) 240–255.
[64] B.-Y. Qu, P.N. Suganthan, S. Das, A distance-based locally informed particle swarm model for multimodal optimization, IEEE Trans. Evol. Comput. 17 (3) (2012) 387–402.
[65] J. Liang, B. Qu, P. Suganthan, A.G. Hernandez Diaz, Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization, Technical Report 201212, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, 2013, pp. 281–295, (34).
[66] Q. Kang, J. Wang, M. Zhou, A.C. Ammari, Centralized charging strategy and scheduling algorithm for electric vehicles under a battery swapping scenario, IEEE Trans. Intell. Transp. Syst. 17 (3) (2015) 659–669.