Swarm and Evolutionary Computation 60 (2021) 100789

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

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

An adaptive particle swarm optimizer with decoupled exploration and exploitation for large scale optimization

Dongyang Li[a], Weian Guo[a,b,e,*], Alexander Lerch[c], Yongmei Li[d], Lei Wang[a], Qidi Wu[a]

[a] Department of Electronics and Information Engineering, Tongji University, Shanghai 201804, China
[b] Sino-German College of Applied Sciences, Tongji University, Shanghai 200092, China
[c] Center for Music Technology, Georgia Institute of Technology, Atlanta, Georgia 30332, USA
[d] State Key Laboratory of Pollution Control and Resource Reuse, College of Environmental Science and Engineering, Tongji University, Shanghai 200092, China
[e] Shanghai Institute of Intelligent Science and Technology, Tongji University, Shanghai 201804, China

ARTICLE INFO

Keywords:
Particle swarm optimization
Balancing exploration and exploitation
Decoupled exploration and exploitation
Local sparseness degree
Large scale optimization

ABSTRACT

As a form of evolutionary computation, particle swarm optimization is less effective in large scale optimization since it is unable to effectively balance exploration and exploitation. To address this problem, first, a learning structure decoupling exploration and exploitation is proposed. This helps simultaneously and independently managing exploration and exploitation in different components. Second, following the proposed learning structure, two novel learning strategies are developed. On the one hand, a local sparseness degree measurement in fitness landscape is proposed to estimate the congestion and distribution of particles, based on which an exploration strategy is built by guiding particles to sparse areas. On the other hand, an adaptive exploitation strategy is developed which can effectively adjust the fitness differences between exemplars and updated particles during the optimization process by employing a multi-swarm strategy and an adaptive sub-swarm size adjustment. Finally, by embedding the two learning strategies into the proposed learning structure, an adaptive particle swarm optimizer with decoupled exploration and exploitation is proposed. Thanks to the novel balancing strategy of exploration and exploitation, the two functions in the proposed algorithm can be independently and simultaneously managed. Furthermore, theoretical analyses are put forward to prove the convergence and computational complexity of the proposed algorithm. Comprehensive experiments are conducted based on the large scale optimization benchmarks from CEC 2010 and CEC 2013 and six state-of-the-art large scale optimization evolutionary algorithms, the results demonstrate the effectiveness of the proposed learning strategies and the competitive performance of the proposed algorithm.

1. Introduction

As a popular meta-heuristics, particle swarm optimization (PSO) has received considerable attention due to its powerful ability in addressing various kinds of optimization problems [1,2]. The canonical PSO (cPSO) employs a swarm of particles to search the global optimum for a given problem. Each particle has two attributes, namely velocity and position, which are iteratively updated according to:

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

$$p_i^d(t+1) = p_i^d(t) + v_i^d(t+1) \quad (2)$$

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; $pbest_i(t)$ represents the best position achieved by the ith particle so far; ω is the inertia weight while c_1 and c_2 are acceleration coefficients; r_1 and r_2 are two random values within (0,1). Due to its simplicity and efficiency, cPSO has been widely applied to various kinds of real world engineering optimization problems [3–5].

For cPSO, it is suffering from the premature convergence [6]. To alleviate this issue, the parameter settings in cPSO must be properly selected to balance the exploration and exploitation to avoid premature convergence [7]. The existing improvements on this issue are conducted on many aspects, including initial population [8–10], swarm size [11], acceleration coefficients [11,12], inertia weight [13], and many others [14–20]. Although cPSO has achieved a great success in dealing with low-dimensional problems [7,21], it is still a big challenge for cPSO to address large scale optimization problems (LSOPs) due to the "curse of dimensionality". To be specific, the increasing dimensionality will cause an explosion of the number of local optima, as well as an extension of the decision space around them, which greatly challenges the efficacy and efficiency of cPSO [22]; on the other hand, due to the limited computa-

* Corresponding author.

https://doi.org/10.1016/j.swevo.2020.100789
Received 4 December 2019; Received in revised form 23 September 2020; Accepted 28 September 2020
Available online 8 October 2020

tional resources, only a small part of the decision space can be explored in LSOPs [23]. Therefore, it is crucial to enhance cPSO's search ability in large scale decision space.

As pointed out by [22], the existing works on improving cPSO for LSOPs can be classified into two categories. First, embedding cPSO in cooperatively coevolutionary (CC) framework is a widely used tactic, where researchers divide a whole dimension into several sub-segments and separately handle each sub-segment to alleviate the adverse impacts caused by the increasing dimensionality [24–27]. Second, significant effort has been put on proposing new learning or updating strategies [6,22,28–31]. Researchers attempt to develop more flexible learning or updating strategies to enhance the diversity for stronger exploration ability. For examples, Zhao et al. adopt multi-swarm methods to promote the information exchange among particles for maintaining swarm diversity [28]; Cheng et al. propose competitive swarm optimizer (CSO) which updates only half of the particles at each generation and employs the mean position of the swarm to guide particles for preserving diversity [6].

However, both of these methods show room for further improvement. For the cPSO variants within the CC framework, their performance not only highly relates to variable grouping techniques [22,26], but also depends on the heavy cost of fitness evaluations (*FEs*) [22,32]. For the cPSO variants with new learning strategies, it is a challenge to calibrate the weight of exploration and exploitation since the two functions are coupled in the algorithm structure. To be specific, the exploration enhancements in the existing works commonly contribute convergence while the existing exploitation managements are always related to diversity preservation. For example, in CSO, the mean position of the swarm is used to enhance the swarm diversity. However, it is shared by all updated particles, which limits its ability of enhancing diversity. Besides, the competitive learning mechanism is designed for exploitation [22], but it allows half of particles directly pass to the next generation. This is beneficial for preserving diversity and reduces the convergence speed.

Therefore, it is still a big challenge for cPSO to handle with LSOPs. In particular, as discussed above, the main obstacle for improvements of the learning strategy is the coupling of exploration and exploitation, leading to difficulties in balancing these two functions during the optimization process. To solve this problem, this paper aims to develop novel learning strategies for balancing exploration and exploitation in PSO. The main contributions of this paper are listed as follows:

1. A novel learning structure decoupling exploration and exploitation for cPSO is proposed in this paper, where the exploration and exploitation are conducted in different components. Therefore, these two functions are decoupled and can be explicitly and independently managed;
2. The traditional way to measure population diversity is based on the whole swarm, and therefore the results cannot bring a detailed improvement for the particles updating. In this paper, a local sparseness degree measurement in fitness landscape is proposed to provide a microscopic view of local crowding information. The proposed measurement is not only independent of the details of the optimization problem, but also can be used to evaluate the sparseness information for each particle by considering both the congestion and distribution of particles. Based on this indicator, an exploration learning strategy is put forward from the perspective of swarm diversity preservation.
3. An adaptive multi-swarm strategy is designed in this paper. On the one hand, thanks to a new particle grouping method, the proposed adaptive multi-swarm strategy is able to adapt to different swarm size settings, which is beneficial for the scalability of the sub-swarm size settings. On the another hand, by adopting an adaptive sub-swarm size adjustment, the proposed adaptive multi-swarm strategy dynamically changes the sub-swarm size during a run. With this multi-swarm strategy, an exploitation learning strategy is proposed by guiding particles with the corresponding best individual in each sub-swarm, where the convergence pressure can be effectively managed by adjusting the sub-swarm size.
4. By incorporating the proposed exploration and exploitation learning strategies into the proposed novel learning structure, an adaptive particle swarm optimizer with decoupled exploration and exploitation (APSO-DEE) is developed. In APSO-DEE, the exploration and exploitation can be independently and simultaneously managed. Furthermore, the employed adaptive sub-swarm size adjustment is able to dynamically balance the exploration and exploitation in the optimization process.

The rest of this paper is organized as follows. Section 2 presents a literature review on popular evolutionary algorithms (EAs), especially PSO, in addressing LSOPs. Section 3 describes the details of the proposed APSO-DEE. Section 4 presents the theoretical analysis of the convergence and computational complexity for the proposed APSO-DEE. In Section 5, based on the benchmarks provided by CEC 2010 and CEC 2013, comprehensive parameter sensitivity analyses are conducted and several peer EAs are employed to compare the proposed algorithm's performance. Finally, we conclude this paper and present our future work in Section 6.

2. Related work

This section will present a comprehensive review on the works of cPSO's variants as well as other widely-used EAs in large scale optimization. Without loss of generality, an optimization problem is regarded as a minimization problem as defined by:

$$min f(x), \quad x = [x^1, ..., x^d, ..., x^D] \tag{3}$$

where D denotes the dimensionality of an optimization problem, x^d represents the dth dimension of a feasible solution.

As introduced by Qiang et al. [22], addressing LSOPs is challenging for cPSO and other meta-heuristics due to the high dimensionality usually causes an explosion of the number of local optima and expands the decision space around them. The existing improvements in this area can be classified into the following two categories.

2.1. Meta-heuristics with CC framework

Incorporating meta-heuristics with CC framework is a hot topic in large scale optimization. Studies in this category focus on dividing decision variables into many sub-segments. Each sub-segments has a low dimension and will be independently optimized. By this means, the difficulties caused by a large scale dimensionality will be overcome.

The first attempt to incorporate cPSO with CC framework is conducted by Van et al., where CCPSO−S_K and CCPSO−S_H are proposed. The former divides the decision variables into D/K sub-segments which are optimized by cPSO while the later alternatively updates the swarm using CCPSO−S_K and the cPSO. However, they just test the proposed algorithms with low dimensionality [24]. Based on CCPSO−S_K, Li et al. propose CCPSO by using a random variables grouping strategy and further refining the solutions obtained by CCPSO−S_K with an adaptive weighting scheme [25]. Then, Li et al. propose CCPSO2 based on CCPSO by adopting Gaussian mutation and Cauchy mutation for exploration and exploitation, respectively, and removing the adaptive weighting. Additionally, an adjustment strategy on variable group size is developed in CCPSO2 showing competitive performance in tackling the interdependency variables [26]. Tang et al. develop AM-CCPSO with increasing the number of context vector to provide robust and effective co-evolution [27]. Recently, Peng et al. propose MMO-CC, where the CC framework is implemented to a modified CMA-ES and a nondominance-based selection scheme is proposed to adaptively select context vector with consideration on both quality and diversity [32].

However, the performance of meta-heuristics with CC framework seriously depends on the variables grouping techniques. To identify the

dependency among variables, several promising grouping methods have been proposed. Chen et al. propose variable interaction learning to dynamically identify the independency between variables in the optimization process [33]. Mahdavi et al. propose meta-modeling decomposition, where a first order RBF-HDMR is used to identify the interactions among variables to create the nonseparable and separable sub-segments [34]. Omidvar et al. present DG to investigate the interaction between two variables based on the changes of the global function fitness [35]. Sun et al. propose XDG to identify the indirect interactions among variables to improve DG for Rosenbrock function [36]. Mei et al. propose GDG which helps address the sensitivity problem of DG by considering the computational errors [37]. Omidvar et al. develop DG2 to reduce the function evaluations in the variables grouping process by introducing an systematic selection of sample points [38].

Nevertheless, there are two main drawbacks behind this kind of methods. On the one hand, these variables grouping techniques commonly cause a heavy *FEs* computation cost [22]. On the other hand, the meta-heuristics with CC framework produce a huge amount of extra *FEs* compared with the methods without CC framework, the reason is that each solution in a sub-segment should be evaluated by interacting with one or more representative solutions [32].

2.2. Meta-heuristics with new learning or updating strategies

The main purpose of designing new learning or updating strategies is to enhance the diversity preservation ability for improving exploration.

Zhao et al. propose DMS-PSO to promote the information exchange among particles for swarm diversity preservation by adopting several grouping tactics to frequently divide the whole population into a large number of sub-swarms during a run [28]. Hsieh et al. present EPUS-PSO allowing each particle's *pbest* to be the leader of other particles to enhance the swarm diversity. Furthermore, the population size of EPUS-PSO will be changed based on the solution searching status to help enhance exploration and save computation budget [29]. Cheng et al. propose FBE to solve LSOPs, where the whole swarm is divided into two sub-swarms. At each generation, a fitness value based competition is employed to identify the superior and inferior particles, then each inferior particle is leaded by the best and another randomly selected particles in the opposite sub-swarm while the superior particles are subjected to a mutation operation [30]. Cheng et al. propose CSO based on the same competition mechanism [6]. In CSO, the superior particles in the competition are employed as the exemplars for the corresponding inferior particles and directly kept to the next generation. Besides, the mean position of the whole swarm is used to guide the inferior particles. With these strategies, the swarm diversity is significantly increased. Inspired by nature phenomenons, Cheng et al. develop SLPSO by allowing particles to learn from each individuals that are better than themselves, by this means, the search flexibility of particles is diversified [31]. Nandar et al. propose HCLPSO to enhance the exploration and exploitation in cPSO, where the swarm is divided into two subpopulations which focus solely on either exploration or exploration by adopting different learning strategies [39]. Yang et al. propose DLLSO by adopting a level-based learning strategy which greatly diversifies the exemplars, besides, the two exemplars of each particle are sorted based on the fitness value to get a compromise between exploration and exploitation [22].

Besides, many researchers attempt to construct new learning or updating strategies by hybridizing cPSO with other algorithms and techniques. Park et al. propose an improved cPSO by employing a chaotic sequences based linearly decreasing inertia weights adjustment and a crossover operation scheme to enhance swarm diversity [40]. Jia et al. develop CGPSO by utilizing chaotic local search and Gaussian optimization. In CGPSO, the chaotic local search is adopted to enhance the exploration ability of cPSO in the initial optimization process while the Gaussian optimization is employed to refine the promising solution in the later stage [41]. Wang et al. propose GOPSO to alleviate the premature convergence based on Cauchy mutation and a generalized opposition-based learning strategy [42]. Tang et al. develop a hybrid cPSO to improve the global search ability of the algorithm, where a memetic algorithm is used to help particles gain some experience before the evolutionary process [43]. Tao et al. propose a hybrid SA-PSO by introducing the metropolis rule of simulated annealing algorithm into cPSO to control the particles' updating during the run and accelerate the convergence speed of the algorithm [44]. Soleimani et al. develop a hybrid optimizer by combining genetic algorithm with cPSO, which can ultilize the advantages of both genetic algorithm and cPSO [45]. Ali et al. propose a hybrid PSO by utilizing the arithmetical crossover operator and mutation operation to increase the swarm diversity [46].

Moreover, except for cPSO, researchers also design various new learning or updating strategies for other kinds of meta-heuristics to solve LSOPs. CMA-ES proposed by Hansen et al. shows competitive performance in global optimization. However, it is less effective in large scale optimization due to the high computational cost for creating the covariance matrix [47,48]. To alleviate this problem, Ros et al. propose sep-CMA-ES to reduce the computation complexity to $O(D)$ by only computing the diagonal elements of the covariance matrix [49]. Loshchilov et al. propose LM-CMA-ES to reduce the computation complexity to $O(MD)$ by computing the covariance matrix only based on M selected direction vectors and a matrix decomposition method, although it's more computation consuming, it shows better performance than sep-CMA-ES [50]. Molina et al. propose Ma-Sw-Chains by employing steady-state GA and a local search method to balance the global and local search behaviour [51]. LaTorre et al. propose MOS which is able to utilize advantages of different optimizers by dynamically operating a set of optimizers in a sequential manner based on designed quality measure [52]. Zhang et al. propose JADE by designing a mutation strategy and an optional archive operation for differential evolution, where the optional archive operation can provide the information of progress direction by utilizing historical information [53]. Brest et al. propose jDElscop by proposing a population size reduction mechanism which is proved to be beneficial for improving the optimization performance [54]. Yang et al. propose GaDE to adaptively adjust DE's parameter for improving its scalability in LSOPs. To balance exploration and exploitation for differential evolution, Ali et al. propose mDE-bES, where different mutation and updating strategies are designed for different sub-populations [55]. Hadi et al. propose a LSADE-SPA memetic Framework, where several optimization techniques are combined together for solving LSOPs [56,57]. Molina et al. develop a DE variant named SHADE-ILS by adopting a population re-start strategy and two large scale local search methods [58].

However, although the above works put great effort on improving meta-heuristics for LSOPs, premature convergence is still challenging. The main reason is that the exploration and exploitation are highly coupled in the existing works. In another word, the diversity and convergence can not be explicitly and independently managed, leading difficulties in balancing them to improve the searching ability. For examples, in CSO, although the mean position will change at every generation, it is obvious that the diversity can not be effectively enhanced by guiding particles with the mean position, because the mean position is shared by all the updated particles, which will inevitably reduce the swarm diversity; for DLLSO, it significantly diversifies the exemplars by a level-based exemplar selection mechanism. However, the distribution of particles is not considered. This means that the particles in crowded decision spaces has the same probability to be selected as the exploration exemplars as the particles in sparse decision spaces, which lacks efficacy to produce high diversity. Besides, DLLSO improves its exploitation ability by increasing the particles number in each level to increase the fitness differences between updated particles and the exemplars. However, more promising particles will be not able to participate in the updating operation under such situation, which potentially contributes the diversity preservation and cannot further benefit the exploitation.

In summary, large scale optimization is still challenging for meta-heuristics. In particular, for the PSO variants with new learning or updating strategies, the coupling of exploration and exploitation lim-

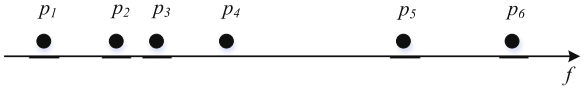


Fig. 1. The potential fitness landscape at generation t under the assumptions that the swarm size is 6 and the particles have been sorted according to the fitness value.

its their abilities of preserving diversity for exploration and refining promising solutions for exploitation. Therefore, balancing exploration and exploitation is still a big challenge for PSO.

3. Proposed algorithm

As discussed in the literature review, due to the coupling of exploration and exploitation in the existing work, cPSO and its variants suffer from the balancing of these two functions. To address this issue, a learning structure decoupling exploration and exploitation for cPSO is proposed first. Then, two novel learning strategies are introduced for exploration and exploitation, respectively. Finally, by embedding the proposed learning strategies into the novel learning structure, an adaptive particle swarm optimizer with decoupled exploration and exploitation is proposed for large scale optimization.

3.1. A learning structure decoupling exploration and exploitation

The proposed novel particles learning structure is shown in the following equations:

$$v_i(t+1) = Inertia(v_i(t)) + Exploration(v_i(t)) + Exploitation(v_i(t)) \tag{4}$$

$$p_i(t+1) = p_i(t) + v_i(t+1) \tag{5}$$

where $Inertia(v_i(t))$ is the inertia component inherited from the cPSO to ensure the stability of the algorithm [6], namely, $\omega v_i(t)$; $Exploration(v_i(t))$ and $Exploitation(v_i(t))$ are the learning strategies conducting exploration and exploitation, respectively; $v_i(t)$ and $p_i(t)$ are the velocity and position of the ith particle at generation t, respectively. In this learning structure, each component on the right side of (4) focuses on different operations. Consequently, the exploration and exploitation are conducted in different components in the proposed learning structure. Therefore, the weights in the three components are explicit in balancing exploration and exploitation.

In short, the exploration and exploitation are decoupled in the proposed learning structure and can be independently and simultaneously managed, which benefits the balance of these two functions.

3.2. Exploration learning strategy

Exploration is the ability of EAs to discover a diverse assortment of solutions that are distributed in different regions of the decision space [59]. Therefore, to improve the exploration ability of cPSO, this paper develops the exploration learning strategy from the perspective of diversity preservation.

As discussed in Section 2, although DLLSO has significantly increased the exemplars diversity, it lacks mechanisms to avoid particles flying together to the congestion areas, which is detrimental to diversity. To be specific, as shown in Fig. 1, since DLLSO only allows particles learn from others that are better than themselves, the fitness landscape between p_4 and p_6 can only be unidirectionally explored by p_5 and p_6. This potentially leads to a poor phenotypic diversity.

To solve this problem, this paper proposes a novel local sparseness degree measurement in fitness landscape taking both the congestion and distribution of particles into consideration, which is computed according to:

$$con(i) = \frac{l_{i,1} + l_{i,2}}{L} \tag{6}$$

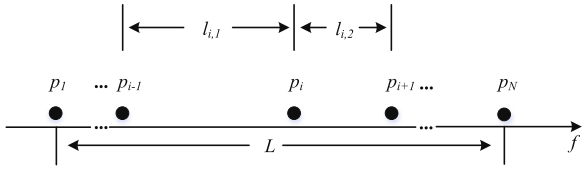


Fig. 2. Computation of $l_{i,1}$, $l_{i,2}$, and L.

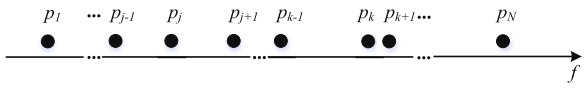


Fig. 3. The potential fitness landscape under the assumption that p_j and p_k are about the same congestion.

$$dis(i) = \frac{min(l_{i,1}, l_{i,2})}{max(l_{i,1}, l_{i,2})} \tag{7}$$

$$LSD(i) = \frac{con(i)}{max(con)} \frac{dis(i)}{max(dis)} \tag{8}$$

where $LSD(i)$ is the proposed local sparseness degree of the ith particle in fitness landscape; $con(i)$ and $dis(i)$ indicate the congestion and distribution of the ith particle, respectively. $l_{i,1}$, $l_{i,2}$, and L are the corresponding intervals shown in Fig. 2, N denotes the swarm size. Note that phenotype diversity, which is computed in fitness space, is a widely adopted diversity measurement in EAs [60]. It is able to not only evaluate the swarm diversity, but also benefit the saving of computational cost on diversity evaluation. Therefore, the LSD in this paper is proposed based on the phenotype diversity for both diversity evaluation and computational simplicity.

The functions of $con(i)$ and $dis(i)$ are clear. First, with the setting of $con(i)$, a large space around the ith particle results in a large value of $LSD(i)$. This indicates that the particles in the sparse areas of the fitness landscape will get larger LSD than that in crowding areas. Second, with the design of $dis(i)$, an uneven distribution of the ith particle and its adjacent neighbours leads to a small value of $LSD(i)$. This is motivated by that particles should uniformly distribute in the fitness landscape. Fig. 3 shows an example in which p_j and p_k are about the same congestion while p_k is closer to its neighbour p_{k+1}, which indicates learning from p_k is less effective to keep an uniform distribution. Since the best and worst particles locate in the boundary scope, there is only one neighbor around them, respectively. Therefore, it is difficult to directly compute the LSD of the best and worst particles. To solve this problem, the two particles are not included in the exploration exemplar selection operation and the corresponding LSD is set to 0 for the following reasons. The best particle should be selected as the exploitation exemplar. On the other hand, it is of little value to employ the worst particle as the exploration exemplar since the worst particle is considered lying in a poor search area. In addition, since a large swarm size is commonly adopted in large scale optimization, it will scarcely impact the searching behavior of the exploration operation without employing the best and worst particles as the exploration exemplars.

With this local sparseness degree, particles in the proposed algorithm will learn from others with higher LSD for diversity preservation. In addition, to prevent the algorithm from premature convergence, this paper suggests that the particles with high LSD should have a larger likelihood to be directly passed to the next generation. Following the above ideas, the exploration learning strategy is designed as follows. Firstly, the LSD of each particle should be computed at each generation. Secondly, an empty set $P_{updated}$ used to contain the particles to be updated at generation t should be defined. Thirdly, a *ranking* vector is computed to store the rank of each particle by sorting their LSD in ascending order. A particle will be put into $P_{updated}$ if its rank is less than or equal to $N \cdot rand$, where $rand$ is randomly generated within (0,1) and N is the swarm size.

The exploration exemplar for a particle in $P_{updated}$ is randomly selected from the particles with larger *LSD*. Finally, the velocity of the ith particle in $P_{updated}$ is updated according to (9) for exploration based on diversity preservation

$$v_i^d(t+1) = \phi r_1(p^d_{exploration,i}(t) - p_i^d(t)). \tag{9}$$

Here, $p^d_{exploration,i}(t)$ is the dth dimension of the exploration exemplar for ith particle in $P_{updated}$ at generation t; $p_i^d(t)$ is the dth dimension of the ith particle's position at generation t; r_1 is a random value within (0,1), ϕ is a parameter set by users. The pseudo-code of the selection strategy for exploration exemplars and updated particles is shown in Algorithm 1 .

Algorithm 1: Pseudo-code of the exploration exemplars and updated particles selection strategy

Input: Swarm $P(t)$, fitness value vector $fitness$, swarm size N
Output: The exploration exemplar set $P_{exploration}$ and the updated particle set $P_{updated}$ at tth generation

$P_{exploration} \leftarrow \emptyset$;
$P_{updated} \leftarrow \emptyset$;
$P \leftarrow$ Sort $P(t)$ according to the fitness value in ascending order;
for $i = 1$ *to* N **do**
  $LSD(i) \leftarrow$ Compute $LSD(i)$ according to (8);
end
$ranking \leftarrow$ Create a vector to store the ranking of each particle that is computed by sorting LSD in ascending order;
for $i = 1$ *to* N **do**
  if $ranking(i) \leq N \cdot rand$ **then**
    Build the set S_e including all the particles with higher LSD than the ith particle;
    $p_{exploration,i}(t) \leftarrow$ Randomly select a particle in S_e;
    $P_{exploration} \leftarrow P_{exploration}(t) \bigcup p_{exploration,i}(t)$;
    $P_{updated} \leftarrow P_{updated} \bigcup$ the ith particle;
  end
end

Apparently, the proposed learning strategy is more efficient in diversity preservation than DLLSO because the space between p_4 to p_6 can be better explored than in DLLSO under the situation shown in Fig. 1. This is benefited from that p_4 and p_5 will be explored by more particles due to their higher *LSD*. To be more general, given any potential fitness landscape, the proposed exploration strategy should outperform DLLSO in uniformly distributing the particles in fitness landscape. The reason is that for one thing, DLLSO fails to efficiently employ particles in sparse areas to guide more particles in crowding areas since DLLSO allows particles attract only the particles worse than them; for another, particles in sparse areas might learn to the particles in crowding areas in DLLSO. In summary, with the proposed exploration strategy, users can vary ϕ to adjust the diversity for exploration function calibration.

3.3. Exploitation learning strategy

Exploitation aims to further refine the promising regions of the decision space in the searching process, which plays a vital role in EAs [23,59]. Therefore, the exploitation operator is also crucial to the performance of cPSO since it helps find better solutions or improve the existing ones. This paper proposes an adaptive multi-swarm based exploitation learning strategy as introduced following.

In multi-swarm strategies with specific swarm size N and sub-swarm size s, the whole swarm will be divided into several sub-swarms. However, if the swarm is equally divided into N/s sub-swarms, it is not conducive to the scalability of s. To this end, given a sub-swarm size s, this paper computes the sub-swarm number g_{num} by (10).

$$g_{num} = \lceil (N/s) \rceil \tag{10}$$

For the sub-swarms, the top $g_{num} - 1$ sub-swarms are randomly assigned with s particles while the last sub-swarm holds the remaining particles. Therefore, the proposed adaptive multi-swarm strategy is able to adapt to different sub-swarm size settings.

With the above settings, the exploitation learning strategy is designed as follows. Firstly, the whole swarm will be divided into several sub-swarms. Secondly, an empty set $P_{updated}$ used to contain the particles to be updated at generation t will be defined. Thirdly, particles in each sub-swarm will be put into $P_{updated}$ except for the best individual. The best particles in each sub-swarm will be directly preserved to the next generation and will be employed as the exploitation exemplar for the remaining particles in the corresponding sub-swarm. Consequently, the ith particle in $P_{updated}$ updates its velocity according to (11) for exploitation

$$v_i^d(t+1) = r_2(p^d_{exploitation,i}(t) - p_i^d(t)). \tag{11}$$

Here, $p^d_{exploitation,i}(t)$ is the dth dimension of the exploitation exemplar for ith particle in $P_{updated}$ at generation t; $p_i^d(t)$ is the dth dimension of the ith particle's position at generation t; r_2 is a random value within (0,1). The pseudo-code of the selection strategy for the exploitation exemplars and updated particles is shown in Algorithm 2 .

Algorithm 2: Pseudo-code of the exploitation exemplars and updated particles selection strategy

Input: Sorted swarm $P(t)$, fitness value vector $fitness$, swarm size N, group size s
Output: The exploitation exemplars $P_{exploitation}$ and the updated particles $P_{updated}$ at tth generation

$P_{exploitation} \leftarrow \emptyset$;
$P_{updated} \leftarrow \emptyset$;
$g_{num} \leftarrow$ Compute the sub-swarm number according to equation (10);
$P_{temp} \leftarrow P$;
for $k = 1$ *to* g_{num} **do**
  if $k < g_{num}$ **then**
    $group_k \leftarrow$ Randomly select s particles in P_{temp};
    Move particles in $group_k$ from P_{temp};
  else
    $group_k \leftarrow P_{temp}$;
  end
end
for $i = 1$ *to* N **do**
  $group_k \leftarrow$ Finding the kth sub-swarm that the ith particle belong to;
  if *the* ith *particle is not the best individual in* $group_k$ **then**
    $p_{exploitation,i}(t) \leftarrow$ the best particle in $group_k$;
    $P_{exploitation} \leftarrow P_{exploitation} \bigcup p_{exploitation,i}(t)$;
    $P_{updated} \leftarrow P_{updated} \bigcup$ the ith particle;
  end
end

As discussed in [6,22], the exploitation ability can be indicated by the fitness differences between exemplars and the updated particles. In the proposed strategy, a swarm with a small s will exploit more solutions, which means small gaps between exemplars and updated particles. This results in a low convergence pressure while more potential solutions can be exploited. Large sub-swarm size s, however, the whole swarm will only exploit few solutions with high qualities leading to increasing fitness differences between exemplars and the updated particles. In other words, increasing the sub-swarm size s results in increasing the convergence pressure.

Therefore, sub-swarm size s can be used to effectively manage the convergence pressure for exploitation. Combining this idea with that a good optimizer should emphasize exploration in the early optimization

stage while paying more attention on exploitation in the later stage [61], an adaptive sub-swarm size adjustment is employed in this paper. To be specific, a set S containing different settings of s will be predefined and the whole optimization process will be divided into $|S|$ stages. In each stage, s is selected from S from small to large according to the cost of *FEs*. For example, if S is {10, 20} and the maximum *FEs* is set to 100, the whole optimization process will be divided into two stages covering *FEs* from 0 to 50 and 51 to 100, respectively, then s should be 10, 20 in each stage, respectively. This means that the sub-swarm size in the proposed exploitation learning strategy will be gradually increasing in order to dynamically adjust the differences between the updated particles and exemplars to manage the convergence pressure.

3.4. APSO-DEE

By embedding the discussed exploration and exploitation formulations into the updating framework presented in (4), a novel learning strategy is proposed which is formulated as:

$$v_i^d(t+1) = \omega v_i^d(t) + \phi r_1(p^d_{exploration,i}(t) - p_i^d(t)) + r_2(p^d_{exploitation,i}(t) - p_i^d(t)) \tag{12}$$

$$p_i^d(t+1) = p_i^d(t) + v_i^d(t+1) \tag{13}$$

where $p_{exploration,i}(t)$ and $p_{exploitation,i}(t)$ are produced by Algorithm 1 and Algorithm 2, respectively; $v_i^d(t)$ and $p_i^d(t)$ are the dth dimension of the ith particle's velocity and position, separately; ω, r_1 and r_2 are set to random values within (0,1); ϕ is set by users. In (12), $\omega v_i^d(t)$ helps ensuring the stability of the algorithm [6] while $\phi r_1(p^d_{exploration,i}(t) - p_i^d(t))$ and $r_2(p^d_{exploitation,i}(t) - p_i^d(t))$ focus on the exploration and exploitation, respectively.

Therefore, the exploration and exploitation are independently managed in different components in the proposed learning strategy, which benefits the control of the balance between exploration and exploitation. Furthermore, by dynamically adjusting the sub-swarm size s during the optimization process, APSO-DEE can get a slow convergence pressure and pay more attention on exploration in the early optimization stage, while just a few solutions with high qualities will be exploited in the later stage resulting in a fast convergence pressure. Therefore, the proposed APSO-DEE is capable to dynamically balance the exploration and exploitation during the optimization process.

In addition, it should be noted that many methods of handling infeasible solutions have been proposed to improve the performance of EAs [62]. However, as the focus of the proposed algorithm is on the decoupling of the exploration and exploitation of cPSO, we chose to implement a simple solution: the decision variables in the particles in APSO-DEE are directly replaced with the corresponding decision space boundaries if they are out of the domain of the decision space. The pseudo-code of APSO-DEE is shown in Algorithm 3 .

Algorithm 3: Pseudo-code of APSO-DEE

```
Input: Swarm size N, set S, parameter φ
Output: gbest: the best solution
Initialize a swarm P;
fitness ← Compute the fitness vector;
t = 1;
while terminal condition is not met do
    Select the sub-swarm size s from S;
    P_exploration, P_updated,1 ← Run Algorithm 1;
    P_exploitation, P_updated,2 ← Run Algorithm 2;
    for i = 1 to N do
        if the ith particle ∈ P_updated,1 ⋂ P_updated,2 then
            Update the ith particle according to (12) and (13);
            Check the feasibility of the ith particle;;
        else
            Pass the ith particle to the next generation without
            doing anything;
        end
    end
    fitness ← Compute the fitness vector;
    gbest ← the current best particle;
    t = t + 1;
end
```

4. Theoretical analysis of APSO-DEE

4.1. Convergence analysis

Convergence stability for EAs is of vital importance. A lot of studies can be found in current literature, several of which are conducted without the stagnation assumption [63,64]. Considering that the exploration and exploitation exemplars in APSO-DEE are selected randomly, this paper investigates the convergence of $E(p(t))$ in APSO-DEE based on the method proposed by [63], where $E(p(t))$ represents the expectation position of an arbitrary particle in APSO-DEE at generation t. According to the theory in [63], the convergence proof is shown as follows.

Since the positions of particles are updated for each dimension independently, the 1−D space convergence stability analysis is presented as follows according to [63]. For simplicity, the updating of a particle at generation t can be rewritten as (14) and (15).

$$v(t+1) = \omega(p(t) - p(t-1)) + \phi r_1(e_1 - p(t)) + r_2(e_2 - p(t)) \tag{14}$$

$$p(t+1) = p(t) + v(t+1) \tag{15}$$

Based on (14) and (15), $p(t+1)$ can be represented by (16), where $l = 1 + \omega - \phi r_1 - r_2$.

$$p(t+1) = lp(t) - \omega p(t-1) + \phi r_1 e_1 + r_2 e_2 \tag{16}$$

Then the expectation of $p(t+1)$ is shown in (17), where μ_ω, μ_{r_1}, μ_{r_2}, μ_{e_1} and μ_{e_2} are the expectation values of the corresponding variables.

$$E(p(t+1)) = E(l)E(p(t)) - \mu_\omega E(p(t-1)) + \phi\mu_{r_1}\mu_{e_1} + \mu_{r_2}\mu_{e_2} \tag{17}$$

Based on (17), a recurrence relation can be obtained as shown in (18).

$$\begin{bmatrix} E(p(t+1)) \\ E(p(t)) \end{bmatrix} = \begin{bmatrix} E(l) & -\mu_\omega \\ 1 & 0 \end{bmatrix} \begin{bmatrix} E(p(t)) \\ E(p(t-1)) \end{bmatrix} + \begin{bmatrix} \phi\mu_{r_1}\mu_{e_1} + \mu_{r_2}\mu_{e_2} \\ 0 \end{bmatrix} \tag{18}$$

For simplicity, we define M as shown in (19).

$$M = \begin{bmatrix} E(l) & -\mu_\omega \\ 1 & 0 \end{bmatrix} \tag{19}$$

According to [63], 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 this paper. Therefore, 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 can be shown in (20).

$$\left|\frac{2 - \phi \pm \sqrt{\phi^2 - 4\phi - 4}}{4}\right| < 1 \tag{20}$$

By analysing (20), the necessary and sufficient condition for convergence of $E(p(t))$ of an arbitrary particle is shown in (21). The detailed analyses of (20) can be found in the appendix.

$$-1 < \phi < 5 \tag{21}$$

Based on this, only $-1 < \phi < 5$ will be employed in this paper to ensure the convergence of $E(p(t))$ of an arbitrary particle in APSO-DEE. However, it must be noted that the above proof does not necessarily ensure the convergence to the global optimum.

4.2. The computational complexity analysis

For the computation complexity analysis for APSO-DEE, the computational complexity proof in [22] is considered. Since all the algorithms should use *FEs* as the terminal criterion in numerical competition, the adopted method assumes using the same swarm size N to compare the computational complexity per generation between different algorithms. With considering that the learning structure of cPSO and APSO-DEE both include three components, only the main additional computational cost introduced by the designed learning strategy at each generation is considered in this paper. According to the proof introduced by Qiang et al. [22], the following three aspects are taken into consideration, to wit, the algorithm structure, the computation cost at each generation, and the required memory cost.

First, APSO-DEE inherits the structure of the cPSO which is convenient for implementation. Second, the extra computational cost at each generation compared to the cPSO is through the addition of Algorithms 1 and 2 as shown in lines 6 and 7 in Algorithm 3. To be specific, in Algorithm 1, it costs $O(Nlog_2(N)+2N)$ to compute *LSD* and identify $P_{exploration}$ and $P_{updated}$; for Algorithm 2, $O(2N)$ is the cost to find the best particle in each sub-swarm and identify $P_{exploitation}$ and $P_{updated}$. Therefore, the additional computational cost for each generation is just $O(Nlog_2(N)+4N)$ compared to cPSO. Finally, because APSO-DEE does not need the information of p_{best}, it has lower memory requirements than cPSO.

In summary, it can be concluded that APSO-DEE remains computationally efficient and it is more conducive to saving memory space than cPSO.

5. Experimental studies

In this part, two widely used large scale optimization test suites provided by CEC 2010 and CEC 2013 are adopted to test APSO-DEE. The benchmark definitions can be found in [65,66]. First, the comparisons between APSO-DEE and six other state-of-the-art large scale optimization EAs are presented to test APSO-DEE; second, we evaluate the effectiveness of the proposed exploration and exploitation learning strategies; finally, for a closer investigation of APSO-DEE, a parameter sensitivity analysis is conducted. All the experiments are executed on a PC with Intel Core i7-8700k CPU, and Microsoft Windows 10 Enterprise 64 bit operating system. The algorithms are written for and run in Matlab 2018a.

Note that for saving computational resources, except for the comparison experiments, the other experiments are conducted based on six typical functions of F_1, F_3, F_6, F_{11}, F_{14}, F_{15} selected from CEC 2013, 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

To confirm the validation of the experiments, APSO-DEE is compared to 6 peer algorithms including CSO [6], SLPSO [31] and DLLSO [22], MA-SW-CHAINs [51], DECC-DG2 [38], and MMO-CC [32]. The first three algorithms are cPSO variants, where DLLSO is proposed in 2018; MA-SW-CHAINs is the winner of the CEC 2010 large scale optimization competition; DECC-DG2 and MMO-CC are two CC framework based EAs. MOS [52] and SHADE-ILS [58] are two competitive methods for LSOPs. However, since only the average results of them can be found and the two algorithms are difficult to be coded for implementation, the comparisons between APSO-DEE and the two algorithms are not made in this paper.

5.1.2. Experimental settings

To ensure a fair comparison, *FEs* is adopted as the terminal condition and the maximum *FEs* is set to $3E+06$ as recommended by Qiang et al. [22]. According to the results in Section 5.4, the APSO-DEE parameters ϕ and swarm size N are set to 0.3 and 1000, respectively. S is set as {2, 4, 8, 10, 20, 25, 40, 50}. For the other six peer algorithms, the parameters are set as recommended in the corresponding reference papers. In addition, as recommended by Qiang et al. [22,32], Wilcoxon rank sum test is adopted to do the statistical analysis between APSO-DEE and the other peer algorithms, where the significant level is set to 0.05 and the p value is the significance factor [22].

5.1.3. Results

Tables 1 and 2 show the experimental results on 1000-dimensional IEEE CEC 2010 and IEEE CEC 2013 benchmark functions. The best results of the average performance are highlighted by gray; " + ", " - ", and " = " above the p values mean that the performance of APSO-DEE is significantly better than, significantly worse than, and statistically equivalent to the compared algorithm on the corresponding benchmark; the p values will be marked in bold font if APSO-DEE performs significantly better than the corresponding algorithms; $w/l/t$ at the bottom of the table represent that how many times APSO-DEE wins/loses/ties in the competitions with comparing to the corresponding algorithms. Note that due to the heavy time consuming, Ma-Sw-CHAINs is only run for 30 times on the CEC 2013 benchmarks, the corresponding results of Ma-Sw-CHAINs on CEC 2010 benchmarks shown in Table 1 are cited from [51], where the Wilcoxon rank sum test between Ma-Sw-CHAINs and APSO-DEE is not conducted.

From the results shown in Table 1 one can find that for the average performance competition, APSO-DEE wins on 10 functions over all other six algorithms out of the 20 benchmarks, and ranks in second on F_1, F_4, F_8, F_{10} and F_{19}. Turn to the 12 multimodal benchmarks of F_2, F_3, F_5, F_6, F_8, F_{10}, F_{11}, F_{13}, F_{15}, F_{16}, F_{18} and F_{20}, APSO-DEE wins on 9 functions. For the rest 8 unimodal benchmarks, APSO-DEE ranks in top three for 5 times. For the 15 partially separable benchmarks of F_4 to F_{18}, APSO-DEE wins on 8 functions and ranks in top three for 12 times. For the fully separable benchmarks of F_{19} and F_{20}, APSO-DEE wins on F_2 and ranks in second on F_{19}.

The Wilcoxon rank sum test results in the bottom of Table 1 also show the competitive performance of APSO-DEE. To be specific, APSO-DEE outperforms CSO, SLPSO, DLLSO, DECC-DG2 and MMO-CC for 19, 18, 16, 16 and 13 times, respectively. Besides, the average performance competition between Ma-Sw-CHAINs and APSO-DEE obviously shows the superior stability of APSO-DEE over Ma-Sw-CHAINs, where APSO-DEE wins Ma-Sw-CHAINs for 15 times.

Table 2 shows the corresponding results on benchmarks from CEC 2013. For the average performance, APSO-DEE wins on 9 benchmarks over all other six algorithms out of the 15 benchmarks and is only slightly outperformed by DLLSO on F_1. With a deep insight, APSO-DEE outperforms other algorithms for 5 benchmarks out of the 8 multimodal functions of F_2, F_3, F_5, F_6, F_7, F_9, F_{10}, F_{12}; for the 7 unimodal functions, APSO-DEE wins for 4 times. For the competition on partially separable benchmarks of F_4 to F_{12}, APSO-DEE wins for 6 times. APSO-DEE also shows competitive results on the overlapping and non-separable benchmarks.

The statistical results obtained by Wilcoxon rank sum test in the bottom of Table 2 also show the competitive performance of APSO-DEE. To be specific, APSO-DEE outperforms CSO, SLPSO, DLLSO, DECC-DG2, MMO-CC and Ma-Sw-CHAINs for 12, 13, 11, 12, 13 and 10 times, respectively.

To show the convergence of these algorithms, the convergence curves of APSO-DEE and the six selected EAs are recorded as shown in Figs. 4 and 5. The results show the competitive convergence of APSO-DEE in comparison with the other algorithms. Especially in the middle and later optimization stages, APSO-DEE has a competitive convergence speed with comparing to other EAs.

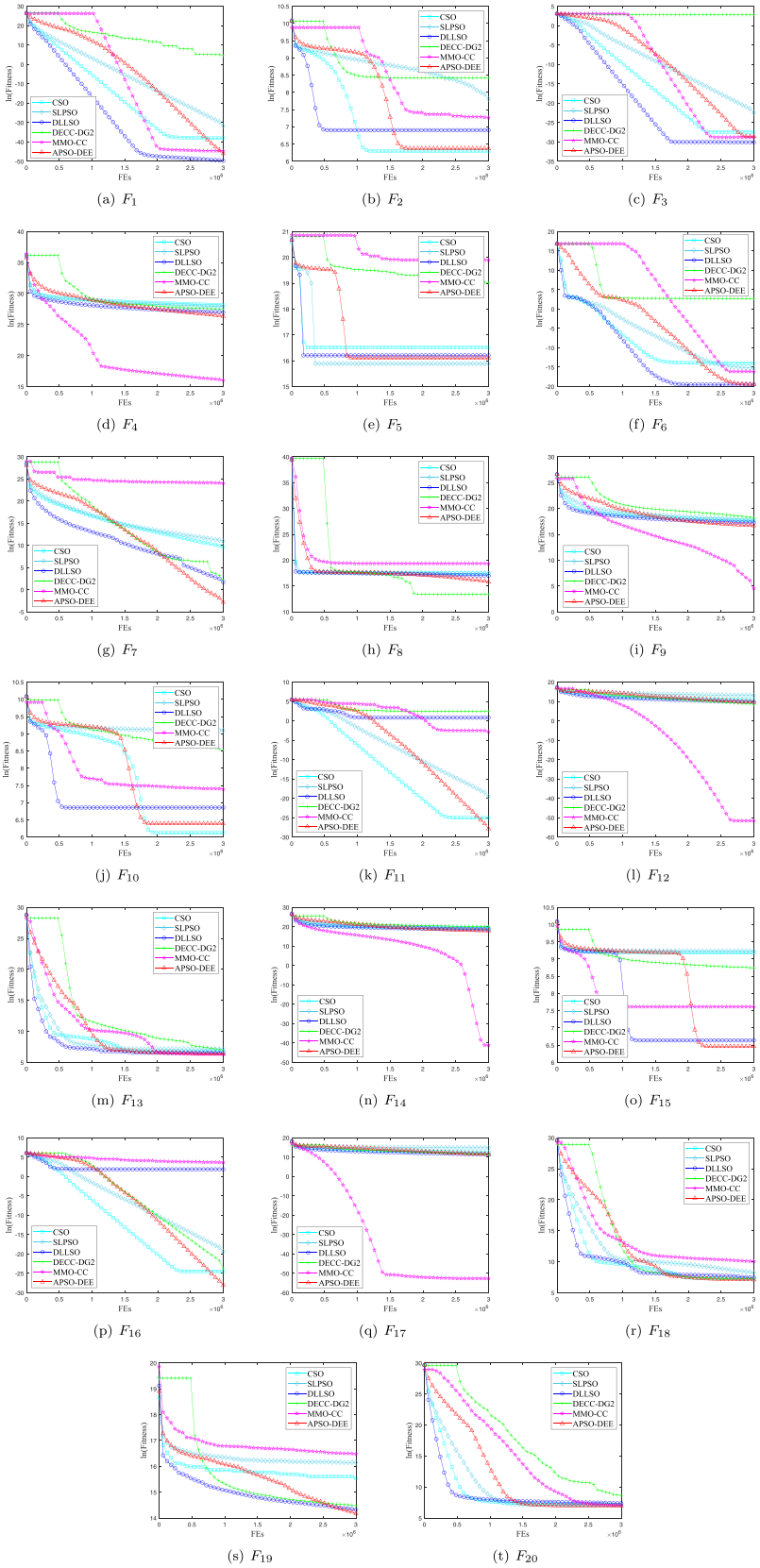
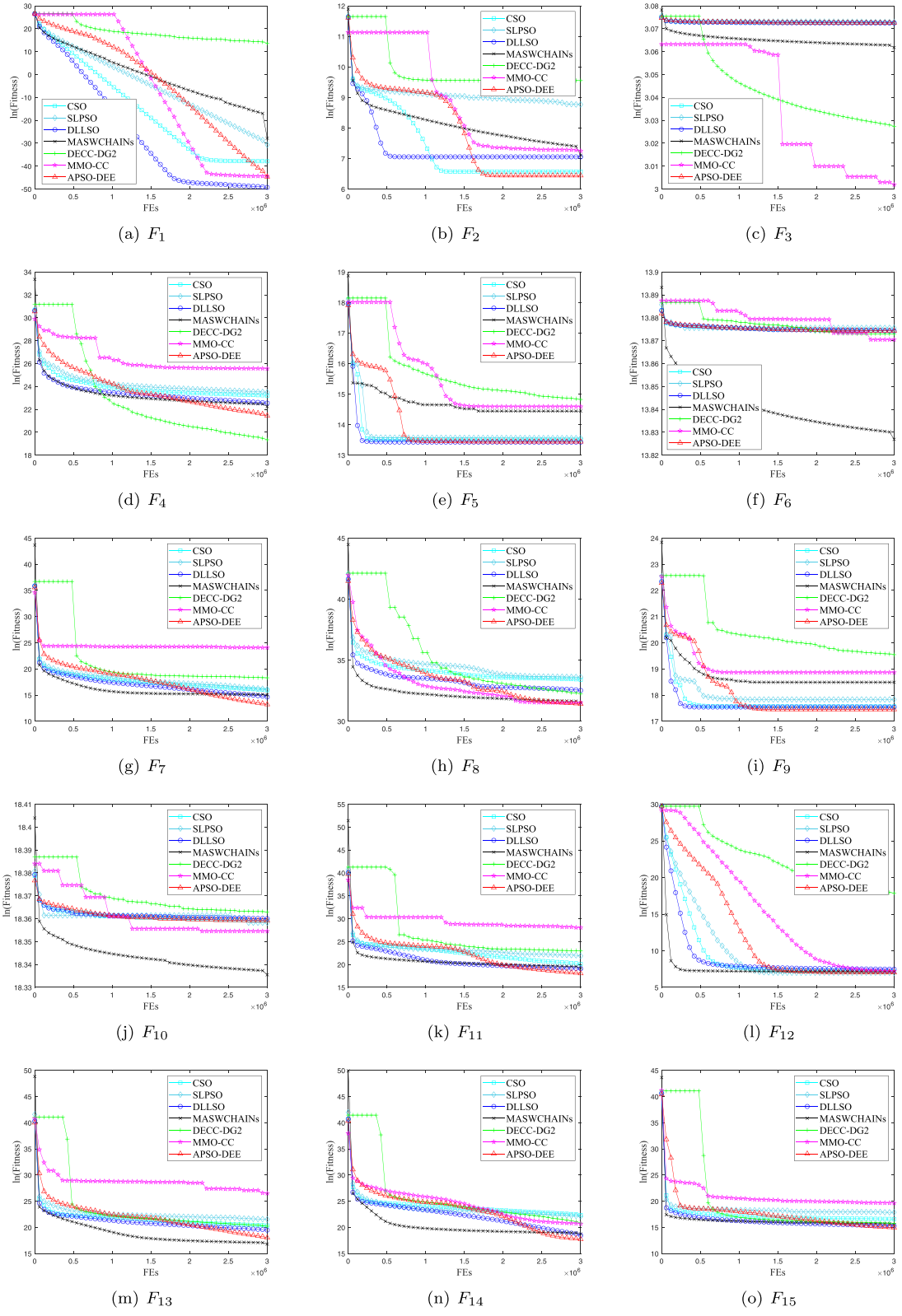


Fig. 4. Convergence profiles of different algorithms obtained on the CEC 2010 test suite with 1000 dimensions. Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.



(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. 5. Convergence profiles of different algorithms obtained on the CEC 2013 test suite 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.

Table 1
The experimental results of 1000-dimensional IEEE CEC 2010 benchmark functions with fitness evaluations of 3e6.

Function	Quality	CSO[6]	SLPSO[31]	DLLSO[22]	DECCDG2[38]	MMOCC[32]	MASWCHAINs[51]	APSO-DEE
F_1	Mean	3.01E-17	1.82E-14	2.64E-22	2.25E+01	4.75E-20	2.10E-14	1.06E-20
	Std	3.17E-19	9.90E-16	8.93E-24	9.76E+00	1.87E-21	1.99E-14	2.55E-22
	p-value	**1.42E-09**$^+$	**1.41E-09**$^+$	1.41E-09$^-$	**1.42E-09**$^+$	**1.42E-09**$^+$	-	-
F_2	Mean	6.28E+02	3.01E+03	9.69E+02	4.45E+03	1.42E+03	8.10E+02	5.86E+02
	Std	5.13E+00	7.85E+01	7.95E+00	2.65E+01	1.49E+01	5.88E+01	4.43E+00
	p-value	**1.49E-06**$^+$	**1.41E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	-	-
F_3	Mean	1.20E-12	1.76E-10	9.40E-14	1.66E+01	3.14E-13	7.28E-13	3.83E-13
	Std	3.76E-15	3.98E-12	1.43E-15	5.17E-02	3.39E-15	3.40E-13	6.86E-15
	p-value	**1.34E-09**$^+$	**1.39E-09**$^+$	1.10E-09$^-$	**1.39E-09**$^+$	2.71E-09$^-$	-	-
F_4	Mean	1.51E+12	1.31E+12	7.89E+11	7.75E+11	7.27E+06	3.53E+11	2.75E+11
	Std	3.51E+10	3.96E+10	3.20E+10	5.87E+10	2.12E+05	3.12E+10	7.76E+09
	p-value	**1.42E-09**$^+$	**1.41E-09**$^+$	**1.42E-09**$^+$	**9.29E-09**$^+$	1.42E-09$^-$	-	-
F_5	Mean	1.25E+07	1.12E+07	1.33E+07	1.38E+08	3.42E+08	1.68E+08	9.91E+06
	Std	3.90E+05	6.42E+05	5.57E+05	4.05E+06	2.89E+07	1.04E+08	3.96E+05
	p-value	**7.89E-04**$^+$	1.30E-01$^=$	**4.26E-06**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	-	-
F_6	Mean	8.70E-07	2.33E-07	2.27E-01	1.54E+01	5.96E-01	8.14E+04	3.73E-09
	Std	6.39E-09	1.48E-09	1.16E-01	8.67E-02	3.30E-01	2.84E+05	9.06E-12
	p-value	**1.42E-09**$^+$	**1.41E-09**$^+$	**1.92E-09**$^+$	**1.42E-09**$^+$	**1.40E-09**$^+$	-	-
F_7	Mean	1.78E+04	6.98E+04	1.50E+03	1.26E+02	1.90E+10	1.03E+02	6.18E-02
	Std	1.19E+03	2.76E+03	1.32E+03	5.07E+01	1.37E+09	8.70E+01	7.92E-03
	p-value	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	-	-
F_8	Mean	3.69E+07	3.85E+07	2.32E+07	7.98E+05	9.64E+07	1.41E+07	7.01E+06
	Std	1.38E+04	1.98E+06	4.51E+04	3.19E+05	4.52E+07	3.68E+07	3.18E+05
	p-value	**1.42E-09**$^+$	**1.13E-09**$^+$	**1.42E-09**$^+$	1.42E-09$^-$	4.73E-01$^=$	-	-
F_9	Mean	4.89E+07	5.71E+07	3.89E+07	6.86E+07	1.01E+02	1.41E+07	1.75E+07
	Std	5.36E+05	9.24E+05	7.41E+05	1.54E+06	1.76E+01	1.15E+06	2.07E+05
	p-value	**1.42E-09**$^+$	**1.41E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	1.42E-09$^-$	-	-
F_{10}	Mean	4.66E+02	8.91E+03	8.96E+02	4.73E+03	1.69E+03	2.07E+03	5.98E+02
	Std	4.15E+00	2.01E+01	9.80E+00	2.59E+01	1.10E+01	1.44E+02	4.23E+00
	p-value	1.42E-09$^-$	**1.39E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	-	-
F_{11}	Mean	1.32E-11	3.29E-09	3.28E+00	1.07E+01	3.72E+00	3.80E+01	8.27E-13
	Std	5.49E-14	1.45E-10	7.32E-01	1.72E-01	1.41E+00	7.35E+00	2.04E-14
	p-value	**1.41E-09**$^+$	**1.41E-09**$^+$	**2.56E-08**$^+$	**1.41E-09**$^+$	**1.41E-09**$^+$	-	-
F_{12}	Mean	4.97E+04	5.40E+05	1.44E+04	4.48E+03	3.64E-23	3.62E-06	1.62E+04
	Std	6.14E+02	1.09E+04	1.40E+02	2.08E+02	1.70E-24	5.92E-07	3.86E+02
	p-value	**1.42E-09**$^+$	**1.41E-09**$^+$	1.43E-04$^-$	1.42E-09$^-$	1.42E-09$^-$	-	-
F_{13}	Mean	1.21E+03	1.23E+03	9.31E+02	1.51E+03	1.48E+04	1.25E+03	5.69E+02
	Std	1.02E+02	9.75E+01	7.68E+01	1.90E+02	9.08E+03	5.72E+02	2.47E+01
	p-value	**1.99E-07**$^+$	**5.02E-07**$^+$	**5.12E-06**$^+$	**7.38E-09**$^+$	**1.65E-06**$^+$	-	-
F_{14}	Mean	1.74E+08	2.71E+08	1.30E+08	4.49E+08	5.68E-19	3.11E+07	5.95E+07
	Std	1.48E+06	2.88E+06	1.86E+06	5.73E+06	1.39E-19	1.93E+06	5.51E+05
	p-value	**1.42E-09**$^+$	**1.41E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	1.42E-09$^-$	-	-
F_{15}	Mean	9.65E+03	1.02E+04	8.45E+02	6.07E+03	2.02E+03	2.74E+03	6.41E+02
	Std	7.72E+00	1.33E+01	9.21E+00	1.65E+01	2.22E+01	1.22E+02	4.08E+00
	p-value	**1.42E-09**$^+$	**7.98E-10**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	-	-
F_{16}	Mean	2.31E-11	3.73E-09	3.93E+00	5.37E-11	2.64E+01	9.98E+01	7.16E-13
	Std	7.54E-14	1.16E-10	4.93E-01	8.95E-13	2.76E+00	1.40E+01	2.23E-14
	p-value	**1.41E-09**$^+$	**1.41E-09**$^+$	**2.55E-08**$^+$	**1.40E-09**$^+$	**1.41E-09**$^+$	-	-
F_{17}	Mean	2.31E+05	2.54E+06	7.91E+04	7.29E+04	5.07E-23	1.24E+00	9.37E+04
	Std	2.07E+03	3.62E+04	8.99E+02	1.11E+03	7.59E-24	1.25E-01	1.34E+03
	p-value	**1.42E-09**$^+$	**1.41E-09**$^+$	1.84E-08$^-$	2.57E-09$^-$	1.42E-09$^-$	-	-
F_{18}	Mean	2.39E+03	2.84E+03	2.55E+03	1.24E+03	3.32E+04	1.30E+03	1.21E+03
	Std	1.57E+02	3.83E+02	1.17E+02	2.45E+01	4.97E+03	4.36E+02	4.43E+01
	p-value	**2.57E-08**$^+$	**4.00E-08**$^+$	**2.29E-09**$^+$	1.40E-01$^=$	**1.42E-09**$^+$	-	-
F_{19}	Mean	5.79E+06	1.07E+07	1.72E+06	1.88E+06	1.58E+07	2.85E+05	1.42E+06
	Std	6.03E+04	1.61E+05	2.30E+04	1.37E+04	3.05E+05	1.78E+04	1.37E+04
	p-value	**1.42E-09**$^+$	**1.40E-09**$^+$	**2.57E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	-	-
F_{20}	Mean	1.29E+03	1.04E+03	1.71E+03	9.86E+03	1.66E+03	1.07E+03	1.02E+03
	Std	2.39E+01	1.28E+01	2.47E+01	2.63E+03	4.53E+02	7.29E+01	9.89E+00
	p-value	**1.59E-09**$^+$	1.86E-01$^=$	**1.41E-09**$^+$	**1.41E-09**$^+$	**2.17E-03**$^+$	-	-
w/l/t		19/1/0	18/0/2	16/0/4	16/3/1	13/6/1	-/-/-	**-**

Although APSO-DEE is not able to outperform all the algorithms in current literature such as MOS [52] and SHADE-ILS [58] by comparing the average results on CEC 2013 benchmarks, APSO-DEE holds two main advantages. On the one hand, APSO-DEE is designed with a simpler structure; on the other hand, the decoupling of exploration and exploitation in cPSO is solved in APSO-DEE, which provides a novel perspective for further studying PSO as well as other kinds of EAs.

In conclusion, APSO-DEE performs competitive in large scale optimization both in results and convergence speed. This is due to the proposed learning strategies and the adaptive multi-swarm size adjustment. On the one hand, APSO-DEE searches the decision space with a low convergence pressure and emphasizes more on exploration in the early optimization stage, which helps APSO-DEE potentially find more promising solutions. On the other hand, the convergence pressure will be increased in the later stage focusing APSO-DEE on refining the searched promising solutions. Therefore, APSO-DEE achieves a reasonable balance between exploration and exploitation in the whole optimization process.

Table 2
The experimental results of 1000-dimensional IEEE CEC 2013 benchmark functions with fitness evaluations of 3e6.

Function	Quality	CSO[6]	SLPSO[31]	DLLSO[22]	DECCDG2[38]	MMOCC[32]	MASWCHAINs[51]	APSO-DEE
F_1	Mean	3.68E-17	3.70E-14	4.32E-22	8.65E+05	4.82E-20	8.49E-13	4.14E-20
	Std	3.89E-19	1.44E-15	3.88E-23	2.18E-13	2.27E+05	1.30E-21	3.62E-21
	p-value	**1.41E-09**$^+$	**1.42E-09**$^+$	1.41E-09$^-$	**1.41E-09**$^+$	**1.27E-02**$^+$	**1.42E-09**$^+$	-
F_2	Mean	7.07E+02	6.70E+03	1.15E+03	1.41E+04	1.51E+03	1.22E+03	6.31E+02
	Std	7.17E+00	4.98E+01	1.31E+01	3.03E+02	8.43E+00	2.28E+01	4.49E+00
	p-value	**9.29E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	-
F_3	Mean	2.16E+01	2.16E+01	2.16E+01	2.06E+01	2.01E+01	2.14E+01	2.16E+01
	Std	1.39E-03	1.14E-03	1.23E-03	1.69E-03	2.36E-03	1.12E-02	8.95E-04
	p-value	4.73E-01$^=$	5.01E-02$^=$	1.07E-01$^=$	1.42E-09$^-$	1.42E-09$^-$	1.42E-09$^-$	-
F_4	Mean	1.14E+10	1.20E+10	5.99E+09	2.51E+08	5.15E+11	4.58E+09	2.05E+09
	Std	2.65E+08	5.54E+08	2.98E+08	1.89E+07	9.71E+10	4.91E+08	5.73E+07
	p-value	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	1.29E-09$^-$	**1.42E-09**$^+$	**1.19E-03**$^+$	-
F_5	Mean	7.38E+05	7.58E+05	7.30E+05	2.74E+06	2.42E+06	1.87E+06	6.50E+05
	Std	2.64E+04	2.14E+04	1.98E+04	5.66E+04	1.14E+05	6.13E+04	2.18E+04
	p-value	**4.34E-03**$^+$	**6.42E-05**$^+$	**1.67E-03**$^+$	**1.16E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	-
F_6	Mean	1.06E+06	1.06E+06	1.06E+06	1.06E+06	1.06E+06	1.01E+06	1.06E+06
	Std	1.91E+02	1.64E+02	2.31E+02	4.50E+02	6.41E+02	3.06E+03	1.82E+02
	p-value	3.13E-01$^=$	**1.56E-03**$^+$	4.73E-01$^=$	6.08E-03$^-$	**3.19E-03**$^+$	1.42E-09$^-$	-
F_7	Mean	8.16E+06	1.73E+07	1.74E+06	8.93E+07	1.28E+10	3.45E+06	5.49E+05
	Std	4.77E+05	1.49E+06	1.67E+05	7.16E+06	1.07E+09	2.53E+05	4.59E+04
	p-value	**1.42E-09**$^+$	**1.42E-09**$^+$	**5.55E-08**$^+$	**1.40E-07**$^+$	**1.42E-09**$^+$	**5.56E-07**$^+$	-
F_8	Mean	3.15E+14	2.89E+14	1.17E+14	1.01E+14	1.54E+14	4.85E+13	4.40E+13
	Std	1.11E+13	1.75E+13	7.87E+12	1.31E+13	4.45E+13	2.03E+12	2.27E+12
	p-value	**1.42E-09**$^+$	**1.42E-09**$^+$	**4.00E-08**$^+$	**9.61E-07**$^+$	**3.84E-04**$^+$	1.57E-01$^=$	-
F_9	Mean	4.42E+07	4.44E+07	4.32E+07	3.08E+08	1.76E+08	1.07E+08	3.80E+07
	Std	1.56E+06	1.47E+06	1.28E+06	1.39E+07	7.03E+06	3.36E+06	1.23E+06
	p-value	**8.32E-03**$^+$	**2.81E-03**$^+$	**8.32E-03**$^+$	**1.40E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	-
F_{10}	Mean	9.40E+07	9.43E+07	9.40E+07	9.44E+07	9.38E+07	9.18E+07	9.40E+07
	Std	4.23E+04	3.99E+04	4.63E+04	5.82E+04	1.02E+05	2.12E+05	4.25E+04
	p-value	6.55E-01$^=$	**5.94E-04**$^+$	6.84E-01$^=$	**3.29E-05**$^+$	4.36E-02$^-$	1.80E-09$^-$	-
F_{11}	Mean	3.56E+08	9.98E+09	1.82E+08	9.93E+09	5.66E+12	2.19E+08	6.85E+07
	Std	1.47E+07	1.82E+09	9.42E+06	3.26E+09	1.09E+12	5.96E+06	2.80E+06
	p-value	**1.42E-09**$^+$	**1.42E-09**$^+$	**2.29E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	-
F_{12}	Mean	1.38E+03	1.13E+03	1.78E+03	5.81E+07	1.14E+11	1.25E+03	1.12E+03
	Std	2.40E+01	2.12E+01	3.06E+01	1.53E+07	6.32E+10	2.11E+01	1.27E+01
	p-value	**1.46E-08**$^+$	8.46E-01$^=$	**1.42E-09**$^+$	**1.05E-09**$^+$	**7.42E-03**$^+$	**1.38E-05**$^+$	-
F_{13}	Mean	8.05E+08	2.05E+09	2.89E+08	6.03E+08	1.32E+12	1.98E+07	6.74E+07
	Std	6.56E+07	2.13E+08	2.41E+07	2.69E+07	2.88E+11	3.64E+05	6.08E+06
	p-value	**1.42E-09**$^+$	**1.42E-09**$^+$	**7.38E-09**$^+$	**1.09E-09**$^+$	**1.42E-09**$^+$	2.30E-08$^-$	-
F_{14}	Mean	6.95E+09	1.60E+10	9.21E+07	1.11E+09	4.12E+11	1.36E+08	4.94E+07
	Std	9.23E+08	1.62E+09	1.39E+07	2.10E+08	1.21E+11	4.22E+06	3.77E+06
	p-value	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.95E-04**$^+$	**1.42E-09**$^+$	**8.29E-07**$^+$	**2.90E-09**$^+$	-
F_{15}	Mean	1.65E+07	6.68E+07	4.25E+06	7.11E+06	4.05E+08	5.71E+06	3.07E+06
	Std	2.20E+05	1.01E+06	5.52E+04	2.70E+05	1.91E+07	1.51E+05	5.37E+04
	p-value	**1.42E-09**$^+$	**1.42E-09**$^+$	**1.80E-09**$^+$	**9.30E-10**$^+$	**1.42E-09**$^+$	**1.42E-09**$^+$	-
w/l/t		12/0/3	13/0/2	11/1/3	12/3/0	13/2/0	10/4/1	-

5.2. The effectiveness of the exploration learning strategy

In this section, several experiments are conducted to demonstrate the diversity preservation ability of the proposed exploration learning strategy. As recommended by Qiang et al. [22], the diversity is computed according to (22) and (23).

In the experiments, ϕ is varied between 0.3 to 0.6 with a step of 0.1 while the swarm size N and sub-swarm size s are set to 1000 and 10, respectively, to show the impacts of ϕ on swarm diversity. The average results of 30 independent runs on each function are shown in Fig. 6.

$$Std(P) = \frac{1}{N}\sum_{i=1}^{N}\sqrt{\sum_{d=1}^{D}(p_i^d - \bar{p}_i^d)^2} \tag{22}$$

$$\bar{p}^d = \frac{1}{N}\sum_{i=1}^{N} x_i^d \tag{23}$$

Fig. 6 clearly confirms that the diversity increases with increasing of ϕ on all the six functions. This proves the applicability of the proposed diversity control method for diversity preservation, which reflects the effectiveness of the proposed exploration learning strategy. With these findings, it is clear that the proposed strategy is able to manage the diversity for exploration with effectiveness.

5.3. The effectiveness of the exploitation learning strategy

To investigate the effectiveness of the proposed exploitation learning strategy, the fitness curves are shown in Fig. 7. In the experiments, the swarm size N and ϕ are set to 1000 and 0.3, respectively. The sub-swarm sizes s are {2, 10, 20, 60} to show the influences on the convergence speed of s. Based on our previous findings, the convergence speed differences can not be shown clearly by the convergence curves obtained with too large value of *FEs*. Therefore, we show the results obtained with *FEs* of $1.2E+05$ in Fig. 7.

Fig. 7 shows clearly that (except for F_3) the convergence speed increases with increasing sub-swarm size s. This can be explained by that with the increasing of s, the exploitation exemplars will get more promising, and more particles tend to exploit around one exploitation exemplar leading to a higher convergence pressure. Therefore, the convergence pressure in APSO-DEE for exploitation during the optimization process can be effectively controlled by using the proposed adaptive sub-swarm

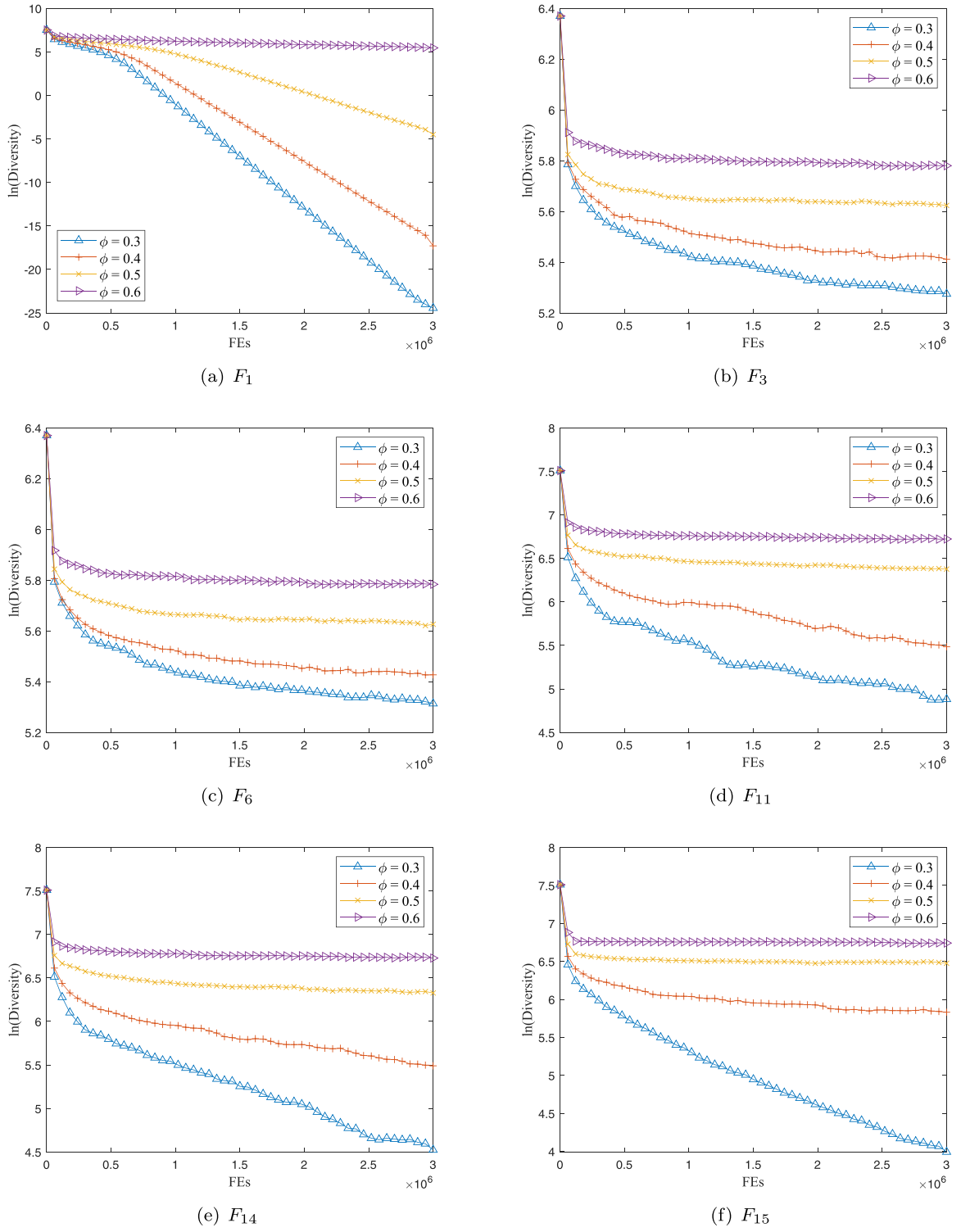


Fig. 6. Swarm diversity comparisons on the selected six functions with FEs of $3E+06$ ($N = 1000$, $s = 10$). Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.

size adjustment which dynamically changes the sub-swarm size from small to large.

5.4. Parametersensitivity analyses

Due to the adopted adaptive sub-swarm size adjustment, APSO-DEE only has two parameters, N and ϕ. For a better understanding of the proposed APSO-DEE, the sensitivity analyses of the swarm size N and parameter ϕ are presented.

5.4.1. *Swarm size* N

Fig. 8 shows the results of APSO-DEE on the six selected functions with different settings of N, where ϕ is set to 0.3. Apparently, except on F_1, APSO-DEE is not sensitive to the settings of N. It is noteworthy that a setting of N to {800, 1000, 1200} outperforms other settings in

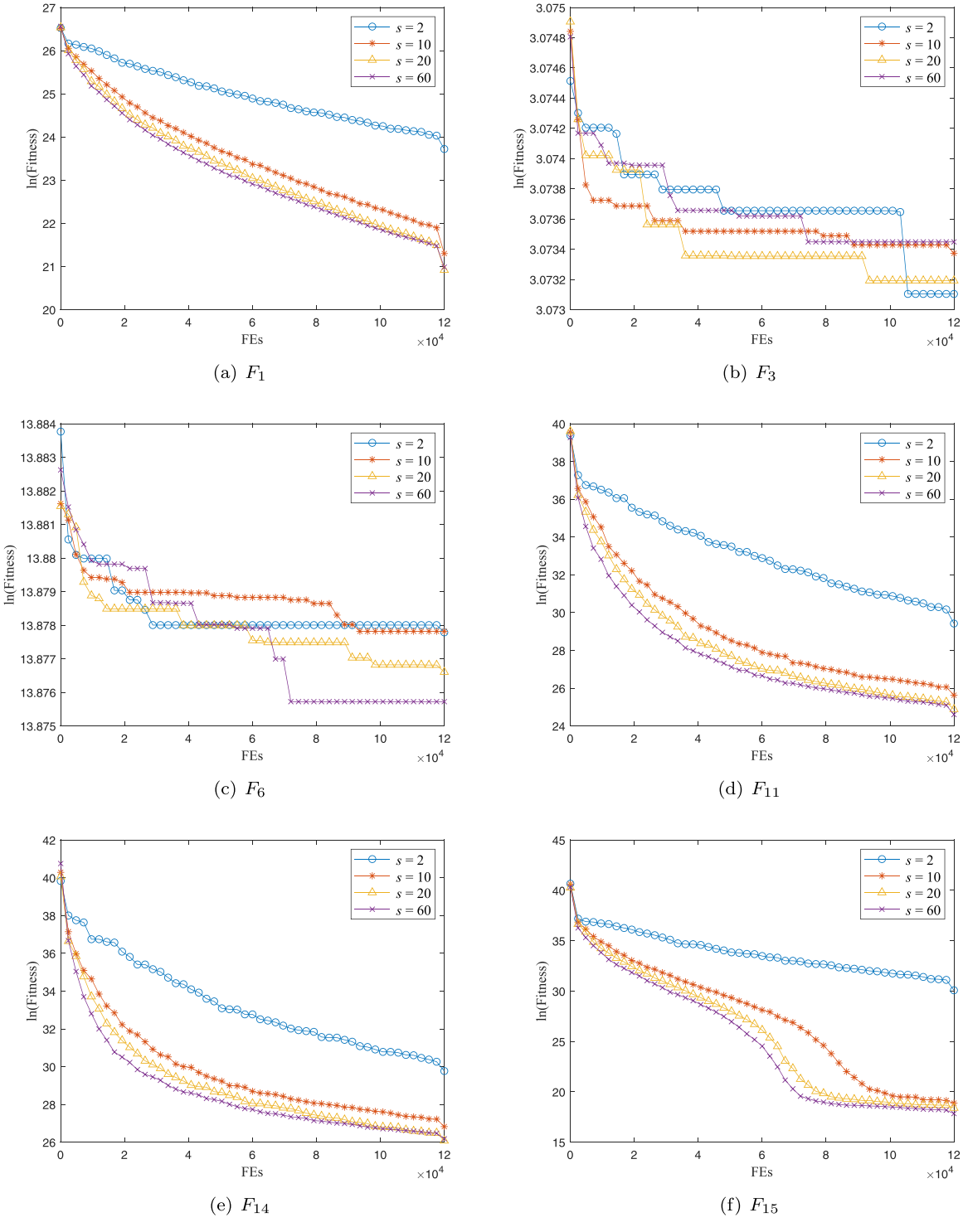


Fig. 7. Fitness comparisons on the selected six functions with FEs of $1.2E+05$ ($N = 1000$, $\phi = 0.3$). Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.

most of the cases. The Friedman ranking test is conducted and the results are shown in Table 3 which demonstrate that N of {800, 1000, 1200} is more suitable for APSO-DEE. Taking into consideration that a large swarm size can ensure the concurrency of EAs, N of {1000, 1200} are suggested in this paper.

Table 3
The Friedman ranking test results on N.

N	Friedman ranking
400	3.33
600	3.67
800	2.50
1000	2.83
1200	2.67

5.4.2. Exploration parameter ϕ

Fig. 9 shows the results of APSO-DEE on the six selected functions with different settings of ϕ, where N is set to 1000. As it depicted in

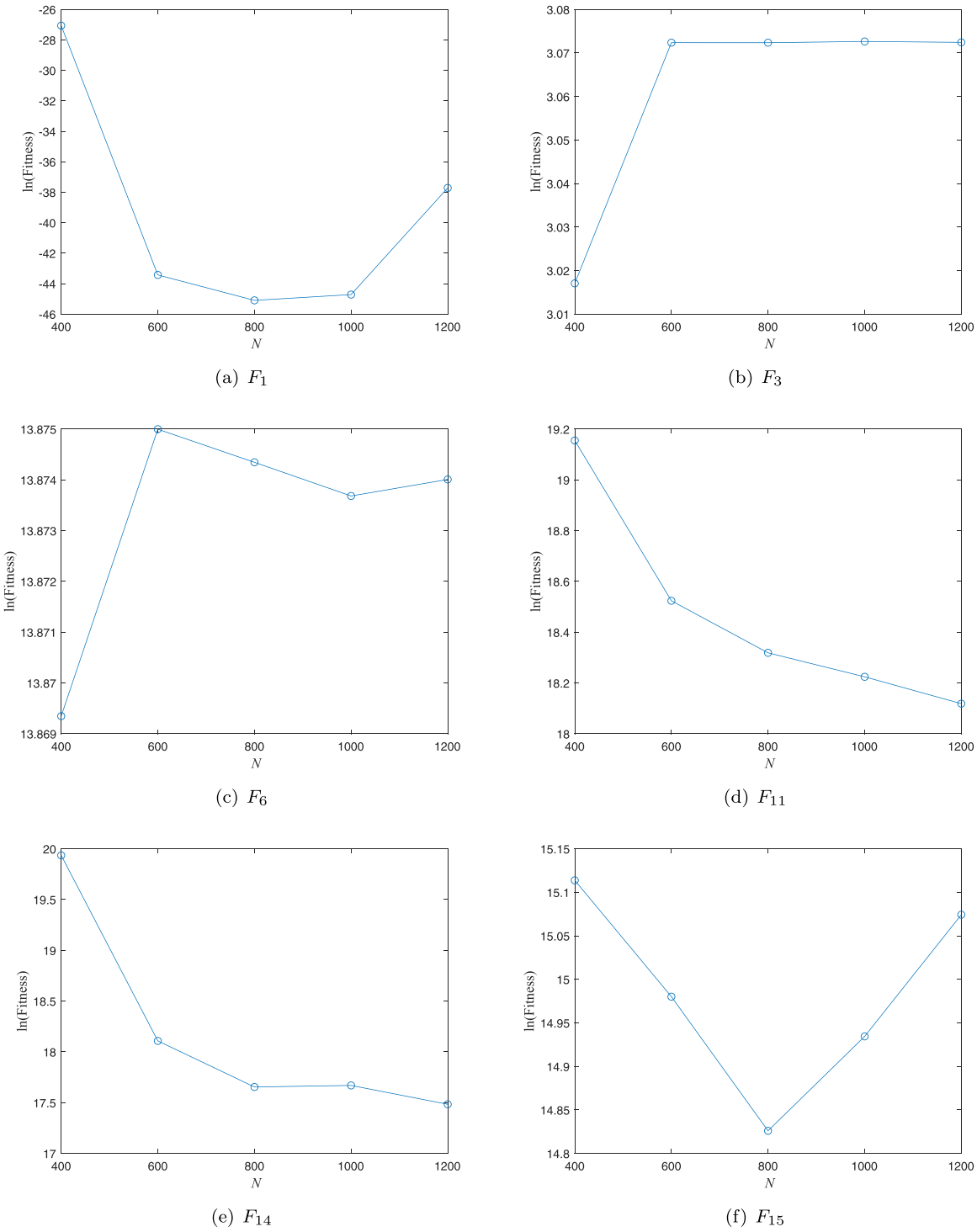


Fig. 8. Average optimization results obtained under different settings of N ($\phi = 0.3$). Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.

Fig. 9, the results are sensitive to ϕ on F_1, where a small ϕ is more suitable because F_1 is a simple convex function for which the exploitation should be emphasized more in the optimization process. For the other five more complex problems, APSO-DEE is not sensitive to ϕ. Setting ϕ to {0.1, 0.2, 0.3} outperforms others in most cases. A Friedman ranking test is also conducted for ϕ. The results are shown in Table 4, which indicates the competitiveness of APSO-DEE with the ϕ of {0.1, 0.2, 0.3}. To ensure a balance between exploration and exploitation, $\phi = 0.3$ is suggested in this paper.

Table 4
The Friedman ranking test results on ϕ.

ϕ	Friedman ranking
0.1	2.83
0.2	2.33
0.3	2.83
0.4	3.33
0.5	4.50
0.6	5.17

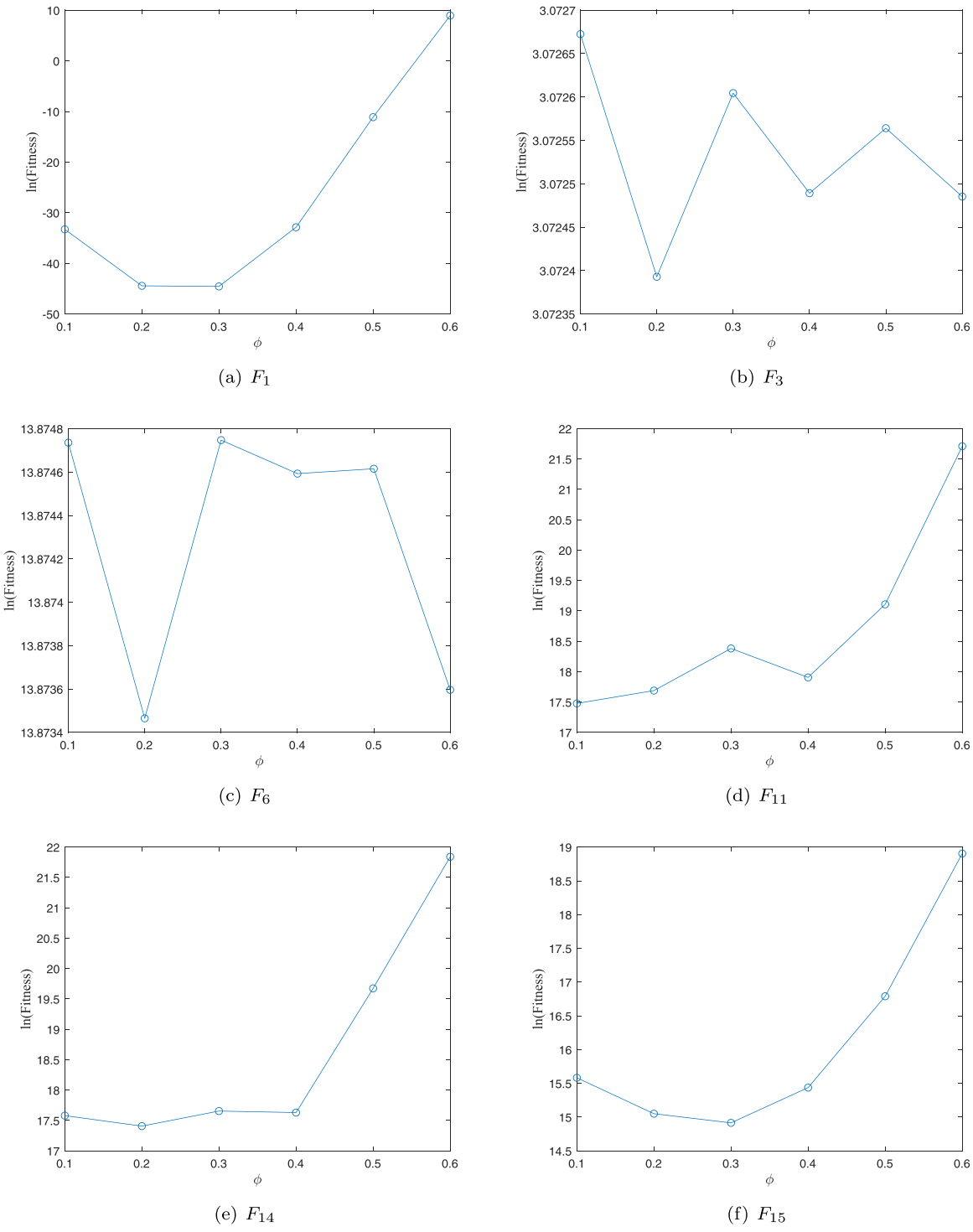


Fig. 9. Average optimization results obtained under different settings of ϕ (N = 1000). Note that the logarithmic values of the average results obtained by 30 independent runs are shown in the above figures for clarity.

5.4.3. Correlation between N *and* ϕ

Although the performance of APSO-DEE is not sensitive to both N and ϕ in most cases as shown in the above results, it is worth to show the correlation between N and ϕ to further investigate the characteristics of APSO-DEE. The experiments results of different combinations of N and ϕ obtained by 30 independent runs are shown in Table 5. From the obtained results one can see that when ϕ is larger than 0.5, N should be adjusted down to 400 to get promising solutions; as ϕ decreases, good solutions can be searched by increasing N. The reason is that a large swarm size will result in a high diversity [22], therefore, a large swarm size should cooperate with a small ϕ to balance the exploration and exploitation in the search process. In summary, the findings indicate that to get good results, a large N is necessary when ϕ is small and vice versa.

Table 5
The experimental results obtained by APSO-DEE, where N is selected in {400, 600, 800, 1000, 1200} and ϕ varies from 0.2 to 0.6. The best results of each combination of N and ϕ are marked in bold font.

ϕ	Func.	N				
		400	600	800	1000	1200
0.2	F_1	5.1171E-15	1.0491E-19	**2.6963E-20**	2.7851E-20	2.6751E-18
	F_3	**2.0558E+01**	2.1594E+01	2.1599E+01	2.1596E+01	2.1591E+01
	F_6	1.0608E+06	1.0608E+06	1.0605E+06	1.0611E+06	**1.0596E+06**
	F_{11}	1.1695E+08	8.6285E+07	4.6035E+07	**3.8061E+07**	4.9421E+07
	F_{14}	3.1277E+08	5.3370E+07	**3.1169E+07**	3.3580E+07	4.3760E+07
	F_{15}	3.6582E+06	3.3575E+06	3.2803E+06	3.2796E+06	3.8680E+06
0.3	F_1	4.8585E-13	1.4651E-19	**3.3858E-20**	5.2154E-20	2.8038E-17
	F_3	**2.0522E+01**	2.1593E+01	2.1599E+01	2.1596E+01	2.1596E+01
	F_6	1.0603E+06	1.0610E+06	**1.0599E+06**	1.0614E+06	1.0606E+06
	F_{11}	1.9023E+08	8.0700E+07	**6.4855E+07**	6.8369E+07	9.2170E+07
	F_{14}	1.4871E+08	9.4878E+07	4.4813E+07	**4.2777E+07**	4.5182E+07
	F_{15}	3.8513E+06	3.6189E+06	**3.0510E+06**	3.1524E+06	3.5136E+06
0.4	F_1	1.5977E-16	**1.6045E-19**	1.0472E-18	4.2024E-15	1.6754E-11
	F_3	**2.0511E+01**	2.1596E+01	2.1601E+01	2.1599E+01	2.1593E+01
	F_6	**1.0598E+06**	1.0613E+06	1.0603E+06	1.0617E+06	1.0617E+06
	F_11	8.9310E+07	5.9223E+07	**4.6339E+07**	6.5600E+07	9.3054E+07
	F_14	5.6938E+07	4.3206E+07	**3.9291E+07**	8.2101E+07	7.1409E+07
	F_15	4.6439E+06	**3.7272E+06**	4.3805E+06	4.7951E+06	6.4888E+06
0.5	F_1	**3.1484E-14**	6.1733E-12	1.6605E-08	1.1495E-05	3.0987E-03
	F_3	**2.0859E+01**	2.1597E+01	2.1592E+01	2.1595E+01	2.1598E+01
	F_6	1.0600E+06	1.0597E+06	1.0614E+06	1.0611E+06	**1.0591E+06**
	F_{11}	**5.7373E+07**	9.3652E+07	1.2903E+08	1.7441E+08	2.8514E+08
	F_{14}	**4.4626E+07**	8.2921E+07	1.4374E+08	4.0139E+08	8.0410E+08
	F_{15}	**6.5776E+06**	9.4012E+06	1.3775E+07	1.9885E+07	2.9165E+07
0.6	F_1	**1.9681E-02**	4.4153E+01	1.8009E+02	6.2036E+03	6.0680E+04
	F_3	**2.1569E+01**	2.1591E+01	2.1599E+01	2.1596E+01	2.1598E+01
	F_6	1.0617E+06	1.0610E+06	**1.0607E+06**	1.0612E+06	1.0613E+06
	F_{11}	**5.7094E+08**	7.9759E+08	1.3773E+09	2.8978E+09	4.4070E+09
	F_{14}	**3.0481E+08**	8.7425E+08	1.1977E+09	3.3076E+09	5.1836E+09
	F_{15}	**2.1710E+07**	5.8421E+07	1.0898E+08	1.7785E+08	2.0098E+08

6. Conclusion

This paper addresses the problem that exploration and exploitation are highly coupled in the existing learning strategies and proposes strategies for balancing these two functions. First, a novel learning structure decoupling exploration and exploitation is proposed in this paper, where the exploration and exploitation operate in different components. This novel structure decouples exploration and exploitation at the learning structure level. Second, two novel learning strategies are proposed. On the one hand, we first propose a novel local sparseness degree measurement for exploration which considers both the distribution and congestion of particles. Furthermore, an exploration strategy is put forward from the perspective of leading particles to the sparse areas. On the other hand, an adaptive multi-swarm strategy is presented for exploitation to dynamically divide the whole swarm into several sub-swarms during the optimization process. In each sub-swarm, the best particle is selected as the exploitation exemplar for others to emphasize exploitation. By this means, the differences between particles and exemplars can be effectively adjusted, resulting in an explicit management of the convergence pressure during a run. Finally, by embedding the two novel learning strategies into the proposed learning structure, APSO-DEE is proposed in this paper, where the exploration and exploitation are decoupled in the learning strategies and can be dynamically balanced during the optimization process.

Furthermore, the convergence stability and computation efficiency of the proposed algorithm are proved theoretically. Comprehensive experiments demonstrate that the proposed learning strategies are able to effectively adjust the swarm diversity and convergence speed, respectively. The comparison results show the competitiveness of the proposed algorithm.

In the future work, more intelligent strategies of adjusting the sub-swarm size s and parameter ϕ will be investigated for better balance of exploration and exploitation. The proposed local sparseness degree measurement is independent of optimization problems and the operations of cPSO and its variants, it is also an interesting study to test its effectiveness in other kinds of EAs. Besides, Due to the simplicities of the proposed learning structure and the novel learning strategies, it is flexible to hybridize APSO-DEE with other optimization techniques. The algorithms are also will be investigated to many other challenge optimization problems in our future work.

Author statement

The main revisions are listed as following.

- More relevant articles are reviewed. The Introduction now is enriched with more explanations for canonical PSO, as well as the differences between small scale optimization and large scale optimization;
- Several references are added for a comprehensive review on the related work;
- Experiments of the statistical are conducted based on Wilcoxon;
- More statistical work are conducted for the parameter sensitivity analysis.
- More work on theoretical analysis are reviewed.
- All of the figures are re-plotted for clearer readability.

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.

Acknowledgment

This work is supported by the National Natural Science Foundation of China under Grant Numbers 71771176 and 61503287, Natural Science Foundation of Shanghai, China under Grant Numbers 19ZR1479000, 20692191200 Fundamental Research Fund for the China Central Universities under Grant Number 22120190202, China Scholarship Council under File Number 201906260030, Science and Technology Project on Winter Olympic, China, under Grant Number 2018YFF0300505.

Appendix A. Appendix

The detailed analyses of (20) are presented as follows.

1. If the eigenvalues of (20) are complex roots.
Then

$$\phi^2 - 4\phi - 4 < 0. \tag{A.1}$$

Then

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

Given that

$$|2 - \phi \pm \sqrt{\phi^2 - 4\phi - 4}| = |2 - \phi \pm \sqrt{4 + 4\phi - \phi^2}\sqrt{-1}| \tag{A.3}$$

$$|2 - \phi \pm \sqrt{\phi^2 - 4\phi - 4}| = |2 - \phi \pm \sqrt{4 + 4\phi - \phi^2}i| \tag{A.4}$$

$$|2 - \phi \pm \sqrt{\phi^2 - 4\phi - 4}| = \sqrt{(2 - \phi)^2 + (\sqrt{4 + 4\phi - \phi^2})} \tag{A.5}$$

$$|2 - \phi \pm \sqrt{\phi^2 - 4\phi - 4}| = 2\sqrt{2}. \tag{A.6}$$

Therefore, for $\forall\phi$,

$$|\frac{2 - \phi \pm \sqrt{\phi^2 - 4\phi - 4}}{4}| < 1. \tag{A.7}$$

With considering (A.2), the feasible domain of ϕ under this scenario are shown in (A.8).

$$2 - 2\sqrt{2} < \phi < 2 + 2\sqrt{2} \tag{A.8}$$

2. If the eigenvalues of (20) are real roots.
Then

$$\phi^2 - 4\phi - 4 \geqslant 0. \tag{A.9}$$

Then

$$\phi \geqslant 2 + 2\sqrt{2} \quad or \quad \phi \leqslant 2 - 2\sqrt{2}. \tag{A.10}$$

Then for $|\frac{2-\phi\pm\sqrt{\phi^2-4\phi-4}}{4}| < 1$, (A.11) should be ensured.

$$-4 < 2 - \phi \pm \sqrt{\phi^2 - 4\phi - 4} < 4 \tag{A.11}$$

For (A.11), the following four situations should be considered.
Firstly,

$$2 - \phi + \sqrt{\phi^2 - 4\phi - 4} < 4. \tag{A.12}$$

Then

$$\phi > -1. \tag{A.13}$$

Secondly,

$$2 - \phi + \sqrt{\phi^2 - 4\phi - 4} > -4. \tag{A.14}$$

Then

$$\phi < 6. \tag{A.15}$$

Thirdly,

$$2 - \phi - \sqrt{\phi^2 - 4\phi - 4} < 4. \tag{A.16}$$

Then

$$\phi > -2. \tag{A.17}$$

Finally,

$$2 - \phi - \sqrt{\phi^2 - 4\phi - 4} > -4. \tag{A.18}$$

Then

$$\phi < 5. \tag{A.19}$$

In summary, the feasible domain of ϕ ensuring the convergence stability of $E(p(t))$ can be obtained based on (A.8) $\cup$ {[(A.13) $\cap$ (A.15) $\cap$ (A.17) $\cap$ (A.19)] $\cap$ (A.10)}, which is shown in (21).

References

[1] R. Eberhart, J. Kennedy, Particle swarm optimization, in: Proceedings of the IEEE International Conference on Neural Networks, 4, 1995, pp. 1942–1948.
[2] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, IEEE, 1995, pp. 39–43.
[3] M. Abido, Optimal power flow using particle swarm optimization, Int. J. Electr. Power Energy Syst. 24 (7) (2002) 563–571.
[4] P. Das, H.S. Behera, B.K. Panigrahi, A hybridization of an improved particle swarm optimization and gravitational search algorithm for multi-robot path planning, Swarm Evol. Comput. 28 (2016) 14–28.
[5] Q. Qin, S. Cheng, X. Chu, X. Lei, Y. Shi, Solving non-convex/non-smooth economic load dispatch problems via an enhanced particle swarm optimization, Appl. Soft Comput. 59 (2017) 229–242.
[6] R. Cheng, Y. Jin, A competitive swarm optimizer for large scale optimization, IEEE Trans. Cybern. 45 (2) (2015) 191–204.
[7] T.V. Sibalija, Particle swarm optimisation in designing parameters of manufacturing processes: a review (2008–2018), Appl. Soft Comput. 84 (2019) 105743.
[8] W.-C. Chen, Y.-C. Tai, M.-W. Wang, H.-C. Tsai, Parameter optimization of etching process for a LGP stamper, Neural Comput. Appl. 23 (6) (2013) 1539–1550.
[9] W.-C. Chen, D. Kurniawan, Process parameters optimization for multiple quality characteristics in plastic injection molding using Taguchi method, BPNN, GA, and hybrid PSO-GA, Int. J. Precis. Eng. Manuf. 15 (8) (2014) 1583–1593.
[10] G. Singh, R.V. Grandhi, D.S. Stargel, Modified particle swarm optimization for a multimodal mixed-variable laser peening process, Struct. Multidisc. Optim. 42 (5) (2010) 769–782.
[11] M. Pant, R. Thangaraj, A. Abraham, Particle swarm optimization: performance tuning and empirical analysis, in: Foundations of Computational Intelligence Volume 3, Springer, 2009, pp. 101–128.
[12] A. Rezaee Jordehi, J. Jasni, Parameter selection in particle swarm optimisation: a survey, J. Exp. Theor. Artif. Intell. 25 (4) (2013) 527–542.
[13] M. Li, H. Chen, X. Shi, S. Liu, M. Zhang, S. Lu, A multi-information fusion riple variables with iterationinertia weight PSO algorithm and its application, Appl. Soft Comput. 84 (2019) 105677.
[14] 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.
[15] 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.
[16] W. Liu, Z. Wang, X. Liu, N. Zeng, D. Bell, A novel particle swarm optimization approach for patient clustering from emergency departments, IEEE Trans. Evol. Comput. 23 (4) (2019) 718–724.
[17] Q. Qin, S. Cheng, Q. Zhang, L. Li, Y. Shi, Particle swarm optimization with interswarm interactive learning strategy, IEEE Trans. Cybern. 46 (10) (2016) 2238–2251.
[18] J. Li, J.Q. Zhang, C.J. Jiang, M.C. Zhou, Composite particle swarm optimizer with historical memory for function optimization, IEEE Trans. Cybern. 45 (10) (2017) 2350–2363.
[19] Y.-J. Gong, J.-J. Li, Y. Zhou, Y. Li, H.S.-H. Chung, Y.-H. Shi, J. Zhang, Genetic learning particle swarm optimization, IEEE Trans. Cybern. 46 (10) (2016) 2277–2290.
[20] 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.
[21] S. Sengupta, S. Basak, R.A. Peters, Particle swarm optimization: a survey of historical and recent developments with hybridization perspectives, Mach. Learn. Knowl. Extract. 1 (1) (2019) 157–191.
[22] Y. Qiang, W.N. Chen, J.D. Deng, L. Yun, T. Gu, J. Zhang, A level-based learning swarm optimizer for large-scale optimization, IEEE Trans. Evol. Comput. 22 (4) (2018) 578–594.
[23] F. Caraffini, F. Neri, G. Iacca, Large scale problems in practice: the effect of dimensionality on the interaction among variables, in: Proceedings of the European Conference on the Applications of Evolutionary Computation, Springer, 2017, pp. 636–652.
[24] F. Van den Bergh, A.P. Engelbrecht, A cooperative approach to particle swarm optimization, IEEE Trans. Evol. Comput. 8 (3) (2004) 225–239.
[25] X. Li, X. Yao, Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms., in: Proceedings of the IEEE Congress on Evolutionary Computation, 2009, pp. 1546–1553.
[26] X. Li, X. Yao, Cooperatively coevolving particle swarms for large scale optimization, IEEE Trans. Evol. Comput. 16 (2) (2012) 210–224.

[27] 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.
[28] 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: Proceedings of the IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 3845–3852.
[29] S.-T. Hsieh, T.-Y. Sun, C.-C. Liu, S.-J. Tsai, Solving large scale global optimization using improved particle swarm optimizer, in: Proceedings of the IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 1777–1784.
[30] R. Cheng, C. Sun, Y. Jin, A multi-swarm evolutionary framework based on a feedback mechanism, in: Proceedings of the IEEE Congress on Evolutionary Computation, IEEE, 2013, pp. 718–724.
[31] R. Cheng, Y. Jin, A social learning particle swarm optimization algorithm for scalable optimization, Inf. Sci. 291 (2015) 43–60.
[32] X. Peng, Y. Jin, H. Wang, Multimodal optimization enhanced cooperative coevolution for large-scale optimization, IEEE Trans. Cybern. 49 (9) (2018) 3507–3520.
[33] W. Chen, T. Weise, Z. Yang, K. Tang, Large-scale global optimization using cooperative coevolution with variable interaction learning, in: Proceedings of the International Conference on Parallel Problem Solving from Nature, Springer, 2010, pp. 300–309.
[34] S. Mahdavi, M.E. Shiri, S. Rahnamayan, Cooperative co-evolution with a new decomposition method for large-scale optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 1285–1292.
[35] 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) (2014) 378–393.
[36] 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 Annual Conference on Genetic and Evolutionary Computation, ACM, 2015, pp. 313–320.
[37] 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 (TOMS) 42 (2) (2016) 13.
[38] 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.
[39] N. Lynn, P.N. Suganthan, Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation, Swarm Evol. Comput. 24 (2015) 11–24.
[40] J.-B. Park, Y.-W. Jeong, J.-R. Shin, K.Y. Lee, An improved particle swarm optimization for nonconvex economic dispatch problems, IEEE Trans. Power Syst. 25 (1) (2009) 156–166.
[41] 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.
[42] H. Wang, Z. Wu, S. Rahnamayan, Y. Liu, M. Ventresca, Enhancing particle swarm optimization using generalized opposition-based learning, Inf. Sci. 181 (20) (2011) 4699–4714.
[43] 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, Inf. Sci. 289 (2014) 162–189.
[44] 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.
[45] H. Soleimani, G. Kannan, A hybrid particle swarm optimization and genetic algorithm for closed-loop supply chain network design in large-scale networks, Appl. Math. Model. 39 (14) (2015) 3990–4012.
[46] 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.
[47] N. Hansen, A. Ostermeier, Completely derandomized self-adaptation in evolution strategies, Evol. Comput. 9 (2) (2001) 159–195.
[48] A. Auger, N. Hansen, Tutorial CMA-ES: evolution strategies and covariance matrix adaptation., in: Proceedings of the GECCO (Companion), 2012, pp. 827–848.
[49] R. Ros, N. Hansen, A simple modification in CMA-ES achieving linear time and space complexity, in: Proceedings of the International Conference on Parallel Problem Solving from Nature, Springer, 2008, pp. 296–305.
[50] I. Loshchilov, A computationally efficient limited memory CMA-ES for large scale optimization, in: Proceedings of the Annual Conference on Genetic and Evolutionary Computation, ACM, 2014, pp. 397–404.
[51] D. Molina, M. Lozano, F. Herrera, Ma-sw-chains: Memetic algorithm based on local search chains for large scale continuous global optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation, IEEE, 2010, pp. 1–8.
[52] A. LaTorre, S. Muelas, J.-M. Peña, Large scale global optimization: experimental results with MOS-based hybrid algorithms, in: Proceedings of the IEEE congress on evolutionary computation, IEEE, 2013, pp. 2742–2749.
[53] J. Zhang, A.C. Sanderson, Jade: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958.
[54] J. Brest, M.S. Mauec, Self-adaptive differential evolution algorithm using population size reduction and three strategies, Soft Comput. 15 (11) (2011) 2157–2174.
[55] M.Z. Ali, N.H. Awad, P.N. Suganthan, Multi-population differential evolution with balanced ensemble of mutation strategies for large-scale global optimization, Appl. Soft Comput. 33 (2015) 304–327.
[56] 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.
[57] D. Molina, A.R. Nesterenko, A. LaTorre, Comparing large-scale global optimization competition winners in a real-world problem, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC), IEEE, 2019, pp. 359–365.
[58] D. Molina, A. LaTorre, F. Herrera, Shade with iterative local search for large-scale global optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC), IEEE, 2018, pp. 1–8.
[59] B. Morales-Castañeda, D. Zaldívar, E. Cuevas, F. Fausto, A. Rodríguez, A better balance in metaheuristic algorithms: Does it exist? Swarm Evol. Comput. (2020) 100671.
[60] Q. Chen, J. Sun, V. Palade, Distributed contribution-based quantum-behaved particle swarm optimization with controlled diversity for large-scale global optimization problems, IEEE Access 7 (2019) 150093–150104.
[61] 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.
[62] F. Caraffini, A.V. Kononova, D. Corne, Infeasibility and structural bias in differential evolution, Inf. Sci. 496 (2019) 161–179.
[63] M.R. Bonyadi, Z. Michalewicz, Stability analysis of the particle swarm optimization without stagnation assumption, IEEE Trans. Evol. Comput. 20 (5) (2015) 814–819.
[64] H. Zhang, A discrete-time switched linear model of the particle swarm optimization algorithm, Swarm Evol. Comput. 52 (2020) 100606.
[65] K. Tang, X. Yao, P.N. Suganthan, C. MacNish, Y.-P. Chen, C.-M. Chen, Z. Yang, Benchmark Functions for the CEC008 Special Session and Competition on Large Scale Global Optimization, 24, Nature Inspired Computation and Applications Laboratory, USTC, China, 2007.
[66] 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.