Egyptian Informatics Journal 23 (2022) 469–481

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

Mutual learning differential particle swarm optimization

Anping Lin [a], Shanglin Li [b,c,*], Rongsheng Liu [a]

[a] School of Physics and Electronic Electrical Engineering, Xiangnan University, Chenzhou 423000, PR China
[b] School of Computer Science and Artificial Intelligence, Xiangnan University, Chenzhou 423000, PR China
[c] Hunan Engineering Research Center of Advanced Embedded Computing and Intelligent Medical Systems, Chenzhou 423000, PR China

ARTICLE INFO

Article history:
Received 9 February 2022
Revised 17 March 2022
Accepted 11 April 2022
Available online 26 April 2022

Keywords:
Mutual learning
Particle swarm optimization
Differential evolution
Elite DE mutation

ABSTRACT

This study proposes a mutual learning strategy to develop a high performance hybrid algorithm based on particle swarm optimization and differential evolution. In the mutual learning strategy, the position information in PSO subswarm is employed for DE mutation, and the DE individuals are used to construct learning exemplar for PSO subswarm together with particles' historical best position. A novel elite DE mutation is proposed to speed up the convergence rate of DE subswarm. Based on mutual learning technique, the mutual learning differential evolution particle swarm optimization (MLDE-PSO) is proposed. To evaluate the performance of MLDE-PSO, three groups of test functions are employed, namely thirteen basic functions, thirteen rotated basic functions and thirty CEC2017 functions. The test results are compared with three state-of-the-art PSO algorithms, three recently PSO algorithms and DE/rand/1. The test results indicate that the proposed MLDE-PSO performs better than the other seven comparison algorithms, especially on rotated functions and CEC2017 functions. The rotation test shows that MLDE-PSO is not very sensitive to rotation transformation.

1. Introduction

Particle swarm optimization (PSO) only uses primary mathematical operations and can obtain a high convergence speed. For PSO converges fast and can obtain high performance, it is widely applied to function optimization and real applications. Since the inception of PSO in 1995, a lot of PSO variants have been reported, the performance of PSO has been improved significantly. However, most of the current PSO variants can only perform well on a group of certain optimization problems, fewer PSO can achieve high performance on various types of optimization problems.

To improve the performance of PSO on different characteristics problem, the evolutionary community racks out their brains. In general, the improved PSO algorithms can be roughly divided into four categories, namely parameters regulating [1], neighborhood topology[1], learning strategy [2,3] and hybridizing with other optimization technique[4]. Houssein et al. [5] summarized the recently development of PSO algorithms.

The learning strategy is an effective approach to improve the performance of PSO. A series of PSO algorithms construct effective learning exemplars to guide the evolution of particles. For example, CLPSO [6] adopts a comprehensive learning strategy in which different dimensions of a particle learns from different particles; Social learning particle swarm optimization (SLPSO) [7]adopts a social learning strategy in which the particles only learn from other particles with better fitness value; Xu et al. [8] introduces a dimensional learning strategy to employ the promising information discovered by the particle swarm; Molaei[9] introduces an enhanced learning strategy in which all the particles are employed to regulate any particle's motion; Lim [10] proposes an elitist learning strategy based on orthogonal experimental designed stochastic perturbation techniques. Yan [11] proposed random learning mechanism to enhance swarm diversity, in random learning mechanism each particle learns from a random neighbor particle with a given probability. Ye [12] proposes dynamic learning strategy by dividing the whole swarm into exploitation oriented ordinary particles and exploration oriented communication particle dynamically. The aforementioned PSO algorithms show that employing the information of elite neighbor particles for updating

* Corresponding author.
E-mail address: lsl@xnu.edu.cn (S. Li).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.eij.2022.04.003
1110-8665/

particles' velocity can enhance swarm diversity and mitigate premature convergence.

To improve the applicability of PSO, kinds of intelligent optimization algorithms are transplanted into PSO. For example, Juang [13] employs PSO and genetic algorithm (GA) to evolve elite particles and proposes hybrid GA and PSO. Javidrad [14] employs simulated annealing (SA) and PSO to evolve the population and proposed hybrid particle swarm and simulated annealing stochastic optimization (PSO-SA). Chen[15] transfers the migration of biogeography-based optimization to PSO and proposes biogeography-based learning particle swarm optimization (BLPSO). Charin [16] introduces random walk distribution into PSO and proposes hybrid of Levy flight and particle swarm optimization (LPSO). Jaferi [17] proposes a hybrid algorithm based on PSO and cultural algorithm for truss structures design. Khan[18] employs (GSA) to overcome premature stagnation of PSO and proposes Hybrid Gravitational Search Particle Swarm Optimization Algorithm (HGSPSO).

Among the hybrid PSO algorithms, hybridizing DE and PSO attracts great attention in EC community. For DE and PSO can complement each other. Since Zhang [19] transfers the DE mutation into PSO algorithm in 2003, a series of hybrid algorithms based on DE and PSO are proposed. The concept of hybrid algorithm is using DE operators to enhance swarm diversity of PSO, or employing PSO to speed up the convergence of DE. The major hybridizing methods including (1) Series structure, employs DE further to evolves the particle swarm [20]; (2) Parallel structure, adopting both DE and PSO to evolve their own subswarm [21]; (3) Crossover, employing DE and PSO simultaneously to generating offspring through crossover [22]. (4) Cascading, employing DE to construct learning exemplar for PSO [23]. More detailed introduction of DE-PSO will be introduced in section 2.3. Due to the opposite characteristics, the performance of some hybrid DE-PSO algorithms on complex multimodal problems may worse than classical DE algorithms. DE needs relatively bigger population than PSO to avoid premature convergence, hence the population is a key parameter for hybrid DE and PSO algorithms. Developing high performance hybrid DE and PSO algorithm still attracts the attention of EC community. To fully use the advantages of DE and PSO algorithms, this study proposed a mutual learning strategy for hybridizing DE and PSO. Furthermore, an elite mutation is introduced to accelerate the convergence speed of DE subswarm. The rest of this paper is organized as follows: Section 2 reviews the related works, Section 3 proposes the methodology, section 4 conducts experiments and section 5 concludes this study.

2. Related works

2.1. Canonical PSO

In PSO, each potential solution is regarded as a particle. The particle flies in the search space to find the global optima under the attractive force from its personal best position (*Pbest*) and the global best position (*Gbest*). The widely used global version PSO updates its velocity and position according to eq .(1), (2) [24].

$$v_{i,d} = \omega v_{i,d} + c_1 \cdot r_{1,d} \cdot (p_{i,d} - pos_{i,d}) + c_2 \cdot r_{2,d} \cdot (g_d - pos_{i,d}) \quad (1)$$

$$pos_{i,d} = pos_{i,d} + v_{i,d} \quad (2)$$

where the velocity vector, *Pbest* vector and position vector of the *i*th particle are denoted by $V_i = [v_{i,1}, v_{i,2}, \ldots, v_{i,D}]$, $P_i = [p_{i,1}, p_{i,2}, \ldots, p_{i,D}]$, and $Pos_i = [pos_{i,1}, pos_{i,2}, \ldots, pos_{i,D}]$, respectively. $G = [g_1, g_2, \ldots, g_D]$ stands for the *Gbest* vector. ω stands for an inertia weight, c_1 and c_2 are two acceleration coefficients. $r_{1,d}$,$r_{2,d} \in [0,1]$are two uniform random numbers. The global version PSO can achieve high convergence rate, however, while it suffers from premature convergence.

2.2. DE

Differential evolution (DE) is one of the most powerful and versatile evolutionary optimizers for the continuous parameter space in recent times [25]. DE moves a population of individual towards the global optima through mutation, crossover and selection operators. The candidate solution vectors are referred to as individuals and denoted by target vector $X_{i,}^G$={$x_{i,1}^G$, $x_{i,2}^G$, $\cdots$, $x_{i,D}^G$}. $i = 1, 2, \ldots, NP$ is the indice of the individual, G is the iteration number and D is the number of dimension. In the initialization, the individuals are initialized randomly. In each generation, a mutant vector V_i^G =-{$v_{i,1}^G$, $v_{i,2}^G$, $\cdots$, $v_{i,D}^G$}is generated by mutation operator for each individual. Five most frequently used mutation strategies [26] implemented in the DE codes are given as below:

1) DE/rand/1.

$$V_i^G = X_{r1}^G + F_i \cdot (X_{r2}^G - X_{r3}^G) \quad (3)$$

2) DE/current-to-best/1.

$$V_i^G = X_i^G + F_i \cdot \left(X_{best}^G - X_i^G\right) + F_i \cdot (X_{r2}^G - X_{r3}^G) \quad (4)$$

3) DE/ best/1.

$$V_i^G = X_{best}^G + F_i \cdot (X_{r1}^G - X_{r2}^G) \quad (5)$$

4) DE/ best/2.

$$V_i^G = X_{best}^G + F_i \cdot \left(X_{r1}^G - X_{r2}^G\right) + F_i \cdot (X_{r3}^G - X_{r4}^G) \quad (6)$$

5) DE/ rand/2.

$$V_i^G = X_{r1}^G + F_i \cdot \left(X_{r2}^G - X_{r3}^G\right) + F_i \cdot (X_{r4}^G - X_{r5}^G) \quad (7)$$

The indices $r_1, r_2, r_3, r_4, r_5 \in [1, NP]$are mutually exclusive integers randomly. F_i is a scaling factor to scale the difference vectors.

After mutation, the crossover operation is conducted between the individual and its mutant vector to generate a trial vector $U_{i,}^G$={$u_{i,1}^G, u_{i,2}^G, \cdots, u_{i,3}^G$}. The crossover operation is executed according to eq.(8).

$$u_{i,j}^G = \begin{cases} v_{i,j}^G if\left(rand_j \le CR\right) or (j = j_{rand}) \\ \quad x_{i,j}^G\, otherwise \end{cases} \quad (8)$$

where $CR \in (0, 1]$is a crossover rate defined by user. $j = 1, 2, \ldots, D$ is the indice of dimension. $rand_j \in [0,1]$ is a uniform distributed random number. $j_{rand} \in [1,D]$ is a random integer to guarantee the trial vector U_i^G will differ from its target vector $X_{i,G}$ in at least one dimension.

After mutation, the fitness value of the trial vector is evaluated and the selection operation is performed. The selection is carried out according to eq.(9) between the trial vector U_i^G and its target vector X_i^G based on their fitness value.

$$X_i^G = \begin{cases} U_i^G iffit(U_i^G) \le f(X_i^G) \\ X_i^G \quad otherwise \end{cases} \quad (9)$$

where $fit(\cdot)$ is the objective function. If the trial vector's fitness value $fit(U_i^G)$ is no worse than its target vector's fitness value $fit(V_i^G)$, the trial vector U_i^G will replace its target vector X_i^G. Otherwise, the target vector will be kept for the next generation. The mutation, crossover and selection operations are carried out repeatedly until the termination criteria is satisfied.

2.3. Hybrid DE and PSO

A serials hybrid DE and PSO are proposed in the past. For example, Pei [27] taking the PSO as auxiliary mutation operator and the DE for crossover operation to avoid the deficiency of using single algorithm. Chen [23] proposed a hybridizing PSO and DE for feature selection by employing differential evolution to breed promising and efficient exemplars for PSO. Sato [28] proposed multi-swarm differential evolutionary particle swarm optimization (MS-DEEPSO) for energy network optimization. Wang [29] introduces a self-adaptive mutation strategy for hybrid differential and PSO algorithm, the selection probability of DE and PSO operators are self-adapted based on their previous performance. Epitropakis proposes a serial structure framework for hybridizing DE and PSO. Sayah et al. [30] proposed a DE-PSO algorithm by incorporating PSO into the conventional DE algorithm as a supplementary mutation operator to improve global search capability. Liu et al. [21]employs PSO to evolve the better half of the DE individuals in each iteration. Seyedmahmoudian [31] proposed a DE-PSO algorithm by adopting DE and PSO to evolve the population alternatively. Chen et al. [32] employs two different differential evolution operators to construct learning exemplar for PSO and proposed particle swarm optimizer with two differential mutations. Chen et al. [33] merge the differential evolution operator into each sub-swarm, and proposed a dynamic multi-swam differential evolution learning particle swam optimizer (DMSDL-PSO). DMSDL-PSO adopts differential mutation to enhance exploration and employs Quasi-Newton method as a local searcher to enhance the exploitation. Xin et al. [34] summarized the existing hybrid algorithms based on DE and PSO.

3. Methodology

3.1. Motivation

As two excellent intelligent optimization algorithms, PSO and DE have their own distinct characteristic. For instance, PSO employs relatively small population and converges fast, while it may be apt to fall into local optima when optimizing complex multimodal problems. Furthermore, PSO is rotation variant, hence its performance on rotated problems is not satisfactory. On the contrary, DE requires a relatively large population size and performs better on complex multimodal problems, while it converges relatively slowly on unimodal problems. For PSO and DE generating offspring in different ways, it is difficult to make good use of their advantages without crapping each other. A few hybrid DE-PSO algorithm seven can't outperform classic DE algorithms on complex multimodal problems. Based on the aforementioned weakness of hybrid DE-PSO algorithms, this study introduces a mutual learning strategy for hybridizing DE and PSO, to fully use the promising information in both DE and PSO subswarm. In the mutual learning strategy, the DE and the PSO subswarm work in parallel, and the elite information is exchanged between two subswarm to improve the population quality. The position vector of PSO subswarm is employed for DE mutation to speed up the response of DE subswarm, furthermore, the elite DE mutation is employed for accelerating the convergence speed of DE subswarm.

3.2. Elite DE

DE/rand/1 has strong exploration, while it converges too slowly. To improve the convergence rate of DE/rand/1, an elite DE (denoted by Elite-DE) is proposed in this study. Elite-DE generates trial vectors according to eq. (10,11).

$$V_i^G = X_{Er1}^G + F_i \cdot (X_{r2}^G - X_{r3}^G) \tag{10}$$

X_{Er1}^G is randomly selected from top N_{elite} individual.

$$N_{elite} = round(NP - (NP - N_{end}) * iter/iter_m) \tag{11}$$

N_{elite} denotes the top N_{elite} individuals in DE subswarm sorted by fitness value. N_{elite} is decreasing linearly during the optimization process, aiming to concentrate the searching in the neighborhood of a few promising individuals. In the early stage of optimization,X_{Er1}^G is randomly selected from the whole population to achieve high exploration. As the increasing of iteration, X_{Er1}^G is selected form a few of high quality individuals to improve convergence speed in the later stage.

3.3. Crossover learning exemplar

The learning exemplar is of vital importance to improve the performance of PSO. For example, CLPSO [6], OLPSO [35], GL-PSO [36] improve the performance of PSO by construct high quality learning exemplars. In this study, individuals of DE subswarm is adopted to generate learning exemplars for PSO subswarm according to Algorithm 1. The crossover is carried out to generate try learning exemplar according to eq. (12,13).

Algorithm 1 Generating learning exemplar of MLDE-PSO

```
For d = 1:D/* Generating try learning exemplar
  Update exemp_i^d according to eq.(12,13)
End
If fit(Exemp_i) < fit(Exemplar_i) /* Selection of learning exemplar
  Exemplar_i = Exemp_i
  Exemplar_val_i = fit(Exemp_i)
End
```

Note: *fit*(·)stand for the fitness function. $Exemp_i$, $Exemplar_i$, $Exemplar\_val_i$ stand for try learning exemplar, learning exemplar, and fitness of learning exemplar of particle *i*, respectively. *d* and *D denote for the indice of dimension and dimension.*

$$exemp_i^d = \begin{cases} Pbest_i^d & (rand <= P_{cr}) \\ X_r^d & (rand > P_{cr}) \end{cases} \tag{12}$$

$$P_{cr} = \begin{cases} 0.9 & (Pbest\,val(i) <= Val(r)) \\ 0.1 & (Pbest\,val(i) > Val(r)) \end{cases} \tag{13}$$

$$v_i^d = \omega v_i^d + c_1 \cdot r_i^d \cdot \left(Exempar_i^d - Pos_i^d\right) + c_2 \cdot r_i^d \cdot (Gbest^d - Pos_i^d) \tag{14}$$

For arbitrarily particle *i*, a random selected DE individual *r* is employed for generating a try learning exemplar through crossover. The crossover probability is determined by eq.(13). According to the guidance of Storm [42], c_r = 0.9 (the crossover probability of DE) can speed convergence. Hence, if *Pbestval*(*i*) (the fitness value of particle *i*)is no worse than *Val*(*r*) (the fitness value of individual *r*), the crossover probability P_{cr} = 0.9, otherwise P_{cr} = 0.1. Based on the crossover, most dimensions of learning exemplar are copied from the better solution between PSO particles and DE individuals, hence the diversity of PSO subswarm is enhanced. In the selection operation, the learning exemplar is replaced by the try learning exemplar if the latter's fitness value is better. With selection operation, the quality of learning exemplar is guaranteed. For the DE subswarm is rotation invariant, employing DE individual for generating PSO learning exemplar can give the PSO subswarm partial rotation invariant feature.

3.4. Elite DE with best position

To further enhance information between DE and PSO subswarm, Elite DE with the best position (denoted by E-DEx) is employed for generating trial vector of DE subswarm.

$$V_i^G = X_{Er1}^G + F_i \cdot (X_{r2}^G - X_{Er3}^G) \tag{15}$$

$X^G{}_{Er3}$ is a random individual chosen from the union DE swarm and Pos_{ibest} (Pos_{ibest} denotes for the current position of the best particle in the PSO subswarm). Pos_{ibest} is updated in every iteration, hence employing Pos_{ibest} for DE mutation can speed up the response of DE subswarm to the landscape of optimization problem.

3.5. The proposed method

The flowchart of the proposed MLDE-PSO is given in Fig. 1. The whole swarm is divides into a DE subswarm and a PSO subswarm. Two subswarms work alternatively and cooperatively according to the mutual learning strategy, and then some promising information can be exchanged sufficiently. As a result, the quality of the whole swarm can be improved. In the initialization, the population of both DE subswarm and PSO subswarm are initialized, and the parameters of DE subswarm and PSO subswarm are assigned will initial value. The number of elite individual, inertial weight and acceleration coefficients are adjusted in every iteration. Then, two subswarms are evolving according to their own ways. The valuable information is exchanged between two subswarms. The position of the best particle is employed for DE mutation, and the DE individuals are utilized for generating learning exemplar in PSO subswarm through crossover. If the number function evaluations *FE* is less than the maximum function evolution, the optimization goes to next iteration, otherwise the optimization is ended and outputs results. Since two subswarms can learn from each other, both the DE subswarms and the PSO subswarms can employ the other subswarm's promising information to improve swarm quality, thereby both the DE subswarm and the PSO subswarm can employ the other swarm's advantages to overcome its own defects. The mutual learning strategy proposed in this study can be applied to other hybrid intelligent optimization algorithm.

4. Experimental work

4.1. Experimental setup

In this study, three group test functions are employed to test the performance of MLDE-PSO. (1) 13 selected basic functions adopted by a few representative PSO variants [36] are employed to compare the performance of MLDE-PSO on basic optimization problems. (2) The aforementioned 13 basic functions are rotated to evaluate the performance of MLDE-PSO on rotated problems. (3) 30 CEC2017 [37] functions adopted by most of recently literature in evolutionary community are employed to test the performance of MLDE-PSO on competitive optimization. Details of 13 basic functions denoted by ($BF_1 \sim BF_{13}$) are given in Table1, BF_1-BF_7 are seven unimodal functions, BF_8-BF_{13} are six multimodal functions. To test the rotation invariance feature of MLDE-PSO, thirteen basic functions are rotated with rotation matrix. The rotation functions can be expressed in eq.(16).

$$RF_i(x) = BF_i(x * M) \tag{16}$$

$i \in [1,13]$ is the index of function. $\boldsymbol{M}$ is a rotation matrix [38], its condition number is 2.

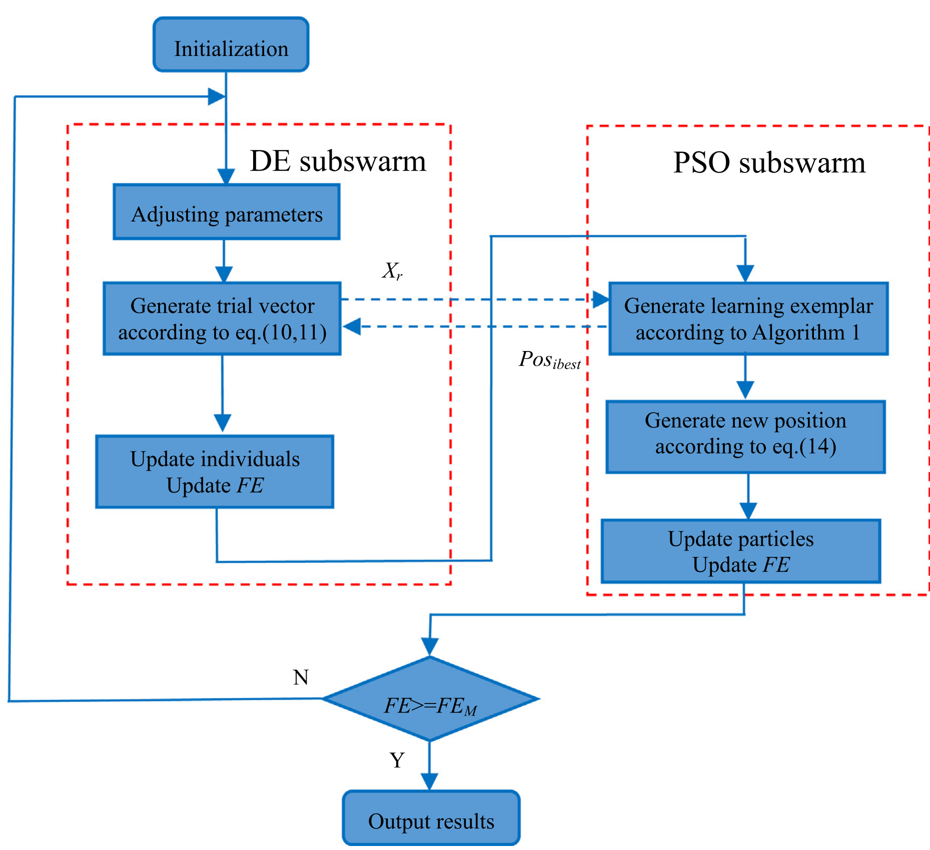


Fig. 1. The flow chart of MLDE-PSO.

Table 1
Basic test function.

No.	Test function	Domain	Name
BF_1	$f_1(x) = \sum_{i=1}^{n} x_i^2$	[-100,100]	sphere
BF_2	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	[-10,10]	Schwefel's P2.22
BF_3	$f_3(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)$	[-100,100]	quadric
BF_4	$f_4(x) = \left\{ \max_i \lvert x_i \rvert, 1 \leqslant i \leqslant 30 \right\}$	[-100,100]	Schwefel's P2.2.1
BF_5	$f_4(x) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	[-10,10]	Rosenbrock's
BF_6	$f_5(x) = \sum_{i=1}^{n} (x_i + 0.5)^2$	[-100,100]	Step
BF_7	$f_7(x) = \sum_{i=1}^{30} x_i^4 random[0,1)$	[-500,500]	Schwefel's
BF_8	$f_7(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$	[-5.12,5.12]	rastrigin
BF_9	$f_8(x) = -20\exp\left(-0.2\sqrt{1/n\sum_{i=1}^{n} x_i^2}\right) - \exp(1/n\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + e$	[-32,32]	ackley
BF_{10}	$f_9(x) = 1/4000\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(x_i/\sqrt{i}) + 1$	[-600,600]	griewank
BF_{11}	$f_{11}(x) = \frac{\pi}{n}\{1 - \sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\}$ $+\sum_{i=1}^{n} u(x_i, 10, 100, 4), y_i = 1 + \frac{1}{4}(x_i + 1)$	[-50,50]	General penalized
BF_{12}	$f_{12}(x) = \frac{1}{10} 10\sin^2(3\pi x_1) + \sum_{i=1}^{n-1}(x_i - 1)^2[1 + 10\sin^2(\pi x_{i+1})] + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	[-50,50]	General penalized
BF_{13}	$f_{13}(x) = \sum_{i=1}^{n}(y_i^2 - 10\cos(2\pi y_i) + 10) + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$	[-5.12,5.12]	Non-continuous Rastrigin

Note: In BF_{12} and BF_{13}, $y_i = \begin{cases} x_i \lvert x_i \rvert \leqslant 0.5 \\ \frac{round(2x_i)}{2} \lvert x \rvert 0.5 \end{cases}$ and $u(x_j, a, k, m) = \begin{cases} k(x_j - a)^m, & x_j > a \\ 0, -a \leqslant x_j \leqslant a \\ k(-x_j - a)^m, & x_j < -a \end{cases}$.

To further test the performance of MLDE-PSO on complex multimodal function, thirty CEC2017 functions (F_1-F_{30}) are tested. The maximum function evolution is set as 10000*D, D is the number of dimension. All the functions are run for 30 independent runs and the mean performance are compared. The experiments are conducted on a PC with AMD Ryzen-3500U 2.1 GHz CPU, 8 GB RAM, MS Windows 10 64-bit OS and Matlab R2018b Compiler.

Seven selective peer algorithms are compared with MLDE-PSO covering three state-of-the-art PSO algorithms, three recent PSO algorithm and one classic DE algorithms. CLPSO [6] adopts comprehensive learning strategy to enhance the exploration of PSO. In comprehensive learning, each particle learns from the whole swarm and different dimensions learn from different particles. CLPSO performs well on multimodal problems. FDR-PSO [39] employs the nearby high fitness particles to guide the motion of learning particle. FDR-PSO avoids premature convergence at the cost of slowing down convergence speed in the early stage. UPSO [40] employs local version PSO and global version PSO simultaneously to achieve good balance between exploration and exploitation. TSL-PSO [8] utilizes two learning strategy to evolve two subswarms respectively. One subswarm adopts dimensional learning strategy to enhance local search while the other subswarm uses comprehensive learning strategy to enhance exploration. GGL-PSOD [41] employs ring topology neighbor particles to construct learning exemplar and linearly adjusting control parameters to balance exploration and exploitation. GGL-PSOD exhibits high adaptability on different kinds of optimization problems. DMSDL-PSO divides the swarm into a few dynamic subswarm and employ DE mutation to construct learning exemplar of PSO, the Quasi-Newton method is employed for local search. DE/rand/1 [42] bears strong exploration while converges slowly, it performs well on complex multimodal problems. For Elite-DE is modified from DE/rand/1, hence DE/rand/1 is adopted for comparison test. The parameters of comparison algorithms presented in Table 2 are in accordance with the original publications.

4.2. Search behavior test

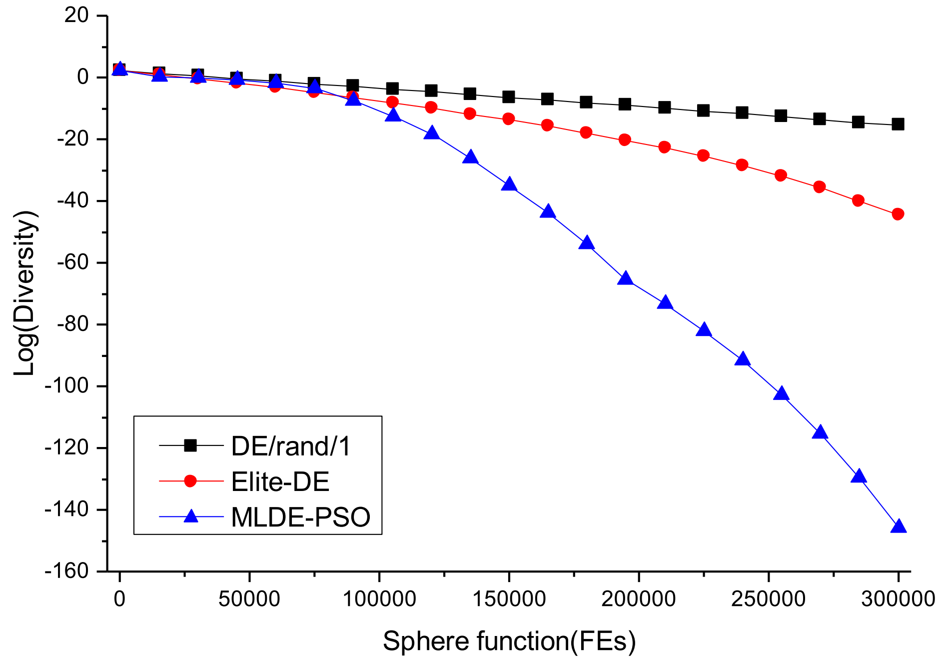
To analyze the search behavior of MLDE-PSO, the diversity curves on Sphere function and Rastrigin function are given in Fig. 2. The diversity curves of DE/rand/1 and Elite-DE are provided for comparison. The diversity is evaluated by mean Euclidean distance between the population and their centriod. The diversity is calculated according to eq. (17).

$$Diversity = \frac{1}{N_P}\sum_{i=1}^{N_P}\sqrt{\sum_{j=1}^{D}(x_{i,j} - \bar{x}_j)^2} \quad (17)$$

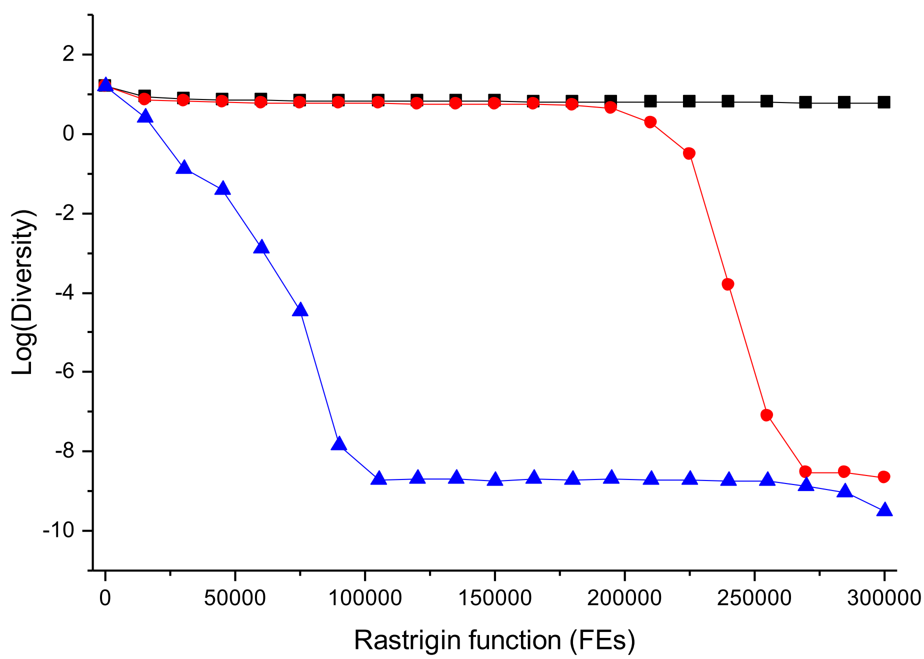
N_P = N_{DE} + N_{PSO}, $\bar{x}_j$ denote for j^{th} dimension mean position of both DE and PSO subswarm. For PSO subswarm, $x_{i,j} = p_{i,j}$. Fig. 2.a shows that the diversity of DE/rand/1 decreases slowly, means the population of DE/rand/1 converges slowly, hence it can't achieve high accuracy on Sphere function. In the starting the diversity curve of Elite-DE is almost overlap with DE/rand/1, while in the later stage, the diversity curve of Elite-DE decreasing faster than DE/rand/1. MLDE-PSO can preserve sufficient diversity in the early stage and converge to the current best solution quickly in the later stage. Thereby MLDE-PSO can obtain high accuracy on Sphere function. Fig. 2.b shows that on Rastrigin functions, the population of DE/rand/1 can't converge to a small area. Elite-DE can't converge in the early stage, while in the later stage, Elite-DE converges to the found best solution quickly. MLDE-PSO converges at suitable speed in the early, while in the later stage, the diversity of MLDE-PSO decreasing slowly. The diversity curves indicate that with PSO subswarm, MLDE-PSO bears stronger exploitation than DE/rand/1, MLDE-PSO can keep sufficient diversity in the early stage and converge to the neighborhood found best solution to achieve high accuracy.

Table 2
Parameters configuration of algorithms.

No.	Algorithm	year	Parameter settings
1	CLPSO[6]	2016	Ps = 40, w = 0.7298, c = 1.49618,m = 5
2	FDR-PSO [39]	2003	Ps = 40, w = 0.7298, c_1 = 1,c_2 = 1, c_3 = 2
3	UPSO-PSO [40]	2005	Ps = 40, w = 0.7298, c_1 = 2.05,c_2 = 2.05
4	TSL-PSOD [8]	2019	N_1 = 15, N_2 = 25,w = 0.9–0.4, c_1 = 1.5,c_2 = 1.5, c_3 = 0.5 $\sim$ 2.5
5	GGL-PSOD [41]	2002	Ps = 50, w = 0.9–0.4, c_1 = 2.5–0.5, c_2 = 0.5–2.5, p_m = 0.01, s_g = 7
6	DMSDL-PSO [33]	2004	Ps = 40, w = 0.7298, c = 1.49618, R = 20, CR = 0.025
7	DE/rand/1 [42]	2006	NP = 100, F = 0.5, CR = 0.9
8	MLDE-PSO	2015	N_{DE} = 100, N_{PSO} = 7, F = 0.5, CR = 0.9, Pcr = 0.9



(a) Diversity curves on Sphere function



(b) Diversity curves on Rasitrigin function

Fig. 2. Diversity curves of MLDE-PSO and DE.

4.3. Comparison test on basic test functions

The comparison test results on basic test functions of MLDE-PSO and seven peer algorithms are presented in Table 3 and Table 4. The statistic wilcoxon signed-rank test is employed for comparing the performance of MLDE-PSO and the other peer algorithm. In Table 3 and Table 4, *FV* stands for mean fitness value in thirty independent runs. *SP* is employed to evaluate the convergence speed of algorithms to achieve the predefined accuracy, it can be expressed in eq.(17).

Table 3
Test results on unimodal basic test functions.

Func.	Statistic	CLPSO	FDR-PSO	UPSO	TSL-PSO	GGL-PSOD	DMSDL-PSO	DE/rand/1	MLDE-PSO
BF_1	FV	>2.182E-30	>1.283E-149	>1.283E-149	>2.557E-203	>1.959E-57	>1.822E-68	>4.924E-31	**0.000E + 00**
	SP	1.231E + 05	1.148E + 05	2.333E + 04	3.313E + 04	1.493E + 05	7.786E + 03	9.012E + 04	**1.844E + 04**
	SR	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	Rank	8	3	4	2	6	5	7	**1**
BF_2	FV	>4.953E-19	>3.696E-63	>3.696E-63	>1.490E-97	>5.398E-33	>1.208E-34	>1.666E-15	**0.000E + 00**
	SP	1.669E + 05	1.225E + 05	3.136E + 04	4.434E + 04	1.657E + 05	7.310E + 04	1.437E + 05	**2.663E + 04**
	SR	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	Rank	7	3	4	2	6	5	8	**1**
BF_3	FV	>1.611E-29	>4.603E-150	>4.603E-150	>2.000E-203	>8.137E-56	>5.657E-65	>9.090E-30	**0.000E + 00**
	SP	1.305E + 05	1.185E + 05	2.588E + 04	3.329E + 04	1.543E + 05	5.510E + 04	1.002E + 05	**2.009E + 04**
	SR	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	Rank	8	3	4	2	6	5	7	**1**
BF_4	FV	>1.302E + 00	>2.447E-04	>2.447E-04	>3.560E-02	>3.383E-06	>4.554E-06	>1.873E-02	**1.377E-131**
	SP	Inf	Inf	3.248E + 05	Inf	5.283E + 05	8.996E + 06	1.271E + 06	**3.422E + 04**
	SR	0.00%	0.00%	83.33%	0.00%	53.33%	3.33%	23.33%	**100.00%**
	Rank	8	4	5	7	2	3	6	**1**
BF_5	FV	<4.505E + 00	<1.313E + 00	<1.313E + 00	<5.104E + 00	<1.337E + 01	**<5.315E-01**	<7.753E-01	2.801E + 01
	SP	Inf	Inf	Inf	Inf	Inf	**3.203E + 05**	Inf	Inf
	SR	0.00%	0.00%	0.00%	0.00%	0.00%	**86.67%**	0.00%	0.00%
	Rank	5	3	4	6	7	**1**	2	8
BF_6	FV	**= 0.000E + 00**	>1.667E-01	>1.667E-01	**= 0.000E + 00**	**= 0.000E + 00**	**= 0.000E + 00**	**= 0.000E + 00**	**0.000E + 00**
	SP	7.062E + 04	1.474E + 05	**1.138E + 04**	1.652E + 04	1.115E + 05	2.296E + 04	3.981E + 04	7.964E + 03
	SR	**100.00%**	83.33%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	Rank	**1**	7	8	**1**	**1**	**1**	**1**	**1**
BF_7	FV	>3.860E-03	>2.532E-03	>2.532E-03	>3.517E-03	>2.471E-03	>2.094E-03	>4.593E-03	**1.953E-04**
	SP	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
	SR	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Rank	7	4	5	6	3	2	8	**1**

Note: FV and SR denote for mean fitness value, and success rate, respectively. SP expressed in eq.(17) stands for the quotient of the average number of function evaluations needed by an algorithm to reach the predefined thresholds.

Table 4
Test results on multimodal basic test functions.

Func.	Statistic	CLPSO	FDR-PSO	UPSO	TSL-PSO	GGL-PSOD	DMSDL-PSO	DE/rand/1	MLDE-PSO
BF_8	FV	**= 0.000E + 00**	>2.802E + 01	>2.802E + 01	**= 0.000E + 00**	= 5.329E-16	**= 0.000E + 00**	= 1.412E + 02	**0.000E + 00**
	SP	2.276E + 05	Inf	Inf	3.349E + 04	2.178E + 05	6.946E + 04	Inf	**2.322E + 04**
	SR	**100.00%**	0.00%	0.00%	**100.00%**	**100.00%**	**100.00%**	0.00%	**100.00%**
	Rank	**1**	6	7	**1**	5	**1**	8	**1**
BF_9	FV	>1.886E + 01	>2.315E-14	>2.315E-14	>3.417E-14	>8.112E-15	>6.454E-15	>5.151E-15	**>8.882E-16**
	SP	Inf	1.300E + 05	4.545E + 04	4.354E + 04	1.688E + 05	7.549E + 04	1.334E + 05	**2.658E + 04**
	SR	0.00%	**100.00%**	96.67%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	Rank	8	5	6	7	4	3	2	**1**
BF_{10}	FV	>7.401E-18	>1.532E-02	>1.532E-02	>6.049E-13	>3.281E-03	**= 0.000E + 00**	**= 0.000E + 00**	**0.000E + 00**
	SP	1.396E + 05	6.346E + 05	1.160E + 05	3.443E + 04	2.423E + 05	6.193E + 04	9.464E + 04	**1.901E + 04**
	SR	**100.00%**	36.67%	80.00%	**100.00%**	76.67%	**100.00%**	**100.00%**	**100.00%**
	Rank	**1**	7	8	5	6	**1**	**1**	**1**
BF_{11}	FV	<3.528E-31	>3.456E-03	>3.456E-03	**<1.571E-32**	**<1.571E-32**	**<1.571E-32**	<6.037E-32	5.525E-04
	SP	1.211E + 05	1.163E + 05	5.832E + 04	**2.561E + 04**	1.376E + 05	4.493E + 04	8.108E + 04	Inf
	SR	**100.00%**	96.67%	93.33%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	0.00%
	Rank	5	7	8	**1**	**1**	**1**	4	6
BF_{12}	FV	<3.601E-30	<1.465E-03	<1.465E-03	**<1.350E-32**	**<1.350E-32**	**<1.350E-32**	<3.510E-31	4.711E-2
	SP	1.283E + 05	1.586E + 05	**2.594E + 04**	2.900E + 04	1.453E + 05	4.872E + 04	8.769E + 04	Inf
	SR	**100.00%**	86.67%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	0.00%
	Rank	5	6	7	**1**	**1**	**1**	4	8
BF_{13}	FV	**= 0.000E + 00**	>9.367E + 00	>9.367E + 00	**= 0.000E + 00**	= 2.368E-16	**= 0.000E + 00**	>1.217E + 02	**0.000E + 00**
	SP	2.401E + 05	Inf	Inf	3.459E + 04	2.188E + 05	7.863E + 04	Inf	**2.764E + 04**
	SR	**100.00%**	0.00%	0.00%	**100.00%**	**100.00%**	**100.00%**	0.00%	**100.00%**
	Rank	**1**	6	7	**1**	5	**1**	8	**1**

$$\mathrm{mean}(SP) = FEMAX * (1 - SR)/SR + \mathrm{mean}(FE \text{ of success runs}) \tag{17}$$

In this study, the predefined accuracy is ε = 1 × 10^{-6}. *SR* denotes for success rate, it equals to the quotient of number of success runs divided by total runs. In the pairwise wilcoxon signed-rank test, the symbol ">","=" and "<" in row of "*FV*" denote for the performance of MLDE-PSO is "significant better than""tie with", "significant worse than" the compared algorithm, respectively. The row of "rank" stands for the rank of mean errors among eight algorithms. The best results are highlighted in bold.

The results in Table 3 indicated that MLDE-PSO ranks the first on six unimodal functions out of seven unimodal functions. Only on BF_5, MLDE-PSO performs worse than other peer algorithms. MLDE-PSO achieves zero errors on BF_1, BF_2, BF_3, and BF_6. On BF_4, MLDE-PSO achieves very high accuracy too. MLDE-PSO yields the lowest *SP* on BF_1-BF_4, BF_6 and BF_7, which means the convergence speed of MLDE-PSO on these functions are faster than other compared algorithms. DMSDL-PSO and UPSO achieve the lowest *SP* on BF_5 and BF_6, respectively. If one algorithm can't achieve the given accuracy in thirty runs, its SP is marked with "INF". In the aspect of success rate "SR", MLDE-PSO generated 100% success rate on all unimodal functions except BF_5. On Rosenbrock function BF_5, MLDE-PSO fails to yield the given accuracy in thirty runs. All the algorithms yield 100% success rate on BF_1-BF_3. On BF_6, only FDR-PSO and UPSO fail to yield 100% success rate. The Wilcoxon signed rank test results show that MLDE-PSO performs significant better than the other algorithms on BF_1-BF_4 and BF_7. On BF_6, MLDE-PSO's performance is better than FDR-PSO and UPSO, tie with the rest peer algorithms. In the criteria of rank of mean errors, MLDE-PSO ranks the first on BF_1-BF_4, BF_6 and BF_7. On BF_5, MLDE-PSO ranks the eighth.

On six multimodal functions, MLDE-PSO achieves the best performance on BF_8-BF_{10} and BF_{13}, on two penalty function BF_{11} and BF_{12}, MLDE-PSO only performs better than other FDR-PSO and UPSO on BF_{11}. MLDE-PSO consumes the lowest SP on BF_8-BF_{10} and BF_{13}. On BF_{11} and BF_{12}, TSL-PSO and UPSO yields the lowest SP, respectively. On the aspect of success rate, MLDE-PSO generates 100% success rate except BF_{11} and BF_{12}. On two penalty functions BF_{11} and BF_{12}, MLDE-PSO fails to achieve the given accuracy in thirty runs, MLDE-PSO can't yield high performance, for than penalty function changed the landscape of the optimization problem. CLPSO, TSL-PSO, GGL-PSOD, DMSDL-PSO and DE/rand/1 achieve 100% SR on $BF_{11,}$ while on BF_{12}, only FDR-PSO and MLDE-PSO fail to achieve 100% SR. The Wilcoxon signed rank test results indicate that MLDE-PSO performs better than other algorithms on BF_9, performs no worsen than other algorithms on BF_8, BF_{10}, and BF_{13}. MLDE-PSO ranks the first on BF_8-BF_{10} and BF_{11}, ranks the sixth, eighth on BF_{11} and BF_{12}, respectively.

The test results on basic test functions are summarized in Table 5. According to the summary of test results in Table 5, MLDE-PSO yields the best performance for 6, 4 and 10 times on unimodal functions, multimodal functions and all thirteen basic functions, respectively. The average ranks show that MLDE-PSO yields the best rank on unimodal functions, DMSDL-PSO yields the best rank on multimodal functions and all thirteen basic functions. The Wilcoxon signed rank test results show that on unimodal functions MLDE-PSO performs better than other peer algorithms on six or five functions. On multimodal functions, MLDE-PSO is outperformed by DMSDL-PSO, CLPSO and TSL-PSO on three, two, and two functions, respectively. The overall performance of MLDE-PSO is worse than DMSDL-PSO and CLPSO, almost the same with TSL-PSO, better than FDR-PSO, UPSO, GGL-PSOD and DE/rand/1. On all thirteen functions, MLDE-PSO outperforms CLPSO, FDR-PSO, UPSO, TSL-PSO, GGL-PSOD, DMSDL-PSO and DE/rand/1 on six, ten, nine, seven, nine, six and eight functions, respectively. For DMSDL-PSO employs local search strategy to enhance convergence, DMSDL-PSO performs better than MLDE-PSO on multimodal functions. The overall performance of MLDE-PSO is better than DMSDL-PSO, DE/rand/1 and other peer algorithms. For MLDE-PSO employs PSO subswarm and Elite-DE to enhance exploitation, it performs better than DE/rand/1. Only on BF_5 and penalty function BF_{11} and BF_{12}, MLDE-PSO performs worse than DE/rand/1.

4.4. Comparison test on rotated basic test functions

In this test, thirteen rotated basic functions are rotated form $BF_1 \sim BF_{13}$ with condition number is 2 matrix to increase the difficulty on finding the global optima. Table 6 indicate that on seven rotated unimodal functions (RBF_1-RBF_7), MLDE-PSO yields the best performance except on RBF_5. With local search strategy, DMSDL-PSO generates the best performance on RBF_5. MLDE-PSO consumes the lowest SP except RBF_1 and RBF_5, DMSDL-PSO achieves the lowest SP on RBF_1 and RBF_5. MLDE-PSO yields 100% success rate except RBF_5. On RBF_1 all algorithms yields100% success rate, on RBF_3 only CLPSO fail to yield 100% success rate. The Wilcoxon signed rank test show that MLDE-PSO performs better than other algorithms except on RBF_5 and RBF_6. On RBF_5, MLDE-PSO performs worse than DMSDL-PSO, better than the rest of peer algorithms. On RBF_6, MLDE-PSO performs better than FDR-PSO and TSL-PSO, performs almost the same with the rest algorithms. MLDE-PSO ranks the first except on RBF_5. On RBF_5, DMSDL-PSO ranks the first and MLDE-PSO ranks the second.

On six rotated multimodal functions, Table 7 shows that MLDE-PSO achieves the best performance except on two penalty functions RBF_{11} and RBF_{12}. DE/rand/1 yield the best performance on RBF_{11} and RBF_{12}. MLDE-PSO consumes the lowest *SP* except on RBF_{11} and RBF_{12}. TSL-PSO consumes the lowest *SP* on RBF_{11} and RBF_{12}. MLDE-PSO achieves the highest success rate on $RBF_{8\text{-}10}$ and RBF_{13}. On RBF_{11} TSL-PSO yields the highest success rate, on RBF_{12}, DMSDL-PSO yields the highest success rate. MLDE-PSO ranks the first on RBF_8,RBF_9, RBF_{10} and RBF_{13}, ranks the fourth on RBF_{11} and ranks the eighth on RBF_{12}. DE/rand/1 ranks the first on RBF_{11} and RBF_{12}.

The summary of test results in Table 8 indicate that MLDE-PSO yields the best performance for six, four and ten times on unimodal functions, multimodal functions and all rotated basic functions. MLDE-PSO achieves the best average rank on rotated unimodal functions, rotated multimodal functions and all rotated basic functions. The Wilcoxon signed-rank test results show MLDE-PSO performs better than other algorithms on unimodal functions, multimodal functions and all rotated function. For MLDE-PSO has DE subswarm, its performance is not impacted significantly by rotate transformation. For DE/rand/1 is rotation invariant, it performs well on rotated basic functions too. For other peer PSO algorithms, their performance are impaired by rotate transformation. Hence MLDE-PSO win more advantages on rotated functions than on basic functions.

4.5. Comparison test on CEC2017 functions

Table 9 indicates that among thirty CEC2017 functions, MLDE-PSO yields the best performance on all three unimodal functions,

Table 5
Summary of test results on basic test functions.

Type	Statistic	CLPSO	FDR-PSO	UPSO	TSL-PSO	GGL-PSOD	DMSDL-PSO	DE/rand/1	MLDE-PSO
Unimodal	Best	1	0	0	1	1	2	1	**6**
	Avg. rank	6.286	3.857	4.857	3.714	4.429	3.143	5.571	**2.000**
	W/T/L	5/1/1	6/0/1	5/1/1	5/1/1	5/1/1	5/1/1	5/1/1	/
Multimodal	Best	2	0	0	4	2	**5**	1	4
	Avg. rank	2.833	6.333	7.167	2.833	3.500	**1.333**	4.667	3.000
	W/T/L	1/3/2	5/0/1	5/0/1	2/2/2	4/0/2	1/3/2	3/1/2	/
All	Best	3	0	0	5	3	7	2	**10**
	Avg. rank	4.692	4.923	5.846	3.308	4.000	**2.308**	5.154	2.462
	W/T/L	6/4/3	11/0/2	10/1/2	7/3/3	9/1/3	6/4/3	8/2/3	/

Table 6
Test results on unimodal rotated basic test functions.

Func.	Statistic	CLPSO	FDR-PSO	UPSO	TSL-PSO	GGL-PSOD	DMSDL-PSO	DE/rand/1	MLDE-PSO
RBF1	FV	>1.428E-12	>2.156E-114	>1.252E-98	>2.340E-121	>1.161E-52	>1.631E-43	>5.373E-28	**0.000E + 00**
	SP	1.796E + 05	1.198E + 05	3.040E + 04	4.859E + 04	1.535E + 05	**7.736E + 03**	1.009E + 05	1.923E + 04
	SR	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	Rank	8	3	4	2	5	6	7	**1**
RBF2	FV	>2.677E + 01	>3.911E-01	>1.094E + 01	>1.123E + 01	>4.965E-04	>1.991E-05	>3.290E-11	**3.523E-152**
	SP	Inf	Inf	Inf	Inf	2.003E + 05	1.458E + 06	2.017E + 05	**2.977E + 04**
	SR	0.00%	0.00%	0.00%	0.00%	93.33%	20.00%	**100.00%**	**100.00%**
	Rank	8	5	6	7	4	3	2	**1**
RBF3	FV	>1.098E + 01	>6.727E-19	>1.288E-10	>1.872E-31	>3.761E-12	>8.727E-12	>5.154E-16	**0.000E + 00**
	SP	Inf	1.919E + 05	1.944E + 05	8.210E + 04	2.307E + 05	2.497E + 05	1.568E + 05	**2.233E + 04**
	SR	0.00%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	Rank	8	3	7	2	5	6	4	**1**
RBF4	FV	>2.665E + 00	>1.685E + 00	>9.204E + 00	>3.958E + 00	>1.507E-05	>4.334E-02	>1.451E-07	**3.737E-136**
	SP	Inf	Inf	Inf	Inf	3.260E + 05	Inf	2.704E + 05	**3.241E + 04**
	SR	0.00%	0.00%	0.00%	0.00%	70.00%	0.00%	**100.00%**	**100.00%**
	Rank	6	5	8	7	3	4	2	**1**
RBF5	FV	>1.306E + 02	>5.435E + 01	>3.000E + 01	>6.366E + 01	>4.256E + 01	**<9.319E-11**	>3.595E + 01	2.881E + 01
	SP	Inf	Inf	Inf	Inf	Inf	**2.751E + 05**	Inf	Inf
	SR	0.00%	0.00%	0.00%	0.00%	0.00%	**100.00%**	0.00%	0.00%
	Rank	5	7	3	8	6	**1**	4	2
RBF6	FV	**= 0.000E + 00**	>4.100E + 00	**= 0.000E + 00**	>9.667E-01	>3.333E-02	**= 0.000E + 00**	**= 0.000E + 00**	**0.000E + 00**
	SP	1.318E + 05	Inf	3.567E + 04	2.204E + 05	1.269E + 05	3.783E + 04	4.726E + 04	**8.631E + 03**
	SR	**100.00%**	0.00%	**100.00%**	96.67%	96.67%	**100.00%**	**100.00%**	**100.00%**
	Rank	**1**	8	**1**	7	6	**1**	**1**	**1**
RBF7	FV	>5.468E-03	>2.637E-03	>1.347E-02	>5.054E-03	>2.128E-03	>3.495E-03	>5.338E-03	**1.612E-04**
	SP	Inf	Inf	Inf	Inf	Inf	Inf	Inf	Inf
	SR	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	**90.00%**
	Rank	8	3	7	5	2	4	6	**1**

Table 7
Test results on multimodal rotated basic test functions.

Func.	Statistic	CLPSO	FDR-PSO	UPSO	TSL-PSO	GGL-PSOD	DMSDL-PSO	DE/rand/1	MLDE-PSO
RBF8	FV	>1.009E + 02	>4.985E + 01	>9.170E + 01	>8.626E + 01	>3.323E + 01	>3.293E + 01	>1.810E + 02	**0.000E + 00**
	SP	Inf	Inf	Inf	Inf	Inf	Inf	Inf	**3.442E + 04**
	SR	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	**100.00%**
	Rank	7	4	6	5	3	2	8	**1**
RBF9	FV	>1.923E + 00	>1.866E + 00	>2.281E + 00	>2.519E + 00	>1.036E-14	>7.832E-09	>1.344E-14	**8.882E-16**
	SP	Inf	4.340E + 06	Inf	7.666E + 05	1.746E + 05	1.833E + 05	1.513E + 05	**2.795E + 04**
	SR	0.00%	6.67%	0.00%	30.00%	**100.00%**	**100.00%**	**100.00%**	**100.00%**
	Rank	6	5	7	8	2	4	3	**1**
RBF10	FV	>2.951E-04	>8.865E-03	>2.712E-03	>6.271E-04	>3.122E-03	>4.138E-12	**= 0.000E + 00**	**0.000E + 00**
	SP	Inf	6.526E + 05	1.835E + 05	5.821E + 04	3.157E + 05	1.358E + 05	1.253E + 05	**2.145E + 04**
	SR	0.00%	36.67%	70.00%	**100.00%**	66.67%	**100.00%**	**100.00%**	**100.00%**
	Rank	4	8	6	5	7	3	**1**	**1**
RBF11	FV	>4.743E-01	>8.296E-02	>3.857E + 00	= 7.509E-01	<3.456E-03	<6.911E-03	**<3.962E-20**	2.218E-02
	SP	Inf	3.727E + 05	2.889E + 06	7.546E + 04	1.629E + 05	2.863E + 05	1.131E + 05	Inf
	SR	0.00%	60.00%	10.00%	**100.00%**	96.67%	93.33%	**100.00%**	0.00%
	Rank	6	5	8	7	2	3	**1**	4
RBF12	FV	<2.934E-05	<3.662E-03	<1.831E-03	<1.748E-03	<3.662E-04	<1.069E-12	**<3.976E-20**	6.654E-1
	SP	Inf	2.795E + 05	1.092E + 05	7.668E + 04	1.650E + 05	1.767E + 05	1.170E + 05	Inf
	SR	0.00%	66.67%	83.33%	96.67%	96.67%	**100.00%**	**100.00%**	0.00%
	Rank	3	7	6	5	4	2	**1**	8
RBF13	FV	>1.104E + 02	>7.262E + 01	>1.165E + 02	>1.017E + 02	>4.790E + 01	>5.904E + 01	>1.523E + 02	**1.282E + 01**
	SP	Inf	Inf	Inf	Inf	Inf	Inf	Inf	1.081E + 05
	SR	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	**90.00%**
	Rank	6	4	7	5	2	3	8	**1**

Table 8
Summary of test results on rotated basic test functions.

Type	Statistic	CLPSO	FDR-PSO	UPSO	TSL-PSO	GGL-PSOD	DMSDL-PSO	DE/rand/1	MLDE-PSO
Unimodal	Best	1	0	1	0	0	2	1	**6**
	Avg. rank	6.286	4.857	5.143	5.429	4.429	3.571	3.714	**1.143**
	W/T/L	6/1/0	7/0/0	6/1/0	7/0/0	6/1/0	5/1/1	6/1/0	/
Multimodal	Best	0	0	0	0	0	0	3	**4**
	Avg. rank	5.333	5.500	6.667	5.833	3.333	**2.833**	3.667	**2.667**
	W/T/L	5/0/1	5/0/1	5/0/1	4/1/1	4/0/2	4/0/2	3/1/2	/
All	Best	1	0	1	0	0	2	4	**10**
	Avg. rank	6.000	5.143	5.714	5.429	3.714	3.143	3.929	**2.1434**
	W/T/L	11/1/1	12/0//1	11/1/1	11/1/1	10/1/2	9/1/3	9/2/2	/

Table 9
Test results on CEC2017 functions.

Func.	CLPSO	FDR-PSO	UPSO	TSL-PSO	GGL-PSOD	DMSDL-PSO	DE/rand/1	MLDE-PSO
F1	> 2.419E + 01	> 1.677E + 03	> 3.377E + 03	> 1.677E + 03	> 2.876E + 03	> 7.447E + 01	> 1.232E-14	**0.000E + 00**
F2	> 1.629E + 12	> 4.914E + 10	> 1.486E + 16	> 4.914E + 10	> 1.509E + 06	> 1.948E + 10	> 4.016E + 07	**1.667E + 00**
F3	> 1.252E + 04	= 4.145E-08	> 4.754E + 01	= 4.145E-08	= 1.103E-07	> 1.199E-05	> 3.416E + 01	**2.519E-09**
F4	> 6.472E + 01	= 2.465E + 01	> 1.240E + 02	= 2.465E + 01	> 6.404E + 01	**< 1.728E + 00**	> 5.893E + 01	4.913E + 01
F5	> 4.242E + 01	> 5.877E + 01	> 8.236E + 01	> 5.877E + 01	> 3.042E + 01	> 3.841E + 01	> 1.738E + 02	**2.584E + 01**
F6	< 2.008E-13	> 1.860E-01	> 1.294E + 00	> 1.860E-01	< 5.917E-08	**< 1.137E-13**	< 5.267E-08	1.091E-07
F7	> 8.487E + 01	> 1.009E + 02	> 1.334E + 02	> 1.009E + 02	= 6.282E + 01	> 7.200E + 01	> 2.041E + 02	**6.198E + 01**
F8	> 4.975E + 01	> 5.860E + 01	> 8.430E + 01	> 5.860E + 01	> 3.279E + 01	> 4.059E + 01	> 1.775E + 02	**2.790E + 01**
F9	> 8.342E + 00	> 1.827E + 01	> 2.641E + 02	> 1.827E + 01	> 6.356E-02	> 2.192E-01	**= 0.000E + 00**	**0.000E + 00**
F10	= 2.276E + 03	> 2.988E + 03	> 3.334E + 03	> 2.988E + 03	= 2.116E + 03	**< 1.874E + 03**	> 6.821E + 03	2.419E + 03
F11	> 5.139E + 01	> 9.403E + 01	> 1.291E + 02	> 9.403E + 01	> 3.314E + 01	> 1.413E + 01	> 6.043E + 01	**1.372E + 01**
F12	> 3.113E + 05	> 2.130E + 04	> 1.094E + 05	> 2.130E + 04	> 2.348E + 04	**< 1.681E + 03**	= 7.365E + 03	6.671E + 03
F13	> 3.158E + 02	> 1.078E + 04	> 8.809E + 03	> 1.078E + 04	> 1.433E + 04	> 1.894E + 03	> 7.967E + 01	**2.367E + 01**
F14	> 2.362E + 04	> 4.989E + 03	> 6.747E + 03	> 4.989E + 03	> 2.235E + 03	> 3.967E + 02	> 6.207E + 01	**1.484E + 01**
F15	> 1.233E + 02	> 5.323E + 03	> 6.231E + 03	> 5.323E + 03	> 4.358E + 03	> 1.037E + 02	> 3.738E + 01	**5.151E + 00**
F16	= 4.912E + 02	= 6.899E + 02	> 8.401E + 02	= 6.899E + 02	**< 2.346E + 02**	< 4.409E + 02	> 6.367E + 02	5.869E + 02
F17	> 1.100E + 02	> 2.257E + 02	> 2.510E + 02	> 2.257E + 02	= 7.616E + 01	> 7.973E + 01	> 7.616E + 01	**6.442E + 01**
F18	> 1.306E + 05	> 7.984E + 04	> 1.753E + 05	> 7.984E + 04	> 8.277E + 04	> 1.475E + 03	> 3.758E + 01	**2.403E + 01**
F19	> 9.529E + 01	> 6.028E + 03	> 4.190E + 03	> 6.028E + 03	> 7.658E + 03	> 2.832E + 02	> 1.863E + 01	**4.270E + 00**
F20	> 1.712E + 02	> 2.360E + 02	> 3.651E + 02	> 2.360E + 02	> 1.273E + 02	> 1.465E + 02	**= 2.665E + 01**	6.688E + 01
F21	> 2.430E + 02	> 2.575E + 02	> 2.814E + 02	> 2.575E + 02	> 2.331E + 02	> 2.312E + 02	> 3.683E + 02	**2.261E + 02**
F22	> 1.966E + 02	> 1.007E + 03	> 6.835E + 02	> 1.007E + 03	> **1.000E + 02**	> 2.288E + 02	**= 1.000E + 02**	**1.000E + 02**
F23	> 3.976E + 02	> 4.150E + 02	> 4.379E + 02	> 4.150E + 02	> 3.822E + 02	> 3.922E + 02	> 5.177E + 02	**3.714E + 02**
F24	> 4.720E + 02	> 4.755E + 02	> 4.906E + 02	> 4.755E + 02	> 4.570E + 02	> 4.888E + 02	> 5.860E + 02	**4.448E + 02**
F25	< 3.867E + 02	> 3.927E + 02	> 4.115E + 02	> 3.927E + 02	> 3.872E + 02	**< 3.810E + 02**	< 3.867E + 02	3.868E + 02
F26	**< 7.658E + 02**	> 1.238E + 03	> 2.010E + 03	> 1.238E + 03	= 8.786E + 02	= 8.941E + 02	> 2.370E + 03	1.105E + 03
F27	> 5.106E + 02	> 5.277E + 02	> 5.616E + 02	> 5.277E + 02	> 5.040E + 02	> 5.062E + 02	**< 4.872E + 02**	4.968E + 02
F28	> 4.105E + 02	> 3.370E + 02	> 4.428E + 02	> 3.370E + 02	> 3.352E + 02	**< 3.000E + 02**	> 3.203E + 02	3.141E + 02
F29	> 5.452E + 02	> 6.040E + 02	> 9.311E + 02	> 6.040E + 02	> 4.515E + 02	> 5.189E + 02	> 5.816E + 02	**4.215E + 02**
F30	> 5.396E + 03	> 4.751E + 03	> 9.404E + 04	> 4.751E + 03	> 4.010E + 03	> 4.095E + 03	**= 1.997E + 03**	2.024E + 03
W/T/L	25/2/3	27/3/0	30/0/0	27/3/0	23/5/2	22/1/7	21/6/3	
Best	1	0	0	0	2	6	5	19

Note: The symbol ">","=" and "<" denote for the performance of MLDE-PSO is "significant better than", "tie with" and "significant worse than" the compared algorithm in the wilcoxon signed rank test with significance level of 0.05.

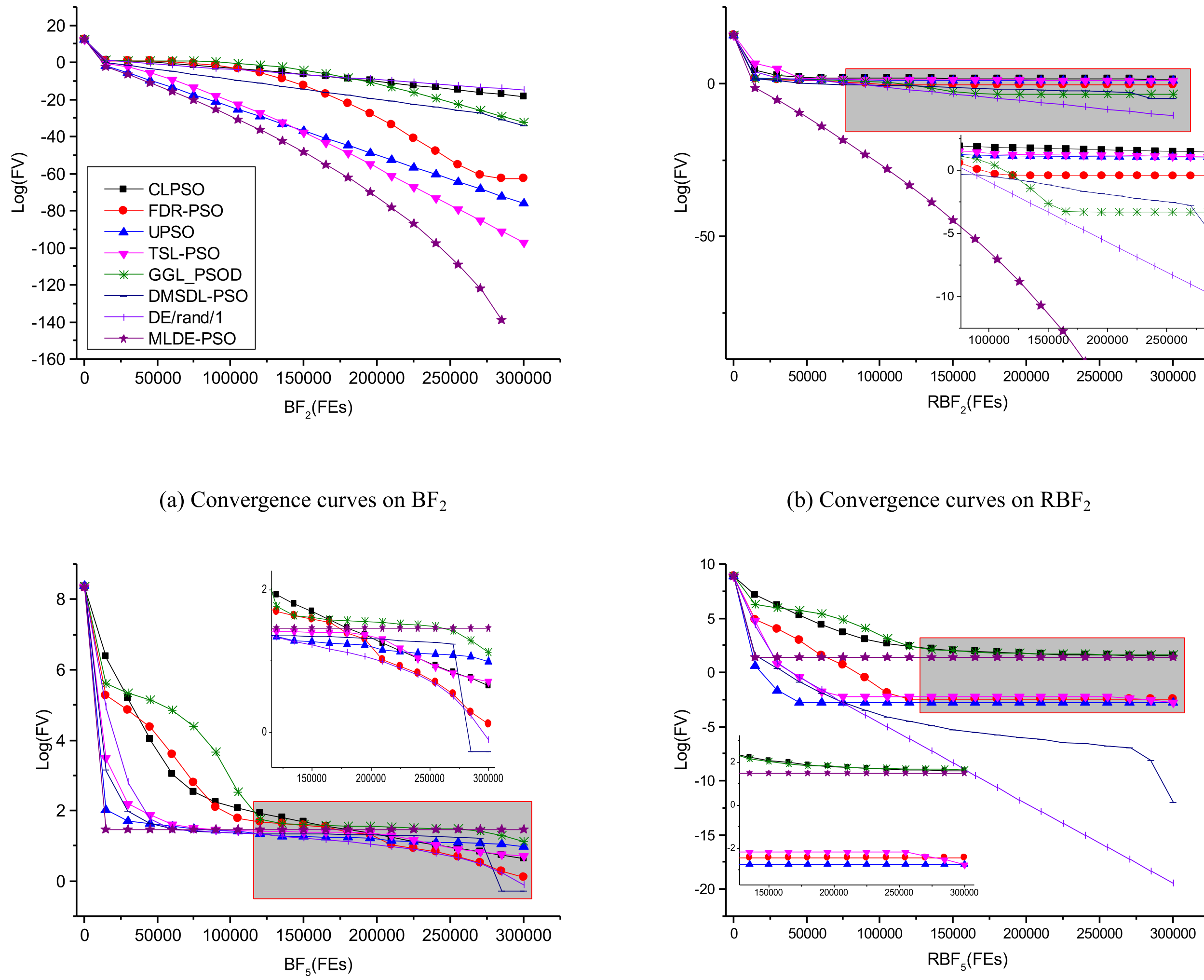
four simple multimodal functions, seven hybrid functions and five composition functions. On functions F_9, MLDE-PSO and DE/rand/1 tie for first, and on function F_{22},GGL-PSOD, DE/rand/1 tie for first. MLDE-PSO win the best performance on nineteen functions out of thirty CEC2017 functions. DMSDL-PSO, DE/rand/1, GGL-PSOD and CLPSO win the best performance on six, five, two and one functions, respectively. MLDE-PSO outperforms CLPSO, FDR-PSO, UPSO, TSL-PSO, GGL-PSO, DMSDL-PSO and DE/rand/1 on twenty five, twenty seven, thirty, twenty seven, twenty three, twenty two and twenty one functions, respectively. MLDE-PSO exhibits high performance on different types of CEC2017 functions. The advantage of MLDE-PSO on unimodal functions, simple multimodal functions and hybrid functions are more significant than on composition functions. The overall performance of MLDE-PSO is better than DE/rand/1, DMSDL-PSO and other peer algorithms. Though DMSDL-PSO employs DE to construct learning exemplar and local search strategy to enhance exploitation, it is still outperformed by MLDE-PSO.

4.6. Convergence speed analysis

To compare the convergence speed of MLDE-PSO with other peer algorithms, the convergence curves four basic functions and four rotated functions are given in Fig. 3. The rotated function RBF_2, RBF_5, RBF_{10} and RBF_{12} are modified form basic function BF_2, BF_5, BF_{10} and BF_{12}, thereby the impact of rotation transformation can be evaluated by comparing the convergence curves of basic function and relevant rotate function. On BF_2, MLDE-PSO converges fast and achieve high accuracy, TSL-PSO, UPSO and FDR-PSO follows MLDE-PSO. With PSO subswarm and elite DE mutation, MLDE-PSO performs much better than DE/rand/1. On RBF_2, MLDE-PSO yield high accuracy and win more advantages, other peer algorithms' performance are impaired by rotation transformation significantly. On BF_5, MLDE-PSO is trapped into local optima, DMSDL-PSO, DE/rand/1 and FDR-PSO occupy top three. On RBF_5, DE/rand/1, DMSDL-PSO yield high accuracy. GGL-PSOD, CLPSO and MLDE-PSO fall behind.

On multimodal functions BF_{10}, MLDE-PSO find the global optima very fast, DMSDL-PSO, DE/rand/1, CLPSO and TSL-PSO achieve high accuracy too, UPSO, GGL-PSOD and FDR-PSO are trapped into local optima. On RBF_{10}, MLDE-PSO, DMSDL-PSO and DE/rand/1 obtain high performance, TSL-PSO and CLPSO are impacted by rotation transformation and can't find the global optima. On BF_{12}, MLDE-PSO fail to achieve high accuracy, for the penalty function misguiding MLDE-PSO to local optima. FDR-PSO falls into local optima too, the rest algorithms find the global optima successfully. On RBF_{12}, DE/rand/1 and DMSDL-PSO obtain high accuracy, UPSO and MLDE-PSO fall behind, MLDE-PSO performs slightly better than UPSO.

The convergence curves indicate that DE/rand/1 and MLDE-PSO are not sensitive to rotation transformation. MLDE-PSO performs well on BF_2, BF_{10}, RBF_2 and RBF_{10}, while on BF_5, BF_{12}, RBF_5 and RBF_{12}, MLDE-PSO can't achieve high accuracy. The penalty function changes the landscape of optimization problem and mislead MLDE-PSO to local optima.



(a) Convergence curves on BF_2

(b) Convergence curves on RBF_2

(c) Convergence curves on BF_5

(d) Convergence curves on RBF_5

Fig. 3. convergence curves of MLDE-PSO and peer algorithms.

4.7. Impact of PSO subswarm size

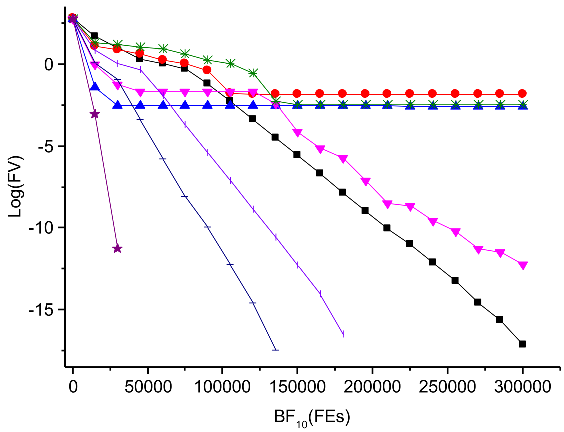
In this section, the PSO subswarm size is set as N_{PSO} = 1, 3, 5, 7, 9 to analyze the impact of PSO subswarm size, other parameters are set according to Table 2. The mean fitness value "*FV*" and rank of mean errors are presented in Table 10. The results in Table 10 show that BF_1, BF_2, BF_3, BF_6, BF_8, BF_9, BF_{10} and BF_{13} are not sensitive to PSO subswarm population size. On the rest functions, the overall performance of N_{PSO} = 3 performs better than other setting, N_{PSO} = 1 and N_{PSO} = 9 performs well too. Increasing the PSO subswarm size can improve the proposed algorithm's performance on BF_5 and BF_{11}.

5. Conclusions

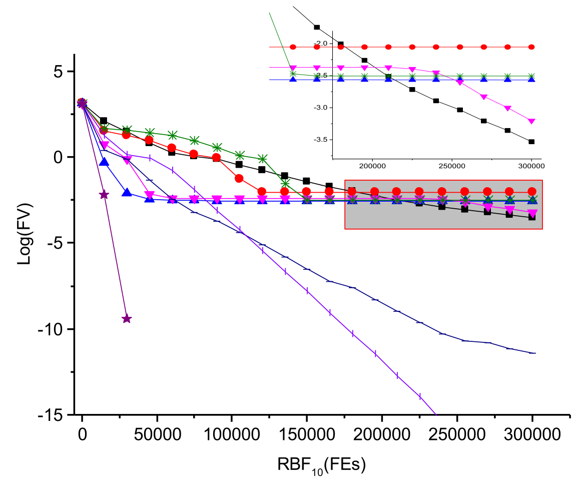
This paper proposed a mutual learning strategy for hybridizing DE and PSO. Based on the mutual learning strategy, mutual learning differential evolution particle swarm optimization (MLDE-PSO) is proposed. MLDE-PSO employs DE and PSO subswarm to evolve their own subswarm and valuable information is exchanged between two subswarms. The position information in PSO subswarm is employed for DE mutation and the DE individuals are employed for constructing learning exemplars for PSO subswarm. To further improve exploitation of DE subswarm, an elite DE mutation is adopted. The experiments on basic functions, rotated basic functions and CEC2017 functions indicated that the overall performance of MLDE-PSO is better than other compared classic PSO algorithms, recently PSO algorithms and classic DE. On complex functions and rotated functions, MLDE-PSO exhibits more advantages. MLDE-PSO is less sensitive to rotation transformation than compared algorithms. On penalty functions, the performance of MLDE-PSO is relatively weak. In the future research, MLDE-PSO will be extended to constraint test functions and real-life optimization problems.

Declaration of Competing Interest

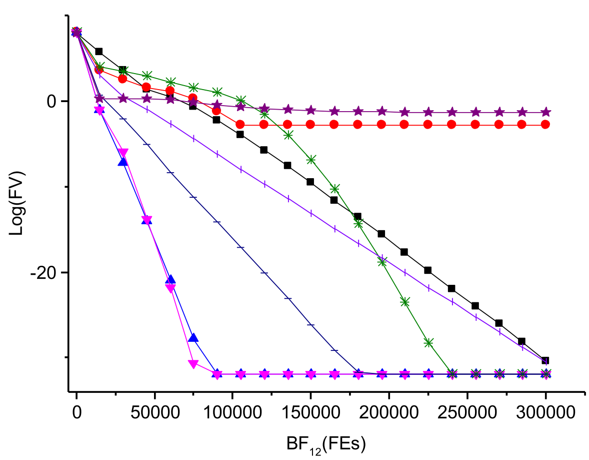
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.



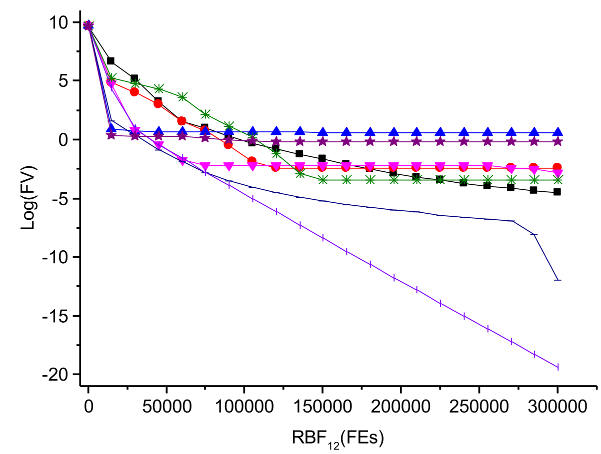
(e) Convergence curves on BF_{10}



(f) Convergence curves on RBF_{10}



(g) Convergence curves on BF_{12}



(h) Convergence curves on RBF_{12}

Fig. 3 (*continued*)

Table 10
Parameters selection of MLDE-PSO.

Func.	N_{PSO} = 1		N_{PSO} = 3		N_{PSO} = 5		N_{PSO} = 7		N_{PSO} = 9	
Statistic	FV	Rank	FV	Rank	FV	Rank	FV	Rank	FV	Rank
BF1	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**
BF2	**0.000E + 00**	1	**0.000E + 00**	1	**0.000E + 00**	1	**0.000E + 00**	**1**	**0.000E + 00**	**1**
BF3	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**
BF4	**3.203E-134**	**1**	1.377E-131	2	1.665E-129	3	1.535E-127	4	6.691E-124	**5**
BF5	2.844E + 01	**5**	2.801E + 01	4	2.717E + 01	3	2.630E + 01	2	**2.552E + 01**	**1**
BF6	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**
BF7	**1.642E-04**	**1**	1.953E-04	2	1.973E-04	5	1.968E-04	**4**	1.960E-04	3
BF8	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**
BF9	**8.882E-16**	**1**	**8.882E-16**	**1**	**8.882E-16**	**1**	**8.882E-16**	**1**	**8.882E-16**	**1**
BF10	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**
BF11	8.078E-03	5	5.523E-04	4	5.736E-07	3	1.970E-12	2	**5.908E-15**	**1**
BF12	5.170E-01	2	**4.711E-2**	**1**	5.320E-02	**3**	5.784E-02	5	5.494E-02	4
BF13	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**	**0.000E + 00**	**1**
Avg.	/	1.693	/	1.615	/	1.923	/	1.923	/	1.692

Acknowledgements

The author thanks the reviewers and other scholars for providing precious suggestion on this manuscript. This work is supported by the Scientific Research Fund of Hunan Provincial Education Department under Grant 20A460 and 20B536, the Scientific Research Start-up Fund for High-level Talents in Xiangnan University, the Applied Characteristic Disciplines of Electronic Science and Technology of Xiangnan University. (Code is available https://www.researchgate.net/publication/360191796_Mutual_learning_differential_particle_swarm_optimization).

References:

[1] Lynn N, Ali MZ, Suganthan PN. Population topologies for particle swarm optimization and differential evolution. Swarm Evol Comput 2018;39:24–35.
[2] Qin Q, Cheng S, Zhang Q, Li Li, Shi Y. Particle swarm optimization with interswarm interactive learning strategy. IEEE Trans Cybern 2016;46(10):2238–51.
[3] Xiang Z, Ji D, Zhang H, Wu H, Li Y. A simple PID-based strategy for particle swarm optimization algorithm. Inf Sci 2019;502:558–74.
[4] Chen X, Tianfield H, Du W. Bee-foraging learning particle swarm optimization. Appl Soft Comput 2021;102:107134.
[5] E. H. Houssein, A. G. Gad, K. Hussain, and P. N. Suganthan, *Major Advances in Particle Swarm Optimization: Theory, Analysis, and Application*, Swarm and Evolutionary Computation, vol. 63, 2021/06/01/ 2021, p. 100868.
[6] Liang JJ, Qin AK, Member S, Suganthan PN, Member S, Baskar S. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evol Comput 2006;10:281–95.
[7] Cheng R, Jin Y. A social learning particle swarm optimization algorithm for scalable optimization. Inf Sci 2015;291:43–60.
[8] Xu G, Cui Q, Shi X, Ge H, Zhan Z-H, Lee HP, et al. Particle swarm optimization based on dimensional learning strategy. Swarm Evol Comput 2019;45:33–51.
[9] S. Molaei, H. Moazen, S. Najjar-Ghabel, and L. Farzinvash, *Particle swarm optimization with an enhanced learning strategy and crossover operator*, Knowledge-Based Systems, vol. 215, 2021/03/05/ 2021, p. 106768.
[10] Lim WH, Mat Isa NA. An adaptive two-layer particle swarm optimization with elitist learning strategy. Inf Sci 2014;273:49–72.
[11] Yan B, Zhao Z, Zhou Y, Yuan W, Li J, Wu J, et al. A particle swarm optimization algorithm with random learning mechanism and Levy flight for optimization of atomic clusters. Comput Phys Commun 2017;219:79–86.
[12] Ye W, Feng W, Fan S. A novel multi-swarm particle swarm optimization with dynamic learning strategy. Appl Soft Comput 2017;61:832–43.
[13] Juang C-F. A hybrid of genetic algorithm and particle swarm optimization for recurrent network design. IEEE Trans Syst Man Cybernet Part B Cybernet A Publ IEEE Syst Man Cybernet Soc 2004;34(2):997–1006.
[14] Javidrad F, Nazari M. A new hybrid particle swarm and simulated annealing stochastic optimization method. Appl Soft Comput 2017;60:634–54.
[15] X. Chen, H. Tianfield, C. Mei, W. Du, and G. Liu, *Biogeography-based learning particle swarm optimization*, Soft Computing, vol. 21, 2017/12/01 2017, pp. 7519-7541.
[16] Charin C, Ishak D, Mohd Zainuri MAA, Ismail B, Mohd Jamil MK. A hybrid of bio-inspired algorithm based on Levy flight and particle swarm optimizations for photovoltaic system under partial shading conditions. Sol Energy 2021;217:1–14.
[17] Jafari M, Salajegheh E, Salajegheh J. Optimal design of truss structures using a hybrid method based on particle swarm optimizer and cultural algorithm. Structures 2021;32:391–405.
[18] T. A. Khan and S. H. Ling, *A novel hybrid gravitational search particle swarm optimization algorithm*, Engineering Applications of Artificial Intelligence, vol. 102, 2021/06/01/ 2021, p. 104263.
[19] Z. Wen-Jun and X. Xiao-Feng, *DEPSO: hybrid particle swarm with differential evolution operator*, in SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance (Cat. No.03CH37483), 2003, pp. 3816-3821 vol.4.
[20] Epitropakis MG, Plagianakos VP, Vrahatis MN. Evolving cognitive and social experience in Particle Swarm Optimization through Differential Evolution: A hybrid approach. Inf Sci 2012;216:50–92.
[21] H. Liu, Z. Cai, and Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization: Elsevier Science Publishers B. V., 2010.
[22] Hao Z, Guo G, Huang H. A particle swarm optimization algorithm with differential evolution. In: 2007 International conference on machine learning and cybernetics. p. 1031–5.
[23] Chen K, Xue B, Zhang M, Zhou F. Hybridising particle swarm optimisation with differential evolution for feature selection in classification. In: 2020 IEEE Congress on Evolutionary Computation (CEC). p. 1–8.
[24] Shi YH, Eberhart R. A modified particle swarm optimizer. New York: IEEE; 1998.
[25] Das S, Mullick SS, Suganthan PN. Recent advances in differential evolution – An updated survey. Swarm Evol Comput 2016;27:1–30.
[26] Qin AK, Huang VL, Suganthan PN. Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans Evol Comput 2009;13(2):398–417.
[27] Xiao-gen P. A new hybrid differential particle swarm optimization algorithm and application. In: 2020 international conference on computer engineering and intelligent control (ICCEIC). p. 157–62.
[28] Sato M, Fukuyama Y, Iizaka T, Matsui T. Total optimization of energy networks in a smart city by multi-swarm differential evolutionary particle swarm optimization. IEEE Trans Sustainable Energy 2019;10(4):2186–200.
[29] Wang S, Li Y, Yang H. Self-adaptive mutation differential evolution algorithm based on particle swarm optimization. Appl Soft Comput 2019;81:105496.
[30] Sayah S, Hamouda A. A hybrid differential evolution algorithm based on particle swarm optimization for nonconvex economic dispatch problems. Appl Soft Comput J 2013;13(4):1608–19.
[31] Seyedmahmoudian M, Rahmani R, Mekhilef S, Maung Than Oo A, Stojcevski A, Soon TK, et al. Simulation and hardware implementation of new maximum power point tracking technique for partially shaded PV system using Hybrid DEPSO method. IEEE Trans Sustainable Energy 2015;6(3):850–62.
[32] Chen Y, Li L, Peng H, Xiao J, Yang Y, Shi Y. Particle swarm optimizer with two differential mutation. Appl Soft Comput 2017;61:314–30.
[33] Chen Y, Li L, Peng H, Xiao J, Wu Q. Dynamic multi-swarm differential learning particle swarm optimizer. Swarm Evol Comput 2018;39:209–21.
[34] Xin B, Chen J, Zhang J, Fang H, Peng Z-H. Hybridizing differential evolution and particle swarm optimization to design powerful optimizers: a review and taxonomy. IEEE Trans Syst Man Cybern Part C Appl Rev 2012;42(5):744–67.
[35] Zhan Z-H, Zhang J, Li Y, Shi Y-H. Orthogonal Learning Particle Swarm Optimization. IEEE Trans Evol Comput 2011;15(6):832–47.
[36] Gong Y-J, Li J-J, Zhou Y, Li Y, Chung H-H, Shi Y-H, et al. Genetic learning particle swarm optimization. IEEE Trans Cybern 2016;46(10):2277–90.
[37] M. Z. A. N.H. Awad, J.J. Liang, B.Y. Qu, P.N. Suganthan. (2016). Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective bound constrained real-parameter numerical optimization. Available: http://www.ntu.edu.sg/home/epnsugan/
[38] Suganthan PN, Hansen N, Liang JJ, Deb K, Chen YP, Auger A, et al. Problem definitions and evaluation criteria for the CEC2005 special session on real-parameter optimization. In: IEEE Congress on Evolutionary Computation. p. 1–50.
[39] Peram T, Veeramachaneni K, Mohan CK. Fitness-distance-ratio based particle swarm optimization. Swarm Intelligence Symp 2003:174–81.
[40] Parsopoulos KE, Vrahatis MN. A unified particle swarm optimization scheme. Int Conf Comput Methods Sci Eng, Ser lecture series on computer & computational sciences Attica. Greece: Vsp International Science; 2004.
[41] Lin A, Sun W, Yu H, Wu G, Tang H. Global genetic learning particle swarm optimization with diversity enhancement by ring topology. Swarm Evol Comput 2018.
[42] Storn R, Price K. Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 1997;11:341–59.