energies

MDPI

Article

Multi-Leader Comprehensive Learning Particle Swarm Optimization with Adaptive Mutation for Economic Load Dispatch Problems

Anping Lin [1,2,3] and Wei Sun [1,2,3,*]

1 College of Electrical and Information Engineering, Hunan University, Changsha 410082, China; anping719@hnu.edu.cn
2 State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body, Hunan University, Changsha 410082, China;
3 Hunan Key Laboratory of Intelligent Robot Technology in Electronic Manufacturing, Changsha 410082, China
* Correspondence: wei_sun@hnu.edu.cn; Tel.: +86-0731-88822756

Received: 5 November 2018; Accepted: 15 December 2018; Published: 29 December 2018

Abstract: Particle swarm optimization (PSO) is one of the most popular, nature inspired optimization algorithms. The canonical PSO is easy to implement and converges fast, however, it suffers from premature convergence. The comprehensive learning particle swarm optimization (CLPSO) can achieve high exploration while it converges relatively slowly on unimodal problems. To enhance the exploitation of CLPSO without significantly impairing its exploration, a multi-leader (ML) strategy is combined with CLPSO. In ML strategy, a group of top ranked particles act as the leaders to guide the motion of the whole swarm. Each particle is randomly assigned with an individual leader and the leader is refreshed dynamically during the optimization process. To activate the stagnated particles, an adaptive mutation (AM) strategy is introduced. Combining the ML and the AM strategies with CLPSO simultaneously, the resultant algorithm is referred to as multi-leader comprehensive learning particle swarm optimization with adaptive mutation (ML-CLPSO-AM). To evaluate the performance of ML-CLPSO-AM, the CEC2017 test suite was employed. The test results indicate that ML-CLPSO-AM performs better than ten popular PSO variants and six other types of representative evolutionary algorithms and meta-heuristics. To validate the effectiveness of ML-CLPSO-AM in real-life applications, ML-CLPSO-AM was applied to economic load dispatch (ELD) problems.

Keywords: multi-leader strategy; comprehensive learning; particle swarm optimization; adaptive mutation; economic load dispatch

1. Introduction

Optimization problems are commonly found in science and engineering applications. Nowadays, these optimization problems are getting more and more complex. Traditional optimization methods such as least square approximation, gradient descent and Newton methods belong to single point optimizers and need gradient information. Hence, most of the traditional optimizers are unfit for complex multimodal problems and non-differentiable optimization problems [1]. To cope with complex optimization problems, several swarm intelligence (SI) algorithms such as particle swarm optimization (PSO) [2,3], ant colony optimization (ACO) [4], artificial bee colony (ABC) [5], cuckoo search algorithm (CS) [6,7], grey wolf optimizer (GWO) [8], et al. have been proposed over the past decades. Brezočnik et al. [9] reviewed the recent development ofSI algorithms. As one of the widely used SI algorithms, PSO is easy to implement and converges fast. Since its inception, PSO has attracted great interest from the evolutionary computation (EC) community, and theoretical

researchers [10–13], and real-life applications [14–18] of PSO have been reported over the past two decades. Zhang et al. [19] have summarized the recent advancements in PSO.

Canonical PSO moves the particle by the attractive force from the global best position (*Gbest*) and the particle's own personnel best position (*Pbest*). This mechanism can a obtain high convergence rate, however, the canonical PSO suffers from premature convergenceon complex multimodal problem. In the early stage of optimization, the diversity of canonical PSO is lost quickly and the algorithm's exploration performance weakens, while in the latter stage, the particles crowd in the neighborhood of *Gbest* and slow down the algorithm's convergence rate.

To enhance the exploration of PSO, many improved PSO variants have been proposed. These PSO variants can be classified into four categories: parameter tuning [20–22], neighborhood topology [23–27], learning strategy [28–31] and hybrid versions [32–38].

Among the aforementioned four categories of improved PSO variants, the learning strategy has a significant impact on the performance of PSO. A lot of PSO variants with novel learning strategies have been proposed recently. For example, inspired by learning principles found in human cognitive psychology, Tanweer et al. [1,39,40] proposed self-regulating particle swarm optimization (SRPSO). In SRPSO, a self-regulating inertia weight is employed by the best particle for better exploration, and self-perception of the global search direction is employed by the rest of particles for intelligent exploitation of the solution space. Qin et al. [41] proposed particle swarm optimization with inter-swarm interactive learning strategy (IIL-PSO). Mo et al. [42] transferred an attractive and repulsive interacting mechanism into PSO and proposed attractive and repulsive fully informed particle swarm optimization based on the modified fitness model (ARFIPSOMF). To solve the premature convergence problem in traditional PSO, Dong et al. [43] proposed an opposition based particle swarm optimization with adaptive mutation strategy (AMOPSO). Wang et al. [44] proposed committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems. Ye et al. [45] presented multi-swarm particle swarm optimization with dynamic learning strategy (PSO-DLS). Dong et al. [46] presented a supervised learning and control method to improve particle swarm optimization algorithms. Yang et al. [12] proposed segment-based predominant learning particle swarm optimizer for large-scale optimization through letting several predominant particles guide the learning of a particle. Cao et al. [11] proposed neighbor-based learning particle swarm optimizer (NLPSO) with short-term and long-term memory for dynamic optimization problems. Wang et al. [47] introduced hybrid particle swarm optimization algorithm using adaptive learning strategy (ALPSO).

To discourage premature convergence, Liang et al. [48] proposed comprehensive learning particle swarm optimization (CLPSO). CLPSO uses a novel comprehensive learning strategy whereby all other particle's *Pbest* is used to update a particle's velocity. The comprehensive learning strategy of CLPSO is widely used to obtained high performance on multimodal problems, however, CLPSO cancels the global learning component and converges slowly on unimodal functions. Nasir et al. [49] retained the global learning component for CLPSO to achieve a high convergence rate, nevertheless, all the particles learning from the sole *Gbest* in the social learning component may impair the algorithm's exploration. Hence to enhance exploitation of CLPSO without significantly weakening its exploration, a multi-leader (ML) strategy is combined with CLPSO and the resultant algorithm is called multi-leader comprehensive learning particle swarm optimization (ML-CLPSO). ML-CLPSO constructs a candidate leader set by using a group of top ranked particles and enables each particle to learn from a randomly selected leader from the candidate leader set in the social learning part. The candidate leader set and the leaders of the particles are refreshed dynamically during the optimization to mitigate premature convergence. To activate the stagnated particles, an adaptive mutation (AM) strategy is incorporated into ML-CLPSO and the resultant algorithm is referred to as multi-leader comprehensive learning particle swarm optimization with adaptive mutation (ML-CLPSO-AM). The adaptive mutation employs the distribution information of the candidate leader set to transfer the stagnated particle to a potentially promising area.

To evaluate the impact of multi-leader and adaptive mutation strategies, they were combined with CLPSO both separately and jointly. The latest CEC2017 test suite was adopted to test the performance of the proposed methods. The test results were compared with ten representative PSO variants, six evolutionary algorithms (EAs) and meta-heuristics. The ten selected PSO variants are CLPSO [48], comprehensive learning particle swarm optimization with *Gbest* (CLPSO-G) [49], particle swarm optimization with constriction factor (PSO-cf) [50], fully informed particle swarm (FIPS) [23], fitness-distance-ratio based particle swarm optimization (FDR-PSO) [28], social learning particle swarm optimization (SL-PSO) [51], enhanced leader particle swarm optimization (ELPSO) [52], ensemble particle swarm optimizer (EPSO) [53], genetic learning particle swarm optimization (GL-PSO) [54], and heterogeneous comprehensive learning particle swarm optimization (HCLPSO) [55]. The six EAs and meta-heuristics include adaptive differential evolution with optional external archive (JADE) [56], artificial bee colony (ABC) [5], evolution strategy with covariance matrix adaptation (CMA-ES) [57], grey wolf optimizer (GWO) [8], across neighbor search (ANS) [58] and salp swarm algorithm (SSA) [59]. The test results showed that the ML strategy can enhance exploitation of CLPSO without significantly weakening its exploration, while the AM strategy can activate the stagnated particles to further improve the performance of ML-CLPSO. The ML strategy and AM strategy are compatible, and combining both of them with CLPSO, the resultant ML-CLPSO-AM yields high performance. ML-CLPSO-AM outperformed ten popular PSO variants, six EAs and meta-heuristics in the CEC 2017 test suite. The test results for ELD problems showed that ML-CLPSO-AM is applicable to real-life optimization problems.

The rest of this paper is organized as follows: Section 2 reviews the related work, Section 3 introduces the methodologies, Section 3 reports the experimental results, Section 5 examines the application to ELD problems, and Section 6 concludes the paper.

2. Related Works

2.1. Canonical PSO

In canonical PSO, every candidate solution is regarded as a particle. The particles fly in the search space to find the optimal solution by the attractive force from *Pbest*s and *Gbest*. The velocities and positions of the particles are updated according to Equations (1) and (2) [60].

$$v_{i,d} = \omega \times v_{i,d} + c_1 \times r_1 \times (pbest_{i,d} - x_{i,d}) + c_2 \times r_2 \times (gbest_d - x_{i,d}) \quad (1)$$

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

where $I = 1, 2, \ldots, Ps$ stands for the index of particle. Ps denotes the swarm size. The velocity and position vector of the ith particle are denoted by $\boldsymbol{Vi} = (v_{i,1}, v_{i,2}, \ldots, v_{i,D})$ and $\boldsymbol{Xi} = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$, respectively. D is the dimensionality. Ω is the inertia weight, c_1 and c_2 are two acceleration coefficients. r_1 and r_2 are two uniformly distributed random numbers within the range of [0,1].

The velocity of canonical PSO is composed of three parts, namely the velocity of the previous iteration, the cognition learning part and the social learning part. In canonical PSO, the particle learns from its own *Pbest* to preserve swarm diversity in the cognition learning part and learns from *Gbest* to enhance exploitation in the social learning part. The whole swarm learns from the sole *Gbest* and can obtain high convergence speed, however, this also causes rapid loss of swarm diversity and premature convergence. The canonical PSO only uses the information of the particle's own *Pbest*s and *Gbest* to move the particles, while the information of other elite neighbor particles is not utilized to enhance exploration, therefore, it does not perform well on complex multimodal problems.

2.2. CLPSO

To enhance exploration of PSO, Liang et al. [48] proposed comprehensive learning particle swarm optimization (CLPSO). CLPSO introduced a novel comprehensive learning (CL) strategy where each dimension of one particle can potentially learn from different particles' *Pbest*s. The velocity of the particles in CLPSO is updated according to Equation (3).

$$v_{i,d} = \omega \times v_{i,d} + c \times rand_{i,d} \times (pbest_{fi(d),d} - x_{i,d}) \quad (3)$$

The comprehensive learning list $\boldsymbol{fi}$ = [fi(1), fi(2), . . . , fi(D)] defines which particles' *Pbest*s the ith particle should follow. $rand_{i,d}$∈[0,1] is a random number. The $Pbest_{fi(d),d}$ records each dimension of one particle should learn from which particles' *Pbest*, and the decision is determined by probability *Pc*, which is referred to as CL probability. The *Pc* is generated according to Equation (4) [48].

$$Pc(i) = a + b \times \frac{(\exp(\frac{10(i-1)}{Ps-1}) - 1)}{\exp(10) - 1} \quad (4)$$

where a and b are two constants. a = 0.05, b = 0.45 [48]. A random number is generated for every dimension of the ith particle. If this random number is below $Pc(i)$, the relevant dimension will learn from the better particle's *Pbest* within two tournament selected particles, otherwise, the relevant dimension will learn from its own *Pbest*. If the fitness value of one particle ceases to improve for a predefined number of continuous iterations, CLPSO will refresh the exemplar of that particle. The predefined number is called the refreshing gap. CLPSO enlarges the search range of PSO and exhibits high performance on multimodal problems. Because CLPSO has no independent global learning component, it converges slowly on unimodal problems. To improve the performance of CLPSO, Lynn et al. [55] employed two sub-groups of swarms to focus solely on either exploration or exploitation and proposed heterogeneous comprehensive learning particle swarm optimization (HCLPSO). HCLPSO employs CLPSO and CLPSO-G for the exploration sub-group and the exploitation sub-group, respectively. The velocity of CLPSO-G is updated according to Equation (5).

$$v_{i,d} = \omega \times v_{i,d} + c_1 \times rand1_{i,d} \times (pbest_{fi(d),d} - x_{i,d}) + c_2 \times rand2_{i,d} \times (gbest_d - x_{i,d}) \quad (5)$$

where $rand1_{i,d}$, $rand2_{i,d}$∈[0, 1] are two random numbers. CLPSO-G added a global learning component to the original CLPSO, so it converges relatively faster. The acceleration coefficients are linearly adjusted in CLPSO-G to obtain better performance. Lynn et al. [53] also adopted the CLPSO-G component in ensemble particle swarm optimizer (EPSO).

2.3. The Social Learning Leader

In the social learning part, the particles learn from other particles' *Pbests*. In this study, the social learning exemplar is referred to as the leader. The selection and utilization of the leader is very important to the performance of PSO. Chen et al. [61] transferred the aging mechanism to PSO and proposed particle swarm optimization with an aging leader and challengers (ALC-PSO). In ALC-PSO, the leader is assigned with limited life of iterations, and the life of the leader is adjusted dynamically during the evolution process according to the evolutionary state of the particles. Once the leader gets too old, a challenger is generated to challenge the old leader. ALC-PSO can mitigate premature convergence to some extent, however, the whole swarm sharing the sole leader may cause the rapid loss of diversity. Zhang et al. [56] presented a novel DE/current-to-$p$best/1 mutation strategy. The DE/current-to-$p$best/1 employs top 100p% individuals in the current swarm to generate a trial vector of differential evolution (DE). The test results prove that the DE/current-to-$p$best/1 performs better than DE/current-to-best/1 and thus it is widely used in many DE variants. Borrowing from DE/current-to-$p$best/1, the idea of employing the top 100p% particles' *Pbest* instead of *Gbest* to guide the motion of particles may yield promising improvements. Yu et al. [62] proposed multiple learning PSO with space transformation perturbation (MLPSO-STP). In MLPSO-STP the particles learn from the top 100p% particles and mean positions of the current swarm. The test results prove MLPSO-STP performs better than classical PSO variants on selected benchmark functions. Tanweer et al. [1] divided the whole swarm into mentors, independents and mentees according to the particle's fitness value and the Euclidian distance between the particle and *Gbest*. The high-quality mentors guide low quality mentees and the manner of learning different dimensions from different particle is adopted.

2.4. Mutation

To enhance swarm diversity and ease premature convergence, the mutation factor of genetic algorithms (GA) [63] can be transferred into the PSO algorithm. To improve the performance of PSO on multimodal problems, Higashi et al. [64] combined particle swarm optimization with Gaussian mutation. The test results showed that the proposed PSO with Gaussian mutation performed better than both GA and PSO on the tested problems. Wei et al. [65] employed Gaussian mutation to increase the diversity of PSO and decrease the risk of plunging into local optima. To mitigate premature convergence, Jordehi [52] employed five succeeding stages of mutations to the swarm leader per iteration, and proposed enhanced leader particle swarm optimization (ELPSO). The Gaussian mutation, Cauchy mutation, dimension by dimension opposition-based mutation, all dimension opposition-based mutation and DE-based mutation are applied to the swarm leader to improve its fitness value. The test results proved that ELPSO obtains high performance in terms of accuracy, scalability and convergence rate. The above mentioned publications indicate that combining mutation with PSO can enhance diversity and improve the performance of PSO on multimodal problem. For more PSO variants with mutation operators, please refer to [41,66–69].

3. Methods

3.1. Multi-Leader Strategy

Section 2.3 analyzed the importance of the social learning leader. In the widely used global version PSO, *Gbest* is the sole leader. However, if the whole swarm learns from the sole leader, this may cause the swarm to quickly lose diversity and suffer from premature convergence in multimodal problems. On the other hand, CLPSO and FIPS have no independent component learning from *Gbest*, so they converge too slowly on unimodal problems. To enhance the exploitation of CLPSO without weakening its exploration, the ML strategy is combined with CLPSO; the resultant PSO algorithm is referred to as ML-CLPSO. In the ML strategy, a group of top ranked particles constitutes the candidate leader set and the candidate leader set is updated dynamically. In the initialization, each particle selects a leader randomly from the candidate leader set. The velocity of ML-CLPSO is updated according to Equation (6).

$$v_{i,d} = \omega \times v_{i,d} + c_1 \times r_1 \times (pbest_{fi(d),d} - x_{i,d}) + c_2 \times r_2 \times (pbest_{Leader(i),d} - x_{i,d}) \quad (6)$$

where *Leader*(*i*) is an index of particle, it records the *i*th particle, which should learn from that particle's *Pbest* in the social learning part. The learning behavior of ML-CLPSO is divided into two parts, the comprehensive learning part and the social learning part. The particles learn from CL exemplars in the comprehensive learning part to enhance exploration and learn from high quality leaders in the social learning part to enhance exploitation. Each particle learns from a different leader and the leader of the particle is updated dynamically during the optimization process. When the particle's fitness value has not been improved within a refreshing gap, the particle's leader will be updated together with its comprehensive learning exemplar. To minimize problems, the improvement in the fitness value means one particle finds a solution with a lower fitness value than its *Pbest*. When updating the leader of one particle, the particles are sorted by fitness value and a group of top ranked particles constructs the new candidate leader set. The particle's new leader is selected from the new candidate leader set by random selection. Rantanweera et al. [70] pointed out "in population-based optimization methods, it is desirable to encourage the individuals to wander through the entire search space, without clustering around local optima, during the early stages of the optimization. On the other hand, during the latter stages, it is very important to enhance convergence toward the global optima, to find the optimum solution efficiently." Hence, to enhance exploration in the early stage and enhance exploitation in the latter stage, the linearly adjusting inertia weight and acceleration coefficients are adopted. ω and c_1 are initialized with high value and decreases linearly, while c_2 is started with low value and increases linearly. The pseudo code of ML-CLPSO is presented in Algorithm 1.

Algorithm 1. The pseudo code of ML-CLPSO. *Note: the line with ***1**, ***2**, ***3** are alternative steps for the AM strategy, which will be introduced in Section 3.2. They are skipped in ML-CLPSO.

```
1   Initialization
2   While ct < Max_FES                                  %  Main loop (Line 2-31)
3     Linearly adjusting ω, c1, c2 according to Table 1  % Adjust control parameters
4     For i = 1:Ps                                      %  Evolve every particle (Line 4-30)
5       Update velocity of the ith particle according to Equation (6)
6       Update position of the ith particle according to Equation (2)
7       Evaluate the fitness value of the ith particle
8       ct = ct + 1
12      If fit(X(i)) < Pbestval(i)                      %  Update Pbest by greedy selection (Line 13,14)
13        Pbestval(i) = fit(X(i))
14        Pbest(i) = X(i)
15        Stag1(i) = 0                                  %  Reset Stag1(i)
16    *1  Stag2(i) = 0                                  %  Reset Stag2(i)
17      Else
18        Stag1(i) = Stag1(i) + 1                       %  Update Stage1(i)
19    *2  Stag2(i) = Stag2(i) + 1;                      %  Update Stage2(i)
20        If Stag1(i) > Stag1m                          %  Update Leader and CL exemplar (Line 21-26)
21          Stag1(i) = 0                                %  Reset Stag1(i)
22          [Sort_val, Sort_id] = sort(Pbestval)        %  Update candidate leader set (Line 22-23)
23          CLS = Sort_id(1:NL)
25          Leader(i) = CLS(ceil(r1*NL))                %  Update Leader
26          Update the comprehensive learning exemplar  %  Update CL exemplar
27    *3 Execute adaptive mutation module according to Algorithm 2   %  Adaptive mutation
28  End
29      End
30    End
31  End
```

In the initialization, the swarm size, search range, position, velocity, *Pbest* and *Gbest* of particles are initialized. The inertia weight and acceleration coefficients are also assigned with initial values. The ω, c_1 and c_2 are linearly adjusted according to Table 1 in Section 4.2. The *ct* and *Max_FES* denote the count of currently consumed function evolutions (*FEs*) and the maximum *FEs*. *Ps* is the swarm size. The $\boldsymbol{X}(i)$, $\boldsymbol{Pbest}(i)$ and *Pbestval*(*i*) stand for the *i*th particle's position, *Pbest* and fitness value, respectively. *fit*(·) is the function for calculating the fitness value. *CLS* is an integer array for recording the indexes of candidate leader particles. $r_1 \in [0,1]$ is a random number. *sort*(·) and *ceil*(·) are two Matlab functions. The *sort*(·) function sorts the input array in ascending order and the *ceil*(·) function returns the input variable's nearest integer in the direction of positive infinity.*Stag*1 and *Stag*2 are two independent integer arrays to count the continuous number of iterations without improving the particles' fitness values.*Stag*1*m* is the refreshing gap for updating the CL exemplar and leader. *Stag*2*m* is the mutation gap, which will be introduced in Section 3.2. N_L is the size of the candidate leader set. The alternative steps are only executed when the AM strategy is employed. In ML-CLPSO, the leader of one particle is updated together with its comprehensive learning exemplar. When one particle's leader needs updating, ML-CLPSO will sort the particles by fitness value to refresh the candidate leader set, then select the particle's new leader randomly from the candidate leader set.

3.2. Adaptive Mutation Strategy

Because PSO adopts greedy selection to update *Pbest*s, in complex multimodal problems some particles may consume many continuous iterations (for example, 30 iterations) without improving their fitness value. These particles are called stagnated particles. The stagnated particles waste numerous

Fes without moving their *Pbest*s towards a promising area, thereby, the performance of the PSO algorithm is degenerated. To activate the stagnated particles, the AM strategy is introduced. If one particle is detected in stagnation, the AM strategy will generate a new *Pbest* for the stagnated particle. Unlike traditional mutation strategy, the AM strategy accepts the new *Pbest* without considering its fitness value. The mean position of the candidate leader set and the normalized mean velocity are employed for the AM strategy. After mutation, the stagnated particle has a chance to search for a new promising area. The new *Pbest* of the stagnated particle is generated according to Equations (7)–(9).

$$pbest_{i,d} = \overline{Leader_d} + \eta \times N(0,1) \times v_{norm} \tag{7}$$

$$v_{norm} = \frac{1}{Ps}\sum_{i=1}^{Ps}\sqrt{\frac{1}{D}\sum_{j=1}^{j=D}(v_{i,d})^2} \tag{8}$$

$$\overline{Leader_d} = \frac{1}{N_L}\sum_{i=1}^{i=N_L} pbest_{Leader(i),d} \tag{9}$$

where $\overline{Leader_d}$ denotes the mean position of the candidate leader set, which may locate the approximate promising area. η is a scale factor to adjust the amplitude of mutation, and it will be discussed in Section 4.4. $N(0,1)$ stands for a normal distributed random number with the standard deviation of 1. v_{norm} denotes the normalized mean velocity in the current swarm. The velocity information of the current swarm is employed to adjust the amplitude of the AM [71]. The AM strategy is described in Algorithm 2.

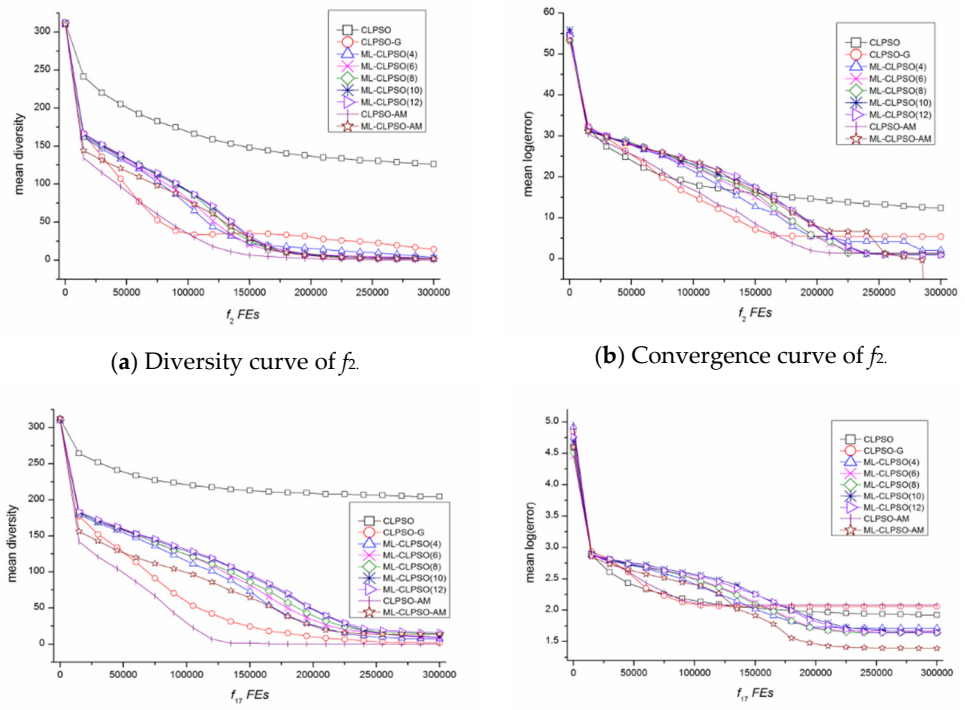
Algorithm 2. The adaptive mutation module

```
If Stag2(i) > Stag2m
Stag2(i) = 0
Pam=1 − ct/Max_FES
    If r2 < Pam
        Generate a new Pbest for the ith particle according to Equations (7)–(9)
    Pbestval(i) = fit(Pbest(i));
    ct = ct+1
  End
End
```

The integer *Stag*2(*i*) introduced in Algorithm 1 is used to detect the stagnated particles. *Stag*2*m* is the mutation gap. Commonly, the *Stag*2*m* is much bigger than *Stag*1*m* to avoid impairing the convergence of algorithm. Once *Stag*2(*i*) is bigger than the mutation gap *Stag*2*m*, the AM may be triggered to generate a new *Pbest* for the *i*th particle. *Pam* is a linearly decreasing probability used to determine whether to execute AM for the *i*th particle. $r_2 \in [0,1]$ is a uniformly distributed random number. If r_2 is smaller than *Pam*, the AM is executed and a new *Pbest* is generated. The stagnated particle accepts the new *Pbest* without considering its fitness value. The AM can transfer the stagnated particle to a potentially promising area by employing the distribution information of the candidate leader set. Conducting AM frequently may impair the convergence of PSO, hence a linearly decreasing probability is employed to control AM. In the early stage, the AM is executed with a higher probability to enhance exploration, while in the latter stage, the AM is carried out with a lower probability to avoid impairing the convergence of the algorithm. The amplitude of mutation is adjusted by the normalized mean velocity. Algorithm 2 is conducted in the alternative step *3 of Algorithm 1. ML-CLPSO-AM executes all the alternative steps in Algorithm 1. The characteristics of ML strategy and AM strategy will be tested in the following section.

3.3. Characteristics of ML and AM Strategies

To investigate the characteristics of the ML and AM strategies, two benchmark functions of the CEC2017 test suite were employed (unimodal function f_2 and multimodal function f_{15}). ML-CLPSOs with different leader size were tested and their diversity curve and convergence curve are given in Figure 1. In this study, the diversity was evaluated by the mean distance between the particles' position and the centroid of the current swarm. The CLPSO, CLPSO-G were also included in this comparison test. The parameter settings of PSO variants were in accordance with Table 1. The leader size of ML-CLPSO was set to 4, 6, 8, 10 and 12. The leader size of ML-CLPSO-AM was set to 10. In Figure 1 the ML-CLPSO(*) denotes ML-CLPSO with the leader size in the bracket, for example ML-CLPSO(4) denotes ML-CLPSO with a leader size of 4.



(**a**) Diversity curve of f_2. (**b**) Convergence curve of f_2.

(**c**) Diversity curve of f_{17}. (**d**) Convergence curve of f_{17}.

Figure 1. The diversity curve and convergence curve of ML-CLPSOs.

Figure 1a shows that on unimodal function f_2, CLPSO keeps the highest diversity, CLPSO-G keeps relatively lower diversity, and the diversity of ML-CLPSOs is higher than CLPSO-G but lower than CLPSO. ML-CLPSO has bigger leader size and keeps slightly higher diversity. The test results show that increasing the leader size is helpful to enhance diversity. The diversity of CLPSO-AM is approximately equal to that of CLPSO-G in the early stage, while in the latter stage, the diversity of CLPSO-AM is lower than CLPSO-G. This is because the AM strategy encourages the motion of the stagnated particles by transferring them to explore the possible promising area while in CLPSO-G, if one particle's fitness value ceases improving, its *Pbest* may stay in the same position unless a better position is found by the particle, hence the diversity of CLPSO-G is higher than CLPSO-AM in the latter stage. The diversity curves of ML-CLPSO(10) and ML-CLPSO-AM are almost overlapping, that is, for unimodal problems, the effect of the AM strategy on the diversity of ML-CLPSO-AM is not significant. Figure 1b shows that CLPSO-G converges quickly because it contains a global learning component, while CLPSO converges a bit more slowly. ML-CLPSOs converge more slowly than CLPSO in the early stage, however, they surpass CLPSO in the middle stage. In the early stage, ML-CLPSO with its smaller leader size converges faster. In the latter stage, ML-CLPSO(6) and ML-CLPSO(8) surpass ML-CLPSO(4). Figure 1b shows that employing moderate leader size leads to

better performance on unimodalf_2. In the end stage, the convergence rate of ML-CLPSO-AM increases significantly and outperforms ML-CLPSO(10) because the AM strategy activates the stagnated particles to explore the potentially promising area and hence, the performance of ML-CLPSO-AM is improved. Because the ML strategy dynamically allocates suitable leaders for the particles, ML-CLPSOs can make good use of the information from the high-quality particles to achieve better performance than CLPSO-G. By adopting the AM strategy, CLPSO-AM and ML-CLPSO-AM yield better performance than the corresponding CLPSO-G and ML-CLPSO(10), respectively.

Figure 1c shows that on multimodal function f_{17}, similar tounimodal function f_2, CLPSO preserves the highest diversity. The diversity of ML-CLPSOs is higher than CLPSO-G but lower than CLPSO. By increasing the leader size, ML-CLPSO is able to preserve more diversity. Compared with unimodal functions, more significant diversity differences can be observed for multimodal functions. Since the AM strategy transfers the stagnated particles to possible promising area, the diversity of CLPSO-AM and ML-CLPSO-AM are relatively lower than the corresponding CLPSO-G and ML-CLPSO(10). Figure 1d shows that in the early stage of optimization, CLPSO and CLPSO-G converge quickly, however, they are surpassed by ML-CLPSOs. ML-CLPSOs achieve almost the same mean error in the end. The convergence curves of CLPSO-AM and CLPSO-G are almost overlapping. Compared with ML-CLPSO(10), ML-CLPSO-AM converges faster and generates a lower mean error. The test results suggest that the solution accuracy is not only dependent on leader size, but also has something to do with the characteristics of the tested problem.

This group of tests show that the ML strategy can enhance the exploitation of CLPSO without significantly impairing its exploration. By using an ML strategy, the ML-CLPSOs yield lower mean error than both CLPSO and CLPSO-G. The leader size has a significant impact on the diversity and performance of ML-CLPSO; increasing the leader size preserves more diversity while adopting a proper leader size can obtains a good trade-off between exploration and exploitation. Because ML-CLPSO employs linear decreasing ω, c_1 and linear increasing c_2, in the early stage, the comprehensive learning part plays the key role in enhancing exploration while in the latter stage, the social learning part plays a leading role in speeding up the convergence rate. Hence, ML-CLPSO converges relatively slowly in the early stage but it converges relatively fast in the latter stage. Regarding the AM strategy, CLPSO-AM performs better than CLPSO-G on f_2 while it performs almost the same as CLPSO-G on f_{17}. The AM strategy can activate the stagnated particle to search potential promising areas. With both ML and AM strategies, ML-CLPSO-AM outperforms ML-CLPSO(10) on both f_2 and f_{17}. For more tests of ML and AM strategies refer to Section 4.3.1.

4. Experimental Works

4.1. Test Problems

To test the performance of the proposed ML-CLPSO-AM, the latest CEC2017 test suite on single objective real-parameter numerical optimization [72] was employed. The test suite is referred to as CEC2017 test suite hereafter. The CEC2017 test suite contains thirty benchmark functions, including three unimodal functions, seven simple multimodal functions, ten hybrid functions and ten composition functions. The exact equations of CEC2017 test suite are not allowed to be used.

According to the instruction of CEC2017 test suite, the maximum *FEs Max_FES* = 10,000**D* and the search range is [−100, 100]. In this study, the number of dimensions was set as *D* = 30. Each algorithm was run for 51 runs and the mean error, average rank and Wilcoxon signed rank test [73–75] were employed to compare the performance of the involved algorithms.

4.2. Parameters Settings

To test the effectiveness of the proposed ML and AM strategy, three PSO variants were proposed in this study, namely, ML-CLPSO, CLPSO-AM, and ML-CLPSO-AM. ML-CLPSO is a combination of ML strategy and CLPSO; CLPSO-AM is developed by transferring AM strategy into CLPSO-G;

ML-CLPSO-AM is merges ML and AM strategies simultaneously into CLPSO. Ten PSO variants, six EAs and meta-heuristics were tested and compared with the proposed methods, including five state-of-the-art PSO variants, five latest PSO algorithms, two classic EAs and three recently reported meta-heuristics. CLPSO-G inherits the comprehensive learning strategy of CLPSO and added a component of learning from *Gbest* to speed up the convergence [55]. Other involved peer PSO algorithms have been introduced as mentioned in Sections 1 and 2. The adaptive differential evolution with optional external archive (JADE) [56] adopts a new mutation strategy "DE/current-to-*p*best" with an optional external archive and updating control parameters in an adaptive manner. Artificial bee colony (ABC) [5] simulates the intelligent foraging behavior of a honeybee swarm. Evolution strategy with covariance matrix adaptation (CMA-ES) [57] solves the rotation problem by adopting covariance matrix adaptation strategy. The grey wolf optimizer (GWO) [8] simulates the leadership hierarchy and hunting mechanism of grey wolves in nature. Across neighbor search (ANS) [58] utilizes a group of individuals to search the solution space cooperatively. A set of superior solutions found by individuals is maintained and refreshed dynamically. The Salp swarm algorithm (SSA) [59] simulates the swarming behavior of salps when navigating and foraging in oceans for solving optimization problems.

All the algorithms employ the relevant author suggested parameters configuration. The parameter settings of the PSO algorithms are given in Table 1.

Table 1. Parameter settings of PSO algorithms.

No.	Algorithm	Year	Parameter Settings
1	ML-CLPSO	/	Ps = 40, ω = 0.9–0.4, c_1 = 2.5–0.5, c_2 = 0.5–2.5, $stag1m$ = 6, N_L = 6
2	CLPSO-AM	/	Ps = 40, ω = 0.9–0.4, c_1 = 2.5–0.5, c_2 = 0.5–2.5, $stag1m$ = 6, $stag2m$ = 40, η = 0.6
3	ML-CLPSO-AM	/	Ps = 40, ω = 0.9–0.4, c_1 = 2.5–0.5, c_2 = 0.5–2.5, $stag1m$ = 6, $stag2m$ = 40 , N_L = 10, η = 0.6
4	PSO-cf [50]	2002	Ps = 40, χ = 0.7298, $c_1 = c_2$ = 2.05
5	FIPS [23]	2004	Ps = 40, χ = 0.7298, φ = 4.1
6	FDR-PSO [28]	2003	Ps = 40, φ_1 = 1, φ_2 = 1, φ_3 = 2, ω = 0.9–0.4
7	CLPSO [48]	2006	Ps = 40, ω = 0.9-0.4, c = 1.49445,
8	CLPSO-G [55]	/	Ps = 40, ω = 0.99-0.2, c_1 = 2.5-0.5, c_2 = 0.5–2.5
9	SL-PSO [51]	2015	M = 100, α = 0.5, β = 0.01, $Ps = M + ceil(D/10)$
10	HCLPSO [55]	2015	Ps = 40, ω = 0.99–0.2, c_1 = 2.5–0.5, c_2 = 0.5-2.5, c = 3–1.5, g_1 = 15, g_2 = 25
11	EL-PSO [52]	2015	Ps = 1500, ω = 0.9–0.4, $c_1 = c_2$ = 2, h = 1, s = 2, F = 1.2
12	GL-PSO [54]	2016	Ps = 50, ω = 0.7298, c = 1.49618, pm = 0.01, s_g = 7
13	EPSO [53]	2017	Reference [53]

4.3. Comparison Tests

4.3.1. Comparison Test of Different Strategies

To show the effects of ML and AM strategies on CLPSO, a comparison test of CLPSO with the ML, AM or both strategies was conducted, and the test results are given in Table 2. The best results are highlighted in bold.

The test results in Table 2 show that on three unimodal functions (f_1–f_3), CLPSO, ML-CLPSO-AM and CLPSO-AM yield the best solutions on f_1, f_2 and f_3, respectively. Because of the ML strategy, ML-CLPSO obtains a higher performance. It achieves a lower mean error than CLPSO and CLPSO-G on two (f_2, f_3) and three (f_1–f_3) functions, respectively. CLPSO-AM performs better than CLPSO-G on f_2 and f_3, performs worse on f_1. Compared with ML-CLPSO, ML-CLPSO-AM performs better on f_2, and performs worse on f_1 and f_3. The Wilcoxon signed-rank test results show that ML-CLPSO-AM performs significantly better than CLPSO, CLPSO-G, ML-CLPSO, CLPSO-AM on 2, 1, 1 and 2 functions, respectively. The ML strategy improves the performance of CLPSO significantly, while the AM strategy doesn't exhibit obvious advantage.

On seven simple multimodal functions (f_4–f_{10}), ML-CLPSO-AM yields the best solutions on six functions (f_5–f_{10}). CLPSO-AM achieves the best solution on f_4. Compared with CLPSO and CLPSO-G, ML-CLPSO obtains lower mean error on four (f_5, f_7–f_9) and five functions (f_5–f_9), respectively. Compared with CLPSO-G, CLPSO-AM yields lower mean error on four functions

(f_4, f_6, f_7, f_9). The Wilcoxon signed rank test results show ML-CLPSO-AM outperforms CLPSO, CLPSO-G, ML-CLPSO on all seven functions. Compared with CLPSO-AM, ML-CLPSO-AM wins on six functions. It only performs worse than CLPSO-AM on f_4. The test results reveal that the ML strategy can improve the performance of CLPSO, and the AM strategy can further improve the performance of ML-CLPSO.

On ten hybrid functions (f_{11}–f_{20}), ML-CLPSO-AM achieves the best solutions on five functions (f_{11}, f_{16}, f_{17}, f_{19}, f_{20}), CLPSO generates the best solutions on f_{13} and f_{15}, ML-CLPSO achieves the best solutions on f_{12} and f_{14}, and CLPSO-AM generates the best solution on f_{18}. Compared with CLPSO and CLPSO-G, ML-CLPSO yields lower mean error on five (f_{11}, f_{12}, f_{14}, f_{16}, f_{17}) and ten functions (f_{11}–f_{20}), respectively. Compared with CLPSO-G, CLPSO-AM yields lower mean error on four functions (f_{13}, f_{14}, f_{16}, f_{18}). According to the Wilcoxon signed rank test results, ML-CLPSO-AM performs significantly better than ML-CLPSO and CLPSO-AM on six (f_{11}, f_{15}–f_{18}, f_{20}) and five functions (f_{11}, f_{15}–f_{17}, f_{20}), respectively. ML-CLPSO-AM and CLPSO-AM obtaineda similar performance on f_{19}. ML-CLPSO performs almost the same as CLPSO while it performs much better than CLPSO-G. For the AM strategy, the performance of CLPSO-AM is similar to CLPSO-G while ML-CLPSO-AM performs slightly better than ML-CLPSO and CLPSO-AM.

Table 2. Test results of CLPSO with different strategies.

Function	CLPSO	CLPSO-G	ML-CLPSO	CLPSO-AM	ML-CLPSO-AM
f_1	**<1.443E+01**	<3.111E+03	<2.394E+03	>4.798E+03	4.348E+03
f	>2.187E+12	>2.502E+05	>1.248E+01	>2.208E+01	**0.000E+00**
f_3	>1.920E+04	<6.147E-04	<4.880E-06	**<1.707E-09**	3.671E-02
f_4	>7.262E+01	>7.312E+01	>8.381E+01	**<6.104E+01**	6.538E+01
f_5	>4.216E+01	>4.107E+01	>2.018E+01	>4.479E+01	**1.437E+01**
f_6	>7.089E-08	>6.614E-04	>4.093E-07	>7.047E-05	**2.190E-08**
f_7	>9.413E+01	>9.434E+01	>7.735E+01	>9.183E+01	**4.469E+01**
f_8	>4.962E+01	>4.712E+01	>1.986E+01	>4.720E+01	**1.377E+01**
f_9	>5.864E+01	>5.802E+00	>2.534E-02	>5.033E+00	**2.175E-02**
f_{10}	>2.214E+03	>2.528E+03	>2.668E+03	>2.567E+03	**2.312E+02**
f_{11}	>4.786E+01	>5.739E+01	>1.625E+01	>6.958E+01	**9.753E+00**
f_{12}	>2.926E+05	<2.341E+04	**<1.603E+04**	<2.473E+04	3.138E+04
f_{13}	**<2.802E+02**	<1.045E+04	<8.177E+03	<9.862E+03	1.755E+04
f_{14}	>2.046E+04	<5.145E+03	**<2.184E+03**	<2.313E+03	6.311E+03
f_{15}	**<1.086E+02**	>4.153E+03	>2.172E+03	>7.085E+03	2.092E+03
f_{16}	>4.137E+02	>6.832E+02	>2.042E+02	>4.018E+02	**2.862E+01**
f_{17}	>8.360E+01	>1.139E+02	>4.646E+01	>1.202E+02	**2.472E+01**
f_{18}	>1.001E+05	>1.146E+05	>1.142E+05	**<7.270E+04**	1.087E+05
f_{19}	<6.657E+01	<6.650E+03	<4.560E+03	=1.122E+04	**1.096E+04**
f_{20}	>1.189E+02	>1.478E+02	>1.223E+02	>1.534E+02	**7.202E+01**
f_{21}	>2.454E+02	>2.456E+02	>2.200E+02	>2.521E+02	**2.142E+02**
f_{22}	=1.034E+02	=1.006E+02	**=1.000E+02**	>2.615E+02	**1.000E+02**
f_{23}	>3.955E+02	>4.059E+02	>3.655E+02	>3.944E+02	**3.522E+02**
f_{24}	>4.786E+02	>4.719E+02	>4.370E+02	>4.644E+02	**4.279E+02**
f_{25}	**=3.869E+02**	=3.874E+02	=3.870E+02	=3.874E+02	**3.869E+02**
f_{26}	**<3.037E+02**	>1.056E+03	<7.546E+02	>1.143E+03	7.926E+02
f_{27}	>5.105E+02	>5.171E+02	=5.041E+02	=5.052E+02	**5.039E+02**
f_{28}	>4.129E+02	<3.444E+02	**<3.319E+02**	<3.505E+02	3.621E+02
f_{29}	>5.151E+02	>5.188E+02	>4.404E+02	>4.936E+02	**4.213E+02**
f_{30}	=4.697E+03	<4.511E+03	**<3.420E+03**	<4.182E+03	4.756E+03
w/t/l	22/3/5	20/2/8	18/3/9	19/3/8	/
best	5	0	5	3	19

Note: 1.443E+01 standards for 1.443×10^1.

Of ten complex composition functions (f_{21}–f_{30}), ML-CLPSO-AM achieves the best solution on seven functions (f_{21}–f_{25}, f_{27}, f_{29}), CLPSO and ML-CLPSO yield the best solution on two (f_{25}, f_{26}) and three functions (f_{22}, f_{28}, f_{30}), respectively. Compared with CLPSO and CLPSO-G, ML-CLPSO generates

lower mean error on eight (f_{21}–f_{24}, f_{27}–f_{30}) and ten functions (f_{21}–f_{30}), respectively. Compared with CLPSO-G, CLPSO-AM achieves lower mean error on five functions (f_{23}, f_{24}, f_{27}, f_{29}, f_{30}). The Wilcoxon signed rank test shows that the performance of ML-CLPSO-AM is significantly better, the same and significantly worse than ML-CLPSO on four (f_{21}, f_{23}, f_{24}, f_{29}), three (f_{22}, f_{25}, f_{27}) and three functions (f_{26}, f_{28}, f_{30}), respectively. Compared with CLPSO-AM, ML-CLPSO-AM performed significantly better, the same, and significantly worse on six (f_{21}–f_{24}, f_{26}, f_{29}), two (f_{25}, f_{27}) and two functions (f_{28}, f_{30}). The test results show that the ML strategy improves the performance of CLPSO significantly, and after combining the AM strategy with ML-CLPSO, the resultant ML-CLPSO-AM performs better than ML-CLPSO and CLPSO-AM.

On all thirty functions, ML-CLPSO-AM, CLPSO, ML-CLPSO and CLPSO-AM achieve the best solutions on nineteen, five, five and three functions, respectively. ML-CLPSO achieves lower mean errors than CLPSO and CLPSO-G on nineteen and twenty-eight functions, respectively. Compared with CLPSO-G, CLPSO-AM achieves lower mean errors on fifteen functions. Compared with CLPSO-AM, ML-CLPSO achieves lower mean errors on twenty-six functions. ML-CLPSO-AM performs significantly better than CLPSO, CLPSO-G, ML-CLPSO and CLPSO-AM on twenty-two functions, twenty functions, eighteen functions and nineteen functions, respectively.

The test results indicate that with the ML strategy, ML-CLPSO performs much better than CLPSO and CLPSO-G, although if incorporating the AM strategy into CLPSO-G, the resultant CLPSO-AM does not show significant advantage over CLPSO-G. The reason is that CLPSO-G cannot preserve sufficient diversity to mitigate premature convergence. Combining ML and AM strategies with CLPSO simultaneously, the resultant ML-CLPSO-AM performs better than both ML-CLPSO and CLPSO-AM. The ML strategy makes good use of high-quality particles to enhance exploitation of CLPSO without impairing its exploration significantly. The AM strategy can further improve the performance of ML-CLPSO by transferring the stagnated particles to potentially promising area. The supplementary file show that the computational complexity of ML-CLPSO-AM is almost the same with CLPSO, while the computing time of ML-CLPSO-AM is shorter than CLPSO (Figure S1).

4.3.2. Comparison Test with PSO Variants

In this test, ML-CLPSO-AM was compared with three state-of-the-art PSO algorithms, namely PSO-cf, FIPS, FDR-PSO, and five recently reported PSO algorithms, namely, SL-PSO, EL-PSO, EPSO, GL-PSO and HCLPSO. The best results are highlighted in bold. The test results in Table 3 show that ML-CLPSO-AM, FIPS, FDR-PSO, SL-PSO, ELPSO, EPSO and HCLPSO yield the best solutions on 16, 2, 1, 1, 2, 3, 7 functions, respectively. PSO-cf and GL-PSO have not yielded the best solution. The Wilcoxon signed rank test results in Table 3 show that compared with PSO-cf, FIPS, FDR-PSO, SL-PSO, EL-PSO, EPSO, GL-PSO and HCLPSO, ML-CLPSO-AM yields a better solution on 29, 23, 21, 23, 22, 16, 19, 18 functions, respectively. The overall performance of ML-CLPSO-AM is superior to any of the peer PSO algorithms. Details of the comparison test results are analyzed as follows.

Table 3. Comparison test results with PSO algorithms.

Function	PSO-cf	FIPS	FDR-PSO	SL-PSO	EL-PSO	EPSO	GL-PSO	HCLPSO	ML-CLPSO-AM
f_1	>5.819E+03	<3.513E+03	>4.773E+03	>5.106E+03	<2.404E+03	<2.683E+03	<2.933E+03	**<7.772E+01**	4.348E+03
f_2	>5.320E+20	>1.108E+15	>1.259E+10	>1.523E+11	>4.634E+14	>2.208E+01	>1.272E+04	>2.145E+06	**0.000E+00**
f_3	<1.717E-08	>4.089E+03	>1.546E-08	>7.047E+03	<3.258E-02	**<8.345E-13**	>1.815E+01	<1.575E-03	3.671E-02
f_4	>1.447E+02	>1.247E+02	<2.850E+01	>7.736E+01	>7.800E+01	**<6.907E+00**	<2.525E+01	>6.848E+01	6.538E+01
f_5	>8.075E+01	>1.375E+02	>5.657E+01	>1.842E+01	>1.070E+02	>4.249E+01	>5.189E+01	>4.277E+01	**1.437E+01**
f_6	>6.666E+00	>2.600E-08	>7.507E-02	>1.682E-06	>2.064E+01	>3.702E-05	>3.085E-04	**<3.956E-13**	2.190E-08
f_7	>1.277E+02	>1.915E+02	>9.450E+01	>1.878E+02	>1.335E+02	>8.188E+01	>8.448E+01	>8.523E+01	**4.469E+01**
f_8	>9.030E+01	>1.361E+02	>5.715E+01	>1.738E+01	>8.856E+01	>4.729E+01	>5.023E+01	>4.372E+01	**1.377E+01**
f_9	>6.503E+02	**<0.000E+00**	>1.907E+01	>8.401E-02	>9.486E+02	>2.398E+01	>7.633E+00	>2.074E+01	2.175E-02
f_{10}	>3.638E+03	>6.305E+03	>3.082E+03	>1.006E+03	>3.423E+03	>2.153E+03	>2.880E+03	>2.070E+03	**2.312E+02**
f_{11}	>1.371E+02	>7.088E+01	>1.039E+02	>2.806E+01	>6.677E+01	>7.407E+01	>6.895E+01	>5.509E+01	**9.753E+00**
f_{12}	>1.554E+06	>5.291E+05	<2.136E+04	>4.592E+04	>7.220E+04	**<2.134E+04**	<2.432E+04	>3.588E+04	3.138E+04
f_{13}	>1.060E+06	<1.330E+04	<1.679E+04	<1.573E+04	<2.707E+03	<3.078E+03	<9.857E+03	**<7.374E+02**	1.755E+04
f_{14}	>4.570E+04	<6.170E+03	<4.440E+03	>1.743E+04	**<9.221E+01**	<2.701E+03	<1.716E+03	<3.442E+03	6.311E+03
f_{15}	>1.457E+04	>1.651E+04	>7.190E+03	<1.890E+03	<1.388E+03	<4.325E+02	>3.388E+03	**<3.305E+02**	2.092E+03
f_{16}	>9.090E+02	>8.411E+02	>6.923E+02	>1.537E+02	>8.995E+02	>6.223E+02	>8.225E+02	>4.413E+02	**2.862E+01**

Table 3. *Cont.*

Function	PSO-cf	FIPS	FDR-PSO	SL-PSO	EL-PSO	EPSO	GL-PSO	HCLPSO	ML-CLPSO-AM
f_{17}	>4.503E+02	>1.626E+02	>1.961E+02	>9.080E+01	>2.324E+02	>1.678E+02	>1.876E+02	>9.811E+01	**2.472E+01**
f_{18}	>1.328E+05	>3.075E+05	<8.228E+04	<1.024E+05	**<2.274E+04**	<6.494E+04	<5.444E+04	<9.355E+04	1.087E+05
f_{19}	>1.302E+04	<5.781E+03	<8.812E+03	<2.226E+03	<3.712E+03	<3.858E+02	<7.901E+03	**<1.580E+02**	1.096E+04
f_{20}	>4.708E+02	>1.927E+02	>2.182E+02	>1.326E+02	>3.312E+02	>1.998E+02	>2.756E+02	>1.267E+02	**7.202E+01**
f_{21}	>2.896E+02	>3.381E+02	>2.595E+02	>2.200E+02	>2.951E+02	>2.341E+02	>2.512E+02	>2.401E+02	**2.142E+02**
f_{22}	>2.555E+03	**=1.000E+02**	>9.137E+02	=1.002E+02	>2.675E+02	=1.005E+02	=1.002E+02	=1.002E+02	**1.000E+02**
f_{23}	>4.392E+02	>4.624E+02	>4.123E+02	>3.706E+02	>4.909E+02	>4.005E+02	>4.037E+02	>3.941E+02	**3.522E+02**
f_{24}	>5.057E+02	>5.636E+02	>4.700E+02	>4.477E+02	>5.172E+02	>4.577E+02	>4.739E+02	>4.673E+02	**4.279E+02**
f_{25}	>4.171E+02	=3.918E+02	=3.882E+02	=3.874E+02	=3.911E+02	=3.875E+02	=3.900E+02	**=3.869E+02**	**3.869E+02**
f_{26}	>2.255E+03	>1.942E+03	>1.355E+03	>1.186E+03	>2.036E+03	<7.263E+02	>1.438E+03	**<4.306E+02**	7.926E+02
f_{27}	>5.892E+02	>5.457E+02	>5.304E+02	>5.195E+02	>5.363E+02	>5.179E+02	>5.170E+02	>5.124E+02	**5.039E+02**
f_{28}	>4.392E+02	>4.066E+02	**<3.132E+02**	>3.755E+02	>4.133E+02	<3.219E+02	=3.595E+02	>3.758E+02	3.621E+02
f_{29}	>9.545E+02	>6.983E+02	>6.613E+02	>4.895E+02	>8.371E+02	>5.972E+02	>6.719E+02	>5.089E+02	**4.213E+02**
f_{30}	>4.606E+04	>2.809E+04	<4.378E+03	**<3.709E+03**	>6.639E+03	<4.083E+03	>5.454E+03	<4.500E+03	4.756E+03
w/t/l	29/0/1	23/2/5	21/1/8	23/2/5	22/1/7	16/2/12	19/3/8	18/2/10	/
best	0	2	1	1	2	3	0	7	16

On three unimodal functions (f_1–f_3), HCLPSO, ML-CLPSO-AM and EPSO achieve the best solutions on f_1, f_2 and f_3, respectively. The Wilcoxon signed rank test results show ML-CLPSO-AM performs better than PSO-cf, FIPS, FDR-PSO, SL-PSO, EL-PSO, EPSO, GL-PSO and HCLPSO on 2, 2, 3, 3, 1, 1, 2, 1 functions, respectively. The performance of ML-CLPSO-AM is better than PSO-cf, FIPS, FDR-PSO, SL-PSO and GL-PSO, and worse than EL-PSO, EPSO and HCLPSO. Compared with five recently reported PSO algorithms, ML-CLPSO-AM does not show significant advantages.

On seven simple multimodal functions (f_4–f_{10}), ML-CLPSO-AM achieves the best solution on four functions (f_5, f_7, f_8 and f_{10}). EPSO, HCLPSO and FIPS obtain the best solutions on f_4, f_6 and f_9, respectively. ML-CLPSO-AM outperforms PSO-cf, FIPS and FDR-PSO on seven (f_4–f_7), six (f_4–f_8, f_{10}), six (f_5–f_{10}) functions, respectively. Compared with recently reported PSO algorithms, ML-CLPSO-AM outperforms SL-PSO, EL-PSO, EPSO, GL-PSO, HCLPSO on seven (f_4–f_{10}), seven (f_4–f_{10}), six (f_5–f_{10}), six (f_5–f_{10}) and six (f_4, f_5, f_7–f_{10}) functions, respectively. ML-CLPSO-AM performs much better than the other PSO algorithms.

On ten hybrid functions (f_{11}–f_{20}), ML-CLPSO-AM generates the best solutions on four functions (f_{11}, f_{16}, f_{17} and f_{20}). HCLPSO, EL-PSO and EPSO yield the best solutions on three (f_{13}, f_{15}, f_{19}), two (f_{14}, f_{18}) and one (f_{12}) functions, respectively. Compared with state-of-the-art PSO algorithms, ML-CLPSO-AM performs significantly better than PSO-cf, FIPS and FDR-PSO on ten (f_{11}–f_{20}), seven (f_{11}, f_{12}, f_{15}–f_{18}, f_{20}), five (f_{11}, f_{15}–f_{17}, f_{20}) functions, respectively. Compared with recently reported PSO algorithms, ML-CLPSO-AM outperforms SL-PSO, EL-PSO, EPSO, GL-PSO and HCLPSO on six ($f_{11}, f_{12}, f_{14}, f_{16}, f_{17}, f_{20}$), five ($f_{11}, f_{12}, f_{16}, f_{17}, f_{20}$), four ($f_{11}, f_{16}, f_{17}, f_{20}$), five ($f_{11}, f_{15}$–$f_{17}, f_{20}$), and five ($f_{11}, f_{12}, f_{16}, f_{17}, f_{20}$) functions, respectively. ML-CLPSO-AM does not exhibit significant advantages over other recently reported PSO variants on hybrid functions.

On ten complex composition functions (f_{21}-f_{30}), ML-CLPSO-AM achieves the best solutions on seven functions (f_{21}–f_{25}, f_{27}, f_{29}). HCLPSO generates the best performance on f_{25} and f_{26}, FIPS, FDR-PSO and SL-PSO wins the best performance on f_{22}, f_{28} and f_{30}, respectively. On f_{22}, FIPS, SL-PSO, EPSO, GL-PSO, HCLPSO and ML-CLPSO-AM generate almost the same mean error. On f_{25}, all the PSO variants yields almost the same mean error except PSO-cf. Compared with PSO-cf, FIPS and FDR-PSO, ML-CLPSO-AM is only outperformed by FDR-PSO on f_{28} and f_{30}. ML-CLPSO-AM only performs worse than SL-PSO, EL-PSO, EPSO, GL-PSO and HCLPSO on one (f_{30}), zeros, three (f_{26}, f_{28}, f_{30}), zeros, two functions (f_{26}, f_{30}), respectively. ML-CLPSO-AM performs better than any other peer PSO algorithms on complex composition functions.

In this group of comparison test, ML-CLPSO-AM performs better than all the involved PSO variants on simple multimodal functions and composition functions. On unimodal functions, ML-CLPSO-AM is outperformed by EL-PSO, EPSO and HCLPSO. On hybrid functions, ML-CLPSO-AM is secondary to EPSO.

To evaluate nine PSO variants' overall performance on 30 CEC2017 functions, they were ranked by mean error in ascending order (the lower rank the better) and the results are given in Table 4.

It can be seen in Table 4 that ML-CLPSO-AM ranks first on most of simple multimodal functions and composition functions, while on unimodal functions and hybrid functions, ML-CLPSO-AM achieves a moderate rank. ML-CLPSO-AM does not perform well on f_{13}, f_{14}, f_{18} and f_{19}. EPSO and HCLPSO showed the highest performance on unimodal functions and hybrid functions, respectively. ML-CLPSO-AM hits the best solutions on more than half of the tested functions. The average ranks of PSO-cf, FIPS, FDR-PSO, SL-PSO, EL-PSO, EPSO, GL-PSO, HCLPSO and ML-CLPSO-AM on 30 CEC2017 functions are 8.233, 6.800, 5.333, 4.033, 6.367, 3.367, 4.667, 3.133 and 2.900, respectively. Evaluating the overall performance of nine PSO algorithms by average ranks, the order is ML-CLPSO-AM, HCLPSO, EPSO, SL-PSO, GL-PSO, FDR-PSO, EL-PSO, FIPS and PSO-cf. The overall performance of ML-CLPSO-AM is better than other comparison PSO algorithms.

Table 4. Rank based analysis of mean performance among PSO algorithms.

Function	PSO-cf	FIPS	FDR-PSO	SL-PSO	EL-PSO	EPSO	GL-PSO	HCLPSO	ML-CLPSO-AM
f_1	9	5	7	8	2	3	4	1	6
f_2	9	8	5	6	7	2	3	4	1
f_3	3	8	2	9	5	1	7	4	6
f_4	9	8	3	6	7	1	2	5	4
f_5	7	9	6	2	8	3	5	4	1
f_6	8	3	7	4	9	5	6	1	2
f_7	6	9	5	8	7	2	3	4	1
f_8	8	9	6	2	7	4	5	3	1
f_9	8	1	5	3	9	7	4	6	2
f_{10}	8	9	6	2	7	4	5	3	1
f_{11}	9	6	8	2	4	7	5	3	1
f_{12}	9	8	2	6	7	1	3	5	4
f_{13}	9	5	7	6	2	3	4	1	8
f_{14}	9	6	5	8	1	3	2	4	7
f_{15}	8	9	7	4	3	2	6	1	5
f_{16}	9	7	5	2	8	4	6	3	1
f_{17}	9	4	7	2	8	5	6	3	1
f_{18}	8	9	4	6	1	3	2	5	7
f_{19}	9	5	7	3	4	2	6	1	8
f_{20}	9	4	6	3	8	5	7	2	1
f_{21}	7	9	6	2	8	3	5	4	1
f_{22}	9	1	8	3	7	6	3	3	1
f_{23}	7	8	6	2	9	4	5	3	1
f_{24}	7	9	5	2	8	3	6	4	1
f_{25}	9	8	5	3	7	4	6	1	1
f_{26}	9	7	5	4	8	2	6	1	3
f_{27}	9	8	6	5	7	4	3	2	1
f_{28}	9	7	1	5	8	2	3	6	4
f_{29}	9	7	5	2	8	4	6	3	1
f_{30}	9	8	3	1	7	2	6	4	5
Average	8.233	6.800	5.333	4.033	6.367	3.367	4.667	3.133	**2.900**
overall	9	8	6	4	7	3	5	2	1

To evaluate the convergence rates of different PSO variants, their convergence curves on six representative functions are given in Figure 2. Six functions are unimodal function f_2, simple multimodal functions f_7, hybrid function f_{12} and f_{17}, composition function f_{23} and f_{29}.

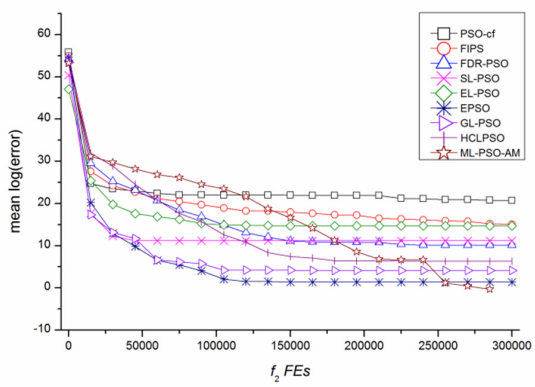
Figure 2a shows that on unimodal function f_2, EPSO and GL-PSO converged fast from the beginning, because EPSO contains a global version HPSO-TAVC component while GL-PSO employs the genetic operator to breed high qualified examples to guide the evolution of the particles. With an independent subgroup to enhance exploitation, HCLPSO performs well too. PSO-cf, FIPS and EL-PSO fell behind. Since ML-CLPSO-AM enhances diversity to explore the search space sufficiently in the early stage, it converges more slowly in the early stage. However, ML-CLPSO-AM gradually changes over to enhance exploitation in the optimization process by linearly adjusting inertia weight and acceleration coefficients, thus it speeds up convergence rate and surpasses a few PSO variants. Figure 1a shows that the convergence rate of ML-CLPSO-AM is accelerated significantly at about

240,000 *FE*s, for the AM strategy can activate the stagnated particles to explore the possible promising areas. The similar phenomenon can be seen in Figure 2b,d,e. In the end, ML-CLPSO-AM achieves the lowest mean error. The AM strategy can improve the performance of ML-CLPSO-AM significantly on f_2.

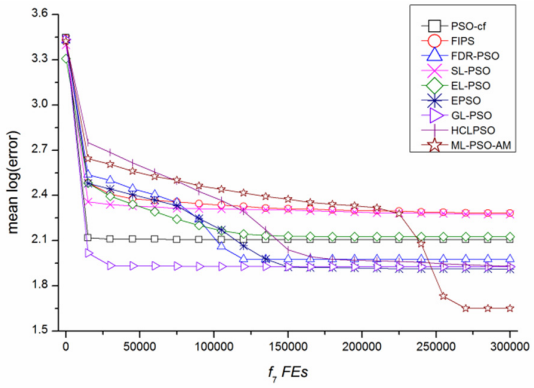
Figure 2b shows that on simple multimodal function f_7, GL-PSO converges fast at the beginning, FDR-PSO, EPSO and HCLPSO speed up the convergence rate and yield almost the same mean error with GL-PSO. SL-PSO and FIPS lag behind. Though ML-CLPSO-AM converges relatively slowly from the beginning, the convergence rate accelerates significantly at about 230,000 *FE*s and it outperforms the other peer PSO variants.

On hybrid functions, Figure 2c shows that on f_{12}, GL-PSO and EPSO converge fast. FDR-PSO keeps a steady convergence rate and catches up with GL-PSO and EPSO in the end. PSO-cf and FIPS converge slowly. Though ML-CLPSO-AM converges relatively slowly at the beginning, it surpasses PSO-cf, FIPS, EL-PSO and SL-PSO in the middle stage. Figure 2d shows that on f_{17}, SL-PSO and HCLPSO converge fast, FIPS, FDR-PSO, EL-PSO, EPSO and GL-PSO follow HCLPSO and achieve almost the same mean error. PSO-cf is trapped in local optima at about 30000 *FE*s and cannot yield high accuracy. After about 100000 *FE*s, the convergence curves of other peer PSO algorithms are almost horizontal. They are trapped in local optima and converge slowly thereafter. Because of the AM strategy, ML-CLPSO-AM escapes from local optima and outperforms other PSO variants.

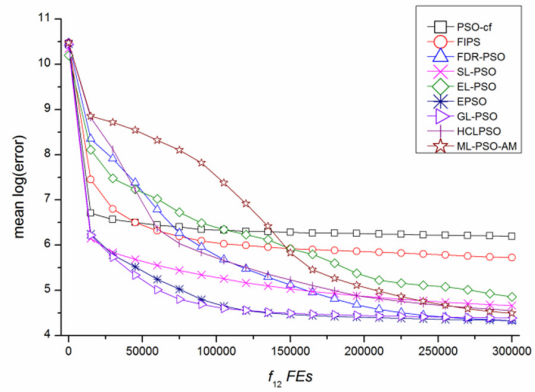
On composition functions, Figure 2e shows on f_{23} SL-PSO converges fast. GL-PSO, HCLPSO and FDR-PSO follow SL-PSO and generate a slightly higher mean error than SL-PSO. EL-PSO and FIPS fall behind. ML-CLPSO-AM converges relatively slowly from the beginning, however, at about 150000 *FE*s, ML-CLPSO-AM speeds up convergence speed significantly by AM strategy and surpasses other peer PSO variants. Figure 2f shows on f_{29}, SL-PSO, EPSO and HCLPSO converges relatively fast. PSO-cf and EL-PSO lag behind. ML-CLPSO-AM keeps steady convergence rate and outperforms other peer PSO variants in the middle stage.



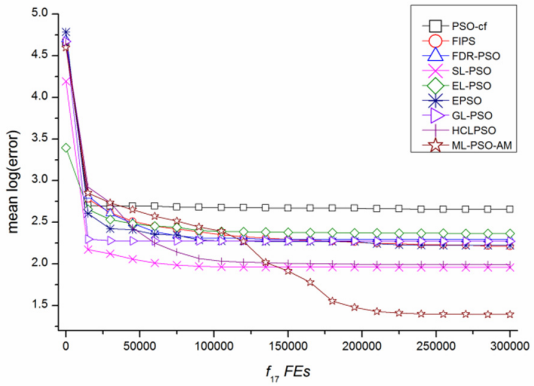
(**a**) Convergence curve of f_2.



(**b**) Convergence curve of f_7.

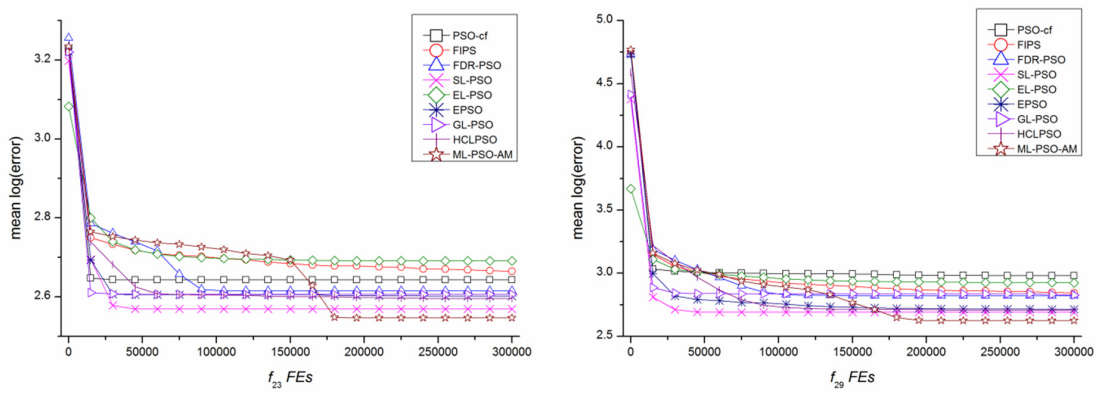


(**c**) Convergence curve of f_{12}.



(**d**) Convergence curve of f_{17}.

Figure 2. *Cont.*



(**e**) Convergence curve of f_{23}. (**f**) Convergence curve of f_{29}.

Figure 2. Convergence curves of the representative functions.

In general, because ML-CLPSO-AM employs linearly adjusted inertia weight and acceleration coefficients to regulate its exploration and exploitation during the optimization process, it converges relatively slowly in the early stage, however, the convergence rate speeds up in the middle stage and achieves high accuracy in the end. The ML strategy can enhance the exploitation of CLPSO without weakening its exploration significantly. The AM strategy can transfer the stagnated particles to search potentially promising areas, hence, ML-CLPSO-AM achieves high accuracy on complex multimodal problems.

4.3.3. Comparison Test with EAs and Meta-Heuristics

In this section, ML-CLPSO-AM is compared with six EAs and meta-heuristics, namely, JADE, ABC, CMA-ES, ABC, GWO, ANS and SSA. All the involved algorithms employ the freely available source codes and authors' suggested parameter configurations. The best test results are highlighted in bold. Table 5 shows that ML-CLPSO-AM yields the best solution on twelve functions, JADE, CMA-ES, ABC and ANS achieve the best solution on seven, seven, four and two functions, respectively. Compared with JADE, ML-CLPSO-AM wins, ties with and loses on sixteen, threeand eleven functions, respectively. ML-CLPSO-AM performs better than JADE on unimodal functions and composition functions, performs similarly to JADE on simple multimodal and hybrid functions. Compared with ABC, ML-CLPSO-AM wins on twenty three functions, loses on seven functions. ABC adopts a one dimension updating strategy and yields the best solution on four composition functions (f_{21},f_{24}–f_{26}). Compared with CMA-ES, ML-CLPSO-AM wins, ties with and loses on eighteen, two, and twelve functions, respectively. CMA-ES generates the best solution on all three unimodal functions. On simple multimodal functions and composition functions, ML-CLPSO-AM outperforms CMA-ES, while on unimodal and hybrid functions, CMA-ES performs better than ML-CLPSO-AM. Compared with GWO, ML-CLPSO-AM wins on all thirty functions. Compared with ANS, ML-CLPSO-AM wins on twenty three functions, ties on two functions and loses on five functions. Compared with SSA, ML-CLPSO-AM only loses on two functions (f_3, f_{14}). ML-CLPSO-AM performs better than ABC, GWO, ANS and SSA on unimodal functions, simple multimodal functions, hybrid functions and composition functions. This group of tests show that ML-CLPSO-AM is competitive compared to six EAs and meta-heuristics.

Table 5. Comparison test results with EAs and meta-heuristics.

Function	JADE	ABC	CMA-ES	GWO	ANS	SSA	ML-CLPSO-AM
f_1	<1.364E-14	<2.410E+02	**<2.842E-14**	>2.409E+09	<3.358E+01	>5.381E+03	4.348E+03
f	>1.219E+04	>5.913E+10	**=0.000E+00**	>8.014E+28	>2.810E+06	>1.024E+06	**0.000E+00**
f_3	>6.864E+03	>1.105E+05	**<1.046E-13**	>3.383E+04	>1.506E+04	<3.429E-08	3.671E-02
f_4	<4.879E+01	<2.243E+01	**<1.472E+01**	>2.022E+02	<3.567E+01	>8.904E+01	6.538E+01
f_5	>2.640E+01	>8.314E+01	>3.057E+02	>9.683E+01	>4.948E+01	>1.121E+02	**1.437E+01**
f_6	**<1.546E-13**	<3.820E-13	>6.355E+01	>8.566E+00	>3.725E-07	>2.884E+01	2.190E-08
f_7	>5.493E+01	>1.043E+02	>7.504E+02	>1.679E+02	>8.234E+01	>1.546E+02	**4.469E+01**
f_8	>2.541E+01	>9.357E+01	>2.248E+02	>8.509E+01	>5.472E+01	>1.200E+02	**1.377E+01**
f_9	**<7.162E-03**	>1.166E+03	>4.697E+03	>7.787E+02	>1.225E+02	>2.018E+03	2.175E-02
f_{10}	>1.872E+03	>2.571E+03	>4.566E+03	>3.009E+03	>2.291E+03	>3.810E+03	**2.312E+02**
f_{11}	>3.489E+01	>7.107E+02	>1.594E+02	>7.469E+02	>2.181E+01	>1.588E+02	**9.753E+00**
f_{12}	**<1.204E+03**	>1.009E+06	<1.721E+03	>5.357E+07	>3.633E+05	>1.342E+06	3.138E+04
f_{13}	**<4.296E+01**	>1.938E+04	<2.147E+03	>2.086E+06	<5.011E+02	>1.036E+05	1.755E+04
f_{14}	<3.709E+03	>1.383E+05	**<1.699E+02**	>3.546E+05	>4.686E+04	<3.435E+03	6.311E+03
f_{15}	>2.541E+03	>4.093E+03	<2.062E+02	>5.967E+05	**<1.346E+02**	>5.081E+04	2.092E+03
f_{16}	>3.719E+02	>6.126E+02	>1.353E+03	>8.156E+02	>6.253E+02	>8.770E+02	**2.862E+01**
f_{17}	>8.579E+01	>1.907E+02	>3.490E+02	>3.132E+02	>1.724E+02	>3.387E+02	**2.472E+01**
f_{18}	<1.865E+04	>2.858E+05	**<1.717E+02**	>1.385E+06	>1.234E+05	>1.760E+05	1.087E+05
f_{19}	<1.552E+03	>1.533E+04	<1.644E+02	>6.058E+05	**<1.495E+02**	>2.986E+05	1.096E+04
f_{20}	>1.207E+02	>2.104E+02	>1.372E+03	>3.321E+02	>2.162E+02	>3.927E+02	**7.202E+01**
f_{21}	>2.267E+02	**<2.107E+02**	>3.976E+02	>2.901E+02	>2.509E+02	>3.108E+02	2.142E+02
f_{22}	**=1.000E+02**	>1.144E+02	>5.830E+03	>2.206E+03	>6.238E+02	>2.229E+03	**1.000E+02**
f_{23}	>3.735E+02	>4.202E+02	>2.306E+03	>4.518E+02	>4.012E+02	>4.644E+02	**3.522E+02**
f_{24}	>4.395E+02	**<3.815E+02**	>5.518E+02	>5.204E+02	>5.234E+02	>5.192E+02	4.279E+02
f_{25}	=3.870E+02	**<3.844E+02**	>3.882E+02	>4.752E+02	=3.850E+02	>3.987E+02	3.869E+02
f_{26}	>1.202E+03	**<2.973E+02**	>2.274E+03	>2.176E+03	>8.069E+02	>1.982E+03	7.926E+02
f_{27}	=5.039E+02	>5.138E+02	**=5.000E+02**	>5.549E+02	>5.145E+02	>5.305E+02	5.039E+02
f_{28}	**<3.478E+02**	>3.987E+02	<3.513E+02	>6.196E+02	>3.977E+02	>3.961E+02	3.621E+02
f_{29}	>4.759E+02	>6.232E+02	>7.836E+02	>8.480E+02	>5.745E+02	>1.049E+03	**4.213E+02**
f_{30}	**<2.157E+03**	>1.549E+04	>5.315E+03	>7.078E+06	=4.724E+03	>1.188E+06	4.756E+03
w/t/l	16/3/11	23/0/7	18/2/10	30/0/0	23/2/5	28/0/2	
best	7	4	7	0	2	0	12

4.4. Parameter Sensitive Analysis

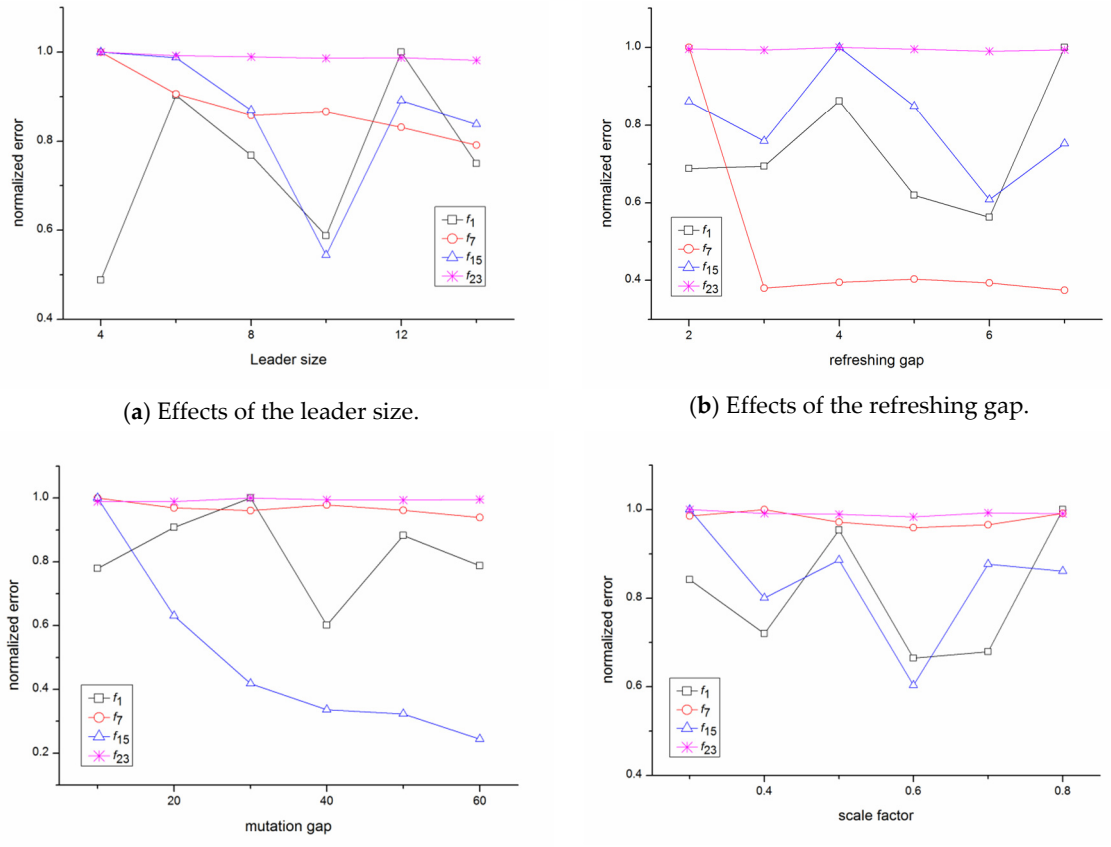
To evaluate the effects of parameters setting on the performance of ML-CLPSO-AM, four representative functions are tested with different parameters setting. In each group of tests, only one parameter is assigned with different value, while the other parameters are set in accordance with Table 1. Each parameter setting on one function is tested for 51 independent runs and the mean error is normalized within the range of [0,1] by dividing the maximum mean error on the tested function. In other words, the lower the normalized error the better.

Figure 3a shows that the composition function f_{23} is not sensitive to the leader size. On unimodal function f_1, the leader size $N_L = 4$ shows the lowest mean error. On simple multimodal functions f_7, the mean error decreases as the leader size increases. The test results show that a smaller leader size favors unimodal functions while a bigger leader size favors multimodal functions. $N_L = 10$ achieves good performance on f_1 and f_{15}; therefore, $N_L = 10$ is the best choice for the leader size.

Figure 3b shows thatf_{23} is insensitive to the refreshing gap, while f_7 achieves almost the same mean error except for the refreshing gap $Stag1m = 2$. $Stag1m = 6$ yields the lowest mean errors on f_1, f_{15} and performs better than the other settings, so $Stag1m = 6$ is employed as the default value of the refreshing gap. A proper refreshing gap can update the particle's comprehensive learning exemplar and social learning leader in a timely way when one particle's fitness value stops improving.

Figure 3c shows that f_7 and f_{23} are insensitive to the mutation gap. The mutation gap $Stag2m = 40$ yields the lowest mean error on f_1 and a relatively low mean error on f_{15}, hence, $Stag2m = 40$ is adopted as the default value of the mutation gap. Too big a mutation gap cannot activate the stagnated particles in time, while too small a mutation gap may impair the algorithm's convergence.

Figure 3d shows that the f_7 and f_{23} are insensitive to scale factor as well. The scale factor $\eta = 0.6$ achieves the lowest mean error on f_1 and f_{15}, hence, $\eta = 0.6$ is the best choice of scale factor. A proper scale factor can improve the efficiency of the AM strategy.



(**a**) Effects of the leader size.

(**b**) Effects of the refreshing gap.

(**c**) Effects of the mutation gap.

(**d**) Effects of the scale factor.

Figure 3. Effects of parameter settings.

5. Application to Economic Load Dispatch Problem

5.1. Problem Definition

The static economic load dispatch (ELD) problem is one of the major optimization problems in power system management. The objective of ELD problem is to minimize the fuel cost of generating units during the specific operation period. The constraints of the power balance, generator, ramp rate and prohibited operating zones are considered in the ELD problem.

Objective Function

The objective functions of ELD problem can be modeled as a quadratic functionas follows in Equation (10).

$$\text{Minimize}: \ F = \sum_{i=1}^{N_G} f_i(P_i) \tag{10}$$

where $f_i(P_i) = a_i P_i^2 + b_i P_i + c_i$, $i = 1, 2, 3, \ldots, N_G$ is the formulation of cost functions corresponding to the ith generator. a_i, b_i and $c_i$are cost coefficients of generator. P_i stands for the ith generator's real power output (in MW) in the period t. N_G denotes the number of total generating units.

Cost Functions with Valve Point Effect

A rippling effect is produced for the steam admission throughthe valve in a turbine, hence it is more practical to take the valvepoint effect into aconsideration of the fuel cost function. The cost function for a generator with valve point loading effects can be expressed in follows in Equation (11).

$$f_i(P_i) = a_i P_i^2 + b_i P_i + c_i + \left| e_i \sin(f_i(P_i^{\min} - P_i)) \right| \tag{11}$$

where e_i and f_i are cost coefficients of valve point loading effects. The constriants of ELD problems are as follows:

Power Balance Constriants

The power balance constraints expressed in Equation (12) are based on the equilibrium between total system generation and total system loads (P_D) and losses (P_L).

$$\sum_{i=1}^{N_G} P_i = P_D + P_L \tag{12}$$

where P_L is calculated according to B-coefficients in the followingEquation (13).

$$P_L = \sum_{i=1}^{N_G} \sum_{j=1}^{N_G} P_i B_{ij} P_j + \sum_{i=1}^{N_G} B_{0i} P_i + B_{00} \tag{13}$$

Generator Constraints

Each generating unit in the power system has a lower and upper bound. The generator constraints are represented by a couple of inequality constraints as below:

$$P_i^{\min} \leq P_i \leq P_i^{\max} \tag{14}$$

where $P_i^{\min}$ and $P_i^{\max}$ correspond to the lower and upper bounds for the power outputs of the ith generator.

Ramp Rate Limits

In practical circumstances, the ramp rate limit restricts the operating range of the online generators when changing-over the generator between two operating periods. The generation may increase or decrease within the relevant upper and lower ramp rate limits. The ramp rate limits are as follows:

$$P_i - P_i^{t-1} \leq UR_i \tag{15}$$

$$P_i^{t-1} - P_i \leq DR_i \tag{16}$$

where P_i^{t-1} is the power output of the ith generator at the previous hour and UR_i and DR_i are the upper and the lower ramp rate limits, respectively.

Prohibited Operating Zones

The generating units have certain prohibited operating zones due to the physical limitations of the machine components or instability such as steam valve or shaft bearings vibration. Hence, discontinuities are produced in the cost objective function by the prohibited operating zones. These zones should be avoided to economize the power production. The prohibited operating zones can be expressed in the following inequality:

$$P_i \leq \breve{P}^{PZ} \text{ and } P_i \geq \breve{P}^{PZ} \tag{17}$$

where $\breve{P}^{PZ}$ and $\breve{P}^{PZ}$ are the lower and upper limits of the prohibited zone for the ith generating unit, respectively.

In this study, three ELD problems were tested. Case 1 is a 6-unit system with transmission losses and valve point effects, the power demand is 1263 MW. Case 2 is a 15-unit system with transmission losses and valve point effects, its power demand is 2630 MW. Case 3 is a 40-unit system without transmission loss, its power demand is 10,500 MW. The experiments for ML-CLPSO-AM and other PSO algorithms were carried out according to the requirements of CEC2011. The maximum *FE*s was set as 150,000 *FE*s. The population size and leader size of ML-CLPSO-AM were set as 100 and 25,

respectively. Other parameter configurations are in accordance with Table 1. Each algorithm was run for 25 independent runs and the test results are given in Sections 5.2 and 5.3.

5.2. Comparison of PSO Algorithms on ELDProblem

In this section, nine PSO algorithms, namely, PSO-cf, FIPS, FDR-PSO, SL-PSO, ELPSO, EPSO, GL-PSO, HCLPSO and ML-CLPSO-AM were tested on an ELD problem. The mean, the standard deviation, the maximum and the minimum costs of 25 independent runs are presented in Table 6, the rank of mean costs is given in Table 7. The row of "average" and "overall" denote the mean rank and the overall rank achieved by one algorithm on three ELD problems, respectively. The best results are highlighted in bold.

Table 6. Comparison test results of PSO algorithms on ELD problems.

Problem	Statistic	PSO-cf	FIPS	FDR-PSO	SL-PSO	EL-PSO	EPSO	GL-PSO	HCLPSO	ML-CLPSO-AM
6 Units	Mean	1.550E+04	1.547E+04	1.547E+04	1.551E+04	1.547E+04	**1.545E+04**	1.548E+04	**1.545E+04**	**1.545E+04**
	St.D	3.773E+01	1.197E+01	9.664E+00	1.897E+01	1.945E+01	2.121E+00	2.154E+01	**2.170E+00**	3.653E+00
	Max	1.560E+04	1.549E+04	1.548E+04	1.554E+04	1.551E+04	1.545E+04	1.552E+04	**1.545E+04**	**1.545E+04**
	Min	1.545E+04	1.545E+04	1.545E+04	1.547E+04	1.545E+04	**1.544E+04**	1.545E+04	1.545E+04	**1.544E+04**
15 Units	Mean	3.310E+04	3.297E+04	3.304E+04	3.298E+04	3.306E+04	3.293E+04	3.294E+04	3.297E+04	**3.273E+04**
	St.D	1.324E+02	5.105E+01	5.911E+01	8.433E+01	8.433E+01	3.745E+01	4.928E+01	**3.347E+01**	3.786E+01
	Max	3.330E+04	3.307E+04	3.317E+04	3.315E+04	3.321E+04	3.303E+04	3.306E+04	3.303E+04	**3.281E+04**
	Min	3.287E+04	3.288E+04	3.295E+04	3.284E+04	3.290E+04	3.289E+04	3.287E+04	3.290E+04	**3.269E+04**
40 Units	Mean	1.441E+05	1.372E+05	1.414E+05	1.329E+05	1.419E+05	1.358E+05	1.406E+05	1.353E+05	1.283E+05
	St.D	6.492E+03	3.093E+03	5.180E+03	3.097E+03	4.586E+03	2.238E+03	4.740E+03	1.674E+03	**6.357E+02**
	Max	1.565E+05	1.424E+05	1.508E+05	1.415E+05	1.490E+05	1.397E+05	1.480E+05	1.374E+05	**1.309E+05**
	Min	1.312E+05	1.325E+05	1.351E+05	1.305E+05	1.358E+05	1.321E+05	1.318E+05	1.320E+05	**1.272E+05**

The test results in Tables 6 and 7 show that EPSO, HCLPSO and ML-CLPSO-AM achieve the lowest mean cost on the 6-unit ELD problem. ML-CLPSO-AM generates the lowest mean cost on 15-unit and 40-unit ELD problems. EPSO and ML-CLPSO-AM yield the lowest minimum cost on the 6-unit ELD problem, ML-CLPSO-AM yields the lowest minimum cost on 15-unit and 40-unit ELD problems. The rank of mean cost in Table 7 shows that EPSO, HCLPSO and ML-CLPSO-AM tie for first on the 6-unit ELD problem. ML-CLPSO-AM ranks first on 15-unit and 40-unit ELD problems. HCLPSO generates the lowest standard deviation on 6-unit and 15-unit ELD problems. The standard deviation of ML-CLPSO-AM is quite close to HCLPSO on 6-unit and 15-unit ELD problems. ML-CLPSO-AM yields the lowest standard deviation on the 40unit ELD problem. Table 7 shows that the average ranks of PSO-cf, FIPS, FDR-PSO, SL-PSO, ELPSO, EPSO, GL-PSO HCLPSO and ML-CLPSO-AM on three ELD problems are 8.667, 4.333, 6.000, 5.667, 6.667, 2.333, 5.333, 2.667 and 1.000, respectively. Evaluating the overall performance of nine PSO algorithms by average rank, the order is ML-CLPSO-AM, EPSO, HCLPSO, GL-PSO, FIPS, FDR-PSO, EL-PSO and PSO-cf. The test results indicate that overall, ML-CLPSO-AM achieves the highest performance on ELD problems.

Table 7. Rank of mean cost among PSO algorithms on ELD problems.

Problem	PSO-cf	FIPS	FDR-PSO	SL-PSO	EL-PSO	EPSO	GL-PSO	HCLPSO	ML-CLPSO-AM
6 Units	8	4	4	9	4	**1**	7	**1**	**1**
15 Units	9	4	7	6	8	**2**	**3**	4	**1**
40 Units	9	5	7	**2**	8	4	6	3	**1**
Average	8.667	4.333	6.000	5.667	6.667	2.333	5.333	2.667	**1.000**
Overall	9	4	7	6	8	2	5	3	**1**

5.3. Comparison with ELD Tailored Algorithms

In this section, ML-CLPSO-AM was compared with several classic and recently ELD tailored meta-heuristics and EAs. The classic meta-heuristics and EAs are simulated annealing (SA) [76], multiple tabu search algorithm (MTS) [77], self-organizing hierarchical particle swarm optimization [78], random drift particle swarm optimization (RD-PSO) [79], ant colony optimization (ACO) [80], classic evolutionary programming (CEP) [81] and Shuffled differential evolution (SDE) [82]. The recent

meta-heuristics are modified crow search algorithm (MCSA) [83], chaotic bat algorithm (CBA) [84], exchange market algorithm (EMA) [85], modified hybrid particle swarm optimization and gravitational search algorithm based on fuzzy logic (FPSOGSA) [86]. The results of the comparison algorithms in Tables 8–10 are cited from the corresponding references. The best results are highlighted in bold.

On 6-unit systems, Table 8 shows that *RD-PSO achieves the lowest best cost and mean cost, while ML-CLPSO-AM achieves the lowest maximum cost. The lowest cost and mean cost of ML-CLPSO-AM are very close to *RD-PSO. CBA generates the highest best cost and mean cost, while SOH-PSO generates the highest maximum cost. The best cost for ML-CLPSO-AM is lower than CBA, SA, MTS, MCSA, SOH-PSO and DE. The performance of ML-CLPSO-AM is comparable to ELD tailed algorithms. Table 11 shows the best optimization results for ML-CLPSO-AM. The minimum cost is $ 15444.1923. The total power production and power loss are 1275.4220 MW, 12.4223 MW, respectively. Subtracting the power loss, the actual power output is 1262.9997 MW.

Table 8. Comparison test results on a 6-unit system.

Algorithm	Best Cost ($/h)	Mean Cost ($/h)	Maximum Cost ($/h)
CBA [84]	15,450.24	15,454.76	15,518.66
SA [77]	15,461.10	15,488.98	15,519.87
MTS [77]	15,450.14	15,451.17	15,453.64
MCSA [83]	15,449.1672	15,449.2358	15,449.3854
SOH-PSO [78]	15,446.02	15,497.35	15,609.64
DE [79]	15,444.9466	15,450.1339	15,472.0651
*RD-PSO [79]	15,442.7575	15,445.0245	15,455.2936
ML-CLPSO-AM	15,444.1923	15,446.5392	15,449.0358

Table 9. Comparison test results on a 15-unit system.

Algorithm	Best Cost ($/h)	Mean Cost ($/h)	Maximum Cost ($/h)
SA [77]	32,786.40	32,869.51	33,028.95
MTS [77]	32,716.87	32,767.21	32,796.15
SOH-PSO [78]	32,751	32,878	32,945
ACSS [87]	32,678.1290	**32,727.6967**	**32,761.3126**
HGPSO [79]	32,864.0501	33,034.1894	33,280.2655
DE [79]	32,818.5792	32,990.8673	33,116.9340
*RD-PSO [79]	**32,652.3357**	32,744.5873	32,959.7592
ML-CLPSO-AM	32,694.1963	32,728.3782	32,813.3675

Table 10. Comparison test results on a 40-unit system.

Algorithm	Best Cost ($/h)	Mean Cost ($/h)	Maximum Cost ($/h)
ACO [80]	121,811.3700	121,930.5800	122,048.0600
CEP [81]	123,488.29	124,793.48	126,902.89
EMA [85]	121,412.5355	121,417.1328	121,426.1548
MCSA [83]	**121,412.140**	**121,413.280**	**121,414.324**
SDE [82]	121,412.54	121,415.72	121,418.58
FPSOGSA [86]	121,412.54211	121,413.5619	121,414.9838
*RD-PSO [79]	128,864.4525	129,482.0970	131,129.0861
ML-CLPSO-AM	127,188.4367	128,319.3124	130,873.5517

Table 11. Best solution achieved by ML-CLPSO-AM on a 6-unit system.

Unit (MW)	Output	Unit (MW)	Output	Economic	Output
*P*1	446.7141	*P*4	143.4893	P_T	1275.4220
*P*2	173.1494	*P*5	163.9166	P_L	12.4221
*P*3	262.7955	*P*6	85.3571	*Cost*	15,444.19

Note: *P*1–*P*6 denote the power output of power generation unit 1–6. P_T and P_L stand for total power production and power loss, respectively. *Cost* stands for fuel cost ($).

On a 15-unit system, the optimization results in Table 9 show that *RD-PSO yields the lowest best cost, and ACSS yields the lowest mean cost and the maximum cost. HGPSO generates the highest best cost, mean cost and the maximum cost. The best cost of ML-CLPSO-AM is lower than SA, MTS, SOH-PSO, HGPSO and DE, and slightly higher than *RD-PSO and ACSS. The performance of ML-CLPSO-AM is very close to *RD-PSO and ACSS. Table 12 reports the best optimization results generated by ML-CLPSO-AM. The best cost is $32,694.1960. The total power production and power loss are 2659.5114 and 29.4981, respectively. Subtracting the power loss, the actual power output is 2630.013 MW.

Table 12. Best solution achieved by ML-CLPSO-AM on 15 units system.

Unit (MW)	Output	Unit (MW)	Output	Unit (MW)	Output
*P*1	454.9034	*P*7	429.9984	*P*13	25.28378
*P*2	379.9322	*P*8	68.50954	*P*14	16.61231
*P*3	129.9973	*P*9	59.56914	*P*15	15.15478
*P*4	129.9356	*P*10	159.8812	P_T	2659.5114
*P*5	169.9948	*P*11	79.82684	P_L	29.4981
*P*6	459.9836	*P*12	79.92868	*Cost*	32,694.1960

On a 40-unit system, Table 10 shows that MCSA achieves the lowest best cost, mean cost and maximum cost. *RD-PSO yields the highest best cost, mean cost and the maximum cost. The best cost of ML-CLPSO-AM is lower than *RD-PSO, but higher than ACO, CEP, EMA, MCSA, SDE and FPSOGSA. With the increase in unit number, the performance of ML-CLPSO-AM deteriorates. The best optimization results of ML-CLPSO-AM in Table 13 indicate that the total power production and fuel cost are 10,499.9561 and 1,271,288.4367, respectively.

Table 13. Best solution achieved by ML-CLPSO-AM on a 40-unit system.

Unit (MW)	Output	Unit (MW)	Output	Unit (MW)	Output
*P*1	110.5773	*P*15	465.5772	*P*29	10.30963
*P*2	110.1358	*P*16	343.512	*P*30	81.38597
*P*3	85.77042	*P*17	389.6023	*P*31	180.0739
*P*4	150.951	*P*18	469.7308	*P*32	189.1115
*P*5	83.85475	*P*19	510.7922	*P*33	181.8391
*P*6	89.80644	*P*20	287.6341	*P*34	158.6189
*P*7	258.3048	*P*21	522.3104	*P*35	193.7992
*P*8	283.6892	*P*22	502.1889	*P*36	197.6772
*P*9	285.4642	*P*23	503.5271	*P*37	108.5601
*P*10	131.3871	*P*24	485.1643	*P*38	109.7186
*P*11	94.59752	*P*25	523.6777	*P*39	108.7393
*P*12	370.3646	*P*26	522.0327	*P*40	525.4569
*P*13	403.3944	*P*27	10.18045	P_T	10,499.9561
*P*14	449.8235	*P*28	10.61457	*Cost*	127,188.4367

In general, ML-CLPSO-AM achieves high performance on 6-unit and15-unit ELD problems. On the 40-unit problem, the performance of ML-CLPSO-AM is a bit weaker than the ELD tailored algorithms. ML-CLPSO-AM performs well on small scale ELD problems. With the increase in unit number, the performance of ML-CLPSO-AM weakens.

6. Conclusions

This paper combined the ML strategy with CLPSO to enhance exploitation without significantly impairing its exploration. The resultant algorithm is denoted as ML-CLPSO. The ML strategy selects the leader of each particle from a group of top ranked particles. In ML-CLPSO, different particles learn from different leaders and the leader of the particle is updated dynamically during the optimization process. Hence, ML-CLPSO can make good use of multiple high-quality particles to enhance exploitation. To activate the stagnated particles, the AM strategy is introduced. The AM strategy employs the distribution information of the candidate leader set to generate new *Pbest* for the stagnated particles, thereby, the stagnated particles can search the potential promising area after mutation.

To test the effects of ML and AM strategies, three novel PSO algorithms, namely, ML-CLPSO, CLPSO-AM and ML-CLPSO-AM were developed by following the principle of CLPSO. The comparison test results on CEC2017 test suite show that after combining the ML strategy with CLPSO, the resultant ML-CLPSO performs significantly better than CLPSO. Combining the AM strategy with CLPSO-G does not result in significant improvement because CLPSO-G cannot preserve sufficient diversity. The ML and AM strategies are compatible. Combining them simultaneously with CLPSO, the resultant ML-CLPSO-AM performs better than ten peer PSO variants including the five recently reported PSO variants: SL-PSO, EL-PSO, EPSO, GL-PSO and HCLPSO. ML-CLPSO-AM performs better than six representative EAs and meta-heuristics as well. Finally, the experiments on ELD problems show that ML-CLPSO-AM is applicable to real-life application.

For future research, more information should be considered in the selection the leader of PSO, such as the Euler distance and the search success rate. Furthermore, the leader size should be adaptively adjusted during the optimization process to achieve better performance.

Supplementary Materials: The following are available online at http://www.mdpi.com/1996-1073/12/1/116/s1, Figure S1: Mean computing time of PSO algorithms (Unit: Second).

Author Contributions: A.L., Conceptualization, Methodology, Software, Writing-Original Draft Preparation; W.S., Supervision.

Funding: Independent Research Project of State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body 71765003, the Hunan Key Laboratory of Intelligent Robot Technology in Electronic Manufacturing Open Foundation Grant 2017TP1011, the National Natural Science Foundation of China under Grant 61573135, the Hunan Provincial Innovation Foundation for Postgraduate Grant CX2017B110.

Acknowledgments: The authors thank the reviewers for their constructive comments, which helped to improve the quality of the manuscript. The authors thank the reference authors for providing their codes.

Conflicts of Interest: The authors declare that there is no conflict of interest.

References

1. Tanweer, M.R.; Suresh, S.; Sundararajan, N. Dynamic mentoring and self-regulation based particle swarm optimization algorithm for solving complex real-world optimization problems. *Inf. Sci.* **2016**, *326*, 1–24. [CrossRef]
2. Eberhart, R.; Kennedy, J. A new optimizer using particle swarm theory. In Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 4–6 October 1995; pp. 39–43.
3. Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In Proceedings of the International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; pp. 1942–1948.
4. Dorigo, M.; Gambardella, L.M. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Trans. Evol. Comput.* **1997**, *1*, 53–66. [CrossRef]

5. Karaboga, D.; Basturk, B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *J. Glob. Optim.* **2007**, *39*, 459–471. [CrossRef]
6. Gandomi, A.H.; Yang, X.S.; Alavi, A.H. Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Eng. Comput.* **2013**, *29*, 17–35. [CrossRef]
7. Chiroma, H.; Herawan, T.; Fister, I.; Fister, I.; Abdulkareem, S.; Shuib, L.; Hamza, M.F.; Saadi, Y.; Abubakar, A. Bio-inspired computation: Recent development on the modifications of the cuckoo search algorithm. *Appl. Soft Comput.* **2017**, *61*, 149–173. [CrossRef]
8. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
9. Brezočnik, L.; Fister, I.; Podgorelec, V. Swarm Intelligence Algorithms for Feature Selection: A Review. *Appl. Sci.* **2018**, *8*, 1521. [CrossRef]
10. Yu, H.; Tan, Y.; Zeng, J.; Sun, C.; Jin, Y. Surrogate-assisted Hierarchical Particle Swarm Optimization. *Inf. Sci.* **2018**, *454*, 59–72. [CrossRef]
11. Cao, L.; Xu, L.; Goodman, E.D. A neighbor-based learning particle swarm optimizer with short-term and long-term memory for dynamic optimization problems. *Inf. Sci.* **2018**, *453*, 463–485. [CrossRef]
12. Yang, Q.; Chen, W.N.; Gu, T.L.; Zhang, H.X.; Deng, J.D.; Li, Y.; Zhang, J. Segment-Based Predominant Learning Swarm Optimizer for Large-Scale Optimization. *IEEE Trans. Cybern.* **2017**, *47*, 2896–2910. [CrossRef]
13. Ge, H.W.; Sun, L.; Tan, G.Z.; Chen, Z.; Chen, C.L.P. Cooperative Hierarchical PSO With Two Stage Variable Interaction Reconstruction for Large Scale Optimization. *IEEE Trans. Cybern.* **2017**, *47*, 2809–2823. [CrossRef]
14. Mistry, K.; Zhang, L.; Neoh, S.C.; Lim, C.P.; Fielding, B. A Micro-GA Embedded PSO Feature Selection Approach to Intelligent Facial Emotion Recognition. *IEEE Trans. Cybern.* **2017**, *47*, 1496–1509. [CrossRef]
15. Chen, P.Y.; Chao, K.H.; Liao, B.J.; Sciubba, E. Joint Operation between a PSO-Based Global MPP Tracker and a PV Module Array Configuration Strategy under Shaded or Malfunctioning Conditions. *Energies* **2018**, *11*, 2005. [CrossRef]
16. Liu, X.; An, H.; Wang, L.; Jia, X. An integrated approach to optimize moving average rules in the EUA futures market based on particle swarm optimization and genetic algorithms. *Appl. Energy* **2017**, *185*, 1778–1787. [CrossRef]
17. Elsayed, W.; Hegazy, Y.; El-Bages, M.; Bendary, F. Improved Random Drift Particle Swarm Optimization with Self-Adaptive Mechanism for Solving the Power Economic Dispatch Problem. *IEEE Trans. Ind. Inform.* **2017**, *13*, 1017–1026. [CrossRef]
18. Xu, D.; Yu, L.; Lv, Z.; Zhang, J.; Fan, D.; Dai, W. Energy Consumption Optimization for the Formation of Multiple Robotic Fishes Using Particle Swarm Optimization. *Energies* **2018**, *11*, 2023. [CrossRef]
19. Zhang, Y.; Wang, S.; Ji, G. A Comprehensive Survey on Particle Swarm Optimization Algorithm and Its Applications. *Math. Prob. Eng.* **2015**, *2015*, 1–38. [CrossRef]
20. Shi, Y.; Eberhart, R.C. Empirical study of particle swarm optimization. In Proceedings of the 1999 Congress on Evolutionary Computation (CEC 99), Washington, DC, USA, 6–9 July 1999; Volume 321, pp. 320–324.
21. Shi, Y.; Eberhart, R.C. Fuzzy adaptive particle swarm optimization. In Proceedings of the 2001 Congress on Evolutionary Computation, Seoul, Korea, 27–30 May 2001; pp. 101–106.
22. Zhan, Z.-H.; Zhang, J.; Li, Y.; Chung, H.S.-H. Adaptive particle swarm optimization. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2009**, *39*, 1362–1381. [CrossRef]
23. Mendes, R.; Kennedy, J.; Neves, J. The fully informed particle swarm: Simpler, maybe better. *IEEE Trans. Evol. Comput.* **2004**, *8*, 204–210. [CrossRef]
24. Liang, J.J.; Suganthan, P.N. Dynamic multi-swarm particle swarm optimizer. In Proceedings of the 2005 IEEE Swarm Intelligence Symposium, Pasadena, CA, USA, 8–10 June2005; pp. 124–129.
25. Sun, S.Y.; Li, J.W. A two-swarm cooperative particle swarms optimization. *Swarm Evol. Comput.* **2014**, *15*, 1–18. [CrossRef]
26. Lim, W.H.; Isa, N.A.M. Particle swarm optimization with increasing topology connectivity. *Eng. Appl. Artif. Intell.* **2014**, *27*, 80–102. [CrossRef]
27. Qu, B.Y.; Suganthan, P.N.; Das, S. A Distance-Based Locally Informed Particle Swarm Model for Multimodal Optimization. *IEEE Trans. Evol. Comput.* **2013**, *17*, 387–402. [CrossRef]
28. Peram, T.; Veeramachaneni, K.; Mohan, C.K. Fitness-distance-ratio based particle swarm optimization. In Proceedings of the 2003 IEEE Swarm Intelligence Symposium, Indianapolis, IN, USA, 26 April 2003; pp. 174–181.

29. Parsopoulos, K.E.; Vrahatis, M.N. Unified Particle Swarm Optimization for solving constrained engineering optimization problems. In Proceedings of the International Conference on Natural Computation, Changsha, China, 27–29 2005; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3612, pp. 582–591.
30. Zhan, Z.H.; Zhang, J.; Li, Y.; Shi, Y.H. Orthogonal Learning Particle Swarm Optimization. *IEEE Trans. Evol. Comput.* **2011**, *15*, 832–847. [CrossRef]
31. Li, C.; Yan, S.; Nguyen, T.T. A self-learning particle swarm optimizer for global optimization problems. *IEEE Trans. Syst. Man Cybern. B Cybern.* **2012**, *42*, 627–646.
32. Juang, C.F. A hybrid of genetic algorithm and particle swarm optimization for recurrent network design. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2004**, *34*, 997–1006. [CrossRef]
33. Shieh, H.L.; Kuo, C.C.; Chiang, C.M. Modified particle swarm optimization algorithm with simulated annealing behavior and its numerical verification. *Appl. Math. Comput.* **2011**, *218*, 4365–4383. [CrossRef]
34. Liu, H.; Cai, Z.X.; Wang, Y. Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization. *Appl. Soft Comput.* **2010**, *10*, 629–640. [CrossRef]
35. Ouyang, H.B.; Gao, L.Q.; Kong, X.Y.; Li, S.; Zou, D.X. Hybrid harmony search particle swarm optimization with global dimension selection. *Inf. Sci.* **2016**, *346–347*, 318–337. [CrossRef]
36. Wei, H.L.; Isa, N.A.M. Teaching and peer-learning particle swarm optimization. *Appl. Soft Comput.* **2014**, *18*, 39–58.
37. Jiang, S.; Ji, Z.; Shen, Y. A novel hybrid particle swarm optimization and gravitational search algorithm for solving economic emission load dispatch problems with various practical constraints. *Int. J. Electr. Power* **2014**, *55*, 628–644. [CrossRef]
38. Khunkitti, S.; Siritaratiwat, A.; Premrudeepreechacharn, S.; Chatthaworn, R.; Watson, N. A Hybrid DA-PSO Optimization Algorithm for Multiobjective Optimal Power Flow Problems. *Energies* **2018**, *11*, 2270. [CrossRef]
39. Tanweer, M.R.; Suresh, S.; Sundararajan, N. Self regulating particle swarm optimization algorithm. *Inf. Sci.* **2015**, *294*, 182–202. [CrossRef]
40. Tanweer, M.R.; Auditya, R.; Suresh, S.; Sundararajan, N.; Srikanth, N. Directionally Driven Self-Regulating Particle Swarm Optimization algorithm. *Swarm Evol. Comput.* **2016**, *28*, 98–116. [CrossRef]
41. Qin, Q.D.; Cheng, S.; Zhang, Q.Y.; Li, L.; Shi, Y.H. Particle Swarm Optimization With Interswarm Interactive Learning Strategy. *IEEE Trans. Cybern.* **2016**, *46*, 2238–2251. [CrossRef]
42. Mo, S.M.; Zeng, J.C.; Xu, W.B. Attractive and Repulsive Fully Informed Particle Swarm Optimization based on the modified Fitness Model. *Soft Comput.* **2016**, *20*, 863–884. [CrossRef]
43. Dong, W.Y.; Kang, L.L.; Zhang, W.S. Opposition-based particle swarm optimization with adaptive mutation strategy. *Soft Comput.* **2017**, *21*, 5081–5090. [CrossRef]
44. Wang, H.D.; Jin, Y.C.; Doherty, J. Committee-Based Active Learning for Surrogate-Assisted Particle Swarm Optimization of Expensive Problems. *IEEE Trans. Cybern.* **2017**, *47*, 2664–2677. [CrossRef]
45. Ye, W.X.; Feng, W.Y.; Fan, S.H. A novel multi-swarm particle swarm optimization with dynamic learning strategy. *Appl. Soft Comput.* **2017**, *61*, 832–843. [CrossRef]
46. Dong, W.Y.; Zhou, M.C. A Supervised Learning and Control Method to Improve Particle Swarm Optimization Algorithms. *IEEE Trans. Syst. Man Cybern. Syst.* **2017**, *47*, 1135–1148. [CrossRef]
47. Wang, F.; Zhang, H.; Li, K.S.; Lin, Z.Y.; Yang, J.; Shen, X.L. A hybrid particle swarm optimization algorithm using adaptive learning strategy. *Inf. Sci.* **2018**, *436*, 162–177. [CrossRef]
48. Liang, J.J.; Qin, A.K.; Member, S.; Suganthan, P.N.; Member, S.; Baskar, S. Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions. *IEEE Trans. Evol. Comput.* **2006**, *10*, 281–295. [CrossRef]
49. Nasir, M.; Das, S.; Maity, D.; Sengupta, S.; Halder, U.; Suganthan, P.N. A dynamic neighborhood learning based particle swarm optimizer for global numerical optimization. *Inf. Sci.* **2012**, *209*, 16–36. [CrossRef]
50. Clerc, M.; Kennedy, J. The particle swarm—Explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evol. Comput.* **2002**, *6*, 58–73. [CrossRef]
51. Cheng, R.; Jin, Y. A social learning particle swarm optimization algorithm for scalable optimization. *Inf. Sci.* **2015**, *291*, 43–60. [CrossRef]
52. Jordehi, A.R. Enhanced leader PSO (ELPSO): A new PSO variant for solving global optimisation problems. *Appl. Soft Comput.* **2015**, *26*, 401–417. [CrossRef]
53. Lynn, N.; Suganthan, P.N. Ensemble particle swarm optimizer. *Appl. Soft Comput.* **2017**, *55*, 533–548. [CrossRef]

54. Gong, Y.J.; Li, J.J.; Zhou, Y.C.; Li, Y.; Chung, H.S.H.; Shi, Y.H.; Zhang, J. Genetic Learning Particle Swarm Optimization. *IEEE Trans. Cybern.* **2016**, *46*, 2277–2290. [CrossRef]
55. Lynn, N.; Suganthan, P.N. Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation. *Swarm Evol. Comput.* **2015**, *24*, 11–24. [CrossRef]
56. Zhang, J.Q.; Sanderson, A.C. JADE: Adaptive Differential Evolution With Optional External Archive. *IEEE Trans. Evol. Comput.* **2009**, *13*, 945–958. [CrossRef]
57. Hansen, N.; Muller, S.D.; Koumoutsakos, P. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evol. Comput.* **2003**, *11*, 1–18. [CrossRef]
58. Wu, G. Across neighbourhood search for numerical optimization. *Inf. Sci.* **2016**, *329*, 597–618. [CrossRef]
59. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191. [CrossRef]
60. Shi, Y.H.; Eberhart, R. A modified particle swarm optimizer. In Proceedings of the IEEE World Congress on Computational Intelligence, 1998 IEEE International Conference on Evolutionary Computation Proceedings, Anchorage, AK, USA, 4–9 May 1998; pp. 69–73. [CrossRef]
61. Chen, W.N.; Zhang, J.; Lin, Y.; Chen, N.; Zhan, Z.H.; Chung, H.S.H.; Li, Y.; Shi, Y.H. Particle swarm optimization with an aging leader and challengers. *IEEE Trans. Evol. Comput.* **2013**, *17*, 241–258. [CrossRef]
62. Yu, K.J.; Wang, X.; Wang, Z.L. Multiple learning particle swarm optimization with space transformation perturbation and its application in ethylene cracking furnace optimization. *Knowl.-Based Syst.* **2016**, *96*, 156–170. [CrossRef]
63. Holland, J.H. Erratum: Genetic Algorithms and the Optimal Allocation of Trials. *Siam J. Comput.* **1973**, *2*, 88–105. [CrossRef]
64. Higashi, N.; Iba, H. Particle swarm optimization with Gaussian mutation. In Proceedings of the 2003 IEEE Swarm Intelligence Symposium, Indianapolis, IN, USA, 26 April 2003; pp. 72–79.
65. Jiao, W.; Liu, G.; Liu, D. Elite Particle Swarm Optimization with mutation. In Proceedings of the Asia Simulation Conference—International Conference on System Simulation and Scientific Computing, Beijing, China, 10–12 October 2018; pp. 800–803.
66. Wang, H.; Wang, W.J.; Wu, Z.J. Particle swarm optimization with adaptive mutation for multimodal optimization. *Appl. Math. Comput.* **2013**, *221*, 296–305. [CrossRef]
67. Peng, Y.; Lu, B.L. A hierarchical particle swarm optimizer with latin sampling based memetic algorithm for numerical optimization. *Appl. Soft Comput.* **2013**, *13*, 2823–2836. [CrossRef]
68. Pehlivanoglu, Y.V. A New Particle Swarm Optimization Method Enhanced With a Periodic Mutation Strategy and Neural Networks. *IEEE Trans. Evol. Comput.* **2013**, *17*, 436–452. [CrossRef]
69. Qiu, C.Y.; Wang, C.L.; Zuo, X.Q. A novel multi-objective particle swarm optimization with K-means based global best selection strategy. *Int. J. Comput. Int. Syst.* **2013**, *6*, 822–835. [CrossRef]
70. Ratnaweera, A.; Halgamuge, S.K.; Watson, H.C. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Trans. Evol. Comput.* **2004**, *8*, 240–255. [CrossRef]
71. Hu, M.; Wu, T.; Weir, J.D. An Adaptive Particle Swarm Optimization With Multiple Adaptive Methods. *IEEE Trans. Evol. Comput.* **2013**, *17*, 705–720. [CrossRef]
72. Awad, N.H.; Ali, M.Z.; Liang, J.J.; Qu, B.Y.; Suganthan, P.N. Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Bound Constrained Real-Parameter Numerical Optimization. Available online: http://www.ntu.edu.sg/home/epnsugan/ (accessed on 15 October 2016).
73. Hollander, M.; Wolfe, D.A. *Nonparametric Statistical Methods*, 2nd ed.; John Wiley & Sons Inc.: New York, NY, USA, 1999; Volume 2, p. 787.
74. Gibbons, J.D.; Chakraborti, S. *Nonparametric Statistical Inference*, 5th ed.; Chapman & Hall: London, UK, 2010. [CrossRef]
75. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [CrossRef]
76. Kirkpatrick, S.; Gelatt, C.D., Jr.; Vecchi, M.P. Optimization by Simulated Annealing. *Science* **1983**, *220*, 671. [CrossRef]

77. Pothiya, S.; Ngamroo, I.; Kongprawechnon, W. Application of multiple tabu search algorithm to solve dynamic economic dispatch considering generator constraints. *Energy Convers. Manag.* **2008**, *49*, 506–516. [CrossRef]
78. Chaturvedi, K.T.; Pandit, M.; Srivastava, L. Self-Organizing Hierarchical Particle Swarm Optimization for Nonconvex Economic Dispatch. *IEEE Trans. Power Syst.* **2008**, *23*, 1079–1087. [CrossRef]
79. Sun, J.; Palade, V.; Wu, X.J.; Fang, W.; Wang, Z. Solving the Power Economic Dispatch Problem with Generator Constraints by Random Drift Particle Swarm Optimization. *IEEE Trans. Ind. Inform.* **2013**, *10*, 222–232. [CrossRef]
80. Pothiya, S.; Ngamroo, I.; Kongprawechnon, W. Ant colony optimisation for economic dispatch problem with non-smooth cost functions. *Int. J. Electr. Power* **2010**, *32*, 478–487. [CrossRef]
81. Sinha, N.; Chakrabarti, R.; Chattopadhyay, P.K. Evolutionary programming techniques for economic load dispatch. *Int. J. Emerg. Electr. Power Syst.* **2003**, *7*, 83–94. [CrossRef]
82. Reddy, A.S.; Vaisakh, K. Shuffled differential evolution for large scale economic dispatch. *Electr. Power Syst. Res.* **2013**, *96*, 237–245. [CrossRef]
83. Mohammadi, F.; Abdi, H. A Modified Crow Search Algorithm (MCSA) for Solving Economic Load Dispatch Problem. *Appl. Soft Comput.* **2018**, *71*, 51–65. [CrossRef]
84. Adarsh, B.R.; Raghunathan, T.; Jayabarathi, T.; Yang, X.S. Economic dispatch using chaotic bat algorithm. *Energy* **2016**, *96*, 666–675. [CrossRef]
85. Ghorbani, N.; Babaei, E. Exchange market algorithm for economic load dispatch. *Int. J. Electr. Power* **2016**, *75*, 19–27. [CrossRef]
86. Duman, S.; Yorukeren, N.; Altas, I.H. A novel modified hybrid PSOGSA based on fuzzy logic for non-convex economic dispatch problem with valve-point effect. *Int J. Electr. Power* **2015**, *64*, 121–135. [CrossRef]
87. Zakian, P.; Kaveh, A. Economic dispatch of power systems using an adaptive charged system search algorithm. *Appl. Soft Comput.* **2018**, *73*, 607–622. [CrossRef]