

Article

APSO-SL: An Adaptive Particle Swarm Optimization with State-Based Learning Strategy

Mingqiang Gao ¹ and Xu Yang ^{2,*}

¹ School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan 114051, China; 18841209319@163.com

² School of Electrical and Control Engineering, Shenyang Jianzhu University, Shenyang 110168, China

* Correspondence: 13998346746@163.com

Abstract: Particle swarm optimization (PSO) has been extensively used to solve practical engineering problems, due to its efficient performance. Although PSO is simple and efficient, it still has the problem of premature convergence. In order to address this shortcoming, an adaptive particle swarm optimization with state-based learning strategy (APSO-SL) is put forward. In APSO-SL, the population distribution evaluation mechanism (PDEM) is used to evaluate the state of the whole population. In contrast to using iterations to just the population state, using the population spatial distribution is more intuitive and accurate. In PDEM, the population center position and best position for calculation are used for calculation, greatly reducing the algorithm's computational complexity. In addition, an adaptive learning strategy (ALS) has been proposed to avoid the whole population's premature convergence. In ALS, different learning strategies are adopted according to the population state to ensure the population diversity. The performance of APSO-SL is evaluated on the CEC2013 and CEC2017 test suites, and one engineering problem. Experimental results show that APSO-SL has the best performance compared with other competitive PSO variants.

Keywords: particle swarm optimization (PSO); adaptive; state-based; complex optimization



Citation: Gao, M.; Yang, X. APSO-SL: An Adaptive Particle Swarm Optimization with State-Based Learning Strategy. *Processes* **2024**, *12*, 400. <https://doi.org/10.3390/pr12020400>

Academic Editor: Francisco Vazquez

Received: 13 December 2023

Revised: 14 January 2024

Accepted: 26 January 2024

Published: 17 February 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Optimization problems occur frequently in various practical engineering problems [1]. Thus, it is crucial to solve optimization problems efficiently [2]. Optimization problems have gradually become an important issue in the industrial field [3]. With the progress of science and technology, optimization problems in the industrial field are becoming more and more complex [4]. They are often accompanied by a large number of discontinuous, non-microscopic, computationally complex, local optimums [5,6]. Finding solutions to optimization problems in complex situations is crucial [7]. Traditional optimization algorithms such as quasi-Newton methods, the steepest descent method and the gradient descent method, etc., require the objective function to meet strict conditions [8]. Therefore, the above methods are difficult to apply in practice [9] and some new methods should be proposed to solve complex optimization problems [10].

At present, evolutionary algorithms have been proposed to address complex optimization problems and have achieved satisfactory results [11,12]. Evolutionary algorithms simulate the evolution of natural species with self-organizing and adaptive characteristics, and do not require the objective function to meet strict conditions. Therefore, evolutionary algorithms are adopted to address complex optimization problems. As a type of evolutionary algorithm, swarm intelligence algorithms have received widespread attention. Examples include PSO [13], differential evolution (DE) [10], artificial bee colony optimization (ABC) [14], ant colony optimization (ACO) [15], etc. In swarm intelligence algorithms, a population can be seen as being composed of independent individuals, each of whom interacts to jointly search for the global optimum. Therefore, use of swarm intelligence

methods to solve complex optimization problems is feasible [16] and they have received widespread attention in recent times [17].

As a popular evolutionary algorithm, PSO has been extensively adopted in different engineering optimization problems since it was proposed in 1995. PSO simulates the foraging behavior of natural animals, and has the advantages of robustness, efficiency and precision. In PSO, a population is composed of some particles, and each particle has two learning samples, namely its own historical best solution (pbest) and a global best solution (gbest). In the process of evolution, particles in the whole population work together to seek gbest. Although PSO has advantages, it still has the problem of poor balance between global and local search. Recently, a large number of PSO variants have been proposed to address the above issue. These can be divided into three classes: adaptive learning parameter, novel updating strategy and new topology mechanism.

Adaptive learning parameter: Parameter setting is an important part of PSO. A fitness-based PSO algorithm (FMPSO) is developed in reference [18], in which different categories of particles have different parameters. In reference [19], the authors use sine and cosine parameters to increase algorithm diversity. In the literature [20], during the process of evolution, the authors use sigmoid function to realize dynamic parameters to make the algorithm search efficiently. Extensive experiments have proved the robustness of the algorithm. In the literature [21], the nonlinear attenuation acceleration coefficient is put forward. The robustness of an algorithm is enhanced by using different acceleration coefficients at different evolutionary stages of the population. In reference [22], inertial weight with chaotic mechanism is put forward. This is used to increase the diversity of the population. Similarly, in reference [23], the dynamic nonlinear inertia weight is adopted to enhance the anti-jamming capacity of the algorithm. Rosso et al. [24] propose an enhanced multi-strategy particle swarm optimization for constrained problems with an evolutionary-strategies-based unfeasible local search operator. This method determines the parameters' values that govern the evolutionary strategy simultaneously during the optimization process. Rosso et al. [25] propose a new constraint-handling approach for PSO, adopting a supervised classification machine learning method: the support vector machine (SVM).

Novel updating strategy: Updating the strategy is the most important part of PSO, as this determines the whole population's evolution direction. E et al. [26] introduce human social learning intelligence into the PSO algorithm to enhance its performance. In the literature [27], five different algorithms are integrated to form a new algorithm in which the algorithm with the best performance is selected to work through a rating function. In order to increase the diversity of learning samples, a comprehensive learning particle swarm optimization algorithm (CLPSO) is put forward [28] in which the entire population is updated based on excellent information. In reference [29], a novel dual population algorithm is proposed. In this approach, two sub-populations are devoted to conduct global and local searches, respectively. Extensive experiments have proved the effectiveness of the algorithm. Xia et al. [30] proposed a multi-learning sample algorithm in which different learning samples and update mechanisms collaborate to work. Li et al. [31] proposed an adaptive cooperative particle swarm optimization with difference learning. In this, performance accuracy can be enhanced by using different learning strategies. In reference [32], according to the spatial distribution, the population state can be obtained. Different learning mechanisms are used in various evolutionary states. Recently, researchers have incorporated other technologies into the PSO algorithm to enhance its performance [33,34].

New topology mechanism: Multiple swarm collaboration methods have been widely adopted recently. In the literature [35], a new multi-swarm interaction mechanism has been put forward (ADPSO). In ADPSO, the proposed strategies can be used to enhance the performance of traditional PSO. Yang and Li [36] combine evolutionary states with the collaborative mechanisms of the whole population for the first time. The experimental results have received praise from peers. In the literature [37], three learning features

are used to enhance the performance of the traditional method. As a famous variant of PSO, dynamic multi-swarm PSO variants achieve excellent results [38] and have gained widespread attention from peers. In the literature [39], a fully informed PSO is proposed (FIPS). In FIPS, each individual can be influenced not only by the global best solution, but also by the particles in its neighborhood. Lu et al. [40] let four sub-swarms work together to find global optimization. Xia et al. [9] put forward an adaptive multi-swarm particle swarm optimization algorithm (MSPSO). In MSPSO, the number of sub-populations can be changed according to population stage.

Although the existing research has made significant improvements to the PSO algorithm, there are still some problems that have not been effectively addressed. The evolutionary state of the whole population is an important indicator. Specifically, in the early stage the whole population should learn more diversity information to increase the global ability. In the later stage, the whole population should learn more accuracy information to increase the local ability. Although a large number of methods have emerged to calculate population spatial distribution to evaluate population evolutionary state, calculating population spatial distribution consumes lots of computing resources. The additional calculations far exceed the complexity of PSO and it is difficult to achieve efficient and fast work. Therefore, how to design a lightweight complexity evaluating population evolutionary state method while obtaining satisfactory results is of great significance. In addition, using different evolutionary strategies in various evolutionary states is worthy. Therefore, it is crucial to adjust the learning strategy adaptively.

Inspired by above discussions, an adaptive PSO with state-based learning strategy (APSO-SL) is put forward. In APSO-SL, two new features are proposed in PSO. The first strategy is the population distribution evaluation mechanism (PDEM), in which the population center position and best position are used for calculating the whole population state. In this way, the whole population state can be evaluated more intuitively and accurately without excessive computation. The second strategy is the adaptive learning strategy (ALS). In ALS, different learning strategies are adopted based on the population state, to ensure the whole population diversity. Specifically, if the population diversity is high, the method conducts a global search to increase the population diversity. Instead, if the population diversity is low, the method carries out local search to increase the population's local search ability. Lots of experiments have been conducted to confirm the effectiveness of APSO-SL, and the main contributions of APSO-SL are as follows:

- (1) Population distribution evaluation mechanism (PDEM) is put forward. In PDEM, instead of using iterations to just population state, APSO-SL only uses the population center position and best position for calculation. Thus, the whole population diversity can be improved without significantly increasing the calculation complexity. Our method can evaluate the population state accurately without increasing computational complexity.
- (2) Adaptive learning strategy (ALS) is put forward in this paper. In ALS, different learning strategies are used according to the population state, to ensure the whole population diversity. In this way, the whole population can achieve a balance between exploration and exploitation.
- (3) An efficient PSO variant, APSO-SL, is proposed, which combines PDEM and ALS, and outperforms 6 competitive optimization approaches. Therefore, APSO-SL is an efficient optimization algorithm.

The organizational framework of this study is as follows: In Section 2, we describe traditional PSO. Section 3 introduces our method: APSO-SL. In Section 4, we conduct ample experiments and discussions. Finally, the study is concluded and some future works are proposed in Section 5.

2. Particle Swarm Optimization (PSO)

PSO is an optimization algorithm with a certain degree of randomness; each particle stands for a result in the D-dimensional feasible domain space. N particles form

a population, and each particle in the population has position $[X_i^1(t), X_i^2(t), \dots, X_i^D(t)]$ and velocity $[V_i^1(t), V_i^2(t), \dots, V_i^D(t)]$. In each iteration, the learning sample of each particle consists of two parts: *pbest* and *gbest*. The update guidelines for PSO are shown in Equations (1) and (2):

$$V_i^d(t+1) = w \times V_i^d(t) + c_1 \times r_1 \times (pbest_i^d(t) - X_i^d(t)) + c_2 \times r_2 \times (gbest^d(t) - X_i^d(t)) \quad (1)$$

$$X_i^d(t+1) = X_i^d(t) + V_i^d(t+1) \quad (2)$$

where w is the inertia weight, d is the current dimension and r_1 and r_2 are two random numbers in the interval $[0, 1]$, c_1 and c_2 stand for the two acceleration constants.

Generally, PSO can be divided into two types: global version and local version. The aforementioned formulas are global PSO. In order to increase the diversity of particles, local optimum is used instead of global optimum. *lbest* stands for the best position in the particle's neighborhood. The velocity update strategy is described as follows:

$$V_i^d(t+1) = \omega \times V_i^d(t) + c_1 \times r_1 \times (pbest_i^d(t) - x_i^d(t)) + c_2 \times r_2 \times (lbest^d(t) - x_i^d(t)) \quad (3)$$

3. APSO-SL

In this part, the proposed APSO-SL is described in detail. Firstly, the population distribution evaluation mechanism (PDEM) is adopted to evaluate the population state. Secondly, the adaptive learning strategy (ALS) is used to achieve a balance between exploration and exploitation. Finally, APSO-SL is described in detail in Algorithm 1. The APSO-SL proposed in this paper includes these two improvements, and we also conduct a detailed analysis of APSO-SL in Section 4.

Algorithm 1. APSO-SL

01. Objective function: $f(x)$
 02. Input: $x_{min}^d, x_{max}^d, v_{min}^d, v_{max}^d, pbest, gbest, D, d, t, Tmax, r1, r2, U, L, w, c1, c2$;
 03. Output: Optimal solution;
 04. Initialization: $x_i^d = x_{min}^d + \text{rand} \times (x_{max}^d - x_{min}^d), v_i^d = x_{min}^d + \text{rand} \times (v_{max}^d - v_{min}^d)$;
 05. while ($t \leq Tmax$)
 06. for $i = 1:N$ do
 07. Calculate the center position of the population based on Equation (4);
 08. Judging the Euclidean distance in space using Equation (5);
 09. Determine the evolutionary state of the population based on Equation (6);
 10. Case 1:
 11. Update particles in the population using Equations (1) and (2);
 12. Case 2:
 13. Update particles in the population using Equations (7) and (2);
 14. Case 3:
 15. Keep the population updating method unchanged.
 16. $t = t + 1$;
 17. end for
 18. end while
-

3.1. Population Distribution Evaluation Mechanism (PDEM)

The evolutionary state of a population is an important indicator. Inspired by the diversity of natural species, different individuals possess different characteristics. As a result, use of different evolutionary strategies in various evolutionary states is worthy. In the initial stage, the whole population is suitable for conducting global search. In contrast, at the end stage the whole population is suitable for small-scale local search. Therefore, this is crucial to judge the population state.

In general, researchers use iterations to judge the evolutionary state. However, there are some problems when using this approach. In the initial iteration, the diversity may be low and suitable for exploitation. In the final iteration, the diversity of the population may

be high and suitable for global search. Recently, researchers have used the spatial distribution of a population to judge its evolutionary state. The population spatial distribution is the most intuitive indicator to evaluate its state. However, calculating the population spatial distribution requires a lot of calculation. The additional calculations far exceed the complexity of PSO, and it is difficult to achieve efficient and fast work. Therefore, the question of how to design a lightweight method of evaluating the population state while obtain satisfying result is of great significance. The calculation process of the algorithm is shown in Equations (4)–(6).

$$X_M^d = \frac{1}{N} \left(\sum_{i=1}^N x_i^d \right), \quad d = 1, \dots, D \quad (4)$$

$$dist_B = \sqrt{\sum_{j=1}^D (X_B^j - X_M^j)^2} \quad (5)$$

$$f_B = \frac{dist_B}{\sqrt{\sum_{j=1}^D (U_j - L_j)^2}} \quad (6)$$

In this section, the population distribution evaluation mechanism (PDEM) is proposed to evaluate the whole population state. Firstly, we calculate the center position of the whole population according to Equation (4). Secondly, the spatial Euclidean distance between the center and best positions is calculated, according to Equation (5). Finally, the population state can be determined based on the above spatial Euclidean distance, based on Equation (6). The whole population evolutionary state can be evaluated without significantly increasing the calculation complexity. Our method can evaluate the population state accurately without increasing computational complexity.

3.2. Adaptive Learning Strategy (ALS)

Based on the above discussion, APSO-SL can evaluate the evolutionary state of the population. We classify f_B into three states: case 1, case 2 and case 3. These represent the states of exploration, exploitation and balance, respectively, and we define the following three cases.

Case (1)—Exploration: In this condition, f_B is a relatively large value (e.g., larger than the threshold 0.4). Specifically, the best individual of the population is far from the center of the population, indicating that the population is currently relatively dispersed and suitable for global search. The proposed method defines this state as exploration state. In this case, exploring as many regions as possible and increasing the population diversity are beneficial to the population evolution. We use traditional PSO algorithms for a large-scale search, based on Equations (1) and (2).

Case (2)—Exploitation: In contrast, a small value of f_B (e.g., smaller than the threshold 0.3) signifies that the current population is in the same region, most of the individuals are around the best individual, and only a few individuals are distributed in other areas. The population distribution is relatively dense, and suitable for small-scale local search. This case is therefore likely to represent the exploitation state. The formula for the population in this case is as follows. Equation (7) consists of three parts, with the first two parts being the same as the basic PSO algorithm, representing the weights learned from inertial learning and the weights learned from one's own historical optimal values. The third item is the weight learned from sample e , where e is a randomly selected sample from the historical optimal values of all particles in the population.

$$V_i^d(t+1) = w \times V_i^d(t) + c_1 \times r_1 \times (pbest_i^d(t) - X_i^d(t)) + c_2 \times r_2 \times (e^d(t) - X_i^d(t)) \quad (7)$$

In formula, e is the new learning sample which is randomly selected from the historical optimum values of all particles in the whole population. In this way, the population can learn high-quality diversity information.

Case (3)—Balance: When fb is a relatively middle value (e.g., smaller than the threshold 0.4 and larger than the threshold 0.3), we maintain the learning strategy of the whole population unchanged to achieve a balance between exploration and exploitation. When the evolutionary state has been estimated, we can control the mutation strategy adaptively. Different learning strategies are used according to the population state, to ensure the whole population diversity. In this way, the whole population can achieve a balance between exploration and exploitation. The flowchart of the algorithm is shown in Figure 1. All input and output parameters have been explained in the corresponding sections of the text, and detailed explanations will not be provided here.

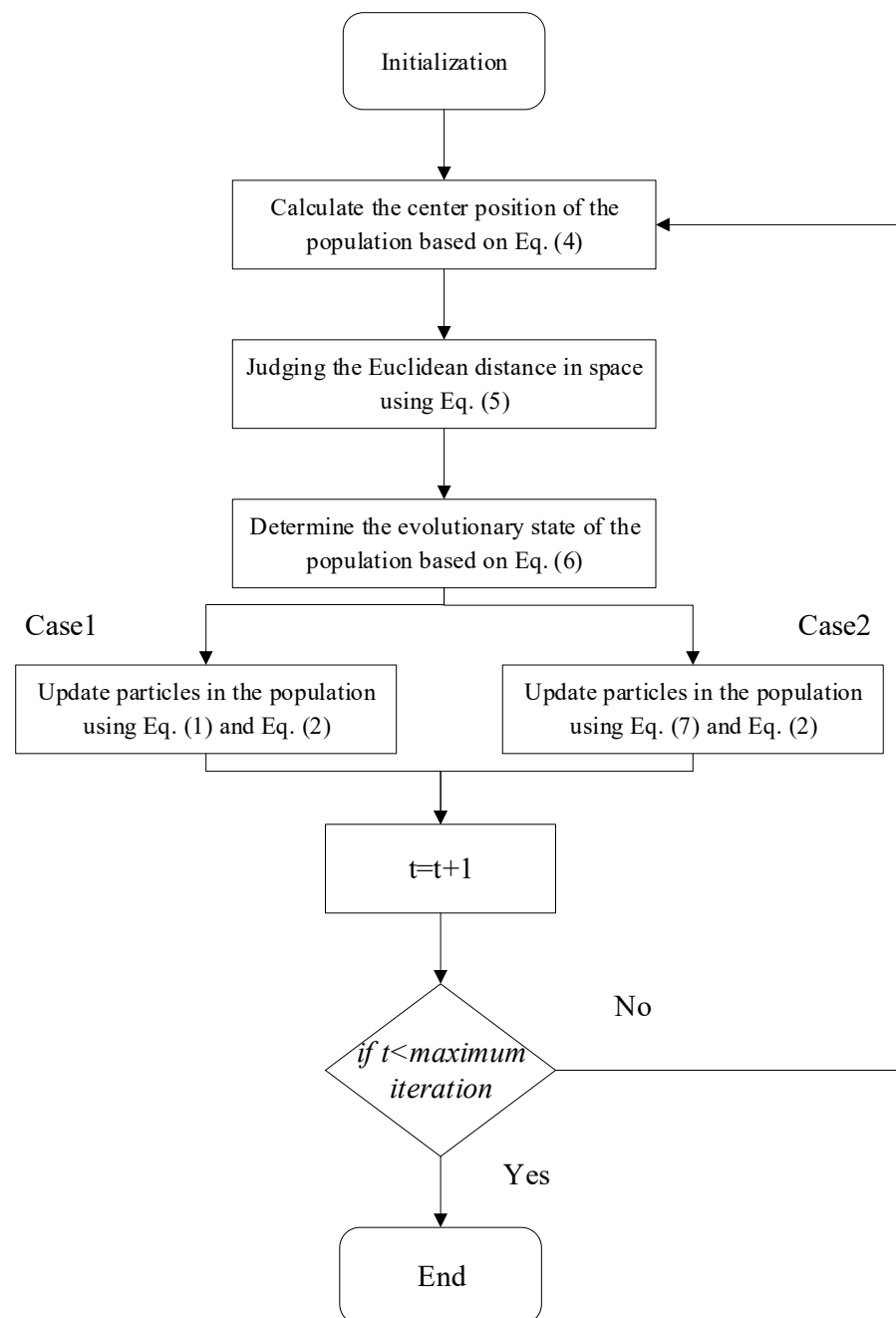


Figure 1. The overall flowchart of the proposed algorithm.

4. Experiments

4.1. Experimental Fundamentals

The APSO-SL is tested on CEC2013 [41,42] test suite, and one practical engineering problem. CEC2013 is a famous test suite with extensive applications, which are listed in Supplementary Materials. CEC2013 is an authoritative dataset in the field of single objective evolutionary computing which includes 28 benchmark functions. Therefore, in the field of evolutionary computing, the CEC2013 dataset is widely used as the platform for evaluating algorithm performance.

The proposed APSO-SL is compared with six competitive PSO variants. The first comparison method is traditional PSO [13]; we can compare APSO-SL and PSO visually. The second peer algorithm is HCLPSO [29], in which two populations collaborate to find the optimum solution. TAPSO [30] is the third peer method, which uses three different strategies to update the population. The fourth method is CLPSO [28], which uses a novel comprehensive learning manner to work. The fifth peer algorithm is DMS-PSO [38], in which multiple populations are used to work together. EPSO [27] is the sixth comparison algorithm, in which five classic algorithms are integrated together, and the best method is selected based on scores during each iteration. The parameter settings of all comparison algorithms are shown in Table 1.

Table 1. Parameters for six peer methods and APSO-SL.

Algorithm	Year	Parameters Settings
PSO [13]	1995	$w = 0.729, c1 = c2 = 1.49445$
HCLPSO [29]	2015	$w = 0.729, c = 3-1.5, c1 = 2.5-0.5, c2 = 0.5-2.5, a = 0, b = 0.25$
TAPSO [30]	2020	$w = 0.7298, pc = 0.5, pm = 0.01, sg = 7$
CLPSO [28]	2006	$w = 0.9-0.2, c = 1.49445, G = 5$
DMS-PSO [38]	2006	$w = 0.729, c1 = c2 = 1.49445$
EPSO [27]	2017	ensemble wPSO, CLPSO, FDRPSO, HPSO-TVAC and LIPS
APSO-SL	2023	$w = 0.729, c1 = c2 = 1.49445$

4.2. Experimental Analysis on CEC2013

To test the effect of APSO-SL, extensive experiments are conducted on CEC2013 and CEC2017 test suites, and the solutions are shown in Tables 2 and 3, where the best results are expressed in bold.

4.2.1. Accuracy Analysis

For the five unimodal functions, the solutions in Tables 2 and 3 show that APSO-SL obtains excellent results in both 30-D and 50-D conditions, followed by TAPSO and HCLPSO. Although HCLPSO does not achieve the best results on a certain function, its overall performance is excellent. For the 15 multimodal functions, APSO-SL gets the best solutions in 30-D conditions, because it gets best solutions on 11 of 15 conditions. In 50-D condition, APSO-SL also obtains excellent performance, which obtains best solutions on 8 of 15 multimodal functions. In addition, TAPSO and HCLPSO have also achieved preferable results in solving multimodal functions. For the eight composition functions, we can see from Tables 2 and 3 that APSO-SL achieves remarkable results in 30-D and 50-D conditions. In conclusion, population distribution evaluation mechanism (PDEM) and adaptive learning strategy (ALS) are effective. In addition, we use box-plot charts to evaluate the capabilities of all comparison methods. Due to the page limit, only some typical test functions are selected for comparison. The performances of APSO-SL and some excellent peer algorithms are shown in Figure 2. According to observation, the box-plot charts show that APSO-SL exhibited a more stable search performance in all conditions.

Table 2. The result of all methods on CEC2013 (D = 30).

	PSO	CLPSO	EPSO	HCLPSO	DMS-PSO	TAPSO	APSO-SL
F1	1.20×10^2	2.20×10^{-13}	2.24×10^{-13}	2.69×10^{-13}	1.72×10^{-13}	3.77×10^{-15}	0.00×10^0
	1.88×10^2	4.21×10^{-13}	4.33×10^{-13}	1.20×10^{-13}	1.09×10^{-13}	1.62×10^{-13}	0.00×10^0
F2	1.32×10^7	1.30×10^6	2.38×10^5	8.69×10^5	1.31×10^6	1.40×10^6	1.25×10^5
	1.28×10^7	7.26×10^5	1.35×10^5	3.79×10^5	6.20×10^5	2.88×10^6	7.85×10^4
F3	2.44×10^8	4.51×10^7	1.16×10^8	2.90×10^7	2.38×10^7	1.99×10^6	7.95×10^6
	2.11×10^8	4.22×10^7	1.39×10^8	2.80×10^7	3.92×10^7	1.85×10^6	1.21×10^6
F4	2.41×10^2	5.45×10^3	3.49×10^2	1.75×10^3	4.40×10^3	1.71×10^2	1.10×10^2
	2.22×10^2	2.52×10^3	2.44×10^2	1.31×10^3	1.54×10^3	1.18×10^2	1.02×10^2
F5	7.11×10^{-13}	3.20×10^{-13}	2.50×10^{-13}	3.99×10^{-13}	1.31×10^{-13}	4.19×10^{-13}	3.79×10^{-13}
	2.45×10^{-12}	8.68×10^{-13}	5.69×10^{-13}	8.44×10^{-13}	2.09×10^{-13}	1.21×10^{-13}	1.02×10^{-13}
F6	4.49×10^1	2.55×10^1	1.51×10^1	1.70×10^1	2.11×10^1	2.66×10^1	7.88×10^{-1}
	2.69×10^1	2.29×10^1	5.63×10^0	2.41×10^0	2.10×10^1	2.45×10^1	2.23×10^0
F7	6.11×10^1	3.77×10^1	3.72×10^1	2.21×10^1	1.11×10^1	3.84×10^0	1.86×10^1
	2.22×10^1	1.66×10^1	1.77×10^1	7.66×10^0	5.92×10^0	2.81×10^0	4.72×10^0
F8	2.09×10^1	2.09×10^1	2.09×10^1	2.09×10^1	2.09×10^1	2.09×10^1	2.09×10^1
	5.78×10^{-2}	5.61×10^{-2}	6.44×10^{-2}	2.62×10^{-2}	4.50×10^{-2}	2.38×10^{-2}	0.00×10^0
F9	2.49×10^1	2.20×10^1	2.51×10^1	1.81×10^1	2.41×10^1	1.77×10^1	1.44×10^1
	4.11×10^0	4.55×10^0	3.40×10^0	3.62×10^0	3.20×10^0	2.32×10^1	4.26×10^0
F10	3.77×10^{-1}	1.82×10^{-1}	2.11×10^{-1}	2.45×10^{-1}	2.62×10^{-1}	1.20×10^{-1}	6.17×10^{-2}
	3.44×10^{-1}	8.43×10^{-2}	1.20×10^{-1}	1.23×10^{-1}	1.45×10^{-1}	4.88×10^{-2}	3.77×10^{-2}
F11	3.31×10^1	2.53×10^1	3.29×10^{-2}	3.95×10^{-2}	3.33×10^1	1.16×10^1	1.34×10^1
	1.44×10^1	6.34×10^0	1.82×10^{-1}	1.86×10^{-1}	1.34×10^1	3.66×10^0	5.35×10^0
F12	8.20×10^1	7.32×10^1	7.31×10^1	6.33×10^1	3.41×10^1	3.92×10^1	3.13×10^1
	2.49×10^1	2.22×10^1	2.35×10^1	1.87×10^1	9.76×10^0	1.90×10^1	1.23×10^1
F13	1.70×10^2	1.31×10^2	1.21×10^2	1.29×10^2	5.48×10^1	1.22×10^2	7.08×10^1
	3.15×10^1	3.11×10^1	3.79×10^1	2.99×10^1	1.67×10^1	5.11×10^1	2.58×10^1
F14	1.61×10^3	1.19×10^3	1.66×10^1	1.75×10^1	2.42×10^3	1.31×10^3	1.49×10^1
	3.33×10^2	2.38×10^2	2.84×10^1	4.25×10^1	4.35×10^2	4.66×10^2	9.94×10^0
F15	3.70×10^3	3.81×10^3	3.62×10^3	3.58×10^3	3.38×10^3	3.23×10^3	2.99×10^3
	6.22×10^2	9.71×10^2	4.55×10^2	5.71×10^2	2.59×10^2	1.78×10^3	5.02×10^2
F16	1.70×10^0	1.24×10^0	1.51×10^0	1.42×10^0	1.31×10^0	1.29×10^0	1.18×10^0
	3.40×10^{-1}	4.75×10^{-1}	2.81×10^{-1}	2.51×10^{-1}	1.77×10^{-1}	2.62×10^{-1}	1.18×10^{-1}
F17	7.24×10^1	6.81×10^1	3.51×10^1	3.18×10^1	6.31×10^1	6.16×10^1	2.55×10^1
	1.66×10^1	1.33×10^1	1.78×10^0	1.90×10^1	9.96×10^0	4.55×10^1	1.33×10^0
F18	9.29×10^1	7.66×10^1	1.34×10^2	8.45×10^1	9.11×10^1	1.96×10^2	7.35×10^1
	2.15×10^1	1.51×10^1	5.59×10^1	1.75×10^1	7.49×10^0	8.66×10^0	2.71×10^0
F19	3.95×10^0	3.29×10^0	1.96×10^0	1.71×10^0	3.19×10^0	2.45×10^0	2.13×10^0
	1.67×10^0	7.41×10^{-1}	3.51×10^{-1}	2.66×10^{-1}	7.60×10^{-1}	5.11×10^{-1}	3.00×10^{-1}
F20	1.37×10^1	1.40×10^1	1.21×10^1	1.12×10^1	1.19×10^1	1.51×10^1	8.99×10^0
	7.55×10^{-1}	1.22×10^0	1.76×10^0	8.85×10^{-1}	5.77×10^{-1}	1.30×10^0	4.77×10^{-1}
F21	8.24×10^2	2.95×10^2	2.31×10^2	2.42×10^2	2.91×10^2	2.18×10^2	2.09×10^2
	4.62×10^2	9.38×10^1	3.72×10^1	4.67×10^1	7.77×10^1	8.22×10^1	7.22×10^1
F22	1.52×10^3	8.45×10^2	1.21×10^2	1.08×10^2	1.90×10^3	9.11×10^1	8.98×10^1
	4.90×10^2	2.19×10^2	5.61×10^1	2.55×10^1	5.42×10^2	2.88×10^1	7.44×10^0
F23	4.62×10^3	4.11×10^3	4.24×10^3	4.23×10^3	3.56×10^3	4.11×10^3	3.32×10^3
	8.51×10^2	6.71×10^2	5.59×10^2	5.66×10^2	2.58×10^2	2.20×10^3	2.51×10^2
F24	2.79×10^2	2.49×10^2	2.56×10^2	2.51×10^2	2.19×10^2	2.20×10^2	2.28×10^2
	1.11×10^1	1.42×10^1	1.22×10^1	9.77×10^0	8.22×10^0	1.69×10^1	8.72×10^0
F25	3.23×10^2	2.78×10^2	2.88×10^2	2.83×10^2	2.71×10^2	2.59×10^2	2.44×10^2
	1.20×10^1	1.30×10^1	7.51×10^0	1.52×10^1	1.17×10^1	7.22×10^0	6.11×10^0
F26	2.50×10^2	2.31×10^2	2.34×10^2	2.20×10^2	2.33×10^2	2.11×10^2	2.00×10^2
	7.41×10^1	4.90×10^1	2.80×10^1	2.55×10^{-2}	4.55×10^1	4.91×10^1	2.11×10^{-3}
F27	8.51×10^2	8.15×10^2	8.09×10^2	5.44×10^2	5.43×10^2	4.72×10^2	4.95×10^2
	9.22×10^1	1.19×10^2	1.50×10^2	8.99×10^1	8.85×10^1	1.35×10^2	9.22×10^1
F28	4.41×10^2	3.45×10^2	2.96×10^2	3.11×10^2	3.00×10^2	2.97×10^2	3.00×10^2
	5.23×10^2	4.52×10^2	3.55×10^1	2.22×10^{-13}	2.79×10^{-13}	4.61×10^1	1.55×10^{-13}

Table 3. The result of all methods on CEC2013 (D = 50).

	PSO	CLPSO	EPSO	HCLPSO	DMS-PSO	TAPSO	APSO-SL
F1	3.41×10^2	5.42×10^{-13}	3.29×10^{-13}	6.51×10^{-13}	2.42×10^{-13}	2.22×10^{-13}	0.00×10^0
	6.32×10^2	1.42×10^{-13}	1.32×10^{-13}	1.27×10^{-13}	0.00×10^0	6.23×10^{-13}	0.00×10^0
F2	2.60×10^7	2.61×10^6	6.77×10^5	1.71×10^6	1.51×10^6	1.31×10^7	3.22×10^5
	2.51×10^7	8.69×10^5	2.66×10^5	4.82×10^5	5.09×10^5	6.70×10^6	1.23×10^5
F3	5.32×10^8	7.80×10^8	4.61×10^8	2.69×10^8	1.20×10^8	3.69×10^7	1.86×10^7
	4.60×10^8	1.35×10^9	4.59×10^8	2.32×10^8	1.40×10^8	6.41×10^7	1.27×10^6
F4	7.51×10^2	1.31×10^4	2.11×10^2	1.70×10^3	7.49×10^3	6.55×10^2	1.86×10^1
	8.66×10^2	3.62×10^3	8.51×10^1	5.91×10^2	2.56×10^3	1.59×10^2	7.77×10^0
F5	1.39×10^{-10}	7.73×10^{-13}	5.37×10^{-13}	7.66×10^{-13}	2.18×10^{-13}	3.21×10^{-13}	1.18×10^{-13}
	4.41×10^{-10}	2.73×10^{-13}	1.33×10^{-13}	1.42×10^{-13}	4.71×10^{-13}	1.88×10^{-13}	4.30×10^{-13}
F6	6.21×10^1	4.50×10^1	4.41×10^1	4.50×10^1	5.37×10^1	4.20×10^1	4.42×10^1
	2.51×10^1	1.97×10^0	1.22×10^0	6.34×10^{-1}	2.33×10^1	4.32×10^0	8.19×10^{-1}
F7	7.48×10^1	7.20×10^1	6.29×10^1	3.92×10^1	3.11×10^1	2.77×10^1	3.20×10^1
	1.62×10^1	2.51×10^1	1.80×10^1	9.95×10^0	8.37×10^0	6.56×10^0	5.41×10^0
F8	2.11×10^1	2.11×10^1	2.11×10^1	2.11×10^1	2.11×10^1	2.11×10^1	2.11×10^1
	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0	0.00×10^0
F9	5.11×10^1	4.52×10^1	4.78×10^1	4.31×10^1	4.43×10^1	2.60×10^1	2.11×10^1
	5.55×10^0	5.66×10^0	4.56×10^0	5.77×10^0	4.33×10^0	4.33×10^0	3.08×10^0
F10	2.20×10^2	2.31×10^{-1}	2.30×10^{-1}	2.31×10^{-1}	2.82×10^{-1}	7.70×10^{-2}	4.21×10^{-2}
	3.71×10^2	1.39×10^{-1}	1.30×10^{-1}	1.46×10^{-1}	1.41×10^{-1}	4.23×10^{-2}	1.91×10^{-2}
F11	6.81×10^1	7.51×10^1	9.82×10^{-2}	2.33×10^{-13}	8.61×10^1	6.33×10^1	2.36×10^1
	2.31×10^1	1.77×10^1	4.11×10^{-1}	6.61×10^{-13}	1.86×10^1	1.23×10^1	6.16×10^0
F12	2.22×10^2	1.49×10^2	1.64×10^2	1.25×10^2	7.66×10^1	1.34×10^2	5.69×10^1
	5.51×10^1	5.27×10^1	4.72×10^1	2.88×10^1	1.86×10^1	4.11×10^1	1.24×10^1
F13	3.50×10^2	2.92×10^2	2.97×10^2	2.57×10^2	1.66×10^2	2.11×10^2	1.37×10^2
	5.11×10^1	4.49×10^1	6.51×10^1	6.88×10^1	2.99×10^1	4.76×10^1	4.95×10^1
F14	2.70×10^3	2.29×10^3	3.42×10^1	1.88×10^2	4.77×10^3	2.33×10^3	2.05×10^2
	6.26×10^2	5.11×10^2	5.25×10^1	1.34×10^2	7.60×10^2	4.55×10^2	1.92×10^1
F15	7.71×10^3	7.80×10^3	7.58×10^3	7.24×10^3	7.23×10^3	6.64×10^3	6.82×10^3
	9.66×10^2	1.33×10^3	6.35×10^2	7.11×10^2	4.09×10^2	2.57×10^3	1.25×10^2
F16	2.60×10^0	1.53×10^0	1.87×10^0	1.71×10^0	1.49×10^0	3.34×10^0	1.49×10^0
	5.45×10^{-1}	3.11×10^{-1}	3.34×10^{-1}	3.51×10^{-1}	2.22×10^{-1}	2.31×10^{-1}	4.36×10^{-1}
F17	1.50×10^2	1.52×10^2	5.78×10^1	5.30×10^1	1.19×10^2	1.41×10^2	8.40×10^1
	2.55×10^1	2.29×10^1	3.64×10^0	1.43×10^{-1}	2.33×10^1	1.55×10^1	6.49×10^0
F18	1.90×10^2	1.62×10^2	2.23×10^2	1.78×10^2	1.82×10^2	3.29×10^2	1.35×10^2
	3.51×10^1	2.73×10^1	1.30×10^2	3.33×10^1	1.50×10^1	6.29×10^1	3.14×10^1
F19	8.90×10^0	7.55×10^0	3.51×10^0	2.40×10^0	6.85×10^0	5.29×10^0	5.20×10^0
	3.14×10^0	1.88×10^0	6.90×10^{-1}	3.95×10^{-1}	1.96×10^0	1.42×10^0	8.41×10^{-1}
F20	2.45×10^1	2.23×10^1	1.90×10^1	2.11×10^1	1.88×10^1	2.12×10^1	1.88×10^1
	8.90×10^{-1}	1.39×10^0	1.09×10^0	8.23×10^{-1}	6.77×10^{-1}	7.65×10^{-1}	2.30×10^{-1}
F21	2.09×10^2	7.70×10^2	3.38×10^2	2.31×10^2	7.31×10^2	7.74×10^2	5.72×10^2
	6.31×10^2	3.55×10^2	2.61×10^2	6.39×10^1	4.20×10^2	1.66×10^2	5.17×10^2
F22	3.22×10^3	2.83×10^3	7.69×10^1	3.55×10^1	5.58×10^3	2.88×10^3	2.32×10^3
	6.77×10^2	6.33×10^2	6.66×10^1	3.70×10^1	9.11×10^2	4.55×10^2	2.15×10^2
F23	9.11×10^3	7.61×10^3	8.70×10^3	8.29×10^3	7.80×10^3	8.11×10^3	7.11×10^3
	1.35×10^3	1.30×10^3	1.41×10^3	1.33×10^3	6.41×10^2	1.22×10^3	1.16×10^2
F24	3.38×10^2	3.33×10^2	3.11×10^2	2.83×10^2	2.70×10^2	3.11×10^2	2.33×10^2
	1.41×10^1	1.71×10^1	1.61×10^1	1.44×10^1	1.74×10^1	1.66×10^1	1.19×10^1
F25	3.77×10^2	3.61×10^2	3.72×10^2	3.51×10^2	3.50×10^2	3.00×10^2	1.88×10^2
	2.22×10^1	1.51×10^1	1.09×10^1	1.72×10^1	1.42×10^1	1.41×10^1	6.66×10^0
F26	3.33×10^2	3.29×10^2	2.05×10^2	2.12×10^2	3.41×10^2	3.49×10^2	1.51×10^2
	1.29×10^2	1.23×10^2	2.66×10^{-2}	5.16×10^{-2}	7.20×10^1	4.44×10^1	6.46×10^0
F27	1.51×10^3	1.44×10^3	1.66×10^3	1.42×10^3	1.18×10^3	1.30×10^3	6.88×10^2
	1.41×10^2	1.76×10^2	2.20×10^2	1.81×10^2	1.72×10^2	8.31×10^1	5.41×10^1
F28	1.49×10^3	5.55×10^2	4.00×10^2	4.00×10^2	4.00×10^2	4.00×10^2	4.00×10^2
	1.51×10^3	1.22×10^3	3.21×10^{-12}	5.11×10^{-13}	0.00×10^0	0.00×10^0	0.00×10^0

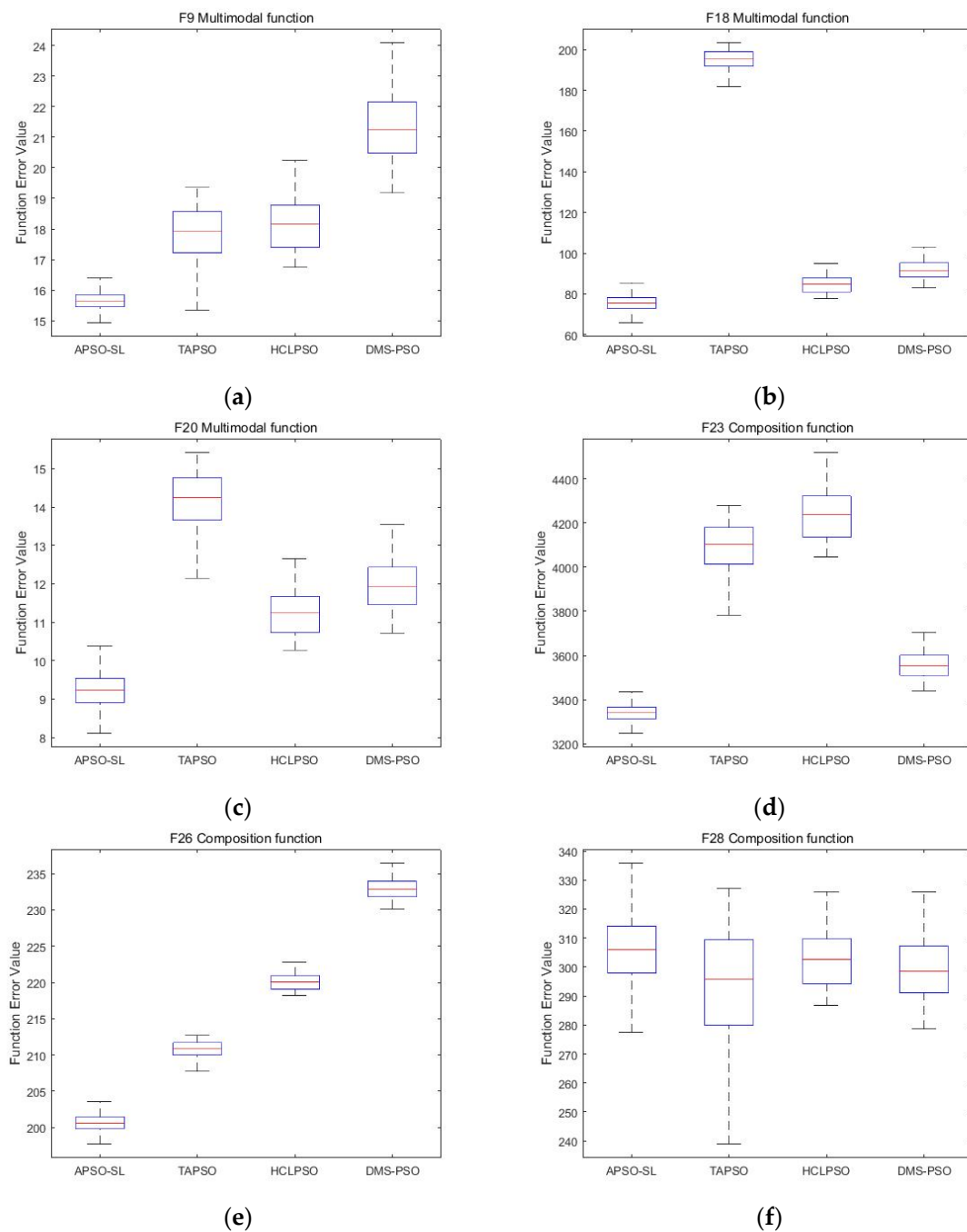


Figure 2. Box-plot figures of peer methods on CEC2013. (a) f09; (b) f18; (c) f20; (d) f23; (e) f26; (f) f28.

4.2.2. Convergence Analysis

In this part, we compare APSO-SL with three competitive peer methods (TAPSO, HCLPSO and DMS-PSO), and the experimental results are displayed in Figure 3. The abscissa is fitness evaluation ($100 \times D$), and the ordinate is the difference between the solution acquired by the peer method and the actual value. To be fair, each method is worked 30 times independently. Due to space limitations, only three multimodal functions (f9, f18, f20) and three complex functions (f23, f26, f28) are selected for analysis and discussion in this section. Through the results in Figure 2, we find APSO-SL has an excellent convergence performance in most functions (f9, f18, f20, f23, f26). In function f28, the convergence character of APSO-SL is the same as DMS-PSO and HCLPSO, but worse than TAPSO.

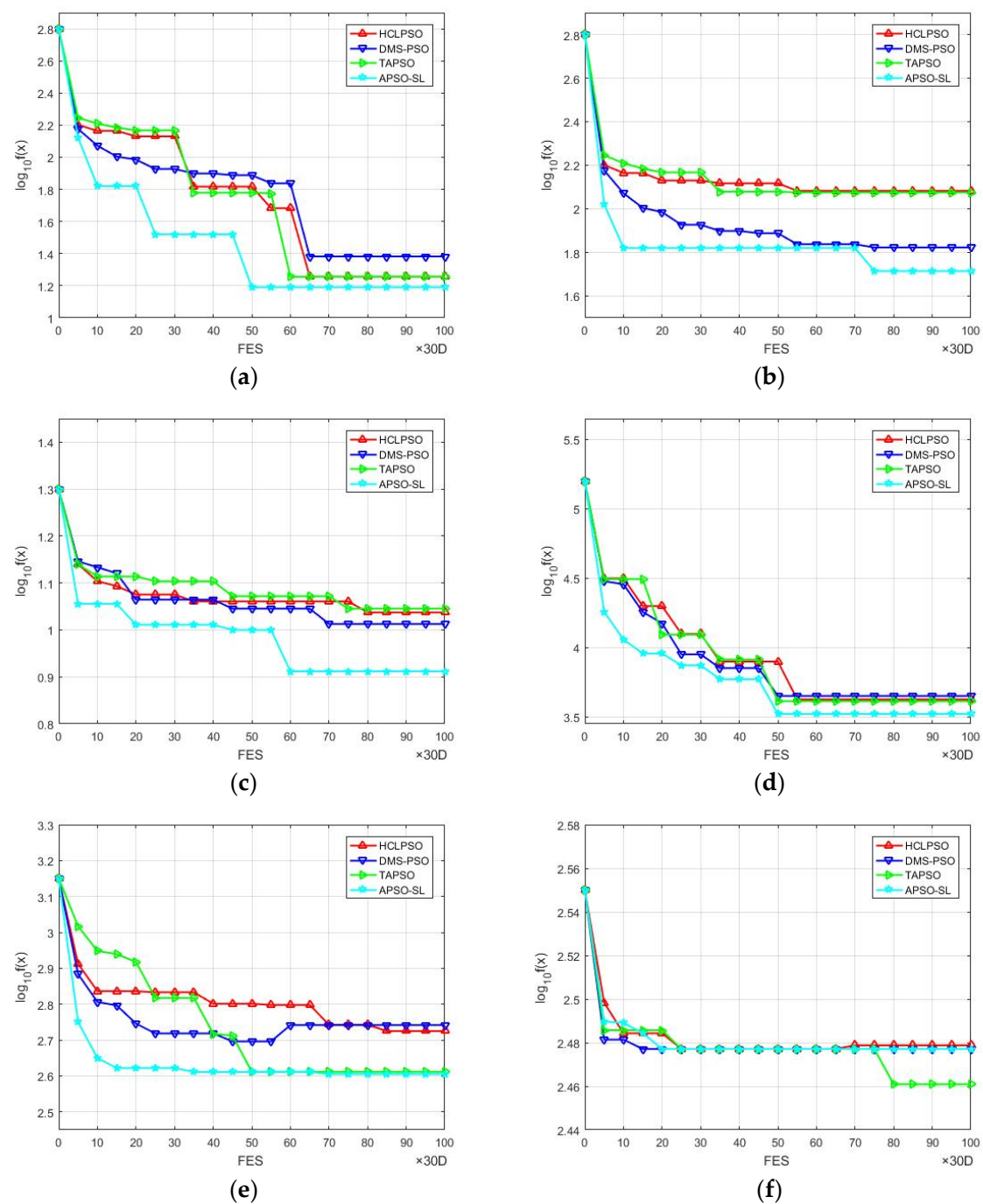


Figure 3. Convergence characteristic of APSO-SL and other methods. (a) f09; (b) f18; (c) f20; (d) f23; (e) f26; (f) f28.

4.2.3. Statistical Analysis

Statistical analysis methods are an important part of data processing. We use the Friedman test to summarize the accuracy of information in all peer algorithms in Tables 2 and 3, and the experimental results are displayed in Table 4. Through the results in Table 4, we find APSO-SL achieves the best performance in all conditions in both dimensions. Hence, APSO-SL has obvious advantages.

Table 4. Friedman test of all compared algorithms on CEC2013 test suite.

Average Rank	All Method	Ranking	30-D Method	Ranking	50-D Method	Ranking
1	APSO-SL	1.74	APSO-SL	1.55	APSO-SL	1.93
2	TAPSO	3.52	TAPSO	3.32	HCLPSO	3.54
3	HCLPSO	3.70	HCLPSO	3.86	TAPSO	3.71
4	DMS-PSO	3.96	DMS-PSO	3.98	EPSO	3.89
5	EPSO	4.04	EPSO	4.14	DMS-PSO	3.93
6	CLPSO	4.88	CLPSO	4.86	CLPSO	4.89
7	PSO	6.20	PSO	6.29	PSO	6.11

4.3. Results on CEC2017

To test the proposed APSO-SL overall, CEC2017 has been used to evaluate all comparison methods performance. In order to reduce the length, only 30-D is selected in this part. Table 5 shows that APSO-SL achieves the best results in unimodal conditions. For seven multimodal functions, APSO-SL gets the best solutions on f5, f6, f8 and f9, in four of the total seven conditions, followed by EPSO and DMS-PSO. In hybrid and composition conditions, APSO-SL obtains preferable results. Experimental results show that APSO-SL performs competitively on CEC2017 test suite.

Table 5. The result of all methods on CEC2017 (D = 30).

		PSO	CLPSO	EPSO	HCLPSO	DMS-PSO	TAPSO	APSO-SL
F1	Mean	4.86×10^7	3.29×10^3	1.20×10^2	8.77×10^1	2.82×10^3	1.92×10^3	1.34×10^{-1}
F3	Mean	4.79×10^1	2.09×10^2	5.57×10^{-11}	7.33×10^{-4}	1.72×10^1	3.33×10^{-2}	1.13×10^0
F4	Mean	6.77×10^1	7.41×10^1	2.10×10^1	7.46×10^1	5.39×10^1	1.22×10^2	3.26×10^1
F5	Mean	6.69×10^1	5.78×10^1	4.41×10^1	3.77×10^1	2.92×10^1	7.69×10^1	2.07×10^1
F6	Mean	2.71×10^{-1}	6.69×10^{-4}	4.33×10^{-13}	3.77×10^{-13}	3.34×10^{-4}	5.24×10^0	2.61×10^{-13}
F7	Mean	1.20×10^2	9.48×10^1	7.39×10^1	8.39×10^1	5.88×10^1	1.19×10^2	9.11×10^1
F8	Mean	6.71×10^1	5.55×10^1	5.01×10^1	4.33×10^1	2.77×10^1	8.67×10^1	1.94×10^1
F9	Mean	6.21×10^1	2.52×10^1	2.31×10^1	1.20×10^1	1.38×10^0	5.77×10^2	0.00×10^0
F10	Mean	3.52×10^3	2.77×10^3	1.58×10^3	2.11×10^3	2.74×10^3	2.69×10^3	2.46×10^3
F11	Mean	7.41×10^1	4.66×10^1	4.75×10^1	5.55×10^1	2.66×10^1	1.20×10^1	9.11×10^1
F12	Mean	1.41×10^6	3.33×10^4	2.50×10^4	3.33×10^4	2.97×10^4	2.40×10^4	4.20×10^3
F13	Mean	1.47×10^4	1.58×10^4	1.34×10^3	4.22×10^2	8.20×10^3	6.54×10^2	4.20×10^2
F14	Mean	6.32×10^2	1.51×10^4	3.70×10^3	5.83×10^3	2.66×10^3	2.45×10^3	1.94×10^3
F15	Mean	7.76×10^2	5.77×10^3	6.20×10^2	2.35×10^2	3.90×10^3	6.81×10^2	5.11×10^2
F16	Mean	8.75×10^2	7.67×10^2	6.34×10^2	4.98×10^2	3.00×10^2	3.11×10^2	1.20×10^2
F17	Mean	3.24×10^2	1.86×10^2	1.48×10^2	1.19×10^2	7.48×10^1	4.19×10^2	6.02×10^1
F18	Mean	3.19×10^4	1.44×10^5	1.32×10^5	8.69×10^4	1.19×10^5	1.66×10^4	3.16×10^3
F19	Mean	4.88×10^3	9.41×10^3	8.30×10^2	1.44×10^2	6.11×10^3	1.23×10^2	3.77×10^3
F20	Mean	2.66×10^2	2.77×10^2	2.04×10^2	1.61×10^2	1.58×10^2	5.16×10^1	1.01×10^2
F21	Mean	2.71×10^2	2.77×10^2	2.50×10^2	2.45×10^2	2.76×10^2	2.99×10^2	2.40×10^2
F22	Mean	1.22×10^2	2.22×10^2	1.99×10^2	1.23×10^2	1.22×10^2	1.11×10^2	1.00×10^2
F23	Mean	4.38×10^2	4.31×10^2	3.90×10^2	3.66×10^2	3.67×10^2	3.68×10^2	3.46×10^2
F24	Mean	4.1×10^2	4.92×10^2	4.55×10^2	4.71×10^2	4.52×10^2	4.41×10^2	4.20×10^2
F25	Mean	4.23×10^2	3.99×10^2	3.99×10^2	3.91×10^2	3.99×10^2	4.22×10^2	7.33×10^2
F26	Mean	4.17×10^3	2.55×10^3	4.22×10^2	3.11×10^2	1.41×10^3	2.18×10^3	1.93×10^3
F27	Mean	4.51×10^2	4.44×10^2	4.54×10^2	4.35×10^2	4.44×10^2	4.22×10^2	3.89×10^2
F28	Mean	4.22×10^2	3.49×10^2	3.08×10^2	3.48×10^2	3.50×10^2	3.70×10^2	3.00×10^2
F29	Mean	6.70×10^2	6.61×10^2	5.55×10^2	5.77×10^2	5.64×10^2	3.83×10^2	2.87×10^2
F30	Mean	3.88×10^3	5.66×10^3	3.67×10^3	3.98×10^3	7.98×10^3	3.69×10^2	2.12×10^3

4.4. APSO-SL for Engineering Application Problem

The optimization problem of tension springs is a classic problem in the field of engineering. The purpose is to find the minimum weight [36] based on material diameter d (x_1), average winding diameter D (x_2) and turns N (x_3).

Its schematic diagram is shown in Figure 4 and the principle of the formula is shown in Equation (8). The experimental results of APSO-SL and six peer algorithms are shown in Table 6 and the results reveal that APSO-SL is competitive.

$$\min f(x) = (x_3 + 2)x_2x_1^2$$

$$s.t. \begin{cases} g_1(x) = 1 - \frac{x_2^3x_3}{71785x_1^4} \leq 0 \\ g_2(x) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} \leq 0 \\ g_3(x) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0 \\ g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \leq 0 \end{cases} \quad (8)$$

where $0.05 \leq x_1 \leq 2$, $0.25 \leq x_2 \leq 1.30$, $2.00 \leq x_3 \leq 15$.

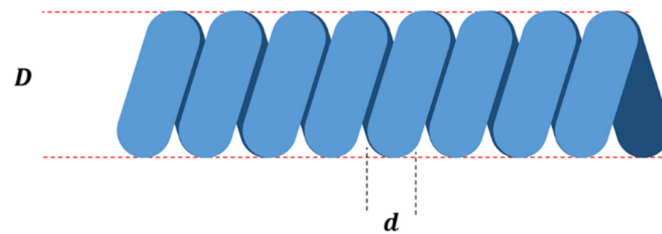


Figure 4. Tension spring optimization problem.

Table 6. Experimental results on engineering issues.

	Optimal Result			Optimal Cost
	x_1	x_2	x_3	
PSO	0.051677	0.3567355	11.28898	0.012671
CLPSO	0.051799	0.3615000	11.00000	0.012655
TAPSO	0.051672	0.3567162	11.28855	0.012677
DMS-PSO	0.053017	0.3895322	9.600166	0.012701
HCLPSO	0.051689	0.3567160	11.28901	0.012665
EPSO	0.051709	0.3571073	11.27082	0.012672
APSO-SL	0.050010	0.3499867	11.84687	0.012233

5. Conclusions

We propose an adaptive PSO with state-based learning strategy (APSO-SL). In PDEM, the population center position and best position are used for calculating the whole population state. In this way, the whole population state can be evaluated more intuitively and accurately without excessive computation. In the second strategy, ALS, different learning strategies are adopted based on the population state to ensure the whole population diversity. Specifically, if the population diversity is high, we conduct a global search. If the population diversity is low, we carry out a local search.

We can draw some conclusions through experimental analysis. First, PDEM can be used to evaluate the population state more intuitively and accurately. Second, ALS can be used to achieve a balance between global and local search. Therefore, the PDEM and ALS strategies proposed in this article can effectively improve the diversity and purposefulness of the population, have good universality, and can be widely promoted. In future work, we can investigate the effectiveness of PDEM and ALS strategies in solving large-scale problems and expensive optimization problems.

However, PDEM and ALS still have limitations, which lacks theoretical proof and sufficient experimentation. In the following work, in terms of methods, we should design more efficient mechanisms. For the applications, we should apply our method to more engineering applications. In addition, we should explore the performance of the proposed method in solving large-scale optimization, expensive optimization, robust optimization, multitasking optimization and dynamic optimization problems.

Supplementary Materials: The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/pr12020400/s1>.

Author Contributions: Conceptualization, X.Y.; Software, X.Y.; Investigation, X.Y.; Resources, M.G.; Writing—original draft, M.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article and supplementary materials.

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

References

1. Yuan, Q.; Sun, R.; Du, X. Path Planning of Mobile Robots Based on an Improved Particle Swarm Optimization Algorithm. *Processes* **2022**, *11*, 26. [CrossRef]
2. Yang, X.; Li, H. Multi-sample learning particle swarm optimization with adaptive crossover operation. *Math. Comput. Simul.* **2023**, *208*, 246–282. [CrossRef]
3. Ali, Y.A.; Awwad, E.M.; Al-Razgan, M.; Maarouf, A. Hyperparameter Search for Machine Learning Algorithms for Optimizing the Computational Complexity. *Processes* **2023**, *11*, 349. [CrossRef]
4. Azrag, M.A.K.; Zain, J.M.; Kadir, T.A.A.; Yusoff, M.; Jaber, A.S.; Abdlrhman, H.S.M.; Ahmed, Y.H.Z.; Husain, M.S.B. Estimation of Small-Scale Kinetic Parameters of *Escherichia coli* (*E. coli*) Model by Enhanced Segment Particle Swarm Optimization Algorithm ESe-PSO. *Processes* **2023**, *11*, 126. [CrossRef]
5. Chen, H.; Cheng, R.; Pedrycz, W.; Jin, Y. Solving Many-Objective Optimization Problems via Multistage Evolutionary Search. *IEEE Trans. Syst. Man Cybern. Syst.* **2019**, *51*, 3552–3564. [CrossRef]
6. Castillo, O.; Melin, P.; Ontiveros, E.; Peraza, C.; Ochoa, P.; Valdez, F.; Soria, J. A high-speed interval type 2 fuzzy system approach for dynamic parameter adaptation in metaheuristics. *Eng. Appl. Artif. Intell.* **2021**, *85*, 666–680. [CrossRef]
7. Li, X.; Mao, K.; Lin, F.; Zhang, X. Particle swarm optimization with state-based adaptive velocity limit strategy. *Neurocomputing* **2021**, *447*, 64–79. [CrossRef]
8. Yang, X.; Li, H.; Yu, X. Adaptive heterogeneous comprehensive learning particle swarm optimization with history information and dimensional mutation. *Multimed. Tools Appl.* **2022**, *82*, 9785–9817. [CrossRef]
9. Xia, X.; Gui, L.; Zhan, Z.-H. A multi-swarm particle swarm optimization algorithm based on dynamical topology and purposeful detecting. *Appl. Soft Comput.* **2018**, *67*, 126–140. [CrossRef]
10. Sun, G.; Yang, B.; Yang, Z.; Xu, G. An adaptive differential evolution with combined strategy for global numerical optimization. *Soft Comput.* **2019**, *24*, 6277–6296. [CrossRef]
11. Jiang, L.; Wang, X. Research on the Participation of Household Battery Energy Storage in the Electricity Peak Regulation Ancillary Service Market. *Processes* **2023**, *11*, 794. [CrossRef]
12. Li, L.; Li, Y.; Lin, Q.; Ming, Z.; Coello, C.A.C. A convergence and diversity guided leader selection strategy for many-objective particle swarm optimization. *Eng. Appl. Artif. Intell.* **2022**, *115*, 105249. [CrossRef]
13. Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995.
14. Hancer, E.; Xue, B.; Zhang, M.; Karaboga, D.; Akay, B. Pareto front feature selection based on artificial bee colony optimization. *Inf. Sci.* **2018**, *422*, 462–479. [CrossRef]
15. 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]
16. Rosso, M.M.; Aloisio, A.; Cucuzza, R.; Asso, R.; Marano, G.C. Structural Optimization with the Multistrategy PSO-ES Unfeasible Local Search Operator. In Proceedings of the International Conference on Data Science and Applications: ICDSA 2022, Kolkata, India, 26–27 March 2022; pp. 215–229.
17. Marano, G.C.; Cucuzza, R. Structural Optimization Through Cutting Stock Problem. In *Italian Workshop on Shell and Spatial Structures*; Springer: Cham, Switzerland, 2023; pp. 210–220.
18. Xia, X.; Xing, Y.; Wei, B.; Zhang, Y.; Li, X.; Deng, X.; Gui, L. A fitness-based multi-role particle swarm optimization. *Swarm Evol. Comput.* **2019**, *44*, 349–364. [CrossRef]

19. Chen, K.; Zhou, F.; Yin, L.; Wang, S.; Wang, Y.; Wan, F. A hybrid particle swarm optimizer with sine cosine acceleration coefficients. *Inf. Sci.* **2018**, *422*, 218–241. [\[CrossRef\]](#)
20. Lin, J.C.-W.; Yang, L.; Fournier-Viger, P.; Hong, T.-P.; Voznak, M. A binary PSO approach to mine high-utility itemsets. *Soft Comput.* **2016**, *21*, 5103–5121. [\[CrossRef\]](#)
21. Tian, D.; Zhao, X.; Shi, Z. Chaotic particle swarm optimization with sigmoid-based acceleration coefficients for numerical function optimization. *Swarm Evol. Comput.* **2019**, *51*, 100573. [\[CrossRef\]](#)
22. Chen, K.; Zhou, F.; Wang, Y.; Yin, L. An ameliorated particle swarm optimizer for solving numerical optimization problems. *Appl. Soft Comput.* **2018**, *73*, 482–496. [\[CrossRef\]](#)
23. Liang, H.; Kang, F. Adaptive mutation particle swarm algorithm with dynamic nonlinear changed inertia weight. *Optik* **2016**, *127*, 8036–8042. [\[CrossRef\]](#)
24. Rosso, M.M.; Cucuzza, R.; Aloisio, A.; Marano, G.C. Enhanced Multi-Strategy Particle Swarm Optimization for Constrained Problems with an Evolutionary-Strategies-Based Unfeasible Local Search Operator. *Appl. Sci.* **2022**, *12*, 2285. [\[CrossRef\]](#)
25. Rosso, M.M.; Cucuzza, R.; Di Trapani, F.; Marano, G.C. Nonpenalty Machine Learning Constraint Handling Using PSO-SVM for Structural Optimization. *Adv. Civ. Eng.* **2021**, *2021*, 6617750. [\[CrossRef\]](#)
26. Jiyue, E.; Liu, J.; Wan, Z. A novel adaptive algorithm of particle swarm optimization based on the human social learning intelligence. *Swarm Evol. Comput.* **2023**, *80*, 101336. [\[CrossRef\]](#)
27. Lynn, N.; Suganthan, P.N. Ensemble particle swarm optimizer. *Appl. Soft Comput.* **2017**, *55*, 533–548. [\[CrossRef\]](#)
28. Liang, J.J.; Qin, A.K.; Suganthan, P.N.; Baskar, S. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Trans. Evol. Comput.* **2006**, *10*, 281–295. [\[CrossRef\]](#)
29. Lynn, N.; Suganthan, P.N. Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation. *Swarm Evol. Comput.* **2015**, *24*, 11–24. [\[CrossRef\]](#)
30. Xia, X.; Gui, L.; Yu, F.; Wu, H.; Wei, B.; Zhang, Y.-L.; Zhan, Z.-H. Triple Archives Particle Swarm Optimization. *IEEE Trans. Cybern.* **2020**, *50*, 4862–4875. [\[CrossRef\]](#)
31. Li, W.; Jing, J.; Chen, Y.; Chen, Y. A cooperative particle swarm optimization with difference learning. *Inf. Sci.* **2023**, *643*, 119238. [\[CrossRef\]](#)
32. Xia, X.; Gui, L.; He, G.; Wei, B.; Zhang, Y.; Yu, F.; Wu, H.; Zhan, Z.-H. An expanded particle swarm optimization based on multi-exemplar and forgetting ability. *Inf. Sci.* **2020**, *508*, 105–120. [\[CrossRef\]](#)
33. Shankar, R.; Ganesh, N.; Ćep, R.; Narayanan, R.C.; Pal, S.; Kalita, K. Hybridized Particle Swarm—Gravitational Search Algorithm for Process Optimization. *Processes* **2022**, *10*, 616. [\[CrossRef\]](#)
34. Ghorbanpour, S.; Jin, Y.; Han, S. Differential Evolution with Adaptive Grid-Based Mutation Strategy for Multi-Objective Optimization. *Processes* **2022**, *10*, 2316. [\[CrossRef\]](#)
35. Yang, X.; Li, H.; Huang, Y. An adaptive dynamic multi-swarm particle swarm optimization with stagnation detection and spatial exclusion for solving continuous optimization problems. *Eng. Appl. Artif. Intell.* **2023**, *123*, 106215. [\[CrossRef\]](#)
36. Yang, X.; Li, H. Evolutionary-state-driven multi-swarm cooperation particle swarm optimization for complex optimization problem. *Inf. Sci.* **2023**, *646*, 119302. [\[CrossRef\]](#)
37. Yang, X.; Li, H.; Yu, X. A dynamic multi-swarm cooperation particle swarm optimization with dimension mutation for complex optimization problem. *Int. J. Mach. Learn. Cybern.* **2022**, *13*, 2581–2608. [\[CrossRef\]](#)
38. Liang, J.; Suganthan, P. Dynamic Multi-Swarm Particle Swarm Optimizer with Local Search. In Proceedings of the 2005 IEEE Congress on Evolutionary Computation, Scotland, UK, 2–5 September 2005.
39. 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, 24–26 April 2003.
40. Lu, J.; Zhang, J.; Sheng, J. Enhanced multi-swarm cooperative particle swarm optimizer. *Swarm Evol. Comput.* **2021**, *69*, 100989. [\[CrossRef\]](#)
41. Jing, L.; Bo-Yang, Q.; Suganthan, P.N. Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization. *Appl. Math. Sci.* **2013**, *7*, 281–295.
42. Awad, N.H.; Ali, M.Z.; Suganthan, P.N.; Liang, J. *Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization*; Technical Report; Nanyang Technological University: Singapore, 2016.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.