symmetry

Article

Improved Chaotic Particle Swarm Optimization Algorithm with More Symmetric Distribution for Numerical Function Optimization

Zhiteng Ma, Xianfeng Yuan *, Sen Han, Deyu Sun, and Yan Ma

School of Mechanical, Electrical & Information Engineering, Shandong University, Weihai 264209, China
* Correspondence: yuanxianfeng@sdu.edu.cn; Tel: +86-152-6312-1688

Received: 4 June 2019; Accepted: 3 July 2019; Published: 3 July 2019

Abstract: As a global-optimized and naturally inspired algorithm, particle swarm optimization (PSO) is characterized by its high quality and easy application in practical optimization problems. However, PSO has some obvious drawbacks, such as early convergence and slow convergence speed. Therefore, we introduced some appropriate improvements to PSO and proposed a novel chaotic PSO variant with arctangent acceleration coefficient (CPSO-AT). A total of 10 numerical optimization functions were employed to test the performance of the proposed CPSO-AT algorithm. Extensive contrast experiments were conducted to verify the effectiveness of the proposed methodology. The experimental results showed that the proposed CPSO-AT algorithm converges quickly and has better stability in numerical optimization problems compared with other PSO variants and other kinds of well-known optimal algorithms.

Keywords: particle swarm optimizer; metaheuristic; nonlinear dynamic weights; dynamic learning factors; numerical optimization functions

1. Introduction

With the development of artificial intelligence and the increasing computational demands of industrial design stimulated by continuously developing industry, various optimization algorithms have been increasingly studied and applied in academia and industry. Optimization algorithms are more often being applied to real-world problems, such as artificial intelligence [1], logic circuit design [2], and system identification [3].

If traditional technology is adopted to solve problems in actual optimization tasks, time would be wasted and finding the solution would be difficult. Therefore, it was essential to develop new optimization algorithms. In the past 20 years, scholars and industry specialists have presented many natural heuristic algorithms, like particle swarm optimization (PSO) [4,5], sine cosine algorithm (SCA) [6], ant colony optimization (ACO) [7], biogeography-based optimization (BBO) [8], grey wolf optimizer (GWO) [9], differential evolution (DE) [10], krill herd (KH) algorithm [11], moth flame optimization (MFO) [12], and the whale optimization algorithm (WOA) [13]. These intelligent algorithms are well-known examples.

As a nature-inspired method, PSO can simulate the social behavior of individuals in fish or bird populations. This method is not only efficient and robust but can also be easily implemented with any computer language [14–16]. Therefore, the industry has already adopted this algorithm to search for the best solutions to stated problems. Researchers compared the performance of PSO with other similar and population-based heuristic optimum techniques (e.g., artificial bee colony (ABC) [17], differential evolution (DE) algorithm [18] and ant lion optimizer (ALO) [19]) and found that the results obtained by PSO were generally better.

How to design a testing method that can provide a fair and objective evaluation of different kinds of optimization algorithms is an interesting and meaningful topic. It is generally known that for deterministic optimization algorithms, rigorous theoretical proof of convergence is essential and important, while for heuristic optimization algorithms, it is very hard to prove convergence theoretically, therefore researchers in this field usually conduct extensive experiments to verify the effectiveness of the proposed methodology [20,21]. Hooker, a pioneer in the integration of optimization and constraint programming technologies, discussed the drawbacks of competitive testing and the benefits of scientific testing [22]. In [23], Sergeyev et al. proposed a new visual technique for a systematic comparison of global optimization algorithms having different natures. The proposed operational zones and aggregated operational zones are powerful tools for reliable comparison of deterministic and stochastic global optimization algorithms. This new testing methodology is a promising method that can be widely applied by researchers from optimization fields in the near future.

The PSO algorithm is simple and efficient, and is widely applied in practical engineering for global optimization. However, in [24], the authors conducted many theoretical derivations and simulation experiments to discuss the performance of global random search algorithms for large dimensions, and they concluded that if the dimension of the feasible domain was large then it was virtually impossible to guarantee that the global minimizer was reached by a general global random search algorithm. Therefore, according to the above conclusion, PSO and its variants are not suitable for solving complex high-dimensional optimization problems theoretically. In addition, PSO has some other shortcomings such as a premature and slow convergence speed. To overcome these deficiencies, many researchers have introduced different methods to enhance the performance of PSO. Compared with the basic PSO, these well-designed PSO variants not only have excellent convergence speed, but also have a strong global optimization ability. However, for special circumstances, such as solving complex optimization problems with different test functions, the robustness, population diversity and the ability to balance local exploitation and global exploration is still insufficient.

2. Review of Previous Work

2.1. Particle Swarm Optimizer

Eberhart and Kennedy proposed the PSO algorithm, which is an evolutionary computational technique invented from the study of the predation behavior of bird flocks. The location of each particle represents a candidate solution, and the fitness of the solution depends on the fitness function. In each iteration, the particles update their speed and position by dynamically tracking the two extremes including the individual best solution $(pbest_i)$ found by the individual and the global final value $(gbest)$ currently found by the entire population.

Set in *D*-dimensional target search space, there is a group of *m* particles, then the *i*-th particle can be represented by a *D*-dimensional vector, and its position can be denoted as $X_i = (x_{i1}, x_{i2}, \cdots, x_{iD})^{\mathrm{T}}, i = 1, 2, \cdots, m$. The flight speed is also a *D*-dimensional vector that can be denoted as $V_i = (v_{i1}, v_{i2}, \cdots, v_{iD})^{\mathrm{T}}$. The optimal position of the *i*-th particle searched so far is $pbest_i$, and the optimal position searched by the entire particle swarm is $gbest$. The *d*-th dimension of the velocity and position of the *i*-th particle in the (*t*+1)-th iteration are updated as follows:

$$V_{i,d}^{t+1} = \omega \times V_{i,d}^{t} + c_1 \times r_1 \times (pbest_{i,d}^{t} - X_{i,d}^{t}) + c_2 \times r_2 \times (gbest_d^{t} - X_{i,d}^{t}) \quad (1)$$

$$X_{i,d}^{t+1} = X_{i,d}^{t} + V_{i,d}^{t+1} \quad (2)$$

where ω is the inertia weight; r_1 and r_2 are random numbers within [0,1]; c_1 and c_2 represent learning factors; $c_1 \cdot r_1 \cdot (pbest_i^d - X_i^d)$ is the individual cognition term, which represents the individual cognitive experience of the particles, and it causes the particles to move toward the best position they experience; and $c_2 \cdot r_2 \cdot (gbest^d - X_i^d)$ is the social cognition term, which denotes the influence of the group experience on the flight path of particles. The social cognition term moves the particles toward the best position found by the group, representing the sharing of information between the particles.

Shi and Eberhart introduced a famous PSO variant with a linear decreasing inertia weight. In [25], the proposed method adds this improvement using Equation (3) to avoid premature convergence. The improved mechanism is as follows:

$$\omega = \omega_{\max} - \frac{M_j}{M_{\max}} \times (\omega_{\max} - \omega_{\min}) \tag{3}$$

where M_j and $M_{\max}$ denote the number of present iterations and the maximum number of iterations defined by the user, respectively.

3. Chaotic Particle Swarm Optimization- Arctangent Acceleration Coefficient Algorithm

Because the PSO algorithm lacks diversity in its search process [26–29], the search is prone to stagnation due to early convergence. Moreover, the PSO algorithm cannot escape from the local optimal solution efficiently and the optimization of the solution is poor. Based on in-depth research on PSO improvements, three well designed improvements were introduced to enhance the performance of the PSO.

3.1. Chaotic Particle Swarm Optimization (CPSO)

The initialization of the particle swarm is important for the convergence speed of the PSO algorithm and the quality of the solution. During the stage of particle swarm initialization, as no prior knowledge is available, the position and velocity of the particles are generally determined by random initialization. Although the random distribution of particle swarms is effective to some extent, some particles may affect the convergence speed of the algorithm because they are too far away from the optimal solution. Chaotic motion occurs in various states in the chaotic attraction domain, and does not lose the required randomness of particle group initialization.

Chaos is a relatively common phenomenon in nonlinear systems. It is characterized by ergodicity and inherent randomness and can traverse all states in a particular range according to its law. Logistic mapping is a typical chaotic system, and the mapping relationship is as follows:

$$z_{i+1} = \mu z_i (1 - z_i), z_i \in (0,1] \tag{4}$$

where μ is the control variable. When $\mu = 4$, the logistic map is completely chaotic, and the chaotic variable z_i generated at this time has better ergodicity.

Figure 1 shows the bifurcation diagram of the logistic chaotic map. The chaotic map can distribute the particle swarm to the random value in the interval [0,1] during the search. However, the traditional chaotic mapping formula in Equation (4) distributes the particles more in the area 0 and 1, as shown in Figure 1. This causes the non-uniformity of the chaotic distribution, which prevents the even distribution of the particles in the [0, 1] interval during the chaotic search process.

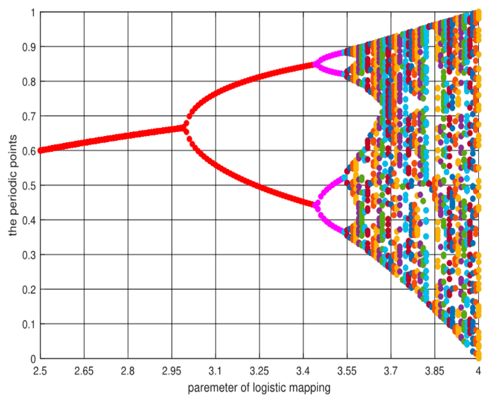

Figure 1. Bifurcation diagram of the logistic map for Equation (4).

Based on Equation (4), we used the cosine function to optimize it, with the goal of more evenly distributing the particles in the [0, 1] interval, and the new mapping relationship was defined as follows:

$$z_{i+1} = \cos\left(\mu z_i (1 - z_i)\right), z_i \in (0,1] \tag{5}$$

The chaos algorithm can be understood as: given an initial value, the pseudo-random sequence is generated by the initial iteration, and each iteration aims to realize a search. Through continuous iterations, the traversal search of the chaotic space can be realized. The chaotic algorithm implements a non-repetitive traversal search, thus fundamentally solving the local extreme problem in the PSO algorithm. Figure 2 illustrates the bifurcation diagram of the logistic map for Equation (5).

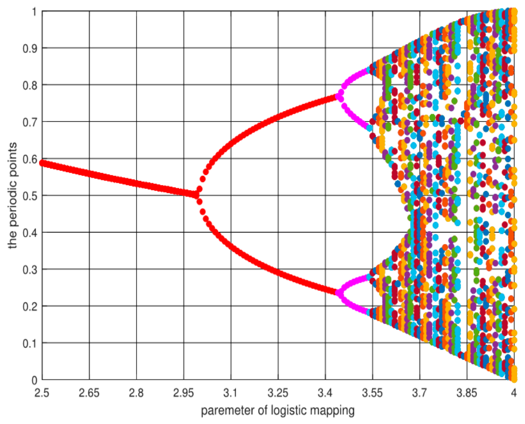

Figure 2. Bifurcation diagram of the logistic map for Equation (5).

We propose two main ways to use the logistic chaotic algorithm:

(1) A random sequence Z between chaotic [0, 1] is generated by an iteration of an initial value between [0, 1] through the iteration of the logistic equation: $a_0, a_1, a_2 \ldots$ Through linear mapping using Equation (6), the chaos is extended to the value range of the optimization variable $X[a,b]$ to achieve traversal of the range of values of the optimized variables.

$$Z \to X : X = a + (b - a) \times \cos(Z) \tag{6}$$

(2) Generate a chaotic random sequence Z between [0, 1] using the logistic equation, and then pass the carrier map in Equation (7), introducing chaos into $gbest$, a nearby area, to achieve local chaotic search:

$$Z \rightarrow Y : X = gbest + R \times \cos(Z) \tag{7}$$

where R is the search radius used to control the range of local chaotic search. Through experiments, we found that there was a good optimization effect when $R \in [0.1, 0.4]$.

3.2. Arc Tangent Acceleration Coefficients (AT)

Using arc tangent acceleration coefficients (AT), we propose a novel cognitive and social component parameter adjustment strategy in this paper. Compared with constant values, AT has a better ability to balance the overall search during the previous period and local convergence in the later stage. In AT, the parameters c_1 and c_2 are defined by Equation (8) and Equation (9), respectively.

$$c_1 = -\partial \times \arctan\left(\left(\frac{M_j}{M_{\max}}\right) \times \sigma\right) + \delta_1 \tag{8}$$

$$c_2 = \partial \times \arctan\left(\left(\frac{M_j}{M_{\max}}\right) \times \sigma\right) + \delta_2 \tag{9}$$

where ∂ and σ, and δ_1 and δ_2 are constant ($\partial = 1.5$, $\sigma = 4$, $\delta_1 = 2.5$, and $\delta_2 = 0.5$).

The range of values of c_1 and c_2 are illustrated in Figure 3, from which we can see c_1 ranged from 2.5 to 0.5 and c_2 ranged from 0.5 to 2.5.

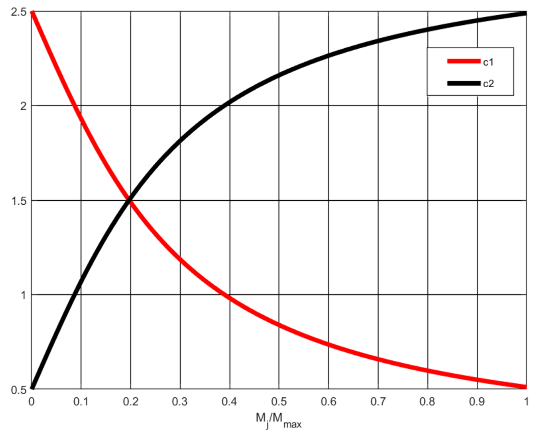

Figure 3. Arc tangent function acceleration coefficients (AT).

3.3. Cosine Map Inertia Weight

When the inertia weight is large, it is convenient for the global search, and when the inertia weight is small, the local search is enhanced. According to previous studies [30,31], the sine function plays an important role in adjusting ω. Inspired by this idea, we proposed an improved cosine function, which was defined by Equation (10).

$$\omega = \varphi \times \cos\left(\left(\frac{M_j}{M_{\max}}\right) \times \pi\right) + \tau \tag{10}$$

where φ and τ are constant ($\varphi = 1/3$, $\tau = 0.6$)

Compared with the traditional PSO algorithm, the improved cosine function can facilitate the PSO algorithm because of its global search in the early stages and local convergence in the later period. After many comparative experiments, we found that the optimal effect was obtained when

φ = 1/3, τ = 0.6. In this case, ω ranged between 0.9333 and 0.2667 and the change curve was depicted in Figure 4.

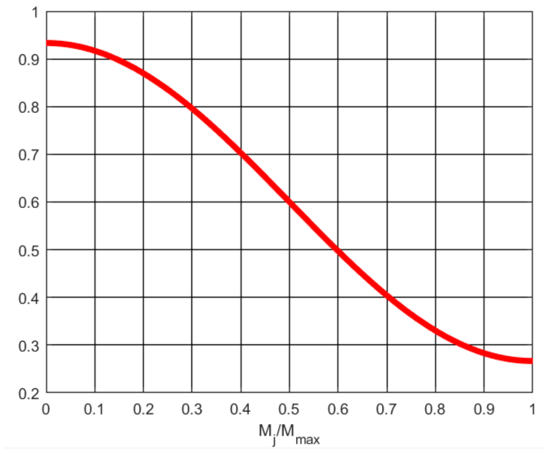

Figure 4. Improved inertia weight image.

Based on the above explanation, the pseudo-code of the proposed hybrid PSO variant CPSO-AT algorithm is shown in Table 1.

Table 1. Pseudo-code of the Chaotic Particle Swarm Optimization- Arctangent Acceleration algorithm.

```
1 Initialize the parameters ($PS$, $D$, $V_{\min}$, $V_{\max}$, $M_{\max}$, $c_1$, $c_2$, $X_{\min}$, $X_{\max}$)
2 The particle swarm positions $X_i (i = 1, 2, \cdots, PS)$ are initialized by chaos theory by Equation (6)
3 Randomly generate $N$ initial velocities within the maximum range
4 PSO algorithm is used to search for individual extremum and global optimal solutions
5 Local search:
6 While $Iter < M_{\max}$ do
7     Update the inertia weight $\omega$ using Equation (10)
8     Using Eqs. (8) and (9) to update the cognitive component $c_1$ and social component $c_2$.
9     for $i$ =1:PS (population size) do
10        The speed $V_i$ of the particle $X_i$ is updated using Equation (1)
11        Use Equation (2) to update the position of particle $X_i$
12        Calculate the fitness values $f_i$ of the new particle $X_i$
13        if $X_i$ is superior to $pbest_i$
14            Set $X_i$ to be $pbest_i$
15        End if
16        if $X_i$ is superior to $gbest$
17        Set $X_i$ to be $gbest$
18        End if
19 Global search:
20 Chaotic search $K$ times near $gbest$
21 Use the following formula $Z \to Y : X = gbest + R \times \cos(Z)$ (Equation (7))
22 $K$ chaotic search points near $pbest_i$ are obtained
23    for $j$=1:$K$ do
24        if $f(X) < f(gbest)$
25        Set $f(X)$ to be $gbest$
26        End if
27    End for
28        $Iter = Iter + 1$
29 End While
```

4. Simulation Experiments, Settings and Strategies

For the basic PSO and its variants listed in Table 2, the maximum iteration steps for all functions were defined as 2000, and the population size was set to 100. For the acceleration parameters for basic PSO and classical PSO, we set $c_1 = c_2 = 2.0$. The dimension values of each classic test function in Table 3 were set to 30 and 50 in turn.

Table 2. Parameter settings for Chaotic Particle Swarm Optimization- Arctangent Acceleration (CPSO-AT) and other Particle Swarm Optimization (PSO) variants.

Algorithms	Population Size	Dimension	Parameter Settings	Iteration
Basic PSO	100	30,50	$c_1 = c_2 = 2.0, \omega = 1, V_{\max} = 0.2 \times Bound$	2000
Classical PSO	100	30,50	$c_1 = c_2 = 2.0, \omega = 0.9 \sim 0.4, V_{\max} = 0.2 \times Bound$	2000
CPSO	100	30,50	$c_1 = c_2 = 2.0, \omega = 0.9 \sim 0.4, V_{\max} = 0.2 \times Bound$	2000
CPSO-AT	100	30,50	$c_1 : 2.5 \sim 0.5, c_2 = 0.5 \sim 2.5, \omega = 0.9333 \sim 0.2667,$ $V_{\max} = 0.2 \times Bound$	2000

Table 3 lists the 10 classical benchmark functions that were applied to verify the performance of the proposed CPSO-AT. In this paper, benchmark functions were separated into two groups. The first group contained five single-modal functions and the second group consisted of five multi-modal functions. There were many local optimizations in f_8 (Rastrigin), which caused more difficulties when searching for the global optimum. f_7 (Ackley) is famous for a rather narrow global optimal basin, which means this function has many local optimums. In Table 3, we used *dim* and *s* to represent the solution space dimension and a subset of R^n, respectively. For each test function, 30 independent experiments were conducted.

Table 3. Multi-dimensional classical benchmark functions.

Name	Test function	*Dim*	*S*	$f_{\min}$	Group
sphere	$f_1(\vec{x}) = \sum_{i=1}^{n} x_i^2$	30 50	$[-100,100]^r$	0	Unimodal
Schwefel's 1.2	$f_2(\vec{x}) = \sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2$	30 50	$[-100,100]^r$	0	Unimodal
Rosenbrock	$f_3(\vec{x}) = \sum_{i=1}^{n-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30 50	$[-30,30]^n$	0	Unimodal
Dixon & Price	$f_4(\vec{x}) = (x_1 - 1)^2 + \sum_{i=2}^{n} i(2x_i^2 - x_{i-1})^2$	30 50	$[-10,10]^n$	0	Unimodal
Sum Squares	$f_5(\vec{x}) = \sum_{i=1}^{n} i x_i^2$	30 50	$[-10,10]^n$	0	Unimodal
Griewank	$f_6(\vec{x}) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(x_i/\sqrt{i}) - 1$	30 50	$[-600,600]^r$	0	Multimodal
Ackley	$f_7(\vec{x}) = -20 \exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + \mathrm{e}$	30 50	$[-32,32]^n$	0	Multimodal

Rastrigin	$f_8(\vec{x}) = \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i) + 10)$	30 50	$[-5.12,5.12]$	0	Multimodal
Levy	$f_9(\vec{x}) = \sin^2 \pi\omega_i \sum_{i=1}^{n-1}(\omega_i - 1)^2[1 + 10\sin^2(\pi\omega_i + 1)] + (\omega_d - 1)^2[1 + 10\sin^2(2\pi\omega_d)]$ where $\omega_i = 1 + \frac{x_i - 1}{4}$, for all $i = 1, \cdots, n$	30 50	$[-10,10]^n$	0	Multimodal
Zakharov	$f_{10}(\vec{x}) = \sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{n} 0.5ix_i\right)^4$	30 50	$[-5,10]^n$	0	Multimodal

5. Experimental Results and Discussion

In this study, the performance of the CPSO-AT algorithm was verified using extensive simulation experiments. These experiments can be divided into two groups. The first group compared CPSO-AT with basic PSO, classical PSO, and CPSO using 10 well-known standard test functions in 30 and 50 dimensions. The simulation results are shown in Tables 4 and 5. The second group compared CPSO-AT with other kinds of well-known optimization algorithms (MFO, KH and BBO) using the same conditions as the first group, and the parameter settings and experimental results are presented in Tables 6, 7 and 8, respectively.

The *k* in Tables 4, 5, 7, and 8 is a performance indicator. When *k* was 0, the performance of the CPSO-AT optimization was worse than that of the other algorithms; when *k* was 1 (non-text bold), the CPSO-AT optimization was slightly better than the other algorithms; when *k* was **1** (bold), the CPSO-AT optimization was superior to the other algorithms and the effect was more obvious. In addition, the best solutions are shown in bold in Tables 4, 5, 7 and 8.

As shown in Table 6, the parameters of MFO, KH, BBO and CPSO-AT were set as follows: the number of iterations was 2000, the population size was 50, and the comparison experiments were performed on the standard test functions (shown in Table 3) in 30 and 50 dimensions.

5.1. Comparison of CPSO-AT with Classical PSO, Basic PSO, and CPSO

We conducted many experiments to verify the effectiveness of a chaotic search theory, nonlinear dynamic learning factor, and nonlinear dynamic inertia weight. The algorithms were tested using 10 benchmark functions with 30 and 50 dimensions, and the results are shown in Tables 4 and 5. Figures 5–16 depict the convergence diagrams of basic PSO, classical PSO, CPSO and CPSO-AT algorithms with 30 and 50 dimensions of test functions. The values indicated in the figures are the mean of the best function values obtained from 30 independent experiments.

Figure 5 and 6 illustrate that the proposed CPSO-AT stood out when compared with the other three algorithms and its optimization accuracy and convergence speed had obvious advantages for unimodal functions f_1 and f_5. From Figures 7, 8, and 10 we can see that the proposed CPSO-AT algorithm showed better performance for multimodal functions f_6, f_7, f_{10}. However, Figures 9 and 15 depict that for f_8 the performance of CPSO-AT was not as good as that of the CPSO algorithm. From Figures 11 and 12, it can be seen that the CPSO-AT showed the fastest convergence speed and stronger robustness compared with the other three methods. Figures 13, 14 and 16 demonstrate that the CPSO-AT had a better ability of escaping from the local optimums.

The same conclusion can be drawn from Tables 4 and 5, which demonstrate the best fitness value (*The best*), worst fitness value (*The worst*), mean results (*mean*) and standard deviations (*S.D.*) of the CPSO-AT algorithm and three other PSO variants. It can be concluded that the proposed CPSO-AT achieved better optimization results for all unimodal functions $f_1 - f_5$. For four multimodal functions (f_6, f_7, f_9, f_{10}), the CPSO-AT algorithm also outperformed the three other PSO variants. However, for multimodal function f_8 all algorithms had very close optimization accuracy. Moreover, the last columns of Tables 4 and 5 revealed that the proposed CPSO-AT provided better robustness and stability than basic PSO and its variants.

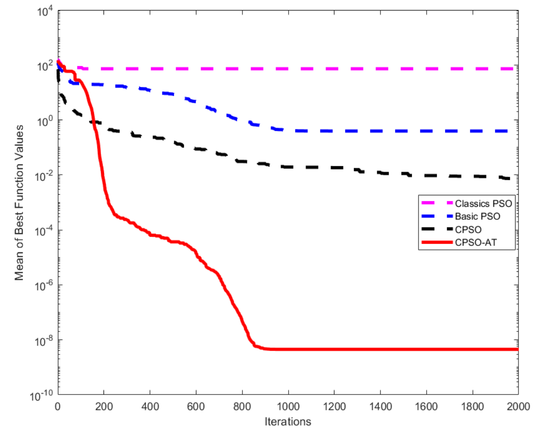

Figure 5. Comparison of the convergence curves of four algorithms for the f_1 (*Dim*=30).

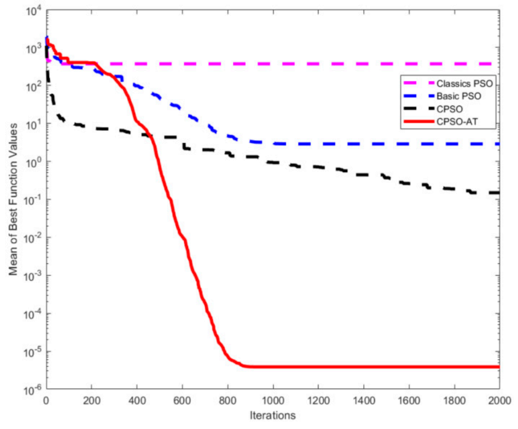

Figure 6. Comparison of the convergence curves of four algorithms for the f_5 function *(Dim=30)*.

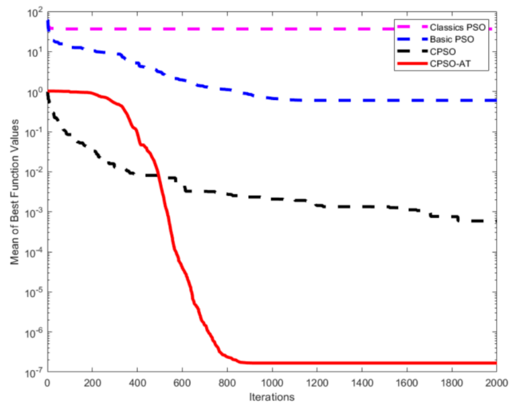

Figure 7. Comparison of the convergence curves of four algorithms for the f_6 (*Dim=30*).

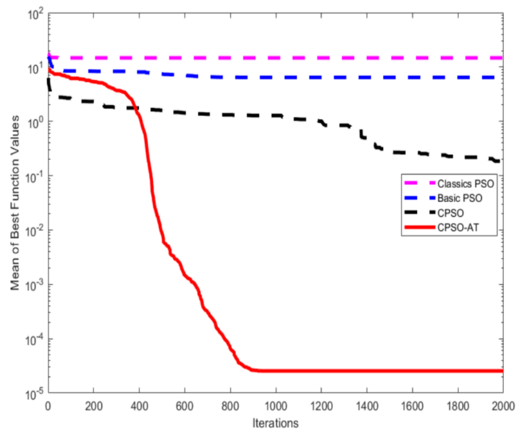

Figure 8. Comparison of the convergence curves of four algorithms for the f_7 (*Dim=30*).

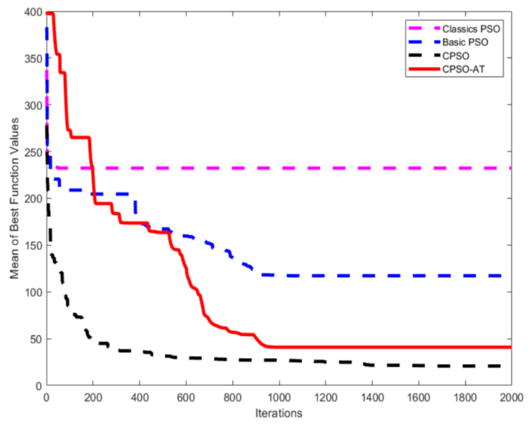

Figure 9. Comparison of the convergence curves of four algorithms for f_8 (*Dim=30*).

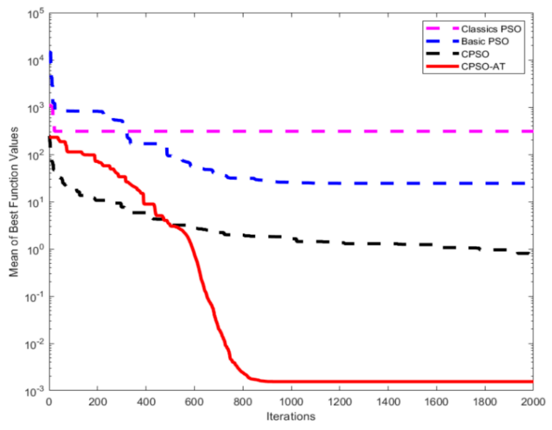

Figure 10. Comparison of the convergence curves of four algorithms for f_{10} (*Dim*=30).

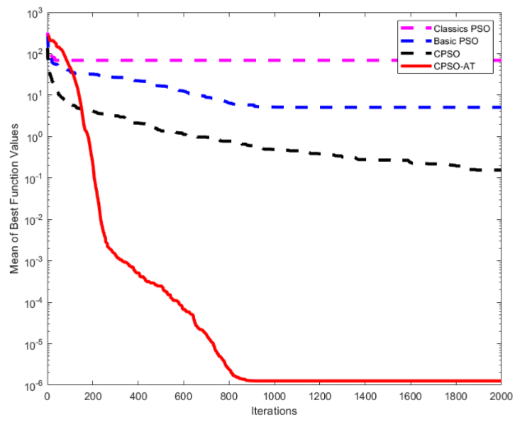

Figure 11. Comparison of the convergence curves of four algorithms for the f_1 (Dim=50).

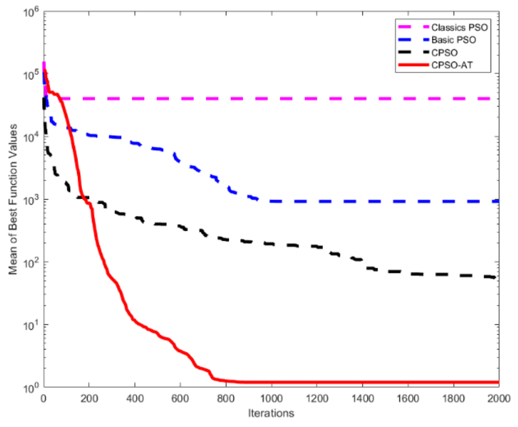

Figure 12. Comparison of the convergence curves of four algorithms for the f_2 (Dim=50).

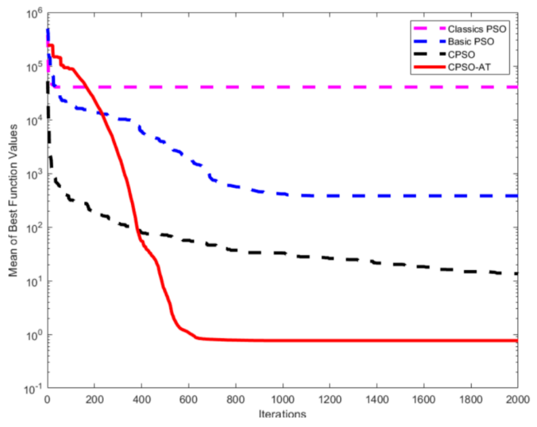

Figure 13. Comparison of the convergence curves of four algorithms for the f_4 (Dim=50).

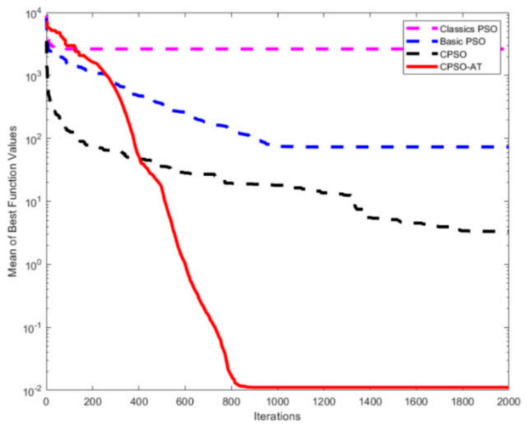

Figure 14. Comparison of the convergence curves of four algorithms for the f_5 (Dim=50).

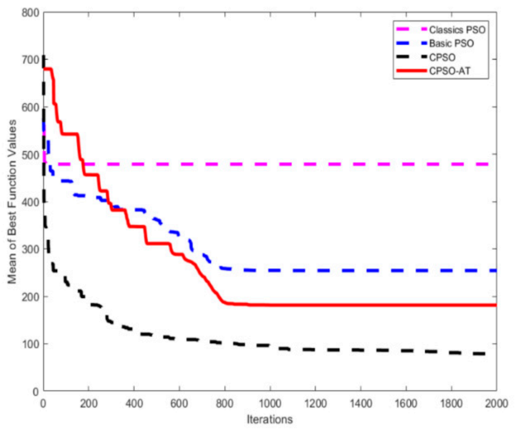

Figure 15. Comparison of the convergence curves of four algorithms for f_8 (Dim=50).

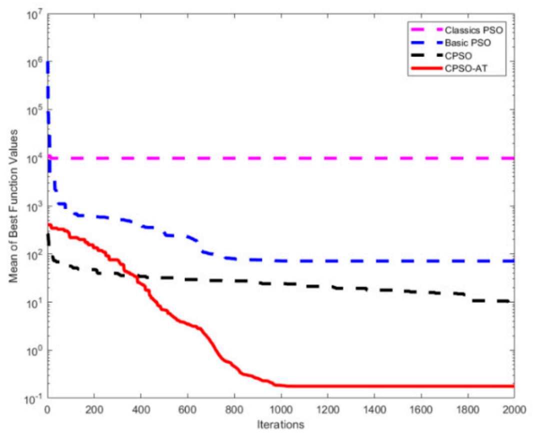

Figure 16. Comparison of the convergence curves of four algorithms for f_{10} (Dim=50).

Table 4. Experimental results obtained by basic particle swarm optimization (PSO), classical PSO, CPSO, and Chaotic Particle Swarm Optimization-Arctangent Acceleration (CPSO-AT) with 30 dimensions.

Function	Algorithm	k	*The best*	*The worst*	*Mean*	*S.D.*
f_1	Basic PSO	**1**	2.6248e+01	7.7947e+01	4.6323e+01	6.7444e+01
	Classical PSO		2.0757e-02	6.6373e-01	1.9719e-01	7.8747e-01
	CPSO		5.2121e-03	2.1287e-02	1.1753e-02	8.2205e-04
	CPSO-AT		**6.3925e-10**	**3.4988e-08**	**4.7194e-09**	**3.4958e-08**
f_2	Basic PSO	**1**	2.2111e+03	9.9471e+03	4.9502e+03	1.0172e+04
	Classical PSO		1.1954e+01	1.6882e+02	5.3816e+01	1.6148e+02
	CPSO		1.1748e+00	5.6324e+00	3.0123e+00	3.0028e-01
	CPSO-AT		**2.4951e-05**	**7.2287e-03**	**1.2410e-03**	**8.8665e-03**
f_3	Basic PSO	1	2.6521e+05	1.6800e+06	9.3596e+05	1.0188e+05
	Classical PSO		3.6194e+02	1.9558e+06	1.1596e+06	1.4155e+05
	CPSO		2.7934e+01	1.4180e+02	5.5210e+01	2.7520e+01
	CPSO-AT		**2.0086e+01**	**2.7560e+01**	**2.4461e+01**	**2.4490e-01**
f_4	Basic PSO	**1**	2.6551e+03	1.1167e+05	2.3649e+04	6.2701e+03
	Classical PSO		7.7642e+00	7.8947e+04	3.3909e+04	6.8874e+03
	CPSO		1.2618e+00	5.2299e+00	2.9283e+00	5.3780e-01
	CPSO-AT		**6.6667e-01**	**6.6939e-01**	**6.6699e-01**	**7.5939e-04**
f_5	Basic PSO	**1**	1.0021e+03	2.0249e+03	1.5684e+03	6.4703e+01
	Classical PSO		1.7245e+01	2.2872e+03	1.3307e+03	1.8101e+02
	CPSO		1.2397e-01	4.4059e-01	2.4581e-01	3.8646e-02
	CPSO-AT		**4.4910e-06**	**4.4064e-04**	**1.1348e-04**	**2.6671e-05**
f_6	Basic PSO	**1**	3.2239e+01	5.5316e+01	4.0062e+01	9.2496e-01
	Classical PSO		1.0188e+00	5.1722e+01	2.7296e+01	2.6616e+00
	CPSO		8.6921e-04	1.1118e-02	3.1795e-03	8.4473e-04
	CPSO-AT		**1.1528e-08**	**5.5167e-04**	**6.0028e-05**	**2.0000e-05**
f_7	Basic PSO	**1**	8.8528e+00	1.6029e+01	1.1915e+01	5.3159e-01
	Classical PSO		7.4734e+00	1.2232e+01	1.1736e+01	2.1728e-01
	CPSO		9.3623e-02	2.0819e+00	5.1825e-01	2.2687e-01
	CPSO-AT		**5.2116e-05**	**1.8759e-04**	**1.0254e-04**	**1.6110e-05**
f_8	Basic PSO	0	2.4279e+02	3.2104e+02	2.6438e+02	3.4627e+00
	Classical PSO		6.8569e+01	3.1272e+02	2.9253e+02	4.5919e+00
	CPSO		**2.9624e+01**	**4.2594e+01**	**3.6522e+01**	**1.9274e+00**
	CPSO-AT		9.2920e+01	1.7194e+02	1.3559e+02	5.0030e+00
f_9	Basic PSO	1	1.0570e+01	5.6268e+01	2.9851e+01	2.6466e+00
	Classical PSO		5.2905e+00	4.9856e+01	2.1421e+01	2.8645e+00
	CPSO		**9.4416e-03**	2.7481e+00	8.9063e-01	**1.4012e-01**
	CPSO-AT		3.5811e-01	**8.0576e-01**	**5.5508e-01**	4.5298e-01
f_{10}	Basic PSO	**1**	8.5199e+01	2.4830e+02	1.4751e+02	2.0541e+01
	Classical PSO		5.3003e-01	1.1802e+02	8.4578e+01	6.7666e+00
	CPSO		5.7915e-01	1.1559e+00	8.3102e-01	5.2794e-02
	CPSO-AT		**1.3389e-04**	**3.6616e-03**	**1.2061e-03**	**2.1842e-04**

Table 5. Experimental results obtained by basic particle swarm optimization (PSO), classical PSO, CPSO, and Chaotic Particle Swarm Optimization- Arctangent Acceleration (CPSO-AT) with 50 dimensions.

Function	Algorithm	*k*	*The best*	*The worst*	*Mean*	*S.D.*
f_1	Basic PSO	**1**	6.2992e+01	1.4072e+02	9.6568e+01	1.1705e+02
	Classical PSO		1.9810e+00	9.4058e+00	4.7912e+00	1.0715e+01
	CPSO		7.9989e-02	1.9665e-01	1.3500e-01	4.0335e-03
	CPSO-AT		**4.3797e-07**	**1.3892e-05**	**2.6618e-06**	**1.3012e-05**
f_2	Basic PSO	**1**	1.7735e+04	4.4191e+04	3.3163e+04	4.0007e+04
	Classical PSO		7.0766e+02	2.4271e+03	1.3298e+03	2.2228e+03
	CPSO		3.4384e+01	1.9495e+02	7.9785e+01	5.0406e+00
	CPSO-AT		**1.3885e-01**	**2.7291e+00**	**7.4383e-01**	**3.4377e+00**
f_3	Basic PSO	**1**	2.9077e+06	1.4292e+07	7.9705e+06	6.3951e+05
	Classical PSO		6.9551e+03	1.6232e+07	6.8072e+06	1.1339e+06
	CPSO		1.0277e+02	3.5589e+02	2.3726e+02	4.4241e+01
	CPSO-AT		**4.5713e+01**	**4.8066e+01**	**4.6730e+01**	**1.3927e-01**
f_4	Basic PSO	**1**	8.5353e+04	6.5400e+05	2.3039e+05	4.4620e+04
	Classical PSO		2.1823e+02	2.5940e+05	1.0611e+05	1.7552e+04
	CPSO		8.5470e+00	3.0465e+01	2.0967e+01	3.9523e+00
	CPSO-AT		**6.6783e-01**	**2.0588e+00**	**8.4647e-01**	**5.6966e-02**
f_5	Basic PSO	**1**	2.9846e+03	7.5971e+03	5.4303e+03	3.7474e+02
	Classical PSO		6.5151e+01	7.6068e+03	4.5427e+03	6.5260e+02
	CPSO		3.1804e+00	5.6937e+00	4.2359e+00	1.9161e-01
	CPSO-AT		**4.7685e-03**	**6.7831e-02**	**2.0803e-02**	**5.0860e-03**
f_6	Basic PSO	**1**	4.3759e+01	1.0136e+02	7.8097e+01	4.6646e+00
	Classical PSO		6.1000e+00	1.0849e+02	8.7209e+01	4.9779e+00
	CPSO		6.9568e-03	1.4624e-02	9.6156e-03	7.3385e-04
	CPSO-AT		**4.4032e-06**	**5.4671e-04**	**8.9087e-05**	**2.3055e-05**
f_7	Basic PSO	**1**	1.3003e+01	1.5286e+01	1.3916e+01	1.7718e-01
	Classical PSO		8.0444e+00	1.5888e+01	1.4502e+01	4.5279e-01
	CPSO		4.6800e-01	1.9761e+00	1.5023e+00	4.2769e-02
	CPSO-AT		**2.0332e-03**	**8.7914e-01**	**9.1370e-02**	**2.9368e-02**
f_8	Basic PSO	0	5.1852e+02	6.0608e+02	5.7156e+02	1.6262e+00
	Classical PSO		1.1434e+02	5.7794e+02	4.5988e+02	8.5234e+00
	CPSO		**7.4647e+01**	**1.4715e+02**	**1.1398e+02**	**7.3222e+00**
	CPSO-AT		2.0437e+02	3.2078e+02	2.6771e+02	8.6923e+00
f_9	Basic PSO	1	3.2829e+01	7.5323e+01	5.1270e+01	3.5370e+00
	Classical PSO		3.8076e+00	4.8866e+01	3.6627e+01	2.9257e+00
	CPSO		1.3024e+00	5.1136e+00	2.8866e+00	4.1186e-01
	CPSO-AT		**6.2676e-01**	**1.4325e+00**	**1.0566e+00**	**8.0876e-01**
f_{10}	Basic PSO	**1**	4.3138e+03	9.8686e+06	1.2329e+06	4.0793e+05
	Classical PSO		7.0378e+01	2.9135e+05	6.9641e+04	2.1887e+04
	CPSO		6.6807e+00	1.3292e+01	1.0134e+01	8.8523e-01
	CPSO-AT		**2.2299e-01**	**1.3807e+00**	**6.4995e-01**	**3.0979e-02**

5.2. Comparison of Chaotic Particle Swarm Optimization-Arctangent Acceleration with Moth-Flame Optimization, Krill Herd and Biogeography-Based Optimization

We compared our CPSO-AT algorithm with the optimization algorithms proposed by other researchers. The parameters for moth-flame optimization algorithm (MFO) [12], krill herd (KH) algorithm [11], and biogeography-based optimization (BBO) [8] are shown in Table 6. The 10 classical test functions listed in Table 3 were used for comparison and their dimensions were set to 30 and 50. For a fair comparison, each experiment was repeated 30 times for each test function. The experimental results are shown in Tables 7 and 8, including the optimal fitness values (*the best*), worst fitness values (*the worst*), average results (*mean*), and standard deviation (*S.D.*) for the MFO, KH, BBO, and CPSO-AT algorithms. The convergence curves of the above algorithms are illustrated in Figures 17–28.

Table 6. Parameter settings for the four studied algorithms.

Algorithm	Population	Maximum Iterations	Dim	other
MFO	50	2000	30,50	t is random number in the range [-2,1]
KH	50	2000	30,50	$N^{max} = 0.01, V_f = 0.02, D^{max} = 0.005$
BBO	50	2000	30,50	$\mathrm{Mu} = 0.005, \mu = 0.8\mathrm{c}$
CPSO-AT	50	2000	30,50	$\omega = \varphi \times \cos\left(\left(\frac{M_j}{M_{max}}\right) \times \pi\right) + \tau$ $c_1 = \partial \times arctan\left(\left(\frac{M_J}{M_{max}}\right) \times \sigma\right) + \delta_1$ $c_2 = -\partial \times arctan\left(\left(\frac{M_j}{M_{max}}\right) \times \sigma\right) + \delta_2$

The experimental results for solving 10 standard test functions in 30 dimensions are provided in Figures 17–22 and Table 7, which show that the CPSO-AT algorithm had obvious advantages compared with the other three well-known algorithms in all unimodal functions $(f_1 - f_5)$. For multimodal functions, the CPSO-AT algorithm found better solutions for f_6, f_7, and f_{10}. However, for f_8 and f_9, the performance was not as good as the BBO and KH algorithms, as shown in Figure 21 and Table 7.

The last two columns of Table 7 show that for the majority of the benchmark functions $(f_1 - f_7, f_{10})$, the proposed CPSO-AT algorithm performed better in both search accuracy and stability. The mean values listed in the sixth column of Table 7 show that for six functions $(f_1, f_2, f_5, f_6, f_7, f_{10})$, the CPSO-AT was obviously superior compared with the three other meta-heuristic optimization algorithms. The BBO algorithm produced the best results for two numerical functions (f_8, f_9), which made it the second best methodology. However, the optimization results of CPSO-AT were very close to those produced by the BBO algorithm for f_8 and f_9. KH was ranked as the third most impactful method by providing preferable solutions on four test functions (f_4, f_3, f_8, f_9), but performed poorly in six functions $(f_1, f_2, f_5, f_6, f_7, f_{10})$. MFO did not perform well on almost all test functions, and was determined to be the least effective method.

Tables 7 and 8 demonstrate that the CPSO-AT algorithm produced the best results among all four methods, followed by BBO, KH, then MFO. Figure 17 depicts the performance of the four algorithms for the f_1 function. During the search, the performance of CPSO-AT was superior. Figures 18–20 and 22 indicate that BBO, KH, and MFO were inferior to CPSO-AT. Figure 21 shows that BBO and KH performed better than CPSO-AT and MFO, but the optimization effect of CPSO-AT was close to that of the BBO algorithm.

Table 7. Experimental results produced by the Moth-Flame Optimization Algorithm (MFO), Krill Herd Algorithm (KH) and Biogeography-Based Optimization Algorithm (BBO) and Chaotic Particle Swarm Optimization-Arctangent Acceleration (CPSO-AT) with 30 dimensions.

Function	Algorithm	*k*	*The best*	*The worst*	*Mean*	*S.D.*
f_1	MFO	**1**	6.3466e-15	1.0000e+04	2.0000e+03	1.2649e+04
	KH		9.6793e-03	4.4021e-02	2.0048e-02	1.3669e-02
	BBO		2.7007e+00	9.8717e+00	4.2946e+00	6.1697e+00
	CPSO-AT		**3.1695e-07**	**2.8893e-06**	**8.6404e-07**	**1.1386e-07**
f_2	MFO	**1**	3.7805e-13	1.1700 e+06	4.7500 e+05	1.4342e+06
	KH		7.4251e+00	7.7804e+02	3.1717e+02	3.4905e+02
	BBO		2.4456e+02	1.0627e+03	6.9743e+02	7.0483e+02
	CPSO-AT		**1.2328e-03**	**1.4670e-01**	**3.7457e-02**	**9.9016e-03**
f_3	MFO	1	1.6385e+03	1.6385e+03	1.6385e+03	1.7316e-12
	KH		2.9288e+01	1.2114e+02	5.5490e+01	4.0249e+01
	BBO		1.2701e+02	6.5835e+02	3.2701e+02	5.1255e+02
	CPSO-AT		**2.3345e+01**	**2.9404e+01**	**2.4992e+01**	**9.1291e-02**
f_4	MFO	1	6.6667e-01	2.7370e+05	3.4679e+04	2.6107e+05
	KH		6.7509e-01	1.2386e+00	8.4426e-01	2.3154e-01
	BBO		4.4060e+00	9.1277e+00	6.9919e+00	4.7954e+00
	CPSO-AT		**6.6664e-01**	**6.6762e-01**	**6.6692e-01**	**2.1520e-04**
f_5	MFO	**1**	**1.1144e-14**	2.9000 e+03	6.9000 e+02	2.6738e+03
	KH		8.0391e-03	2.6790e-01	7.1299e-02	1.1182e-01
	BBO		3.3696e-01	6.0732e-01	4.6529e-01	2.5268e-01
	CPSO-AT		1.9718e-05	**1.8382e-04**	**8.3644e-05**	**2.0107e-05**
f_6	MFO	**1**	**9.3259e-15**	9.0535e+01	2.7080e+01	1.3078e+02
	KH		2.0406e-03	2.3335e-02	1.0695e-02	8.9323e-03
	BBO		1.0110e+00	1.0698e+00	1.0388e+00	4.8482e-02
	CPSO-AT		8.4554e-08	**1.5809e-06**	**3.6554e-07**	**5.6011e-08**
f_7	MFO	**1**	**7.0957e-09**	1.9963e+01	9.4662e+00	2.9308e+01
	KH		2.0726e-03	1.6499e+00	7.0606e-01	7.0425e-01
	BBO		7.0705e-01	1.3181e+00	9.6088e-01	4.6179e-01
	CPSO-AT		2.9201e-04	**1.2453e-03**	**5.9240e-04**	**2.0650e-04**
f_8	MFO	0	7.6612e+01	1.9331e+02	1.3427e+02	9.8256e+01
	KH		5.9930e+00	1.7947e+01	1.2761e+01	4.3635e+00
	BBO		**1.0071e+00**	**2.2005e+00**	**1.7721e+00**	**1.2180e+00**
	CPSO-AT		9.9508e+00	3.7812e+01	2.0800e+01	2.5423e+00
f_9	MFO	0	2.5135e+01	3.7618e+01	3.0205e+01	9.4966e+00
	KH		2.2932e-04	8.1506e-01	2.0720e-01	2.2620e-01
	BBO		**8.6060e-03**	**2.8077e-02**	**2.1078e-02**	**2.0072e-02**
	CPSO-AT		2.6859e-01	8.0576e-01	5.3717e-01	6.5790e-01
f_{10}	MFO	**1**	1.1036e-02	4.7619e+02	2.6749e+02	4.3400e+02
	KH		6.6967e+01	1.3059e+02	9.7460e+01	2.5847e+01
	BBO		2.3055e+01	4.4664e+01	3.3702e+01	2.0264e+01
	CPSO-AT		**9.4240e-05**	**2.1734e-04**	**1.7080e-04**	**4.8481e-06**

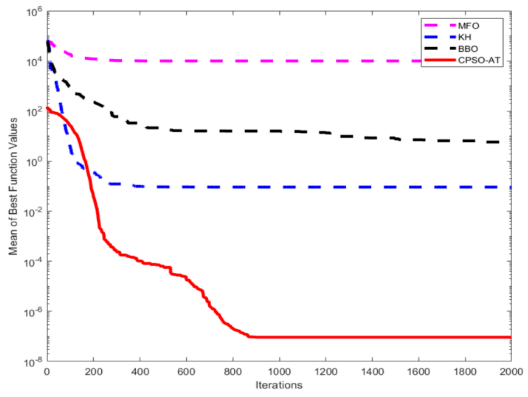

Figure 17. Comparison of convergence curves of the four algorithms for the f_1 (*Dim=30*).

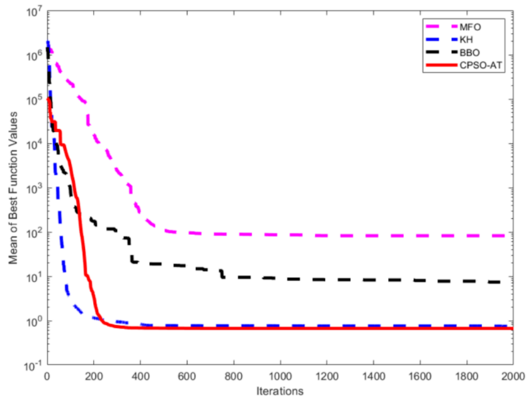

Figure 18. Comparison of convergence curves of the four algorithms the f_4 (*Dim=30*).

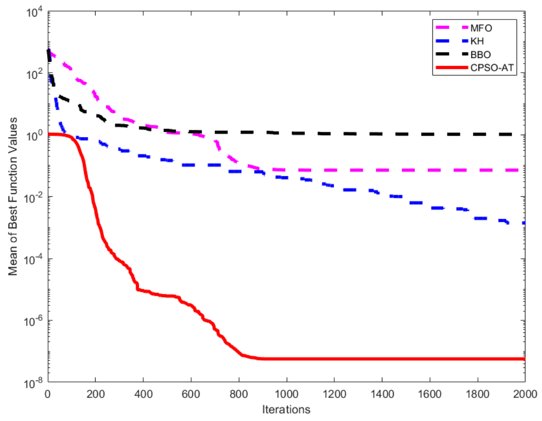

Figure 19. Comparison of convergence curves of the four algorithms for the f_6 (*Dim=30*).

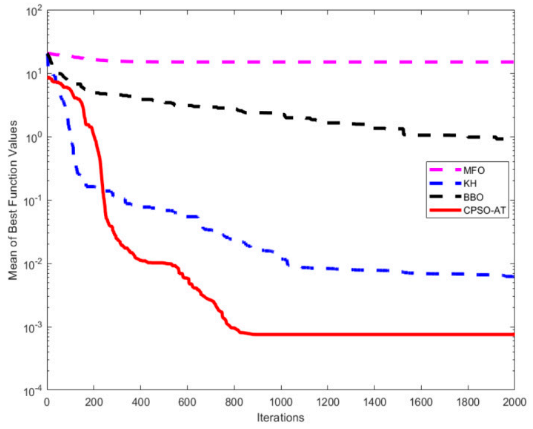

Figure 20. Comparison of convergence curves of the four algorithms for the f_7 (*Dim=30*).

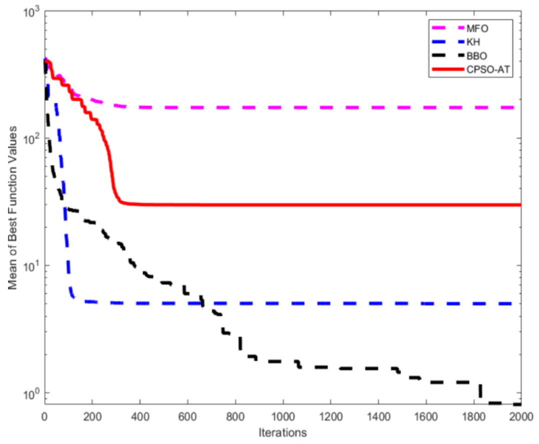

Figure 21. Comparison of convergence curves of the four algorithms for f_8 (*Dim=30*).

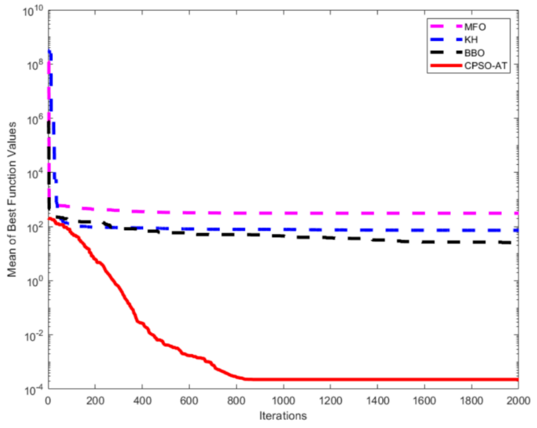

Figure 22. Comparison of convergence curves of the four algorithms for f_{10} (*Dim=30*).

Table 8 presents the optimization results of 10 standard test functions with 50 dimensions, and we concluded that the CPSO-AT algorithm provided a considerable advantage for solving unimodal function problems $(f_1 - f_5)$. However, for multimodal functions f_8 and f_9, BBO and KH algorithms performed slightly better than CPSO-AT. For the rest of the multimodal functions (f_6, f_7, f_{10}), the proposed CPSO-AT outperformed the other three heuristic optimization algorithms.

The convergence curves of the mean values of the numerical experiment functions $(f_1, f_2, f_5, f_7, f_8, f_{10})$ with 50 dimensions are illustrated in Figures 23–28. Figures 23 and 24 indicate that the CPSO-AT method provided the best search result compared with the other three methods, whereas the KH method was able to quickly converge to the global optimum. However, Figure 27 shows that the two algorithms (KH and BBO) can find similar solutions, and Figure 28 shows that the BBO algorithm has a powerful ability to globally search to evade the local optimum solution in the 50-dimensional f_{10} function. For most situations, the proposed CPSO-AT algorithm provided superior convergence accuracy and a steadier search ability than the other three methods when solving numerical optimization problems.

Table 8. Experimental results of the Moth-Flame Optimization Algorithm (MFO), Krill Herd Algorithm (KH) and Biogeography-Based Optimization Algorithm (BBO) , and Chaotic Particle Swarm Optimization-Arctangent Acceleration (CPSO-AT) with 50 dimensions.

Function	Algorithm	k	*The best*	*The worst*	*Mean*	*S.D.*
f_1	MFO	**1**	9.0418e-05	2.0000e+04	1.0000e+04	2.0000e+04
	KH		9.1848e-02	3.8079e-01	2.2432e-01	1.0846e-01
	BBO		6.8687e+01	1.4266e+02	9.9904e+01	6.6770e+01
	CPSO-AT		**1.7020e-05**	**5.5462e-05**	**4.3225e-05**	**8.3417e-07**
f_2	MFO	**1**	1.0000e+04	5.0200e+06	1.8190e+06	4.6469e+06
	KH		1.0638e+03	8.3849e+03	3.1824e+03	3.0333e+03
	BBO		1.6172e+04	5.0529e+04	2.7620e+04	2.7312e+04
	CPSO-AT		**1.0216e+00**	**4.4131e+00**	**2.3541e+00**	**2.6353e-01**
f_3	MFO	**1**	2.7684e+03	2.7686e+03	2.7684e+03	1.3105e-01
	KH		6.0262e+01	1.8692e+02	1.3323e+02	4.8077e+01
	BBO		1.4617e+03	4.8529e+03	2.5030e+03	2.7857e+03
	CPSO-AT		**4.5596e+01**	**1.0267e+02**	**5.2237e+01**	**2.1782e+00**
f_4	MFO	**1**	6.2704e+00	4.6889e+05	1.8365e+05	4.7211e+05
	KH		2.3253e+00	5.6925e+00	3.2332e+00	1.4030e+00
	BBO		4.7422e+01	1.0958e+02	7.4611e+01	4.8684e+01
	CPSO-AT		**6.6895e-01**	**8.9973e-01**	**7.4677e-01**	**4.9214e-02**
f_5	MFO	**1**	**1.8984e-05**	9.7000e+03	2.2300e+03	8.7864e+03
	KH		5.1658e-02	4.7215e-01	2.2164e-01	1.6813e-01
	BBO		2.1286e+01	3.1796e+01	2.6284e+01	9.5861e+00
	CPSO-AT		3.1182e-03	**3.4762e-02**	**1.5271e-02**	**2.2791e-03**
f_6	MFO	**1**	2.6725e-05	9.0793e+01	2.7162e+01	1.3102e+02
	KH		8.7800e-03	5.1455e-02	2.3202e-02	1.7232e-02
	BBO		1.8720e+00	2.5471e+00	2.2098e+00	5.5828e-01
	CPSO-AT		**2.7021e-06**	**7.4758e-03**	**7.5390e-04**	**2.4893e-04**
f_7	MFO	**1**	3.1461e+00	1.9963e+01	1.7381e+01	1.5640e+01
	KH		1.1614e+00	2.9876e+00	1.8086e+00	6.9580e-01
	BBO		2.9456e+00	3.5964e+00	3.2432e+00	5.4301e-01
	CPSO-AT		**2.4651e-03**	**5.1956e-03**	**3.7252e-03**	**4.6693e-04**
f_8	MFO	0	2.0610e+02	4.0942e+02	2.7372e+02	1.7473e+02
	KH		**1.6014e+01**	3.4316e+01	2.4666e+01	6.5266e+00
	BBO		1.6638e+01	**2.8529e+01**	**2.4282e+01**	9.1892e+00
	CPSO-AT		2.4890e+01	4.4884e+01	3.4785e+01	**9.8221e-01**
f_9	MFO	0	1.2546e+01	8.8610e+01	3.2530e+01	6.3687e+01
	KH		4.4904e-01	2.7557e+00	1.0771e+00	8.7791e-01
	BBO		**3.0271e-01**	**6.0636e-01**	**4.4337e-01**	**2.8994e-01**
	CPSO-AT		5.3723e-01	1.3432e+00	9.6706e-01	7.5773e-01
f_{10}	MFO	**1**	6.2671e+02	9.7623e+02	7.9242e+02	4.5047e+02
	KH		3.0410e+02	3.9719e+02	3.4173e+02	3.5971e+01
	BBO		7.6162e+01	1.3712e+02	1.0980e+02	4.7729e+01
	CPSO-AT		**9.2969e-03**	**6.3677e-02**	**2.5705e-02**	**1.2072e-02**

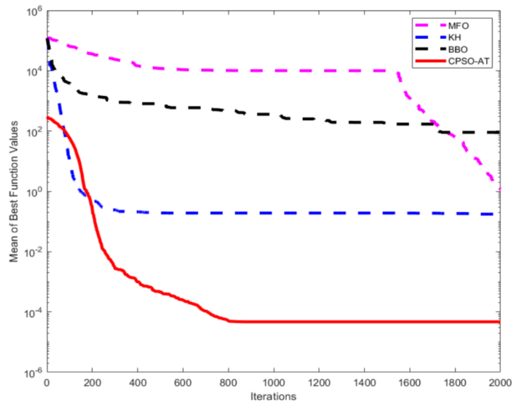

Figure 23. Comparison of the convergence curves of the four algorithms for the f_1 (*Dim=50*).

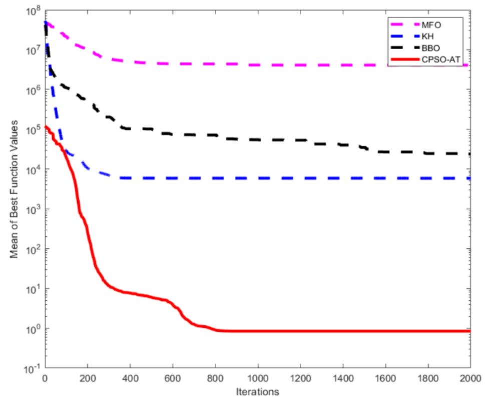

Figure 24. Comparison of the convergence curves of the four algorithms for the f_2 (*Dim=50*).

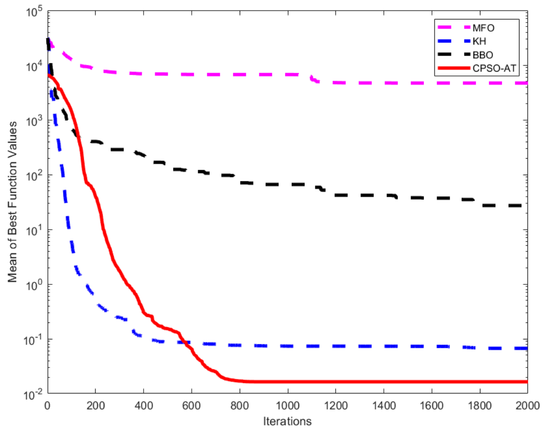

Figure 25. Comparison of the convergence curves of the four algorithms for the f_5 (*Dim=50*).

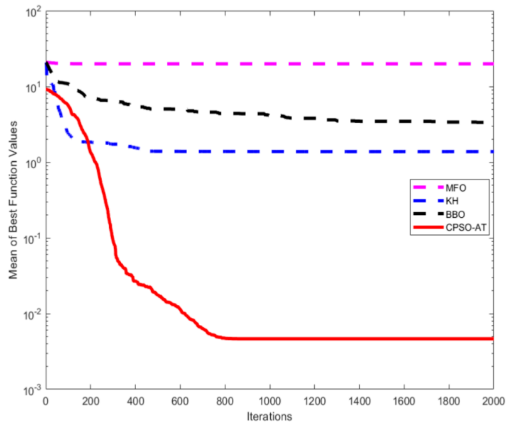

Figure 26. Comparison of the convergence curves of the four algorithms for the f_7 (*Dim=50*).

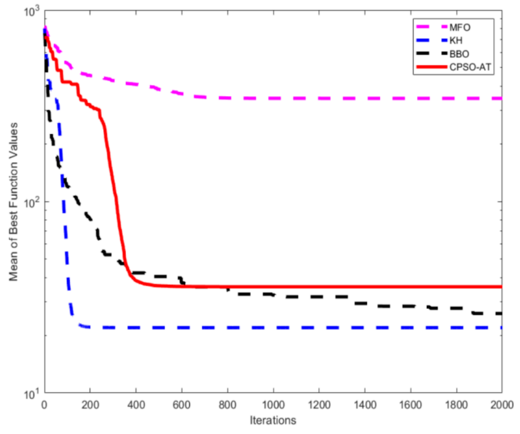

Figure 27. Comparison of the convergence curves of the four algorithms for f_8 (*Dim=50*).

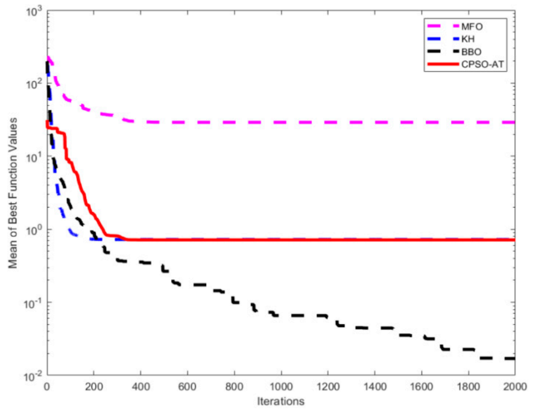

Figure 28. Comparison of the convergence curves of the four algorithms for the f_{10} (*Dim=50*).

From Tables 7 and 8 and Figures 17–28, we conclude that for numerical function optimization problems, the proposed CPSO-AT algorithm was superior compared with MFO, KH, and BBO. In particular for unimodal functions, the proposed method had obvious advantages. Therefore, when solving single-modal function problems, we recommend using the CPSO-AT algorithm. For multimodal functions, CPSO-AT also performs well in both optimization accuracy and convergence speed. In summary, the CPSO-AT provides a good balance between global exploration and local exploitation capabilities, which indicates that the proposed nonlinear dynamic weights, dynamic learning factors, linear mapping, and chaotic particle position updating method are effective for enhancing the performance of CPSO-AT.

6. Conclusion and Future Work

In this paper, we proposed a novel CPSO-AT algorithm that could improve the overall performance of the traditional PSO method. To obtain a good balance between global and local search capabilities, nonlinear dynamic weights, dynamic learning factors, linear mapping, and chaotic particle position updating methods were combined in the proposed method. Extensive experiments were conducted using ten 30-dimensional and 50-dimensional classical benchmark functions to verify the effectiveness of the proposed method. In the first set of experiments, CPSO-AT was compared with basic PSO and its two kinds of variants, and with three other well-known optimization algorithms in the second group of experiments. The experimental results indicated that applying dynamic weights and dynamic learning factors in the proposed CPSO-AT algorithm were beneficial and better results were obtained for most of the classical benchmark functions. Therefore, the proposed CPSO-AT is a good choice for solving numerical optimization problems.

In the future, the diversity of particles and the convergence speed of the CPSO-AT algorithm needs to be further studied. We plan to apply the CPSO-AT algorithm to solving some real-world problems in future work.

Author Contributions: Zhiteng Ma and Xianfeng Yuan designed the experiments; Zhiteng Ma, Sen Han and Deyu Sun performed the experiments; Yan Ma analyzed the data; Zhiteng Ma and Xianfeng Yuan wrote the paper.

Acknowledgments: This work was supported by The National Natural Science Foundation of China (Grant No. 61803227, 61773242, 61603214, 61573213), National Key R & D Program of China (Grant No.2017YFB1302400), Key Program of Scientific and Technological Innovation of Shandong Province (Grant No. 2017CXGC0926), Key Research and Development Program of Shandong Province (Grant No. 2017GGX30133, 2018GGX101039), Independent Innovation Foundation of Shandong University (Grant No. 2082018ZQXM005).

Conflicts of Interest: The authors declare that there were no conflicts of interests regarding the publication of this paper.

References

1. Wang, G.; Guo, J.M.; Chen, Y.P.; Li, Y.; Xu, Q. A PSO and BFO-based Learning Strategy applied to Faster R-CNN for Object Detection in Autonomous Driving. *IEEE Access* **2019**, *7*, 18840–18859.
2. Siano, P.; Citro, C. Designing fuzzy logic controllers for DC–DC converters using multi-objective particle swarm optimization. *Electr. Power Syst. Res.* **2014**, *112*, 74–83.
3. Yu, Z.H.; Xiao, L.J.; Li, H.Y.; Zhu, X.L.; Huai, R.T. Model Parameter Identification for Lithium Batteries using the Coevolutionary Particle Swarm Optimization Method. *IEEE Trans. Ind. Electron.* **2017**, *64*, 569–5700.
4. Chen, K.; Zhou, F.Y.; Liu, A.L. Chaotic dynamic weight particle swarm optimization for numerical function optimization. *Knowl. Based Syst.* **2018**, *139*, 23–40.
5. Lee, C.S.; Wang, M.H.; Wang, C.S.; Teytaud, O.; Liu, J.L.; Lin, S.W.; Hung, P.H. PSO-based Fuzzy Markup Language for Student Learning Performance Evaluation and Educational Application. *IEEE Trans. Fuzzy Syst.* **2018**, *26*, 2618–2633.
6. Mirjalili, S. SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowl. Based Syst.* **2016**, *96*, 120–133.
7. Dorigo, M.; Maniezzo, V.; Colorni, A. Ant system: Optimization by a colony of cooperating agents. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2002**, *26*, 29–41.
8. Simon, D. Biogeography-Based Optimization. *Trans. Evol. Comput.* **2008**, *12*, 702–713.
9. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61.
10. Das, S.; Mullick, S.S.; Suganthan, P.N. Recent advances in differential evolution–An updated survey. *Swarm Evol. Comput.* **2016**, *27*, 1–30.

11. Gandomi, A.H.; Alavi, A.H. Krill herd: A new bio-inspired optimization algorithm. *Commun. Nonlinear Sci. Numer. Simul.* **2012**, *17*, 4831–4845.
12. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl. Based Syst.* , **2015**, 89, 228–249.
13. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67.
14. Chen, K.; Zhou, F.Y.; Wang, Y.G.; Yin, L. An ameliorated particle swarm optimizer for solving numerical optimization problems. *Appl. Soft Comput.* **2018**, *73*, 482–496.
15. Bonyadi, M.R.; Michale, Z. Particle Swarm Optimization for Single Objective Continuous Space Problems: A Review. *Evol. Comput.* **2017**, *25*, 1–54.
16. Zhang, X.M.; Feng, T.H.; Niu, Q.S.; Deng, X.J. A Novel Swarm Optimisation Algorithm Based on a Mixed-Distribution Model. *Appl. Sci. Basel* **2018**, *8*, 632.
17. Ozturk, C.; Hancer, E.; Karaboga, D. A novel binary artificial bee colony algorithm based on genetic operators. *Inf. Sci.* **2015**, *297*, 154–170.
18. Tey, K.S.; Mekhilef, S.; Seyedmahmoudian, M.; Horan, B.; Oo, A.M.T.; Stojcevski, A. Improved Differential Evolution-Based MPPT Algorithm Using SEPIC for PV Systems Under Partial Shading Conditions and Load Variation. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4322–4333.
19. Mirjalili, S. The Ant Lion Optimizer. *Adv. Eng. Softw.* **2015**, *83*, 80–98.
20. García-Ródenas, R.; Linares, L.J.; López-Gómez, J.A. A Memetic Chaotic Gravitational Search Algorithm for unconstrained global optimization problems. *Appl. Soft Comput.* **2019**, *79*, 14–29.
21. Tian, M.; Gao, X. Differential evolution with neighborhood-based adaptive evolution mechanism for numerical optimization. *Inf. Sci.* **2019**, *478*, 422–448.
22. Hooker, J.N. Testing heuristics: We have it all wrong. *J. Heuristics* **1995**, *1*, 33–42.
23. Sergeyev, Y.D.; Kvasov, D.E.; Mukhametzhanov, M.S. On the efficiency of nature-inspired metaheuristics in expensive global optimization with limited budget. *Sci. Rep.* **2018**, *8*, 453. doi:10.1038/s41598-017-18940-4.
24. Pepelyshev, A.; Zhigljavsky, A.; Žilinskas, A. Performance of global random search algorithms for large dimensions. *J. Glob. Optim.* **2018**, *71*, 57–71.
25. Shi, Y.; Eberhart, R. Modified Particle Swarm Optimizer. In Proceedings of the IEEE International Conference on Evolutionary Computation, Anchorage, AK, USA, 4-9 May 1999; pp. 69-73.
26. Khatami, A.; Mirghasemi, S.; Khosravi, A.; Lim, C.P.; Nahavandi, S. A new PSO-based approach to fire flame detection using K-Medoids clustering. *Expert Syst. Appl.* **2017**, *68*, 69–80.
27. Lin, C.W.; Yang, L.; Fournier-Viger, P.; Hong, T.P.; Voznak, M. A binary PSO approach to mine high-utility itemsets. *Soft Comput.* **2017**, *21*, 5103–5121.
28. Zhou, Y.; Wang, N.; Xiang, W. Clustering Hierarchy Protocol in Wireless Sensor Networks Using an Improved PSO Algorithm. *IEEE Access* **2017**, *5*, 2241–2253.
29. Chouikhi, N.; Ammar, B.; Rokbani, N.; Alimi, A.M. PSO-based analysis of Echo State Network parameters for time series forecasting. *Appl. Soft Comput.* **2017**, *55*, 211–225.
30. Wang, G.G.; Guo, L.; Gandomi, A.H.; Hao, G.S.; Wang, H. Chaotic Krill Herd algorithm. *Inf. Sci.* **2014**, *274*, 17–34.
31. Niu, P.; Chen, K.; Ma, Y.; Li, X.; Liu, A.; Li, G. Model turbine heat rate by fast learning network with tuning based on ameliorated krill herd algorithm. *Knowl. Based Syst.* **2017**, *118*, 80–92.