Improved Particle Swarm Optimization with Mean-based Prediction

Juncheng Guo, Yonghong Zhang

***Abstract*—This paper introduces a novel particle swarm optimization algorithm called Mean Prediction Particle Swarm Optimization (MPSO). In MPSO, a potential position is predicted firstly by using the mean value of the population, and then the new movement equation is derived from the difference between the current position and the potential position. In addition, to improve the accuracy of MPSO, a chaos-based local search strategy is adopted. To evaluate the effectiveness of MPSO, we conducted experiments on various benchmark functions and compared its performance with some other existing PSO algorithms. The experimental results demonstrate that MPSO not only achieves better convergence performance but also exhibits higher accuracy.**

***Index Terms*—Particle swarm optimization; Optimization; Swarm intelligence; Mean-based prediction**

I. Introduction

SIMILAR to other swarm intelligence algorithms [1], particle swarm optimization (PSO) was presented in 1995 [2] based on the collective behavior of birds and fish. Since then, it has become a powerful optimization algorithm that finds extensive applications in many fields, including engineering problems [3-5], wireless networks [6], cloud computing [7], image processing [8], and medical care problems [9], and so on.

PSO is an algorithm that guides a population of particles through a search space to find the optimal solution. Each particle adjusts its position based on its own experience and that of its neighbors. Although PSO has made significant progress since it was introduced, it still has some limitations. One major drawback is it converges too early to a suboptimal solution, failing to identify the global optimum. To overcome this limitation, researchers have proposed various improvement measures to enhance the performance of PSO.

For example, Wang et al. proposed a hierarchical particle swarm optimization based on the mean value (mHPSO) [10]. Chen et al. aimed to enhance the search ability of PSO by using two different crossover operations to breed promising exemplars that guide the particles [11]. Wang and Song introduced an improved PSO (CNPSO) based on two new formulas [12]. Additionally, some scholars have suggested using neighborhood topologies to enhance the exploration and exploitation abilities of PSO.

Manuscript received June 20, 2023; revised December 21, 2023.

This work was supported by the Key cultivation project of Xianyang Normal University (XSYK21044);the Key Scientific Research Projects in Universities in Henan Province (24B110006);the Research Project of Henan Polytechnic (2023ZK22).

Juncheng Guo is an associate professor of the Basic Education Department, Henan Polytechnic, Zhengzhou, 450046, PR China, e-mail:acheng2090@126.com

Yonghong Zhang is an associate professor of the School of Mathematics and Statistics, Xianyang Normal University, Xianyang, 712000, PR China, email:zhangyonghong09@126.com

In this paper, we propose a novel particle swarm algorithm MPSO. The main idea behind MPSO is to utilize the mean method to predict the optimal solution's location for the next population. This prediction is based on the optimal solutions of the population from the past three generations.

Furthermore, we introduce a new movement equation by taking into account the difference between the predicted optimal position and the current position. This equation guides the particles to search for the optimal solution.

To compare the convergence and accuracy of MPSO, it is compared with several other PSOs based on a set of well-known test functions.

The structure of the paper is as follows. In Section II, we provide a brief review of the algorithm PSO. Section III gives the details of the algorithm MPSO , including the process of mean prediction and the derivation of the movement equations. Section IV presents the comparison results between MPSO and several other PSOs.

II. Basic PSO algorithm

In PSO, particles move through the search space to find the optimal solution. Each particle adjusts its position based on its own experience and the experience of the whole population.

Assume that the dimension of the search space is D, and the swarm size is ps. In PSO, the initial population is generated in the search space in a random manner firstly. Then, each particle evaluates its fitness value based on the objective function, and updates its position and velocity according to the following equations:

$$v_i^{t+1} = wv_i^t + c_1 r_1 (pbest_i - x_i^t) + c_2 r_2 (gbest - x_i^t), \quad (1)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1}, \quad (2)$$

where v_i^t and x_i^t represent the velocity and position of particle i at time t, respectively; $pbest_i$ is the historical optimal position of particle i, and $gbest$ is the current optimal position of the entire population. c_1, c_2, and w are the acceleration coefficients, and r_1 and r_2 are random numbers between 0 and 1. In this paper, ω adopts a nonlinear decreased way, which is defined as the following formula:

$$\omega = \omega_{min} + (\omega_{max} - \omega_{min}) e^{-0.6t} \quad (3)$$

where the parameters ω_{min} and ω_{max} denote the minimum and maximum inertia weight, respectively.

To provide a clear illustration of the iterative process of PSO, the pseudo-code and the flowchart of PSO are given in Algorithm 1 and Fig. 1, respectively.

Algorithm 1: Pseudo-code of PSO

01:Initialize the population size ps, the maximum number of iterations $ItMax$, c_1 and c_2. For each particle i, initialize the position x_i, and velocity v_i; the minimum inertia weight ω_{min} and the maximum inertia weight ω_{max}; the minimum velocity v_{min} and the maximum velocity v_{max}.
02: Set $pbest_i = x_i (i = 1, \cdots, ps)$ and find $gbest$, the iteration counter $it = 1$.
03: **While** $it < ItMax$ **do**
04: For $i = 1$ to ps **do**
05: By (1) and (2), update the velocity and the position of each particle.
06: If $f(x_i) < f(pbest_i)$
07: $pbest_i = x_i$;
08: End if
09: If $f(x_i) < f(gbest)$
10: set $gbest = x_i$,
11: End if
12: End for
13: $it = it + 1$
14: By (3), update the inertia weight ω.
15: End while
16: Output the final result.

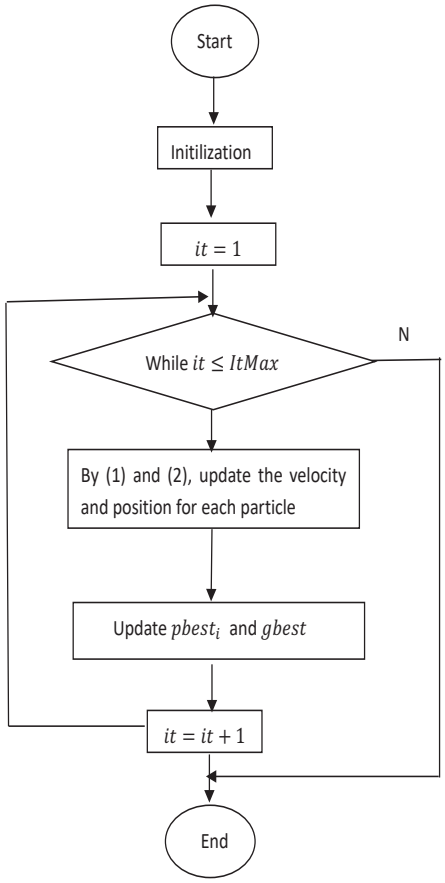


Fig. 1 Flowchart of PSO

The flowchart illustrates the iterative process of PSO, where the population of particles is updated until the termination condition is met.

III. Improved PSO based on mean prediction (MPSO)

Although the basic PSO has been proven to be effective in solving various problems, it does have some limitations that can hinder its performance. One of the major shortcomings is that it fails to fully utilize the information contained in the population, and leads to slow convergence speed.

The aim of MPSO is to improve PSO by integrating the collective experience of the previous population into the movement equation for each particle. This integration can guide each particle to search the promising regions.

A. Mean-based prediction method

The mean prediction method is a simple technique used in data analysis and prediction, which aims to estimate or predict future values by utilizing the mean value of different iterations. The process of mean prediction can be described as follows.

Let $gbest(t-2)$, $gbest(t-1)$, $gbest(t)$ be the current optimal positions of the whole population at times $t-2$, $t-1$, and t, respectively. Define $\widehat{g}$ as follows:

$$\widehat{g} = \frac{1}{3}(gbest(t-2) + gbest(t-1) + gbest(t)), \qquad (4)$$

which is a prediction value of the global optimal value at iteration $t+1$.

Based on $\widehat{g}$, we construct a new movement equation as follows:

$$v_i^{t+1} = wv_i^t + c_1 r_1 (pbest_i - x_i^t) + c_2 r_2 (gbest - x_i^t) + c_3 r_3 (\widehat{g} - x_i^t), \qquad (5)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1}. \qquad (6)$$

To reduce computational time, MPSO uses the mean-based prediction method only after a certain number of iterations.

Algorithm 2: Pseudo-code of MPSO

01:Initialize the population size ps, the maximum number of iterations $ItMax$, c_1 and c_2. For each particle i, initialize the position x_i, and velocity v_i; the minimum inertia weight ω_{min} and the maximum inertia weight ω_{max}; the minimum velocity v_{min} and the maximum velocity v_{max}; the number K of chaos search near $gbest$.
02: Set $pbest_i = x_i (i = 1, \cdots, ps)$ and find $gbest$, the iteration counter $it = 1$.
03: **While** $it < ItMax$ **do**
04: If $mod(it, 200) == 0$
05: For $i = 1$ to ps **do**
06: By (4)-(6), update the velocity and the position of each particle.
07: If $f(x_i) < f(pbest_i)$
08: $pbest_i = x_i$;
09: End if
10: If $f(x_i) < f(gbest)$
11: set $gbest = x_i$,
12: End if
13: End for
14:Else 15: For $i = 1$ to ps **do**
16: By (1)-(2), update the velocity and the position of each particle.
17: If $f(x_i) < f(pbest_i)$
18: $pbest_i = x_i$;
19: End if
20: If $f(x_i) < f(gbest)$
21: set $gbest = x_i$,
22: End if
23: End for
24: End if
25: By (7)-(9), generate K candidate solutions, and update $gbest$ by the greedy rule.
26: By (3), update the inertia weight ω.
27: $it = it + 1$
28: End while
29: Output the final result.

B. Chaos-based local search strategy

Generally speaking, the current optimal solution represents a location with significant potential, so strengthening the search around it can improve the accuracy of the solution. Additionally, to avoid premature convergence, the algorithm should be able to escape from local optima. To achieve these goals, MPSO incorporates K iterations of a chaos-based local search mechanism at the current best solution. The specific process is given as follows:

$$\begin{aligned} &ch(1) = rand, \\ &ch(k+1) = \tfrac{\alpha}{4} \times \sin(\pi \times ch(k-1)), \end{aligned} \qquad (7)$$

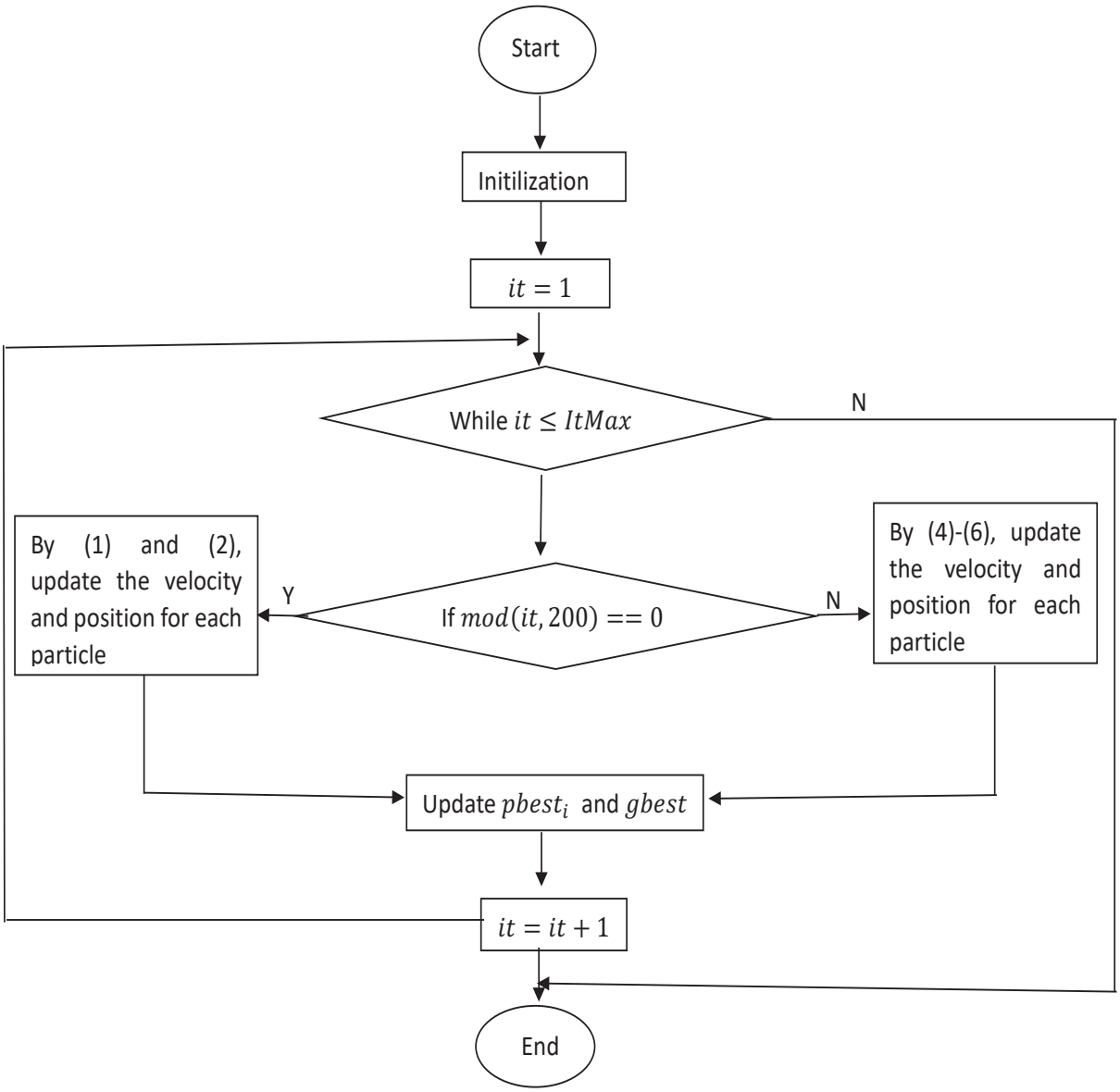


Fig. 2 Flowchart of MPSO

where $ch(k)$ is the kth chaotic number, K is the iteration counter, and it is set to 5 in this paper. In addition, α is set to 4, which means the chaotic search is in a complete chaotic state.

Mapping the above chaotic numbers to the search space as follows:

$$Ch(k) = l + ch(k) \times (u - l). \tag{8}$$

where l and u represent the lower and upper bounds of variables, respectively.

Then, a local search is carried out near the current best individual to improve the precision:

$$g'_{best} = (1 - \xi) \times g_{best} + \xi \times Ch(k), \tag{9}$$

where $\xi = \frac{ItMax - t + 1}{ItMax}$. The pseudo-code of MPSO is given in Algorithm 2.

The flowchart of MPSO algorithm is given in Fig. 2:

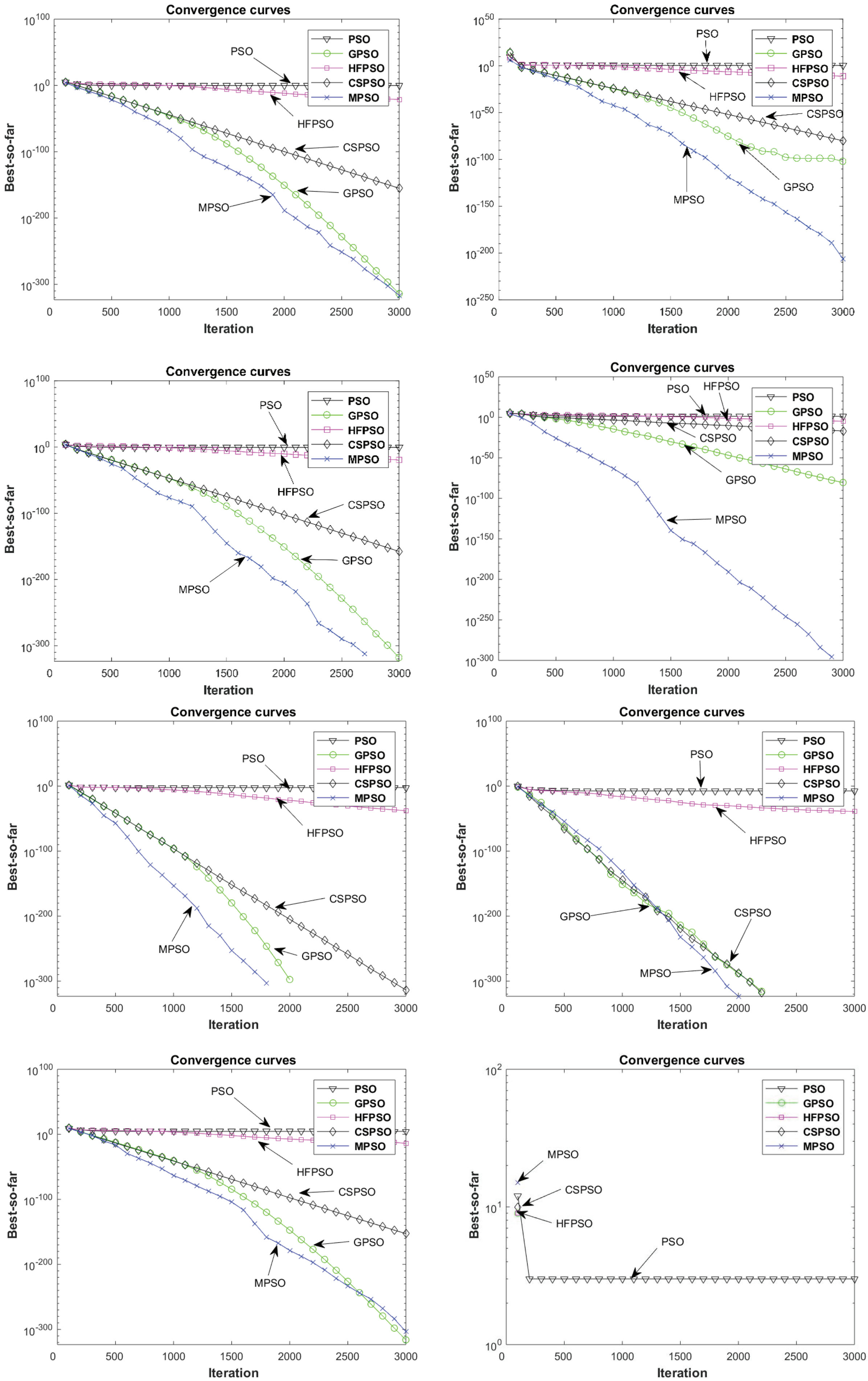
IV. Numerical experiments

To verify the performance of MPSO, it is compared with and four PSO variants: PSO [2], GPSO [11], HFPSO [13], and CSPSO [14] on 17 benchmark functions. The experiments are executed on a computer with an Intel (R) Core (TM) i7-6500U CPU @ 2.50 GHz, 16 GB memory, Windows 10 system, and the experiments are written in Matlab 2017a.

A. Benchmark functions

Table I shows the 17 benchmark functions. In Table I, D, range and optimal value are used to represent dimensions, bounds of the search space and global minimum values of these functions, respectively. Among these benchmark functions, f_1-f_{10} are unimodal functions, f_{11}-f_{15} are multimodal functions, f_{16}-f_{17} are two shifted functions.

For fair comparison, each algorithm runs independently 30 times on each function, and the population size is set to 50. The maximum number of iterations is set to 3000,


Convergence curves
Best-so-far
Iteration
PSO
GPSO
HFPSO
CSPSO
MPSO
PSO
HFPSO
CSPSO
GPSO
MPSO
Convergence curves
Best-so-far
Iteration
PSO
GPSO
HFPSO
CSPSO
MPSO
PSO
HFPSO
CSPSO
MPSO
GPSO
Convergence curves
Best-so-far
Iteration
PSO
GPSO
HFPSO
CSPSO
MPSO
PSO
HFPSO
CSPSO
GPSO
MPSO
Convergence curves
Best-so-far
Iteration
PSO
GPSO
HFPSO
CSPSO
MPSO
PSO
HFPSO
CSPSO
GPSO
MPSO
Convergence curves
Best-so-far
Iteration
PSO
GPSO
HFPSO
CSPSO
MPSO
PSO
HFPSO
CSPSO
MPSO
GPSO
Convergence curves
Best-so-far
Iteration
PSO
GPSO
HFPSO
CSPSO
MPSO
PSO
HFPSO
GPSO
CSPSO
MPSO
Convergence curves
Best-so-far
Iteration
PSO
GPSO
HFPSO
CSPSO
MPSO
PSO
HFPSO
CSPSO
GPSO
MPSO
Convergence curves
Best-so-far
Iteration
PSO
GPSO
HFPSO
CSPSO
MPSO
MPSO
CSPSO
HFPSO
PSO

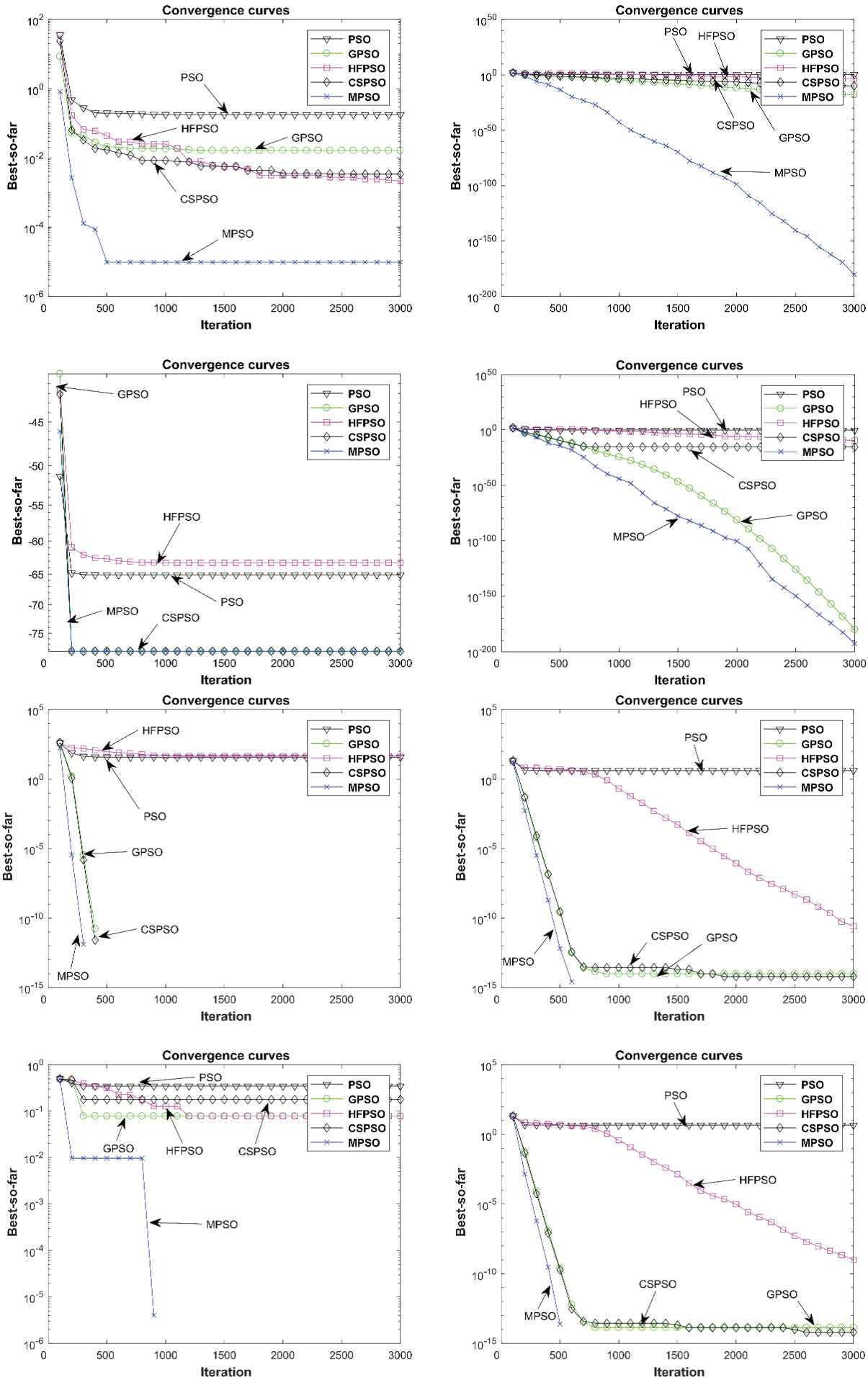


Fig. 3 The convergence curves of different algorithms

TABLE I: Benchmark test functions

Functions	Range	D	Optimal value
$f_1 = \sum_{i=1}^{D} x_i^2$	[-100,100]	30	0
$f_2 = \sum_{i=1}^{D} \mid x_i \mid + \prod_{i=1}^{D} \mid x_i \mid$	[-10,10]	30	0
$f_3 = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	[-100,100]	0	0
$f_4 = \sum_{i=1}^{D} i x_i^2$	[-10,10]	30	0
$f_5 = \sum_{i=1}^{D} i x_i^4$	[-1.28,1.28]	30	0
$f_6 = \sum_{i=1}^{D} \mid x_i \mid^{(i+1)}$	[-1,1]	30	0
$f_7 = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} x_i^2$	[-100,100]	30	0
$f_8 = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	[-1.28,1.28]	30	0
$f_9 = \sum_{i=1}^{D} i x_i^4 + random[0,1)$	[-1.28,1.28]	30	0
$f_{10} = \max\{\mid x_i \mid, 1 \leq i \leq n\}$	[-100,100]	30	0
$f_{11} = \frac{1}{D} \sum_{i=1}^{D} (x_i^4 - 16x_i^2 + 5x_i)$	[-5,5]	30	-78.332
$f_{12} = \sum_{i=1}^{D} \mid x_i sin(x_i) + 0.1x_i \mid$	[-10,10]	30	0
$f_{13} = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$	[-5.12,5.12]	30	0
$f_{14} = -20\exp(-0.2\sqrt{\sum_{i=1}^{D} x_i^2/D}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + e$	[-32,32]	30	0
$f_{15} = 0.5 + \frac{\sin(\sqrt{\sum_{i=1}^{D} x_i^2)^2 - 0.5})}{(1+0.001\sum_{i=1}^{D} x_i^2)^2}$	[-100,100]	30	0
$f_{16} = -20\exp(-0.2\sqrt{\sum_{i=1}^{D} z_i^2/D}) - \exp(\sum_{i=1}^{D}\cos(2\pi z_i/D) + 20 + e,\ z = Mx$	[-32,32]	30	0
$f_{17} = \sum_{i=1}^{D} (1000z_1)^2 + \sum_{i=2}^{D} z_i^2,\ z = Mx$	[-100,100]	30	0

TABLE II: The comparison results for different algorithms

Functions	PSO			GPSO			HFPSO			CSPSO			MPSO		
	Min	Mean	SD	Min	Mean	SD	Min	Mean	SD	Min	Mean	SD	Min	Mean	SD
f_1	8.93e-01	2.27e+00	1.24e+00	7.11e-174	7.11e-174	0	1.11e-16	4.42e-16	3.89e-16	8.56e-96	2.28e-95	1.38e-95	0	3.95e-302	0
f_2	9.88e-01	2.15e+00	1.03e+00	1.09e-121	1.09e-121	0	3.63e-12	8.79e-12	4.24e-12	3.81e-84	9.54e-84	7.19e-84	2.78e-213	4.37e-190	0
f_3	4.27e+00	1.59e+01	7.02e+00	9.48e-83	9.48e-03	0	1.29e-06	8.61e-06	6.56e-06	2.24e-19	4.09e-18	4.60e-18	0	5.02e-291	0
f_4	1.07e+00	2.39e+00	1.17e+00	0	0	0	6.59e-22	6.55e-21	4.64e-21	6.83e-165	2.30e-164	0	0	0	0
f_5	6.99e-05	9.85e-04	9.34e-04	0	0	0	1.52e-41	5.74e-39	7.52e-39	0	0	0	0	0	0
f_6	3.74e-10	2.85e-07	7.12e-07	0	0	0	3.04e-39	2.45e-36	6.77e-36	0	0	0	0	0	0
f_7	5.48e+02	3.91e+04	3.18e+04	0	0	0	3.71e-16	5.78e+03	1.36e+04	1.86e-158	2.21e-157	3.25e-157	0	0	0
f_8	1.00e+00	2.90e+00	1.72e+00	0	0	0	0	0	0	0	0	0	0	0	0
f_9	2.47e-02	1.20e-01	7.01e-02	1.73e-02	1.73e-02	0	1.04e-03	2.42e-03	1.21e-03	1.09e-03	2.17e-03	7.86e-04	1.65e-05	1.88e-04	1.58e-04
f_{10}	1.50e+00	2.50e+00	9.89e-01	6.84e-21	6.84e-21	0	2.52e-05	3.56e-04	3.81e-04	3.44e-11	1.30e-09	1.37e-09	3.32e-201	2.86e-184	0
f_{11}	-70.64	-67.27	2.20e+00	-78.33	-78.33	0	-68.90	-67.49	1.42e+00	-78.33	-78.33	1.49e-14	-78.33	-78.33	2.59e-08
f_{12}	1.65e-01	2.23e+00	2.15e+00	8.32e-16	8.32e-16	0	2.63e-12	3.99e-10	9.88e-10	1.28e-88	5.66e-16	5.39e-16	2.66e-215	4.11e-05	1.30e-04
f_{13}	3.12e+01	4.37e+01	1.16e+01	0	0	0	2.68e+01	4.01e+01	1.51e+01	0	0	0	0	0	0
f_{14}	2.69e+01	4.23e+01	1.09e+01	6.21e-15	6.21e-15	0	4.62e-12	9.39e-12	3.73e-12	6.21e-15	8.70e-15	3.37e-15	-8.88e-16	-8.88e-16	0
f_{15}	2.73e-01	3.68e-01	5.50e-02	1.27e-01	1.27e-01	0	7.82e-02	9.28e-02	2.36e-02	7.82e-02	1.38e-01	3.96e-02	0	2.92e-03	4.69e-03
f_{16}	2.50e+00	4.79e+00	1.51e+00	1.33e-14	1.33e-14	0	5.41e-11	1.25e-10	5.32e-11	9.41e-15	8.70e-15	3.37e-15	-8.88e-16	-8.88e-16	0
f_{17}	1.37e+00	4.54e+00	2.14e+00	2.96e-323	2.96e-323	0	2.16e-17	8.35e-17	3.91e-17	9.87e-161	4.47e-159	6.92e-159	0	0	0

which is also used as the termination condition. The other parameters are used as the comparison algorithms suggested. The comparison results are summarized in Table II.

B. Comparison results

Table II clearly demonstrates that MPSO outperforms the other four PSOs on almost all 17 benchmark functions. With the exception of f_3-f_7 and f_{12}, MPSO consistently achieves higher accuracy compared to the other algorithms.

In the case of f_3 and f_6, both MPSO and GPSO exhibit the same level of precision. Similarly, for f_4-f_5, MPSO, GPSO, and CSPSO demonstrate identical precision. For f_7, MPSO, GPSO, HFPSO, and CSPSO exhibit the same level of precision. Finally, for f_{12}, MPSO, GPSO, and CSPSO achieve the same precision.

To intuitively compare the convergence rate of MPSO and the other four PSOs, the convergence curves (benchmark functions f_1-f_{16}) of these algorithms are shown in Fig. 3. From Fig. 3, it is easy to see that MPSO can find a better solution when the algorithm terminates. For these functions, MPSO converges to the optimal solution at a faster speed. This implies that the mean prediction mechanism is effective.

V. Conclusion

In this paper, to provide a more favorable direction, we introduced empirical knowledge into PSO algorithm, which help the particles find the candidates with high quality. Numerical experiments showed the mechanism is very useful. In the next step, this method will be used to solve some practical problems.

References

[1] C.F. Wang, P.P. Shang, L.X. Liu, "Improved artificial bee colony algorithm guided by experience," *Engineering Letters*, vol. 30, no. 1 pp. 261-265, 2022.

[2] J. Kennedy, R. Eberhart, "Particle swarm optimization," *in Proceedings of the IEEE International Conference on Neural Networks*, Perth, vol. 4, pp. 1942-1948, 1995.

[3] M.A. Abido, "Optimal power flow using particle swarm optimization", *International Journal of Electrical Power and Energy Systems*, vol. 24, no. 7, pp. 563-571, 2002.

[4] P. Das, H.S. Behera, B.K. Panigrahi, "A hybridization of an improved particle swarm optimization and gravitational search algorithm for multi-robot path planning", *Swarm and Evolutionary Computation*, vol. 28, pp. 14-28, 2016.

[5] Q. Qin, S. Cheng, X. Chu, X. Lei, Y. Shi, "Solving non-convex/non-smooth economic load dispatch problems via an enhanced particle swarm optimization", *Applied Soft Computing*, vol. 59, pp. 229-242, 2017.

[6] R. Chaudhry, S. Tapaswi, N. Kumar, "FZ Enabled multi-objective PSO for multicasting in IoT based wireless sensor networks", *Information Science*, vol. 498, pp. 1-20, 2019.

[7] J.A.J. Sujana, T. Revathi, T.S.S. Priya, K. Muneeswaran, "Smart PSO-based secured scheduling approaches for scientific workflows in cloud computing", *Soft Computing*, vol. 23, pp. 1745-1765, 2019.

[8] A. Alizadeh Naeini, M. Babadi, S.M.J. Mirzadeh, S. Amini, "Particle swarm optimization for object-based feature selection of VHSR satellite images", IEEE Geosci. Remote Sens. Lett. vol. 15, pp. 379-383, 2018.

[9] R. Sheikhpour, M.A. Sarram, R. Sheikhpour, "Particle swarm optimization for bandwidth determination and feature selection of kernel density estimation based classifiers in diagnosis of breast cancer", *Applied Soft Computing*, vol. 40, pp. 113-131, 2016.

[10] C.F. Wang, P.P. Shang, X.D. Wu, "Hierarchical particle swarm optimization based on mean value," *IAENG International Journal of Applied Mathematics*, vol. 53, no. 2, pp. 695-703, 2023.

[11] Y. Chen, L. Li, et al. "Particle swarm optimizer with crossover operation", *Engineering Applications of Artificial Intelligence*, vol. 70, pp. 159-69, 2018.

[12] C.F. Wang, W.X. Wen, "A modified particle swarm optimization algorithm based on velocity updating mechanism", *Ain Shams Engineering Journal*, vol. 10, pp. 847-866, 2019.

[13] I.B. Aydilek, "A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems", *Applied Soft Computing*, vol. 66, pp. 232-249, 2019.

[14] W.F. Gao, S.Y. Liu, L.L. Huang, "Particle swarm optimization with chaotic opposition-based population initialization and stochastic search technique", *Communications in Nonlinear Science and Numerical Simulation*, vol. 7, no. 11, pp. 4316-4327, 2012.