Performance of Composite PPSO on Single Objective Bound Constrained Numerical Optimization Problems of CEC 2022

Bo Sun
School of Information Engineering
JiangXi University of Science and Technology
Ganzhou, China
sunbo@mail.jxust.edu.cn

Wei Li
School of Information Engineering
JiangXi University of Science and Technology
Ganzhou, China
liwei@jxust.edu.cn

Ying Huang
School of Mathematical and Computer Science
Gannan Normal University
Ganzhou, China
nhwshy@whu.edu.cn

***Abstract*—Particle swarm optimization has been extensively noticed for its fast convergence speed with few parameters. However, it would be plagued by the premature convergence only affected by the global particles. In this study, Composite Proactive Particles in Swarm Optimization (Co-PPSO) is proposed. In Co-PPSO, the composite strategy framework is embedded into Proactive Particles in Swarm Optimization (PPSO), that is, three learning strategies are proposed to evaluate their differences and select the most suitable one for each particle. In addition, an elite group is constructed to make the particles jump out of the situation that they are only affected by the global best one in the particle swarm, and further improve the convergence accuracy. CEC2022 competition of single objective bound-constrained numerical optimization is employed to test the effect of 10-*D* and 20-*D* optimization, and four well-known PSO variants were used for comparison. The experimental results show that the Co-PPSO has certain competitiveness to improve premature convergence.**

***Index Terms*—particle swarm optimization, composite strategy framework, elite group, CEC2022**

I. INTRODUCTION

Particle Swarm Optimization (PSO) is a meta-heuristic for global optimization that uses a population-based approach. It was inspired by the flocks of birds moving together [1]. According to numerous research, the PSO is more efficient and reliable in obtaining the optimal solution. Moreover, the PSO is well known for its few parameters and easy implementation. Different parameter settings, as we all know, play a critical part in the PSO's optimization. In the early research work, the impact of the linear decline of inertia weight on global exploration and local exploitation capacity is analyzed [2]. In a later study on the linearly decreasing inertia weight, a variation of the inertia weight from 0.9 to 0.4 resulted in an improved performance [3]. Some researchers have proposed an algorithm with time-varying control parameters. Based on the classic PSO, the approach includes the notion of mutation and constructs a time-varying acceleration coefficient and a time-varying inertia weight factor to effectively regulate the algorithm's local search and get the global best solution. [4].

There is an effective way to choose the best parameter setting for solving complex optimization problems, i.e., the fuzzy system will be used to evaluate and guide particle behavior. To solve dynamic optimization challenges, a fuzzy system was designed to modify the inertia weight [5]. The effectiveness of the personal history best candidate solution and the current inertia weight are utilized as inputs to the fuzzy system in each iteration to evaluate the next iteration inertia weight of the entire particle swarm in its fuzzy version. Furthermore, to overcome the problem of premature convergence, a Fuzzy Adaptive Turbulence in Particle Swarm Optimization was presented. [6]. A fuzzy logic controller integrated with the turbulence of the PSO algorithm adaptively tunes the parameter and the lowest velocity threshold of the particles. The PSO algorithm, as an optimizer, was employed to search for a set of optimal fuzzy rule weights [7]. The value of each weight is obtained as a dimension in particle swarm optimization, and the particle thus represents a possible solution to the given set of weight values. Using PSO for searching a set of optimal weight values, the accuracy of classification is improved. Besides, the fuzzy logic was utilized to propose a proactive particles in swarm optimization (PPSO) [8]. The algorithm utilizes fuzzy logic to evaluate the inertia weight, individual cognitive factor, and social cognitive factor. Therefore, the algorithm does not need to manually set the parameters. Moreover, the algorithm automatically adjust parameters according to the particle history information. Later, a fuzzy self-tuning PSO was proposed [9]. It uses fuzzy logic to calculate the inertia, cognitive and social factors, as well as the particle's minimum and maximum velocities, independently. The use of fuzzy rules to dynamically set parameters can significantly increase particle swarm optimization's searchability.

The above-improved algorithms are the adjustment of PSO parameters, which can be promoted to the optimization affected by PSO. At the same time, some studies have shown that fuzzy logic is a viable way of adjusting particle swarm optimization settings [7, 8, 9]. However, the above-mentioned algorithm ignores some other factors such as the particle speed is only affected by individual history information and global optimal particle information that are easy to premature the convergence of PSO.

In this study, by considering the influence of parameter setting and information which is obtained by other populated particles the Co-PPSO is proposed. The composite strategy framework is embedded into the PPSO to enhance and balance the local exploitation and global exploration abilities. Furthermore, an elite group seeks to make the population avoid the circumstance of learning only from global best particles and lower the probability of population premature convergence.

The remainder of this study's chapters are organized as follows. The background of PSO and variants are studied in Section II. In Section III, the proposed Co-PPSO is thoroughly described. The CEC2022 competition on single objective bound constrained numerical optimization are tested

experimentally in Section IV. The prospects of the research work is summarized in SectionV.

II. BACKGROUND

A. Prticle Swarm Optimization

Each particle of the PSO has two vectors (velocity V_i = (v_{i1}, v_{i2}, ..., v_{iD}) and position X_i = (x_{i1}, x_{i2}, ..., x_{iD}). For a problem in *D*-dimensional space, a particle *i* which with the position has a velocity when the particle is moving. During the search process, the velocity and the position can be updated by the Eq. (1) and Eq. (2):

$$V_i^{t+1} = \omega \cdot V_i^t + c_{cog} \cdot r_1 \cdot (Pbest_i^t - X_i^t) + c_{soc} \cdot r_2 \cdot (Gbest - X_i^t) \quad (1)$$

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

where V_i^t is the velocity of the *i-th* particle, X_i^t is the current position. ω is the inertia weight, r_1 and r_2 are two random numbers, and the range is [0, 1]. The role of random numbers is to maintain population diversity, c_{cog} and c_{soc} denote cognitive factor and social factor. $Pbest_{ij}^t$ = ($pbest_{i1}$, $pbest_{i2}$, ..., $pbest_{iD}$) is the individual historical optimal position. *Gbest* is the global optimal position of the population.

B. PPSO

Nobile et al. proposed an approach that the proactive particles in swarm optimization had their inertia weight, cognitive and social factors [8]. Compared with the standard PSO, The algorithm's convergence efficiency was enhanced. In PPSO, each particle uses a specific setting determined by utilizing a fuzzy rule-based system approach to make the simple reactive individual of the PSO an active optimization agent. Hence, the velocity updating equation is modified as follows:

$$V_i^{t+1} = \omega_i^t \cdot V_i^t + c_{cog_i^t} \cdot r_1 \cdot (X_i^t - Pbest_i^t) + c_{soc_i^t} \cdot r_2 \cdot (X_i^t - Gbest_i^t) \quad (3)$$

where ω_i^t , $c_{cog_i^t}$, and $c_{soc_i^t}$ denote, respectively, the inertia, cognitive, social factors of *i-th* particle during the iteration of *t*, respectively. Three elements are dynamically calculated using fuzzy rules based on two fundamental concepts: the particle's distance from the *Gbest* and a function that assesses the particle's fitness improvement from the initial state. The distance between particles *i* and *j* can be calculated by Eq. (4):

$$\delta(x_i(t), x_j(t)) = \left\| x_i(t) - x_j(t) \right\|_2 = \sqrt{\sum_{d=1}^{D} (x_{i,d}(t) - x_{j,d}(t))^2} \quad (4)$$

where $x_{i,d}(t)$, $x_{j,d}(t)$ denote the he *d-th* dimension of the position X_i, X_j. The normalized fitness incremental factor function can be calculated by Eq. (5):

$$\phi(x_i(t), x_i(t-1)) = \frac{\min\{f(x_i(t), f_\Delta)\} - \min\{f(x_i(t-1), f_\Delta)\}}{|f_\Delta|} \cdot \frac{\delta(x_i(t), x_i(t-1))}{\delta_{\max}} \quad (5)$$

where $\delta_{\max}$ is the diagonal length of the hyper-rectangle defined by the search space. Considering the optimization problem under inquiry, $|f_\Delta|$ denotes the predicted lowest fitness value.

The calculated values by the above-mentioned functions are used as the input of fuzzy rules to obtain three output parameters. These three parameters correspond to three different linguistic values which are Low, Medium, and High. Then, the Sugeno inference method [10] is used to calculate the final numerical value of the output variable in Eq. (6) as the weighted average of all output of all rules.

$$output = \frac{\sum_{r=1}^{R} \rho_r z_r}{\sum_{r=1}^{R} \rho_r} \quad (6)$$

where ρ_r is the input variable's membership degree of the *r*-th rule. As stated in TABLE II, z_r signifies the matching output value.

In Eq. (1) and Eq. (3). The PPSO is slightly different from classical PSO. This study makes the fuzzy rules based on Eq. (1). Hence, the parameter setting of the membership function used in this study is slightly different from PPSO, so the fuzzy rules and membership function are extensively described in detail in CO-PPSO section. The algorithm flow of PPSO is summarized as the following:

Step1: Calculate Eq. (4) and Eq. (5) to obtain the δ_i and ϕ_i (i is 1, 2, …, N, N is the population size).

Step2：Take δ_i and ϕ_i as inputs into the membership function to calculate the membership of δ_i and ϕ_i .

Step3：The membership of δ_i and ϕ_i are brought into the fuzzy rules to get three output variables.

Step4：Calculate Eq. (6) to get the final parameters of each particle.

III. PROPOSED CO-PPSO

A. Fuzzy Rules and Membership Function

In PPSO, the Eq. (4) and Eq. (5) are used to calculate each particle, corresponding to two input variables respectively δ_i , ϕ_i . Thus, nine fuzzy rules as shown in TABLE I are formulated, corresponding to three output variables respectively: $Inertia_i$, $Cognitive_i$, $Social_i$.

TABLE I. FUZZY RULES

Rule	Output variable	Linguistic values	Rule definition
1	$Inertia_i$	Low	If ϕ_i is Worse or δ_i is Medium or δ_i is High
2		Medium	If ϕ_i is Unvaried or δ_i is Low
3		High	If ϕ_i is Better
4	$Cognitive_i$	Low	If δ_i is High
5		Medium	If ϕ_i is Unvaried or ϕ_i is Worse or δ_i is Low or δ_i is Medium
6		High	If ϕ_i is Better
7	$Social_i$	Low	If ϕ_i is Better or δ_i is Medium
8		Medium	If ϕ_i is Unvaried
9		High	If ϕ_i is Worse or δ_i is Low or δ_i is High

TABLE II shows the precise output weights corresponding to the fuzzy inputs. In this study, the output values corresponding to cog_i are adjusted to 1, 2, and 3 respectively, which are different from the PPSO.

TABLE II. OUTPUT VARIABLES AND THEIR DEFUZZIFICATION

Output variable	Linguistic values	values
$Inertia_i$	Low	0.3
	Medium	0.5
	High	1
$Cognitive_i$	Low	1
	Medium	2
	High	3
$Social_i$	Low	0.1
	Medium	1.5
	High	3

For linguistic values of the distance from the global best *Gbest*, the following functions can be stated as follows:

$$Low = \begin{cases} 1, 0 < \delta_i < \delta_1 \\ (\delta_2 - \delta_i)/(\delta_2 - \delta_1), \delta_1 < \delta_i < \delta_2 \\ 0, \delta_2 < \delta_i < \delta_{\max} \end{cases} \quad (7)$$

$$Medium = \begin{cases} 0, 0 < \delta_i < \delta_1 \\ (\delta_i - \delta_1)/(\delta_2 - \delta_1), \delta_1 < \delta_i < \delta_2 \\ (\delta_3 - \delta_i)/(\delta_3 - \delta_2), \delta_2 < \delta_i < \delta_3 \\ 0, \delta_3 < \delta_i < \delta_{\max} \end{cases} \quad (8)$$

$$High = \begin{cases} 0, 0 < \delta_i < \delta_2 \\ (\delta_i - \delta_2)/(\delta_3 - \delta_2), \delta_2 < \delta_i < \delta_3 \\ 1, \delta_3 < \delta_i < \delta_{\max} \end{cases} \quad (9)$$

where, $\delta_{\max}$ in this study is the distance between the best and worst particles in each generation. The notion of a fuzzy length between the best particle and the global best is described according to δ_1, δ_2, and δ_3. In this study, we refer to the value set of FST-PSO [9], which $\delta_1 = 0.2 \cdot \delta_{\max}$, $\delta_2 = 0.4 \cdot \delta_{\max}$, $\delta_3 = 0.6 \cdot \delta_{\max}$. The numbers δ1, δ2, and δ3 are determined by the size of the search space. These can be used to apply a broad and nebulous definition of distance from the global optimal solution.

The following membership functions are used to normalize the fitness increment factor variable as can be described as the following:

$$Better = \begin{cases} 1, \phi_i = -1 \\ -\phi_i, -1 < \phi_i < 0 \\ 0, 0 \le \phi_i \le 1 \end{cases} \quad (10)$$

$$Un\,var\,ied = \begin{cases} 0, -1 \le \phi_i < \phi_1 \\ (\phi_1 - \phi)/\phi_1, \phi_1 \le \phi_i < 0 \\ (\phi_2 - \phi)/\phi_2, 0 \le \phi_i < \phi_2 \\ 0, \phi_2 \le \phi \le 1 \end{cases} \quad (11)$$

$$Worse = \begin{cases} 0, -1 < \phi_i < 0 \\ \phi_i, 0 \le \phi_i < 1 \\ 1, \phi_i = 1 \end{cases} \quad (12)$$

where ϕ_1, and ϕ_2 are set to $\phi_1 = -0.0025$, and $\phi_2 = 0.0025$, respectively.

B. Co-PPSO

In this study, the composite strategy framework is proposed by inspiring CoDE [11]. The principles of the composite strategy framework is proposed. The principles followed by the composite strategy framework are shown in Fig. 1. Strategy 1 is used to improve the population's global exploration ability, Strategy 2 is used to improve the population's exploitation ability, and Strategy 3 is used to balance the two abilities. With the mentioned motivation, three learning strategies are proposed in this study. Each particle performs three learning strategies to pre-move. By evaluating the pre-moved particles, the suitable particles are retained.

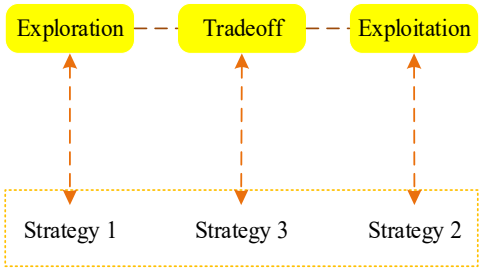


Fig. 1. Motivation of the designed search algorithm

An elite group for particle swarm optimization is created in each iteration to get out of the situation of learning exclusively from the global best particles in the particle swarm optimization process. Considering that if the size of the elite population is large, the effect of improving the convergence speed cannot be achieved. If smaller, it may cause rapid population convergence leading to premature convergence. To construct the elite group, the fitness values of particles are sorted. The top five particles of sorted set are selected as the initial elements of elite group. Within the iteration process, the scale of the elite group gradually decreases, and the final scale of the elite group is becomes two. This method can increase the diversity of the elite groups in the early stage. In the later stage, the population tends to local exploitation. The scaled change equation of the elite group can be expressed in Eq. (13):

$$elite\_group = \lfloor 5 - itrn \cdot (3 / Ma\,x\,itrn) \rfloor \quad (13)$$

where, *itrn* denotes the current iterations, and *Maxitrn* is the maximum iterations.

The first learning strategy is to improve the global exploration ability of the population, mainly for particles in inferior positions and reduce the influence of historical experience. In this study, the generation range of rand value is limited to [0.5, 1] which enhances the influence of elite particles. It causes the particles in the inferior position move towards the elite particles quickly to increase the convergence speed of particles. The velocity update formula is described in Eq. (14):

$$V1_i^{t+1} = \omega_i \cdot V_i^t + c_{soc} \cdot (1 - 0.5 \cdot rand) \cdot (elite_1 - X_i^t) \quad (14)$$

where $elite_1$ is an elite particle which is randomly selected from the elite group. It is worth noting that the elite group can be updated in each iteration.

The second learning strategy is to enhance the local exploitation ability of particle swarm. This strategy reduces the influence of elite particles and tend to a step-by-step exploitation. Similarly, in this study, the generation range of rand value is limited to [0.5, 1], so as to enhance the influence

of their own historical experience. The velocity update formula is described in Eq. (15):

$$V2_i^{t+1} = \omega_i^j \cdot V_i^t + c_{cog} \cdot (1 - 0.5 \cdot rand) \cdot (Pbest_i^t - X_i^t) \quad (15)$$

The third learning strategy is proposed to balance the global exploration and the local exploitation of particle swarm. It maintains a balance based on the the Strategies 1 and 2. The velocity update formula is described in Eq. (16):

$$V3_i^{t+1} = \omega_i^j \cdot V_i^t + c_{cog} \cdot rand \cdot (Pbest_i^t - X_i^t) + c_{soc} \cdot rand \cdot (elite_2 - X_i^t) \quad (16)$$

where, $elite_2$ is an elite particle which is randomly selected from the elite group. The use of elite particles in Eq. (16) greatly minimizes the possibility of particle swarms falling into local optimization.

For a better understanding, a schematic is drawn in Fig. 2 which shows the composite strategy framework. Three learning strategies are implemented for the particle X_i to obtain three pre-moved particles: ($X1_i$, $X2_i$ and $X3_i$). The fitness values of the three particles are evaluated. It is concluded that the Strategy 3 is suitable for X_i. Therefore, the particles executed by Strategy 3 are retained.

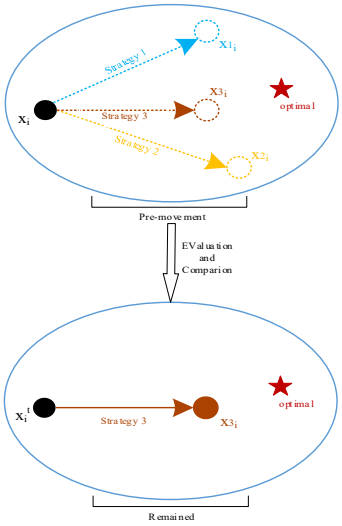


Fig. 2. The framework of composite strategy

IV. RESULTS

A. Experimental Setting

In this section, the performance of Co-PPSO is discussed on the CEC2022 test suits [12]. The test set includes twelve test functions, and contains four categories as TABLE III. TABLE III contains function categories, function names, and optimal values. Where the F1 function is unimodal functions, the F2-F5 functions are basic functions, the F6-F8 functions are hybrid functions, and the F9-F12 functions are composition functions. The search scope is [-100, 100]^D. Test the functions in 10-*D* and 20-*D* respectively and the global optima of the functions are Recorded in TABLE III [12]. In TABLE III, the F1-F5 functions are obtained by a function through a shift and partial or complete rotation. The variables are randomly separated into certain subcomponents on the F6-F8 functions, and then different basic functions are applied for different subcomponents. And the number of basic functions contained in each hybrid function is different. On the F9-F10 functions, They better combine the features of the sub-functions and keep the global or local optima consistent. And shifted and rotated functions are the basic functions that have been used in composition functions. This makes the functions more complex and poses a greater challenge to the performance of the algorithm.

In this study, each function runs 30 times independently. The termination criteria for the evaluation of the objective function for each run is a maximum of 200000 for 10-*D* and 1000000 for 20-*D* (*MaxFES*). These experiments are performed on a computer with a Win_64 bit, Intel (R) Core (TM) i5 - 6300hq CPU@ 2.30 GHz and 8 GB RAM. The version of MATLAB is MATLAB R2019a, which was used to follow through the relevant experimental. For this competition, the population size N=100 for D=10 and N=200 for D=20.

TABLE III. SUMMARY OF CEC2022 SINGLE OBJECTIVE BOUND CONSTRAINED NUMERICAL OPTIMIZATION

	No.	Functions	F_i^*
Unimodal Function	1	Shifted and full Rotated Zakharov Function	300
Basic Functions	2	Shifted and full Rotated Rosenbrock's Function	400
	3	Shifted and full Rotated Expanded Schaffer's f6 Function	600
	4	Shifted and full Rotated Non-Continuous Rastrigin's Function	800
	5	Shifted and full Rotated Levy Function	900
Hybrid Functions	6	Hybrid Function 1 (N = 3)	1800
	7	Hybrid Function 2 (N = 6)	2000
	8	Hybrid Function 3 (N = 5)	2200
Composition Functions	9	Composition Function 1 (N = 5)	2300
	10	Composition Function 2 (N = 4)	2400
	11	Composition Function 3 (N = 5)	2600
	12	Composition Function 4 (N = 6)	2700
Search range: [−100,100]^D			

B. Optimization Performance

The performance of Co-PPSO on the test set through error (the difference between the global optimal value determined by the algorithm and the objective function value) is analyzed in this section. The error values obtained for 10-*D*, 20-*D* are tabulated in TABLE IV and V. The error was evaluated as 0 when the difference between the ideal solution and the best solution found by the method was 1E-8 or less.

In TABLE IV, the F1 function of Co-PPSO on 10-*D* has good robustness and has excellent performance. Within the basic functions, although the F2 function fails to get the optimal value, the convergence accuracy has been improved to a certain extent. Its convergence accuracy can reach 1e-6. This indicates its poor stability. The F3 and F5 functions have excellent performance and can get the optimal value. The five indicators of the F3 and F5 functions have achieved the best results. Within the hybrid functions, the error value shows that the convergence accuracy of the F7 function can get the optimal value. However, it still lacks certain stability. The F8 function also has a certain improvement effect on convergence accuracy. Its convergence accuracy can reach 1e-1. Within the composition functions, the F11 can get the optimal value. TABLE III shows that the F11 function is composed of five complex functions, which shows that Co-PPSO has certain

advantages in solving complex optimization problems. However, running the F11 function 30 times, sometimes did not reach the actual optimal value. This also shows that the stability of Co-PPSO on the F11 function is insufficient. The convergence accuracy of the other three composition functions is similar. The convergence accuracy of the F9, F10, F12 functions has reached 1E + 2. Although the three functions can not get the optimal value, they have a certain global exploration ability. The main reason is that the composition functions are composed of different numbers of complex functions, and each composite function has a considerable number of local optimal values, which is a challenge to the optimization ability of Co-PPSO.

In TABLE V. the F1 function of Co-PPSO on 20-*D* still has good robustness. Within the basic functions, the F3 and F5 functions can get the optimal value, but their stability is not good. The worst accuracy of the F3 function is 1e-3, which shows that the algorithm sometimes fails to escape the local optimum. Like the F3 function, some results of the F5 function fail to escape the local optimum. This shows that with the increase of dimension, the search difficulty also increases, which is a test for the optimization ability of the algorithm. Within the hybrid functions, the convergence accuracy of the other three composition functions is similar. However, the performance of the F7 function in 20-*D* is inferior to that in 10-*D*, failing to search for the actual optimal value. Within the composition functions, It's worth mentioning that the F9 function performs better in 20-*D* than it does in 10-*D*, which shows that the proposal of Co-PPSO is suitable for the high-dimensional search of the F9 function. The F11 function has good performance in convergence accuracy and can get the optimal value as that in 10-*D*. The F12 function failed to get the optimal solution, mainly because the function form is particularly complex, which is composed of six functions, and there are many local optimal values. The F12 function still shows some advantages in global exploration ability.

TABLE IV. PERFORMANCE ON 10-*D*

Func.	Best	Worst	Median	Mean	Std.
F1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F2	5.57E-06	9.04E+00	1.71E-03	1.70E+00	2.42E+00
F3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F4	1.99E+00	1.39E+01	6.96E+00	6.70E+00	2.60E+00
F5	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F6	7.00E+00	1.76E+03	1.11E+02	3.33E+02	4.30E+02
F7	2.04E-08	2.20E+01	1.80E+00	8.84E+00	9.81E+00
F8	1.02E-01	2.24E+01	2.06E+01	1.54E+01	9.20E+00
F9	2.30E+02	2.30E+02	2.30E+02	2.30E+02	8.70E-02
F10	1.00E+02	1.00E+02	1.00E+02	1.00E+02	6.54E-02
F11	0.00E+00	3.00E+02	0.00E+00	2.50E+01	7.96E+01
F12	1.63E+02	1.65E+02	1.65E+02	1.65E+02	4.31E-01

TABLE V. PERFORMANCE ON 20-*D*

Func.	Best	Worst	Median	Mean	Std.
F1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F2	1.40E+01	1.94E+01	1.65E+01	1.66E+01	1.16E+00
F3	0.00E+00	1.48E-03	2.54E-06	6.48E-05	2.71E-04
F4	5.97E+00	3.38E+01	1.74E+01	1.92E+01	6.02E+00
F5	0.00E+00	4.00E+00	8.95E-02	5.36E-01	9.21E-01
F6	4.41E+01	2.18E+04	3.91E+03	5.59E+03	5.62E+03
F7	2.03E+01	4.88E+01	2.67E+01	2.96E+01	7.31E+00
F8	2.04E+01	2.64E+01	2.14E+01	2.18E+01	1.29E+00
F9	1.80E+02	1.82E+02	1.80E+02	1.80E+02	3.76E-01
F10	1.00E+02	4.89E+02	1.00E+02	1.17E+02	7.37E+01
F11	0.00E+00	4.00E+02	3.00E+02	3.13E+02	9.73E+01
F12	1.95E+02	1.99E+02	1.96E+02	1.96E+02	1.00E+00

C. Comparisons with advanced PSO algorithms

To verify the performance of Co-PPSO in solving the CEC2022 test suite, 4 advanced comparison algorithms are used to compare with Co-PPSO. These algorithms are recently proposed state-of-art algorithms that attract many researchers' attention：PPSO [8], DSPSO [13], XPSO [14], MPSO [15]. The parameters settings and the experimental setup for the peer algorithms are the same as that in the corresponding papers. For fairness, each algorithm was run 30 times independently and the mean and standard deviation were recorded.

TABLE VII is the comparison result of 5 algorithms in 10-*D*, TABLE VIII is the comparison result of 20-*D*. From the 10-*D* statistical results, it can be seen that Co-PPSO ranks first in 7 functions, and DSPSO also ranks first in 7 functions. But it is not difficult to find that the performance of DSPSO is mainly manifested in unimodal functions and basic functions. However, Co-PPSO not only performs well on unimodal and basic functions, but also outperforms the other four algorithms on hybrid functions and composition functions. More importantly, the statistical results show that the solution accuracy of Co-PPSO is better than PPSO in both 10- and 20-*D*.

TABLE VI. PARAMETER SETTINGS OF THE 5 COMPARISON PSO VARIANTS.

No.	Algorithm	Year	Parameter settings
1	PPSO	2017	N=floor(10+sqrt(D))
2	DSPSO	2019	c_1=2.0, F_{min}=0.7
3	XPSO	2020	η=0.2, $Stag_{max}$=5, p=0.2
4	MPSO	2020	c_1=c_2=2.0
5	Co-PPSO		N=100 (10-D), 200 (20-D)

TABLE VII. *COMPARISONS WITH COMPONENT ALGORITHMS ON 10-D*

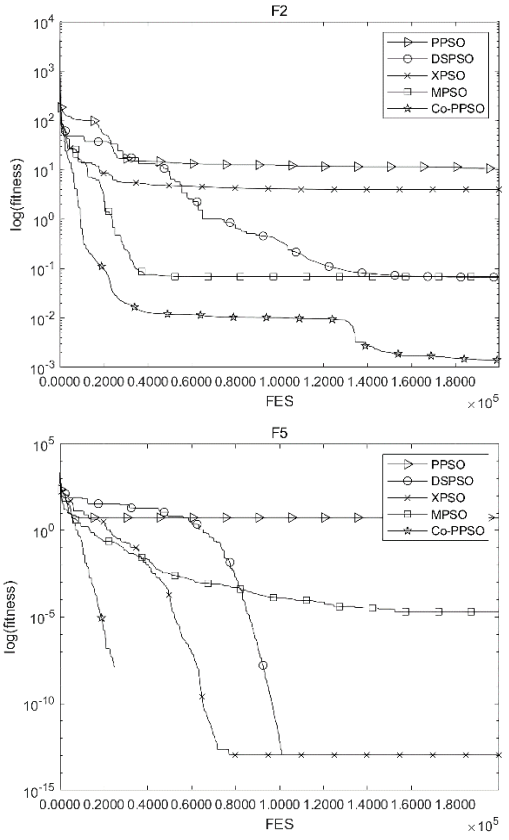
Func.		PPSO	DSPSO	XPSO	MPSO	Co-PPSO
F1	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std.	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Rank	1	1	1	1	1
F2	Mean	3.55E+01	**1.10E+00**	5.07E+00	5.41E+00	1.70E+00
	Std.	3.34E+01	2.76E+00	3.70E+00	1.29E+01	**2.42E+00**
	Rank	5	1	3	4	2
F3	Mean	2.10E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std.	2.68E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Rank	2	1	1	1	1
F4	Mean	2.23E+01	**1.89E+00**	7.16E+00	1.01E+01	6.70E+00
	Std.	7.60E+00	**1.42E+00**	2.82E+00	5.13E+00	2.60E+00
	Rank	5	1	3	4	2
F5	Mean	1.05E+01	**0.00E+00**	2.98E-03	1.81E-01	**0.00E+00**

	Std.	1.17E+01	**0.00E+00**	1.63E-02	3.92E-01	**0.00E+00**
	Rank	4	1	2	3	1
	Mean	1.77E+03	6.31E+02	1.16E+03	**1.20E+02**	3.33E+02
F6	Std.	1.71E+03	5.37E+02	1.24E+03	**2.78E+02**	4.30E+02
	Rank	5	3	4	1	2
	Mean	2.44E+01	7.66E+00	**7.53E+00**	1.19E+01	8.84E+00
F7	Std.	**7.72E+00**	9.10E+00	9.52E+00	9.82E+00	9.81E+00
	Rank	5	2	1	4	3
	Mean	6.72E+01	1.74E+01	1.75E+01	1.75E+01	**1.54E+01**
F8	Std.	5.93E+01	7.84E+00	7.83E+00	**7.14E+00**	9.20E+00
	Rank	4	2	3	3	1
	Mean	2.47E+02	**2.30E+02**	2.31E+02	2.29E+02	**2.30E+02**
F9	Std.	2.81E+01	3.62E-01	2.36E-01	9.21E-02	**8.70E-02**
	Rank	4	1	3	2	1
	Mean	1.90E+02	1.50E+02	1.58E+02	1.67E+02	**1.00E+02**
F10	Std.	1.23E+02	5.37E+01	5.51E+01	5.53E+01	**6.54E-02**
	Rank	5	2	3	4	1
	Mean	2.44E+02	**2.00E+01**	7.67E+01	8.53E+01	2.50E+01
F11	Std.	1.50E+02	**7.61E+01**	1.43E+02	1.09E+02	7.96E+01
	Rank	5	1	3	4	2
	Mean	1.97E+02	1.66E+02	**1.65E+02**	1.70E+02	**1.65E+02**
F12	Std.	3.46E+01	1.08E+00	6.24E-01	8.97E+00	**4.31E-01**
	Rank	4	2	1	3	1
	Ave Rank	4.08	1.50	2.33	2.83	1.50
	Total Rank	5	1	3	4	1

TABLE VIII. *COMPARISONS WITH COMPONENT ALGORITHMS ON 20-D*

Func.		PPSO	DSPSO	XPSO	MPSO	Co-PPSO
	Mean	4.04E-02	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F1	Std.	2.06E-01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Rank	2	1	1	1	1
	Mean	1.66E+02	4.93E+01	4.85E+01	3.63E+01	**1.66E+01**
F2	Std.	1.10E+02	**1.83E-01**	1.92E+01	2.10E+01	1.16E+00
	Rank	5	4	3	2	1
	Mean	1.38E+01	**0.00E+00**	**0.00E+00**	4.84E-02	6.48E-05
F3	Std.	6.88E+00	**0.00E+00**	**0.00E+00**	2.18E-01	2.71E-04
	Rank	4	1	1	3	2
	Mean	6.30E+01	**6.96E+00**	2.20E+01	3.23E+01	1.92E+01
F4	Std.	1.76E+01	**2.56E+00**	1.01E+01	9.79E+00	6.02E+00
	Rank	5	1	3	4	2
	Mean	5.06E+02	**3.03E-02**	8.88E-01	2.12E+01	5.36E-01
F5	Std.	3.97E+02	**1.15E-01**	1.15E+00	2.72E+01	9.21E-01
	Rank	5	1	3	4	2
	Mean	3.53E+03	**1.24E+03**	3.55E+03	1.71E+03	5.59E+03
F6	Std.	4.10E+03	**1.45E+03**	4.49E+03	1.48E+03	5.62E+03
	Rank	4	1	3	2	5
	Mean	8.00E+01	**2.25E+01**	2.37E+01	4.00E+01	2.96E+01
F7	Std.	4.55E+01	**5.83E+00**	6.93E+00	1.23E+01	7.31E+00
	Rank	5	1	2	4	3
	Mean	1.13E+02	2.93E+01	**2.16E+01**	2.98E+01	2.18E+01
F8	Std.	1.16E+02	2.99E+01	**1.09E+00**	3.00E+01	1.29E+00
	Rank	5	3	1	4	2
F9	Mean	2.09E+02	1.81E+02	1.82E+02	1.82E+02	**1.80E+02**
	Std.	5.28E+01	3.72E-01	**2.01E-01**	2.28E+00	3.76E-01
	Rank	5	2	3	4	1
F10	Mean	6.91E+02	1.70E+02	2.00E+02	1.82E+02	**1.17E+02**
	Std.	4.91E+02	7.94E+01	1.48E+02	1.34E+02	**7.37E+01**
	Rank	5	2	4	3	1
F11	Mean	1.62E+03	3.27E+02	3.40E+02	4.02E+02	**3.13E+02**
	Std.	7.01E+02	**4.50E+01**	4.98E+01	2.81E+02	9.73E+01
	Rank	5	2	3	4	1
F12	Mean	4.06E+02	2.79E+02	2.69E+02	2.66E+02	**1.96E+02**
	Std.	7.21E+01	1.76E+01	2.16E+01	1.77E+01	**1.00E+00**
	Rank	5	4	3	2	1
	Ave Rank	4.58	1.92	2.50	3.17	1.83
	Total Rank	5	2	3	4	1

To verify the improvement of Co-PPSO in terms of convergence speed, we selects 4 functions from Basic, Hybrid and Composition functions for analysis. In Fig. 3, it is obvious that Co-PPSO converges faster on the F2 function than the other four algorithms. The performance on the F5 and F11 functions is even better, and Co-PPSO can search for the best value in the early stage. On the F10 function, the convergence effect of Co-PPSO and MPSO is not much different. In Fig. 4, it is not difficult to find that the convergence speed of Co-PPSO on the four functions is significantly better than that of the other four algorithms. Co-PPSO converges rapidly in the early stage of the F2 function, and the final convergence accuracy is high. It can be seen from the F7, F8, and F12 functions that DSPSO has a poor convergence speed in the early stage, but can jump out of the local part in the later stage, thereby speeding up the convergence speed. To sum up, Co-PPSO shows excellent convergence ability, and both the convergence speed and the convergence accuracy have been effectively improved.



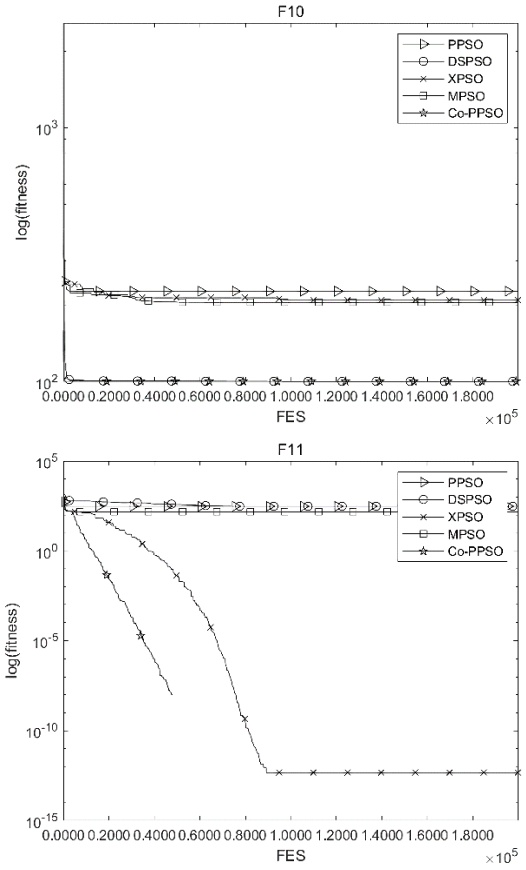
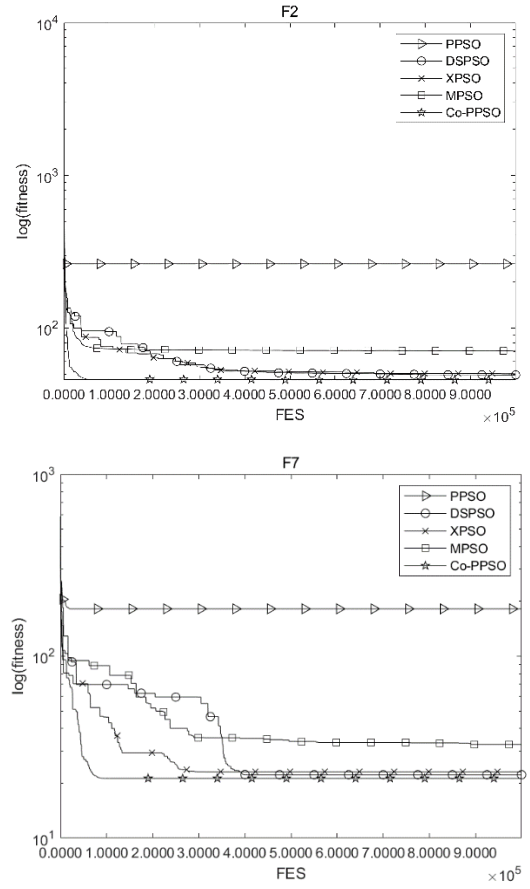


Fig. 3. Convergence effect of 5 algorithms on 10-*D*



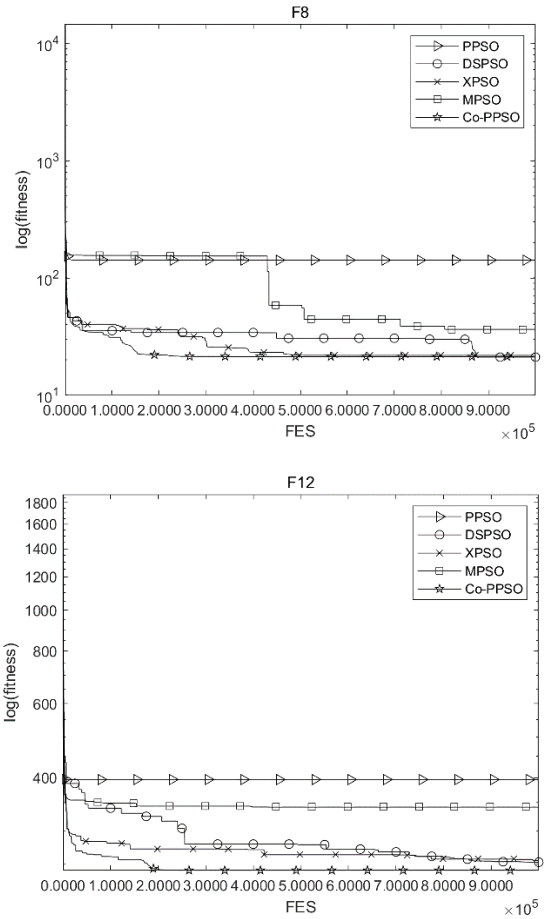


Fig. 4. Convergence effect of 5 algorithms on 20-*D*

In short, Co-PPSO has certain robustness in optimization. The main reason is that Co-PPSO can choose the appropriate learning strategy for each particle under the proposed composite framework. When the particle diversity is insufficient, the global exploration ability is enhanced through Strategy 1. When the particle exploitation ability is insufficient, the local exploitation ability is enhanced through Strategy 2. Strategy 3 applies to particles that need to balance the two abilities. In this way, the combination of composite framework and PPSO can effectively complement each other, so as to improve the search accuracy and speed.

D. Time Complexity

For the calculation of algorithm time complexity, reference [12] gives detailed calculation steps, and the calculation rules are as follows:

1. Run a test program [12] and computing time for the above = *T0*:

2. Evaluate the computing time just for Function 1. The number of evaluations is 200000 with D-dimension, it is *T1*;

3. Evaluate the complete computing time for Co-PPSO. The number of evaluations is 200000 with D-dimension, it is *T2*;

4. Execute step 3 five times. The mean of the obtained values of *T2=mean* (*T2*).

The calculation of time complexity has three parameters: *T2*, *T1*, *T0*, and (*T2*-*T1*)/*T0*. The computational complexity is tabulated in TABLE IX. TABLE IX shows that T0 is small,

and *T2* is larger than *T1* because of the increased time complexity in building the elite population. And in the composite strategy framework, each particle will be evaluated three times and compared. In TABLE IX, the time complexity increases with the increase of dimension and evaluation times. However, this increase is not very obvious. It approves that the Co-PPSO can be employed as an effective algorithm to solve different complex problems.

TABLE IX. TIME COMPLEXITY

	T0	*T1*	*T2*	(*T2*-*T1*)/*T0*
D=10	0.0625	0.7656	6.7093	95.0992
D=20	0.0625	0.7813	7.8031	112.3488

V. CONCLUSION

In this study, a variant Co-PPSO is proposed to test CEC2022 single objective bound constrained numerical optimization. The composite strategy framework is embedded into PPSO to solve the problem of premature convergence. In the composite strategy framework, firstly, an elite group is formed to avoid the circumstance where the population only learns from the global best particle. Secondly, three learning strategies are designed to improve the global exploration and local exploitation capabilities of particle swarm, and finally the both capabilities are balanced. Through the research and analysis of the results of the two-dimensional problems, it is concluded that the Co-PPSO has certain competitiveness in improving the convergence accuracy and speed, and shows low time complexity. It is observed that the Co-PPSO does not effectively cooperate with the learning strategy in parameter setting, and further modifications are needed to improve the convergence effect.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (Grant Nos. 62066019, 61903089), the Natural Science Foundation of Jiangxi Province (Grant Nos. 20202BABL202020, 20202BAB202014) and the National Key Research and Development Program of China (Grant No. 2020YFB1713700) and the Graduate Innovation Foundation of JiangXi University of Science and Technology (Grant No. XY2021-S094).

REFERENCES

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in IEEE International Conference on Neural Networks, pp. 1942-1948, Perth, WA, Australia, 1995.

[2] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence, pp. 69-73, Anchorage, AK, USA, 1998.

[3] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in Proceedings of the Congress on Evolutionary Computation, pp. 84-88, La Jolla, CA, USA, 2000.

[4] A. Ratnaweera, S. K. Halgamuge, H. C. Watson, Self-Organizing Hierarchical Particle Swarm Optimizer with Time-Varying Acceleration Coefficients. IEEE Transactions on Evolutionary Computation, vol. 8(3). 2004, pp. 240-255.

[5] Y. Shi and R. C. Eberhart, "Fuzzy adaptive particle swarm optimization," in Proceedings of the Congress on Evolutionary Computation, pp. 101-106, Seoul, South Korea, 2001.

[6] A. Abraham, H. Liu, Turbulent particle swarm optimization using fuzzy parameter tuning. Foundations of Computational Intelligence Vol. 3. Springer, Berlin, Heidelberg, 2009, pp. 291-312.

[7] T. Chen, Q. Shen, P. Su, C. Shang, Fuzzy rule weight modification with particle swarm optimisation. Soft Computing, vol. 20(8). 2016, pp. 2923-2937.

[8] M. S. Nobile, G. Pasi, P. Cazzaniga, D. Besozzi, R. Colombo, G. Mauri, Proactive particles in swarm optimization: a self-tuning algorithm based on fuzzy logic. 2015 IEEE International Conference on Fuzzy Systems. pp. 1-8, 2015.

[9] M. S. Nobile, P. Cazzaniga, D. Besozzi, R. Colombo, G. Mauri, G. Pasi, Fuzzy Self-Tuning PSO: A settings-free algorithm for global optimization. Swarm and evolutionary computation, vol. 39. 2018, pp. 70-85.

[10] M. Sugeno, Industrial Applications of Fuzzy Control. New York, NY:Elsevier Science Inc., 1985.

[11] Y. Wang, Z. Cai, Q. Zhang, Differential evolution with composite trial vector generation strategies and control parameters. IEEE transactions on evolutionary computation, vol. 15(1). 2011, pp. 55-66.

[12] A. Kumar, K. V. Price, A. W. Mohamed, A. A. Hadi, P. N. Suganthan. Problem Definitions and Evaluation Criteria for the CEC 2022 Special Session and Competition on Single Objective Bound Constrained Numerical Optimization. 2022.

[13] X. Zhang, X. Wang, Q. Kang, Cheng J. Differential mutation and novel social learning particle swarm optimization algorithm. Information Sciences, 2019, 480: 109-129.

[14] X. Xia, L. Gui, G. He, B. Wei, Y. Zhang, F. Yu, H. Wu, Z. Zhan. An expanded particle swarm optimization based on multi-exemplar and forgetting ability. Information Sciences, 2020, 508: 105-120.

[15] H. Liu, X. W. Zhang, L. P. Tu. A modified particle swarm optimization using adaptive strategy. Expert systems with applications, 2020, 152: 113353.