An Improved Chicken Swarm Optimization Algorithm with Fireworks Factor

Baofeng Zheng[1], Xiuxi Wei[2](✉), and Huajuan Huang[2]

[1] College of Electronic Information, Guangxi University for Nationalities, Nanning 530006, China

[2] College of Artificial Intelligence, Guangxi University for Nationalities, Nanning 530006, China

Abstract. Considering the problem that the original chicken swarm optimization algorithm ethnic diversity and easy to fall into local optimum problems, an improved chicken swarm optimization algorithm based on fireworks (FW-ICSO) algorithm was proposed. In this algorithm, roulette was introduced to die out some chickens, produce new chicken's methods is implemented in fireworks algorithm, and adding an inertia factor to balance search capability. Finally, FW-ICSO is tested with twenty benchmark function and compared with other similar algorithms to confirm their effectiveness in terms of the accuracy and convergence rate of the results.

Keywords: Chicken swarm optimization (CSO) algorithm · Roulette algorithm · Fireworks optimization (FWA) algorithm · Function optimization

1 Introduction

Swarm intelligence optimization algorithms are novel type of computational method by simulating the swarm behavior or predatory behavior of natural creatures. At present, researchers have proposed a variety of swarm intelligence optimization algorithms, such as particle swarm optimization (PSO) [1], Genetic algorithm (GA) [2], firefly algorithm (FA) [3].

The chicken swarm optimization (CSO) algorithm was originally presented by Meng et al. [4] at the Fifth International Conference on swarm intelligence (ICSI) in 2014. The algorithm simulates the regular pattern of foraging and the hierarchy order in the chicken swarm, the proposed algorithm has become the focus of a growing number of scholars, they are extensively used in applications. Such as, Dinghui Wu et al. [5] have used the Markov chain to analyze the convergence of clustering algorithm and verified the global convergence of CSO; Deb S et al. [6] summarized the research progress of chicken swarm olony optimization algorithm;Hafez A I et al. [7] proposed an innovative approach for feature selection based on chicken swarm optimization; Cui D [8] proposed projection pursuit model for evaluation of flood and drought disasters based on chicken swarm optimization algorithm; Osamy W et al. [9] proposed the Chicken Swarm Optimization based Clustering Algorithm to improve energy efficiency in Wireless Sensor Network.

Although the CSO algorithm has fast convergence rate and high optimum accuracy, there is easy to trap in the local optimum. Therefore, the optimization of the CSO algorithm becomes the focus of the researcher. Such as, Dinghui Wu et al. [10] proposed a method, which added the part of chicken learning cock, and introduced inertia weight and learning factor; Wang J et al. [11] proposed the improved Chicken Swarm Optimization algorithm to solve the interruptible load scheduling scheme; Bin Li et al. [12] proposed algorithm is a combination of the grey wolf optimizer (GWO) and the particle swarm optimization (PSO) algorithm with CSO; N. Bharanidharan et al. [13] proposed a method, which Improved chicken swarm optimization to classify dementia MRI images using a novel controlled randomness optimization algorithm.

In this paper, an improved chicken swarm optimization algorithm based on fireworks (FW-ICSO) algorithm was developed. Firstly, the roulette algorithm is used to select and eliminate individuals. Secondly, combines the advantage of the firework algorithm (FWA) [14] to generate new particles and improve the population diversity. Then, adding inertia factor to balance search capability. Finally, FW-ICSO is tested with a set of test functions and compared with other algorithms. The experimental data show that FW-ICSO has obvious advantages.

The rest of the paper is organized as follows: Sect. 2 introduces the idea of the chicken swarm optimization algorithm and update formula. The improvements of the FW-ICSO algorithm are introduced in Sect. 3. In Sect. 4, simulation results of FW-ICSO are presented. Finally, conclusions and expectation are stated in Sect. 5.

2 CSO Algorithm Introduction

The CSO algorithm was proposed by observing chicken swarm behavior.The best quality food is taken as the target point, through the continued transport of position information among chickens of different grades in each subgroup, and comparison with their best position, the next direction of each chicken is determined and the food finally funds. The central idea of CSO is as follows:

1) The chicken swarm is divided into multiple subgroups, which are composed of roosters, hens, and chicks.
2) Each subgroup consists of only one rooster and several hens and chicks, roosters have the best fitness and act as the leader in a subgroup. The individuals with the worst fitness values will be defined as chicks and the rest are hens.
3) The hierarchy and mother-child relationships are updated every few times.
4) The roosters lead its subgroup foraging, hens always looking for food follow the roosters, each chick follows their mothers to search for food.

In the algorithm, the entire swarm population was set to N, then N_R, N_M, N_H, N_C were the number of roosters, mother hens, hens, and chicks. x denotes the position of each chicken, then the position update of the rooster can be expressed as:

$$x_{i,j}(t+1) = x_{i,j}(t) \times \left[1 + Randn\left(0, \sigma^2\right)\right] \tag{1}$$

$$\sigma^2 = \begin{cases} 1, \ \text{if } f_i \le f_k \\ \exp(\frac{f_k - f_i}{|f_i| + \varepsilon}), \quad \text{otherwise} \\ k \in [1, N_R], \quad k \ne i \end{cases} \tag{2}$$

In formula (1), $x_{i,j}(t)$ denote the position of the i th rooster on the j th dimension in the t th iteration($j \in [1, d]$, d is the dimension of the search space); $Randn\left(0, \sigma^2\right)$ is a Gaussian distribution with mean 0 and standard deviation σ^2. Formula (1) simulates the rooster's position moves.

In formula (2), k is a rooster's index, which is randomly selected from the roosters' group ($k \ne i$), ε is the smallest constant in the computer. Formula (2) simulates the competitive relationship between different roosters.

The position update of the hens (include mother hens) can be expressed as:

$$\begin{aligned} & x_{i,j}(t+1) = x_{i,j}(t) + C_1 \times \text{Rand} \times \left(x_{r_1,j}(t) - x_{i,j}(t)\right) + C_2 \times \\ & \text{Rand} \times \left(x_{r_2,j}(t) - x_{i,j}(t)\right) \end{aligned} \tag{3}$$

$$C_1 = \exp(\frac{f_i - f_{r_1}}{abs(f_i) + \varepsilon}) \tag{4}$$

$$C_2 = \exp(f_{r_2} - f_i) \tag{5}$$

In the formula (3), $Rand$ represents a random number between [0,1]; r_1 is the rooster in the same subgroup, r_2 is chicken(rooster or hen), which randomly calculated in the entire swarm($r_1 \ne r_2$), and the fitness of r_2 is better than the fitness of i.

The position update of the chicks can be expressed as:

$$x_{i,j}^{t+1} = x_{i,j}^t + FL\left(x_{m,j}^t - x_{i,j}^t\right) \tag{6}$$

In the formula (6), m is the index of the mother hen of chicks i; FL is the adjustment parameter of chick following its mother, $FL \in [0, 2]$.

The algorithm CSO is shown in Algorithm 1.

Algorithm 1. CSO

```
Initialization pop, M, G, Dim, rPercent, hPercent, mPercent, t;
Initialization x, save best_fitness, best_location;
While  $t \leq M$ do
    If  $t\%G$ do
        Sort fitness;
        Role assignment;
        subgroup division;
    End
    For each chicken $x$ do
        If $x$ is a rooster do
            Update $x$ location using formula (1) ~ (2);
        End
        If $x$ is a hen do
            Update $x$ location using formula (3) ~ (5);
        End
        If $x$ is a chick do
            Update $x$ location using formula (6);
        End
        Update fitness based on $x$ ;
        Save best_fitness, best_location;
    End
End
```

3 FW-ICSO Algorithm Introduction

Although convergence accuracy and rate of the traditional CSO algorithm is quite remarkable, the ethnic diversity is relatively single, and the chicks easily fall into local optimum, which leads to the depression of algorithm efficiency. Therefore, this paper proposed the FW-ICSO algorithm, additional the elimination mechanism, and use the FWA algorithm to generate new individuals, which is conducive to jumping out of the local optimum; additional the inertia factor, and balancing the searchability.

3.1 Elimination Mechanism

In this paper, using the roulette algorithm, after the *G* cycle, the poorly fitness part of chickens will be eliminated. Roulette algorithm: Link the probability of each individual being selected to its fitness, with better fitness comes a lower probability of being selected, and with worse fitness comes a greater probability of being selected.

Firstly, calculate the probability of each individual being selected:

$$P(x_i) = \frac{f(x_i)}{\sum_{j=1}^{N} f(x_j)} \tag{7}$$

Secondly, calculate the cumulative probability:

$$Q(x_i) = \sum\nolimits_{k=1}^{i} P(x_k) \tag{8}$$

If the individual's fitness is poor, the corresponding selection probability will be greater. After the selection probability is converted to the cumulative probability, the corresponding line segment will be longer, and the probability of being selected will be greater.

Finally, generate a random number *judge*, $judge \in [rPercent, 1]$. If $Q(x_i) > judge$, eliminate the individual.

3.2 Creates New Individuals

In order to create new individuals, this paper introduces the FWA [12] algorithm. Assuming that n individuals are eliminated in part 3.1, then set the first n individuals with better fitness as the center. Calculate the explosive strength, explosive amplitude and displacement, then create new individuals, and select individuals with good fitness to join the algorithm.

(a) Explosive strength
This parameter is used to determine the number of new individuals. Good fitness individuals will create more new individuals, and poor fitness individuals will create fewer new individuals.

In formula (9): S_i represents the number of new individuals produced by the i th individual; m is the constant used to control the number of new individuals of the maximum explosive amplitude, $Y_{\max}$ is the worst fitness.

$$S_i = m \frac{Y_{\max} - f(x_i) + \varepsilon}{\sum_{i=1}^{N} (Y_{\max} - f(x_i)) + \varepsilon} \tag{9}$$

(b) Explosive amplitude
The explosive amplitude is used to limit the range of individuals to create new individuals. Good fitness solutions can be very close to the global solution, so the smaller the explosive amplitude, on the contrary, the larger the explosive amplitude.

$$A_i = \hat{A} \frac{f(x_i) - Y_{\min} + \varepsilon}{\sum_{i=1}^{N} (f(x_i) - Y_{\min}) + \varepsilon} \tag{10}$$

In formula (10): A_i represents the explosive amplitude of the i th individual creating new individuals; $\hat{A}$ is the constant of the maximum explosive amplitude; $Y_{\min}$ is the best fitness.

(c) Displacement operation
With the explosive strength and explosive amplitude, *and according to the* Displacement operation, S_i new individuals can be created.

$$\Delta x_i^k = x_i^k + rand(0, A_i) \tag{11}$$

In formula (11): x_i^k represents the location of the ith individual; Δx_i^k represents the location of the new individual;$rand(0, A_i)$ is the random displacement distance.

3.3 Inertial Factor

In the test without adding inertial factor ω, although the FW-ICSO algorithm can achieve better results than other algorithms, its stability is not strong, so the inertial factor ω is added to solve this problem.

The updated roosters' position formula is:

$$x_{i,j}(t+1) = \omega \times x_{i,j}(t) \times \left[1 + N\left(0, \sigma^2\right)\right] \tag{12}$$

The updated hens' position formula is:

$$x_{i,j}(t+1) = \omega \times x_{i,j}(t) + C_1 \times \text{Rand} \times \left(x_{r_1,j}(t) - x_{i,j}(t)\right) + C_2 \times \text{Rand} \times \left(x_{r_2,j}(t) - x_{i,j}(t)\right) \tag{13}$$

The updated chicks' position formula is:

$$x_{i,j}^{t+1} = \omega \times x_{i,j}^t + F\left(x_{m,j}^t - x_{i,j}^t\right) \tag{14}$$

3.4 The Flow of the FW-CSO Algorithm

The algorithm FW-ICSO is shown in Algorithm 2.

Algorithm 2. FW-ICSO

```
Initialization pop, M, G, Dim, rPercent, hPercent, mPercent, t;
Initialization x, save best_fitness, best_location;
While ( $t \le M$ )
    If $t\%G$ then
        If $t \neq 1$ do
            Select individual elimination using formula (7) ~ (8);
            Creates new individuals using formula (9) ~ (11);
        End
        Select individuals with good fitness and save to $x$ ;
        Sort fitness;
        Role assignment;
        subgroup division;
    End
    For each chicken $x$ do
        If $x$ is a rooster do
            Update $x$ location using formula (12);
        End
        If $x$ is a hen do
            Update $x$ location using formula (13);
        End
        If $x$ is a chick do
            Update $x$ location using formula (14);
        End
    End
    Update fitness based on $x$ ;
    Save best_fitness, best_location;
End
```

4 Experimental Comparison and Analysis

4.1 Experimental Parameter Settings

Fifteen benchmark functions in Table 1 are applied to compare FW-ICSO, CSO, ISCSO[15], PSO, BOA [16]. Set the $D = 50$; The search bounds are [−100,100]; The total particle number of each algorithm to 100; The maximum number of iterations is 1000; The algorithms run 50 times independently for each function. The parameters for algorithms are listed in Table 2.

Table 1. Fifteen benchmark functions

ID	Function name	ID	Function name
F1	Bent Cigar	F9	Discus
F2	Sum of different power	F10	Ackley's
F3	Zakharov	F11	Powell
F4	Rosenbrock's	F12	Griewank's
F5	Rastrigin's	F13	Katsuura
F6	Rotated Rastrigin's	F14	HappyCat
F7	Levy	F15	HGBat
F8	High conditioned elliptic		

Table 2. The main parameter settings of the five algorithms

Algorithm	Parameters
FW-ICSO	$N_R = 0.15 * NN_H = 0.7 * NN_C = N - N_R - N_H N_M = 0.5 * N_H$ $G = 10, F \in (0, 2), m = 50, \hat{A} = 40, \omega = 0.8$
CSO	$N_R = 0.15 * NN_H = 0.7 * NN_C = N - N_R - N_H N_M = 0.5 * N_H$ $G = 10, F \in (0, 2)$
ISCSO	$N_R = 0.15 * NN_H = 0.7 * NN_C = N - N_R - N_H N_M = 0.5 * N_H$ $G = 10, F \in (0, 2)$
PSO	$c1 = c2 = 1.5, \omega = 0.8$
BOA	$p = 0.8, \alpha = 0.1, c = 0.01$

4.2 Experiment Analysis

As can be seen from Figs. 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29 and 30 the FW-ICSO has a considerable convergence speed. In Fig. 1, 15, and 17, the convergence speed of the FW-ICSO algorithm is much faster, the PSO and BOA has fallen into the local optimum. CSO and ISCSO converge at the same speed, but compared with FW-ICSO, the speed is much slower, and it falls into the local optimum. Therefore, FW-ICSO can avoid falling into the local optimal solution.

In Fig. 3, 7, 9, 11, and 23, FW-ICSO has a considerable speed of convergence. Although other algorithms perform well, they are still slower than FW-ICSO in terms of speed. Therefore, the convergence speed of the FW-ICSO algorithm is excellent. It can be seen from their variance graphs, the variance of FW-ICSO is small and stable. Therefore, FW-ICSO is not only fast, but also very stable.

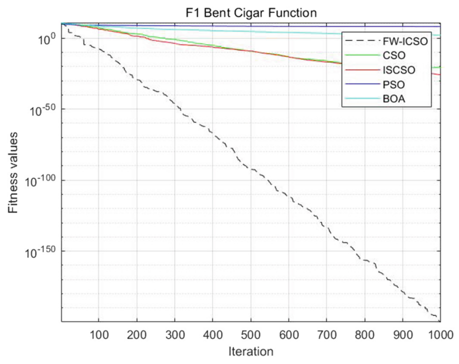

Fig. 1. Comparison of algorithm convergence in F1 function

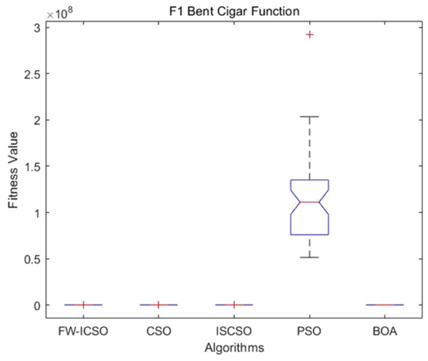

Fig. 2. Comparative ANOVA tests of algorithms in F1 function

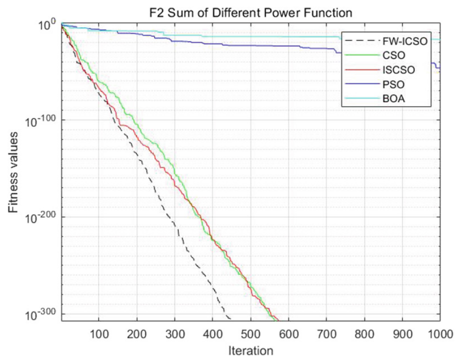

Fig. 3. Comparison of algorithm convergence in F2 function

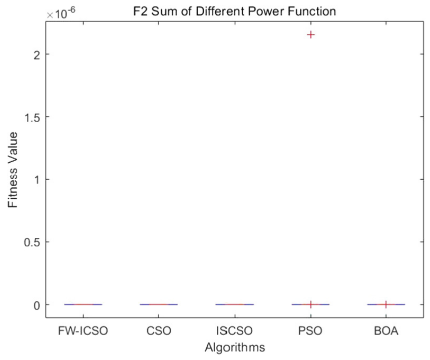

Fig. 4. Comparative ANOVA tests of algorithms in F2 function

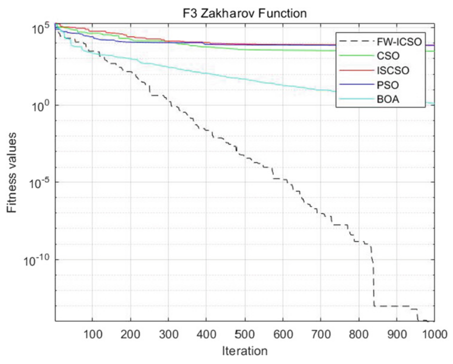

Fig. 5. Comparison of algorithm convergence in F3 function

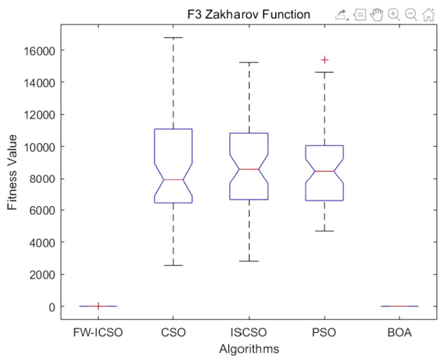

Fig. 6. Comparative ANOVA tests of algorithms in F3 function

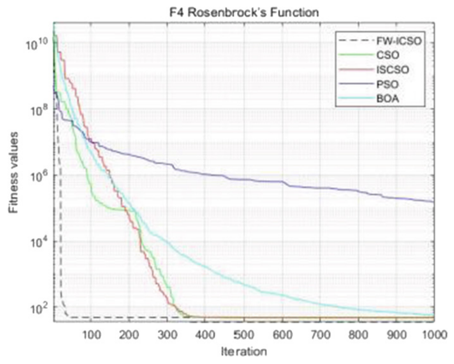

Fig. 7. Comparison of algorithm convergence in F4 function

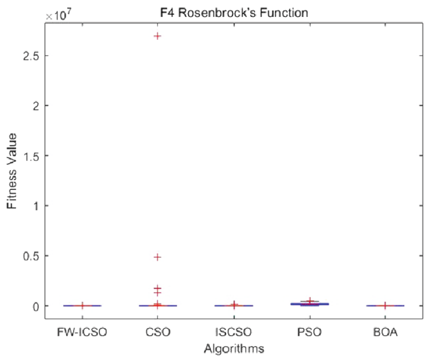

Fig. 8. Comparative ANOVA tests of algorithms in F4 function

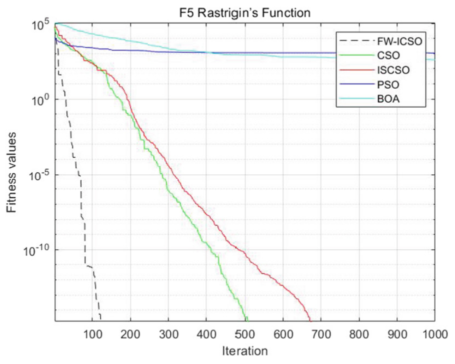

Fig. 9. Comparison of algorithm convergence in F5 function

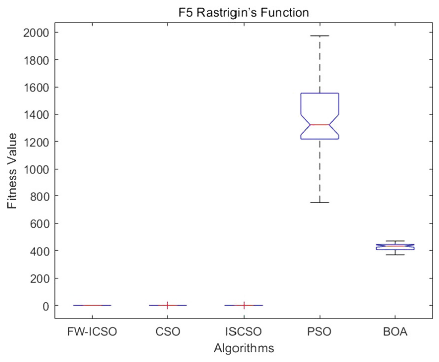

Fig. 10. Comparative ANOVA tests of algorithms in F5 function

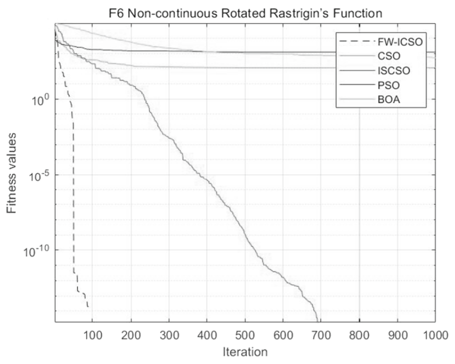

Fig. 11. Comparison of algorithm convergence in F6 function

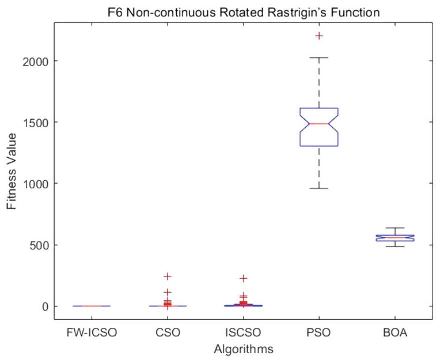

Fig. 12. Comparative ANOVA tests of algorithms in F6 function

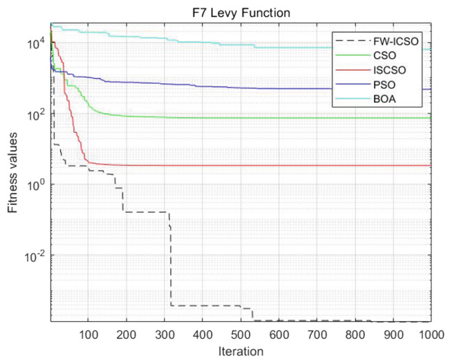

Fig. 13. Comparison of algorithm convergence in F7 function

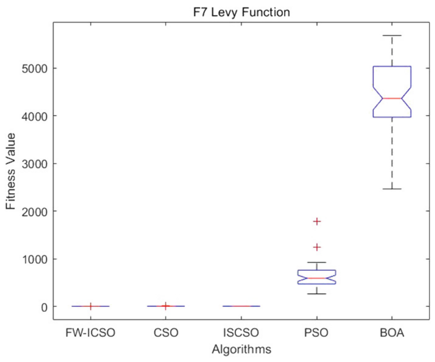

Fig. 14. Comparative ANOVA tests of algorithms in F7 function

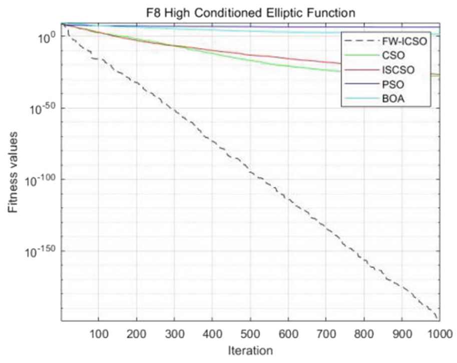

Fig. 15. Comparison of algorithm convergence in F8 function

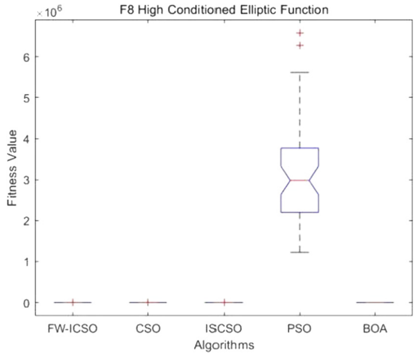

Fig. 16. Comparative ANOVA tests of algorithms in F8 function

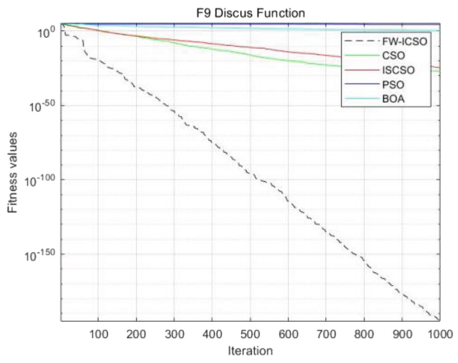

Fig. 17. Comparison of algorithm convergence in F9 function

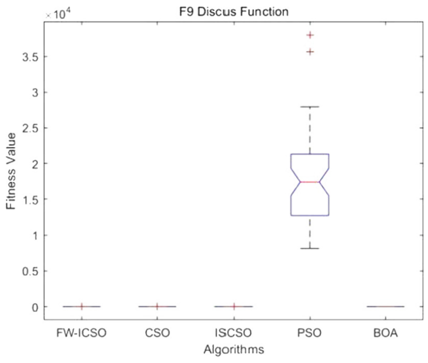

Fig. 18. Comparative ANOVA tests of algorithms in F9 function

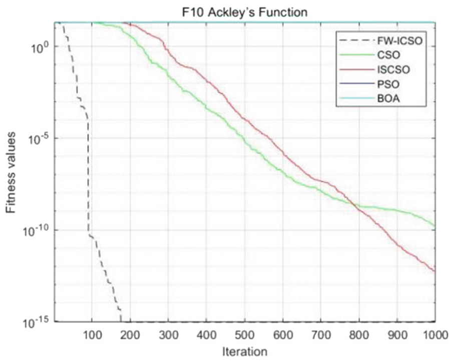

Fig. 19. Comparison of algorithm convergence in F10 function

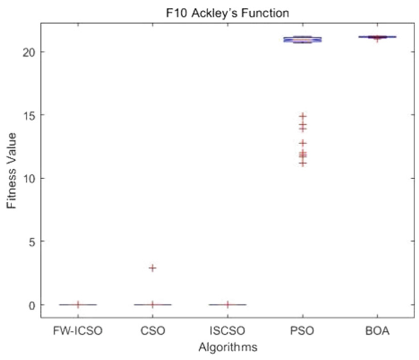

Fig. 20. Comparative ANOVA tests of algorithms in F10 function

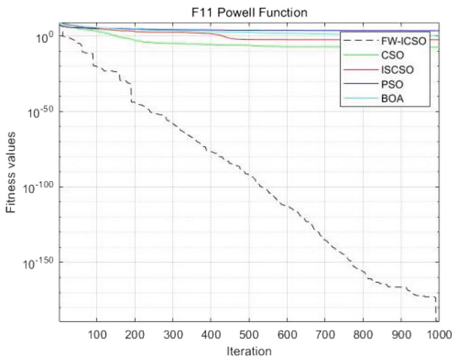

Fig. 21. Comparison of algorithm convergence in F11 function

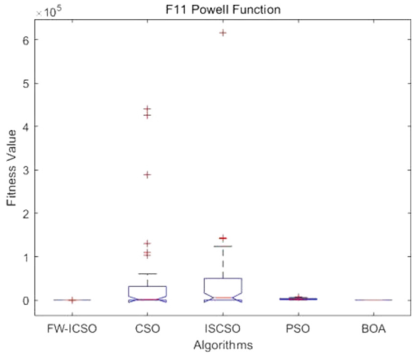

Fig. 22. Comparative ANOVA tests of algorithms in F11 function

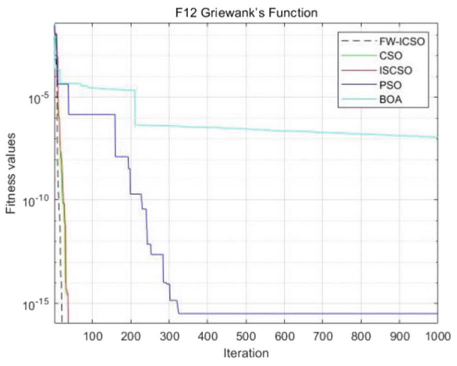

Fig. 23. Comparison of algorithm convergence in F12 function

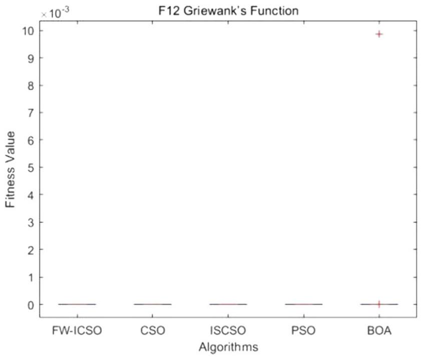

Fig. 24. Comparative ANOVA tests of algorithms in F12 function

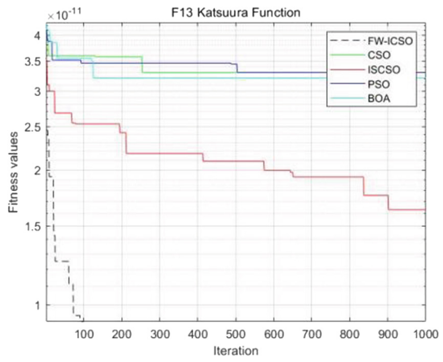

Fig. 25. Comparison of algorithm convergence in F13 function

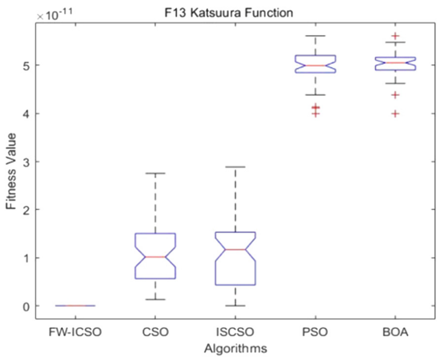

Fig. 26. Comparative ANOVA tests of algorithms in F13 function

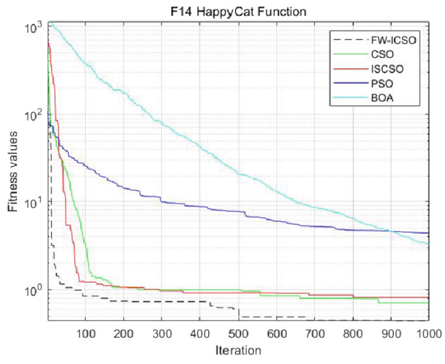

Fig. 27. Comparison of algorithm convergence in F14 function

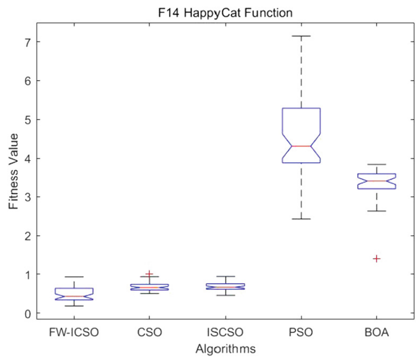

Fig. 28. Comparative ANOVA tests of algorithms in F14 function

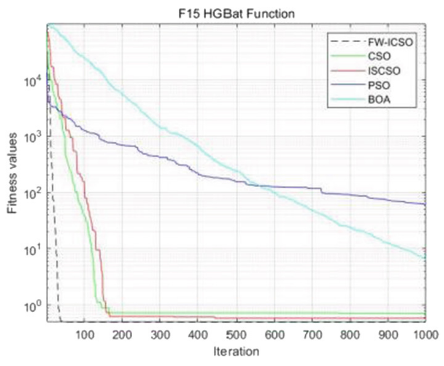

Fig. 29. Comparison of algorithm convergence in F15 function

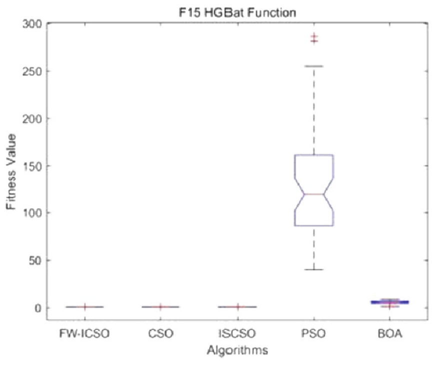

Fig. 30. Comparative ANOVA tests of algorithms in F15 function

It can be seen from Table 3 that the FW-ICSO is better than the data of other algorithms in the Worst, Best, Mean, and Std of each function except F4 and F14. In function F4, the standard deviation of FW-ICSO was worse than that of BOA, and the worst fitness of FW-ICSO was better than the best fitness of BOA. although the standard deviation is worse than BOA, it is also within the acceptable range. In function F14, the standard deviation of FW-ICSO was larger than the standard deviation of ISCSO, but the value is relatively close, and other values are better than ISCSO.

In terms of time consumption, since the algorithm time of CSO itself is longer than other algorithms, the speed of FW-ICSO algorithm is 0.1 to 0.2 s slower than that of CSO algorithm, but when the time loss is acceptable, the accuracy will be greatly improved.

Table 3. Accuracy comparison table

Function	Algorithm	Worst	Best	Mean	Std.	Time
F1	FW-ICSO	6.1894E−186	1.13E−200	2.104E−187	0	1.051045
	CSO	3.40331E−20	6.238E−27	2.0407E−21	7.003E−21	0.957194
	ISCSO	4.06791E−19	8.924E−27	1.3122E−20	6.049E−20	1.309967
	PSO	292452978.1	51403345	114868034	47144279	0.182071
	BOA	212.5405395	95.420394	152.697557	26.59569	0.832887
F2	FW-ICSO	0	0	0	0	1.114456
	CSO	0	0	0	0	1.047371
	ISCSO	0	0	0	0	1.454144
	PSO	2.15657E−06	1.199E−72	4.3131E−08	3.05E−07	0.244820
	BOA	2.17614E−14	1.312E−23	1.1895E−15	4.188E−15	0.991472
F3	FW-ICSO	4.90599E−11	2.075E−41	2.3296E−12	7.68E−12	0.968975
	CSO	16779.68466	2552.4041	8772.98815	3746.6	0.815973
	ISCSO	15233.94168	2817.5755	8725.16405	2847.7072	1.276224
	PSO	15385.06415	4684.2087	8844.41871	2761.0037	0.154354
	BOA	1.694109795	0.5827842	1.1987125	0.2434447	0.775721
F4	FW-ICSO	48.68285915	13.8433763	45.4735067	7.94407003	1.074267
	CSO	26919621.79	46.715618	730510.944	3854974.8	0.935579
	ISCSO	107363.5283	46.806922	2401.37179	15161.043	1.376185
	PSO	481169.9612	32591.93	183831.55	112823.36	0.256701
	BOA	59.03452539	52.354011	56.1893463	1.2720791	0.956245
F5	FW-ICSO	0	0	0	0	0.895254
	CSO	5.32907E−15	0	4.9738E−16	1.296E−15	0.818512
	ISCSO	3.55271E−15	0	1.0658E−16	5.571E−16	1.341090
	PSO	1973.247062	750.83661	1365.0541	271.31223	0.272896
	BOA	469.8530387	369.29832	427.495741	24.207126	1.028247

(*continued*)

Table 3. (*continued*)

Function	Algorithm	Worst	Best	Mean	Std.	Time
F6	FW-ICSO	0	0	0	0	2.454195
	CSO	240.7162391	0	10.1121123	37.742581	2.361683
	ISCSO	223.6098307	0	11.7887804	35.079087	3.135543
	PSO	2205.574795	960.40339	1491.23701	298.63115	1.622343
	BOA	637.9790385	485.56995	556.661562	36.401121	3.875971
F7	FW-ICSO	0.003964923	4.788E−10	0.00055943	0.0009584	1.279563
	CSO	13.84271935	2.7653595	3.56474086	1.5273858	1.079681
	ISCSO	4.026353675	2.6930204	3.30681594	0.3092122	1.609634
	PSO	1786.141083	263.95178	626.270917	251.02204	0.390569
	BOA	5684.007969	2463.7525	4403.996	735.67077	1.363668
F8	FW-ICSO	1.208E−190	6E−211	2.744E−192	0	2.201493
	CSO	6.38208E−23	1.92E−30	2.0776E−24	9.318E−24	1.829214
	ISCSO	6.84164E−21	1.806E−29	1.383E−22	9.674E−22	2.293385
	PSO	6581731.942	1223689.8	3233924.77	1330427.8	1.028134
	BOA	47.33434907	21.019203	35.3138648	7.0328411	2.601756
F9	FW-ICSO	2.2114E−196	1.42E−212	5.33E−198	0	1.145240
	CSO	3.61323E−24	9.983E−31	1.41E−25	5.668E−25	0.962248
	ISCSO	7.48824E−24	1.185E−31	3.0069E−25	1.185E−24	1.389841
	PSO	37983.62703	8127.6735	18080.3354	6694.1586	0.191719
	BOA	3.057654745	0.5670763	1.85472434	0.4892557	0.871098
F10	FW-ICSO	4.44089E−15	8.882E−16	9.5923E−16	5.024E−16	1.158565
	CSO	2.883589661	1.643E−13	0.05767179	0.4078012	1.036261
	ISCSO	1.81687E−07	7.994E−15	3.7897E−09	2.568E−08	1.527546
	PSO	21.21403019	11.214801	19.6998291	3.0778906	0.373748
	BOA	21.24352341	21.027682	21.1774251	0.0441896	1.193409
F11	FW-ICSO	3.3649E−149	2.416E−211	6.73E−151	4.76E−150	1.266613
	CSO	381039.5695	3.8513E−19	49370.8543	85243.512	0.983339
	ISCSO	378745.4295	5.2773E−33	42672.9934	88703.772	1.418165
	PSO	6861.689017	368.528453	2175.08293	1208.4472	0.310457
	BOA	2.898785526	0.36426493	1.5823845	0.5714033	1.156489

(*continued*)

Table 3. (*continued*)

Function	Algorithm	Worst	Best	Mean	Std.	Time
F12	FW-ICSO	0	0	0	0	1.279460
	CSO	0	0	0	0	1.234044
	ISCSO	0	0	0	0	1.674844
	PSO	0	0	0	0	0.420399
	BOA	0.009864687	6.013E−12	0.0003946	0.0019527	1.299393
F13	FW-ICSO	0	0	0	0	24.39417
	CSO	2.75E−11	1.33E−12	1.11E−11	6.365E−12	24.27090
	ISCSO	2.89E−11	1.53E−13	1.03E−11	7.749E−12	25.52178
	PSO	5.61E−11	4.11E−11	5.10E−11	4.205E−12	23.32075
	BOA	5.61E−11	4.63E−11	5.08E−11	2.203E−12	47.55279
F14	FW-ICSO	0.93545686	0.1853164	0.47910787	0.1902337	1.033239
	CSO	1.003017025	0.5063184	0.67737218	0.1142708	0.868092
	ISCSO	0.945762978	0.461541	0.69000236	0.1085319	1.357010
	PSO	7.144064581	2.4240018	4.53295436	0.9515144	0.142176
	BOA	3.844123963	1.3945829	3.34472732	0.3847439	0.806925
F15	FW-ICSO	0.500000001	0.4950851	0.49980061	0.0008948	1.052220
	CSO	0.586131408	0.4261263	0.48958174	0.0265692	0.826003
	ISCSO	0.683222701	0.4419355	0.49830781	0.0404395	1.349768
	PSO	286.5622008	39.581567	132.858233	58.787859	0.128205
	BOA	8.357891343	0.8503189	5.2778622	1.9406607	0.767732

5 Conclusion and Future Work

In order to improve the CSO algorithm, this paper proposes the FW-ICSO algorithm to select individuals for elimination through the roulette algorithm, introduces the explosion strength, explosion amplitude, and displacement operations in the FWA algorithm, generates new individuals to join the algorithm, and introduces the searchability of the inertial balance algorithm. Finally, tested FW-ICSO with Fifteen benchmark functions, the results demonstrate that it is true, the convergence speed, accuracy, and robustness of the FW-ICSO algorithm are considerable.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (61662005). Guangxi Natural Science Foundation (2018GXNSFAA294068) ;Research Project of Guangxi University for Nationalities (2019KJYB006); Open Fund of Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis (GXIC20–05).

References

1. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: IEEE Proceedings of ICNN'95-International Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995)
2. Yong-Jie, M.A., Yun, W.X.: Research progress of genetic algorithm. Appl. Res. Comput. **29**(4), 1201–1206 (2012)
3. Iztok, F., et al.: A comprehensive review of firefly algorithms. Swarm Evol. Comput. **13**, 34–46 (2013)
4. Meng, X., Liu, Y., Gao, X., Zhang, H.: A new bio-inspired algorithm: chicken swarm optimization. In: Tan, Y., Shi, Y., Coello, C.A.C. (eds.) ICSI 2014. LNCS, vol. 8794, pp. 86–94. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11857-4_10
5. Wu, D., Shipeng, X., Kong, F.: Convergence analysis and improvement of the chicken swarm optimization algorithm. IEEE Access **4**, 9400–9412 (2016)
6. Deb, S., Gao, X.-Z., Tammi, K., Kalita, K., Mahanta, P.: Recent studies on chicken swarm optimization algorithm: a review (2014–2018). Artif. Intell. Rev. **53**(3), 1737–1765 (2019). https://doi.org/10.1007/s10462-019-09718-3
7. Ahmed Ibrahem, H., et al.: An innovative approach for feature selection based on chicken swarm optimization. In: 2015 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR), pp. 19–24. IEEE (2015)
8. Cui, D.: Projection pursuit model for evaluation of flood and drought disasters based on chicken swarm optimization algorithm. Adv. Sci. Technol. Water Resour. **36**(02), 16–23+41 (2016)
9. Osamy, W., El-Sawy, A.A., Salim, A.: CSOCA: chicken swarm optimization based clustering algorithm for wireless sensor networks. IEEE Access **8**, 60676–60688 (2020)
10. Wu, D., et al.: Improved chicken swarm optimization. In: 2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), pp. 681–686. IEEE (2015)
11. Wang, J., Zhang, F., Liu, H., et al.: A novel interruptible load scheduling model based on the improved chicken swarm optimization algorithm. CSEE J. Power Energy Syst. **7**(2), 232–240 (2020)
12. Bin, L.I., Shen, G.J., Sun, G., et al.: Improved chicken swarm optimization algorithm. J. Jilin Univ. (Eng. Technol. Ed.) **49**(4), 1339–1344 (2019)
13. Bharanidharan, N., Rajaguru, H.: Improved chicken swarm optimization to classify dementia MRI images using a novel controlled randomness optimization algorithm. Int. J. Imaging Syst. Technol. **30**(3), 605–620 (2020)
14. Tan, Y., Zhu, Y.: Fireworks algorithm for optimization. In: International Conference in Swarm Intelligence, vol. 6415, pp. 355-364. Springer, Heidelberg (2010)
15. Tong, B., et al.: Improved chicken swarm optimization based on information sharing. J. Guizhou Univ. (Nat. Sci.) **38**(01), 58–64+70 (2021)
16. Fan, Y., et al.: A self-adaption butterfly optimization algorithm for numerical optimization problems. IEEE Access **8**, 88026–88041 (2020)