A Novel Cultural Algorithm based on Particle Swarm Optimization and Whale Optimization Algorithm

Ya Shen¹, Chen Zhang¹.²*, Xu Bai¹, ChongQing Zhang¹

¹ Department of Artificial Intelligence and Big Data, Hefei University, Hefei, Anhui, 230601, China
²* Guochuang Software Co., Ltd, Hefei, Anhui, 230094, China

*Corresponding author’s e-mail: zhangchen@hfuu.edu.cn

Abstract. An ameliorative cultural algorithm (CA) based on particle swarm optimization (PSO) and whale optimization algorithm (WOA) is raised (CA-PSOWOA), so as to conquer the defects of WOA and PSO, such as poor global exploration ability and easy fall into local optimal solution. Firstly, a nonlinear inertia weight strategy is leaded to optimize the PSO and WOA, then CA is introduced to regulate the ability of global exploration and local exploitation of PSO and WOA. By testing on benchmark functions, it is proved that CA-PSOWOA improves the global exploration ability and solution accuracy, and its performance is better than the traditional PSO and WOA, and other algorithms.

1. Introduction

Swarm intelligence optimization algorithm imitates the cooperation between animals in nature to solve a problem, such as PSO [1], firefly algorithm (FA) [2] and WOA [3].

However, swarm intelligence optimization algorithm has its disadvantages, such as low global exploration ability and poor optimization accuracy. Many scholars and experts have studied these problems. Koyuncu et al. [4] prove the necessity of chaotic mapping for PSO through 13 benchmark functions, then PSO based on Gaussian mapping is proposed. Kiani, et al. [5] propose a dynamic inertia weight strategy based on double exponential function, which can reduce the premature convergence problem of traditional PSO. Li et al. [6] use tent chaotic mapping to improve the WOA, increase the variety of the initial population, and adopt the tournament selection tactic to improve the accuracy of WOA. Chao et al. [7] propose a linear decreasing inertia weight strategy and applied it to the WOA. Xu et al. [8] combine differential evolution (DE) and FA, and adjust the firefly's moving step adaptively by using vector angle parameters to avoid local optimum. DE is used to change the moving direction of FA and enhance exploration ability. Han et al. [9] put forward an algorithm based on PSO and artificial bee colony algorithm (ABC). ABC can surmount the defect that PSO is likely to sink into local optimal. Meanwhile, ABC has fine global exploration ability and can rise the chance of getting the optimal solution. Saad et al. [10] combine the CA and ABC, and propose a new algorithm (CB-ABC). CB-ABC consists of two parts, one is updated by the preventient optimal solution, the other is updated according to the adaptive information. Jafari et al. [11] propose an algorithm based on CA and PSO. When PSO is trapped at the local optimal solution, the CA can correct the moving direction of PSO and make it move closer to the optimal solution.

We put forward the CA-PSOWOA algorithm in this paper. Firstly, a nonlinear inertia weight strategy is used to improve the ability of exploitation and exploration. Then, PSO and WOA are combined with
cultural algorithm framework to regulate the exploitation and exploration. Experiments show that CA-PSOWOA can get better effect than the basic WOA and PSO, and other algorithms.

2. Materials and Methods
Although PSO and WOA have good optimization ability, they are still easy to lose the ability of global exploration and fall into local optimal solution. Therefore, we propose a nonlinear inertia weight strategy and use cultural algorithm to combine PSO and WOA. See the following for a detailed description.

2.1 Nonlinear inertia weight
The traditional linear inertia weight is difficult to regulate the ability of exploitation and exploration. Thus, we put forward a nonlinear inertia weight strategy as shown in Eq (1) and give the movement modes of PSO and WOA based on nonlinear inertia weight.

\[ w = 2 - e^{\left(\frac{t}{\text{max iteration}}\right)} \]  

(1)

In Eq (1), \( t \) indicates the current iteration, \( \text{max iteration} \) indicates the total iterations.

The movement mode of PSO is shown in Eq (2) and Eq (3):

\[ V_{t+1}^i = wV_t^i + c_1 r_1 (P_b^i - x_t^i) + c_2 r_2 (G_b^i - x_t^i) \]  

(2)

\[ X_{t+1}^i = wX_t^i + V_{t+1}^i \]  

(3)

Where \( r_1 \) and \( r_2 \) are random numbers between [0,1], \( c_1 \) and \( c_2 \) are two constants, \( P_b^i \) is the optimal solution that particle has ever experienced at current iteration and \( G_b^i \) indicates the global optimal solution.

The movement mode of WOA is shown in Eq (4) and Eq (5):

\[ X_{t+1}^i = \begin{cases} wX_t^i - A \cdot D & \text{if } p < 0.5 \\ D' \cdot e^{b \cdot \cos(2\pi l)} + wX_t^i & \text{if } p \geq 0.5 \end{cases} \]  

(4)

\[ X_{t+1}^i = wX_{\text{rand}}^i - A \cdot D \]  

(5)

Where \( X_{\text{rand}}^i \) is a whale randomly select from the current population.

By analyzing Eq (1), \( w \) decreases slowly in the early phase of algorithm, which enables the algorithm to have sufficient ability to conduct global exploration. In the late phase of algorithm, \( w \) rapidly decreases to a value close to 0, let the algorithm have strong exploitation ability and accelerate the convergence speed.

2.2 Cultural Algorithm based on PSO and WOA
Cultural Algorithm (CA) is an algorithm framework that allows two algorithms to run in parallel. We propose the novel cultural algorithm based on PSO and WOA. CA is formed of population space and belief space. Generally speaking, population space has strong global exploration ability, and regularly contributes its elite individuals to belief space. Belief space has the strong ability of local exploitation. Belief space can produce better individuals and influence population space by constantly evolving elite individuals contributed by population space. Through the framework of cultural algorithm, PSO and WOA are combined and run in parallel, thus forming a mechanism of co-evolution. Fig.1 shows the model of CA-PSOWOA.
As PSO has fast convergence speed and strong local exploitation ability, PSO is used as belief space. WOA is population space because of its strong global exploration ability.

2.3 The description of CA-PSOWOA
The description of CA-PSOWOA is shown in Table 1.

|Algorithm 1: CA-PSOWOA|
|---|
|Initialize the population of PSO and WOA|
|Calculate the fitness of PSO and WOA|
|\(P_{\text{best}}^*\) and \(W_{\text{best}}^*\) are the best individuals of PSO and WOA, respectively|
|\(\text{Best}^*\) is the best individual of CA_PSOWOA from the best of \(P_{\text{best}}^*\) and \(W_{\text{best}}^*\)|
|while \((t < \text{maxiter})\)|
|for each individual in PSO|
|PSO is updated by Eq (2) and Eq (3)|
|Calculate the fitness of PSO and update the \(P_{\text{best}}^*\)|
|end for|
|for each individual in WOA|
|WOA is updated by Eq (4) and Eq (5)|
|Calculate the fitness of WOA and update the \(W_{\text{best}}^*\)|
|end for|
|if \((t \% \text{Acceptance}) = = 0\)|
|WOA gives some of its best individuals to PSO|
|end if|
|if \((t \% \text{Influence}) = = 0\)|
|PSO gives some of its best individuals to WOA|
|end if|
|Update the \(\text{Best}^*\) from \(P_{\text{best}}^*\) and \(W_{\text{best}}^*\)|
|\(t = t + 1\)|
|end while|
|return the \(\text{Best}^*\)|

3. Results & Discussion
We select 10 benchmark functions from [12] to assess the performance of CA_PSOWOA. We compare CA-PSOWOA with basic PSO [2] and WOA [4] and their two variants: PSO with nonlinear inertia
weight (NIPSO) and WOA with inertia weight (IWOA) [13], and two other swarm intelligence optimization algorithms: salp swarm algorithm (SSA) [14] and grey wolf optimization (GWO) [15]. NIPSO is an improved version of PSO, which use the nonlinear inertia weight strategy that propose in this paper.

3.1 Benchmark functions
The benchmark functions used for testing are shown in Table 2:

| Function | Name          | Dimension | Range      | F_{opt} |
|----------|---------------|-----------|------------|---------|
| f1       | Sphere        | 30        | [-100,100] | 0       |
| f2       | Step          | 30        | [-100,100] | 0       |
| f3       | Rosenbrock    | 30        | [-30,30]   | 0       |
| f4       | Schwefel 2.22 | 30        | [-10,10]   | 0       |
| f5       | Schwefel 1.2  | 30        | [-100,100] | 0       |
| f6       | Schwefel 2.21 | 30        | [-100,100] | 0       |
| f7       | Ackley        | 30        | [-32,32]   | 0       |
| f8       | Quartic       | 30        | [-1.28,1.28]| 0       |
| f9       | Griewank      | 30        | [-600,600] | 0       |
| f10      | Rastrigin     | 30        | [-5.12,5.12]| 0       |

Unimodal functions (f1~f6) can assess the ability of exploitation of all algorithms, and multimodal functions (f7~f10) can assess the ability of exploration of all algorithms.

3.2 Parameter setting
In CA-PSOWOA, the number of individuals in the belief space accounts for 30% of the total individuals, and the population space is 70%, and the individuals exchanged each time are set to 50% of the belief space. Set the Acceptance = 5 and Influence = 5 + fix((maxiter − t) * 5/maxiter). When \(t \% \text{Acceptance} = 0\), WOA contributes its best individuals to PSO and when \(t \% \text{Influence} = 0\), PSO passes on its current best individuals to WOA. The exchange number of individuals in PSO and WOA are same. Where \(t\) indicates the current iteration of algorithm, and \(\text{maxiter}\) indicates the total iterations. The function \(\text{fix}(\cdot)\) is used to round a number.

Parameters of other algorithms for comparison are shown in Table 3:

| Algorithms   | Parameters                          |
|--------------|-------------------------------------|
| PSO          | \(c_1 = c_2 = 2, w = 0.9\)         |
| WOA          | \(b = 1\)                           |
| NIPSO        | \(c_1 = c_2 = 2, w = 2 - e^{\frac{t}{\text{maxiter} + 1 \ln 2}}\) |
| IWOA         | \(b = 1\)                           |
| SSA          | \(c_1 = 2e^{\frac{t}{\text{maxiter} + 1 \ln 2}}\) |
| GWO          | \(r_1 = r_2 = \text{random}(0,1)\) |
| CA-PSOWOA    | \(c_1 = c_2 = 2, b = 1, w = 2 - e^{\frac{t}{\text{maxiter} + 1 \ln 2}}\) |
3.3 Results and Analysis
Set the number of iterations to 500 and the number of populations to 30, so as to ensure an impartial comparison. Meanwhile, in order to avert the contingency of experimental results, all functions run 30 times and then their standard deviation (Std) and mean are analyzed, which are shown in Table 4.

| Function | Mean (Std)       | Mean (Std)       | Mean (Std)       | Mean (Std)       | Mean (Std)       | Mean (Std)       | Mean (Std)       |
|----------|------------------|------------------|------------------|------------------|------------------|------------------|------------------|
| f1       | 2.61E-13 (6.06E-06) | 2.50E-73 (1.13E-72) | 1.38E-07 (1.50E-07) | 1.20E-308 (0.00E+00) | 6.14E-158 (2.50E-157) | 1.01E-27 (1.05E-27) | 0.00E+00 (0.00E+00) |
| f2       | 6.44E+00 (8.28E-01) | 4.04E-01 (2.22E-01) | 2.12E-07 (2.22E-07) | 2.81E+00 (8.89E-01) | 7.78E-01 (3.01E-01) | 7.20E-01 (3.01E-01) | 1.06E-04 (3.01E-01) |
| f3       | 3.42E-03 (1.62E-04) | 2.82E+01 (4.46E-01) | 3.18E+02 (3.60E+02) | 2.88E+01 (5.99E-02) | 2.84E+01 (2.73E-01) | 2.74E+01 (7.16E-01) | 5.19E-03 (3.79E-05) |
| f4       | 1.33E+01 (1.12E-01) | 8.62E-50 (1.78E-09) | 2.21E+00 (3.86E+15) | 9.14E-01 (2.99E-01) | 5.48E-85 (2.99E-84) | 7.24E-27 (1.05E-27) | 0.00E+00 (0.00E+00) |
| f5       | 6.31E+03 (6.36E+03) | 4.31E+04 (6.36E+03) | 4.31E+04 (1.64E+04) | 1.56E+03 (8.95E+02) | 1.18E-308 (0.00E+00) | 2.18E-103 (1.19E-102) | 1.17E-05 (4.42E-05) |
| f6       | 2.22E+00 (1.31E+00) | 5.76E+01 (2.48E+01) | 6.42E+01 (7.92E+01) | 1.17E+01 (4.07E+00) | 9.14E-157 (3.86E-156) | 6.42E-155 (3.43E-154) | 4.17E-65 (1.64E-64) |
| f7       | 3.62E-02 (2.83E-02) | 4.81E-03 (2.83E-02) | 1.72E-01 (6.24E-02) | 7.64E-04 (6.75E-04) | 8.98E-05 (7.50E-05) | 2.00E-03 (1.44E-03) | 3.47E+00 (5.05E+00) |
| f8       | 3.07E+00 (0.00E+00) | 1.66E-02 (0.00E+00) | 0.00E+00 (0.00E+00) | 0.00E+00 (0.00E+00) | 0.00E+00 (0.00E+00) | 0.00E+00 (0.00E+00) | 0.00E+00 (0.00E+00) |
| f9       | 4.66E+01 (0.00E+00) | 5.25E+01 (0.00E+00) | 0.00E+00 (0.00E+00) | 0.00E+00 (0.00E+00) | 0.00E+00 (0.00E+00) | 3.17E+00 (0.00E+00) | 0.00E+00 (0.00E+00) |

Table 4 describes the running results of all algorithms, in which the best results are marked in bold. We know that CA-PSOWOA produces excellent results on functions f1 and f3 ~f7 and f9 ~f10, IWOA obtains good results on f8 and f10, SSA obtains good results on f2, NIPSO obtains good results on functions f7 and f9 ~f10, WOA gets good results on functions f9 ~f10. PSO and GWO do not find any better or the same solution than other algorithms. Through data analysis, we find that CA-PSOWOA shows better performance than other algorithms.

4. Conclusions
We propose a nonlinear inertia weight strategy to enhance the accuracy of PSO and WOA, then use cultural algorithm to combine PSO and WOA (CA-PSOWOA). Through experiments, we find that CA-PSOWOA has stronger ability of exploitation and exploration than the basic PSO and WOA, the variation of PSO and WOA, and other algorithms.

CA-PSOWOA is still prone to sink into local optimal solutions on some functions, such as f2 and f8. In the future, we will continue to enhance the algorithm to improve its accuracy and global exploration ability.

Acknowledgments
This study is sustained by the National Natural Science Foundation of China under Grant 61806068, and Anhui Provincial University Outstanding Talent Cultivation Project under Grant gxgnfx2020117.
References

[1] Kennedy, J. (2011). Particle swarm optimization. In Encyclopedia of machine learning (pp. 760-766). Springer, Boston, MA. DOI: https://doi.org/10.1007/978-0-387-30164-8_630

[2] Yang, X.S. (2009) Firefly algorithms for multimodal optimization, in: O.Watanabe, T. Zeugmann (Eds.), Stochastic Algorithms: Foundations and Applications, Springer Berlin Heidelberg, Berlin, Heidelberg, pp.169–178.

[3] Mirjalili, S, & Lewis, A. (2016). The whale optimization algorithm. Advances in Engineering Software, 95, 51-67. DOI: https://doi.org/10.1016/j.advengsoft.2016.01.008

[4] Koyuncu, H. (2020) GM-CPSO: A New Viewpoint to Chaotic Particle Swarm Optimization via Gauss Map. Neural Processing Letters., 52:241–266.

[5] Kiani, T, Nadeem, F, Ahmed, A, Khan, I, & Das, N. (2020) Optimal PV Parameter Estimation via Double Exponential Function-Based Dynamic Inertia Weight Particle Swarm Optimization. Energies, 13(15),4037.

[6] Li, Y.C., Han M.X., & Guo Q.L. (2020) Modified Whale Optimization Algorithm Based on Tent Chaotic Mapping and Its Application in Structural Optimization. KSCE Journal of Civil Engineering., 24:3703–3713.

[7] Chao, I, & Liu, J. (2020) Improved Whale Optimization Algorithm Based on Inertia Weights for Solving Global Optimization Problems. Advances in Technology Innovation, 5(3),147-155.

[8] Xu, C.Y., Meng, H.P., & Wang, Y.F., (2020) A Novel Hybrid Firefly Algorithm Based on the Vector Angle Learning Mechanism. IEEE Access., 8:205741 - 205754.

[9] Han, Z, & Liang, J. (2018) Numerical Improvement for the Mechanical Performance of Bikes Based on an Intelligent PSO-ABC Algorithm and WSN Technology. IEEE Access, 6,32890-32898.

[10] Saad, E, & Haikal, Y. (2019) Culture-based Artificial Bee Colony with heritage mechanism for optimization of Wireless Sensors Network. Applied Soft Computing, 79,59-73.

[11] Jafari, M, & Salajegheh, J. (2021) Optimal design of truss structures using a hybrid method based on particle swarm optimizer and cultural algorithm. Structures, 32,391-405.

[12] Li, Y.T., Han, T., Han, B.J., & Wei, Z. (2019) Whale Optimization Algorithm with Chaos Strategy and Weight Factor. Journal of Physics Conference Series, 1213,032004.

[13] Hu, H P, Bai, Y P, & Xu, T. (2016). A whale optimization algorithm with inertia weight. WSEAS Trans. Comput, 15, 319-326.

[14] Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. Advances in Engineering Software, 114, 163-191. DOI: https://doi.org/10.1016/j.advengsoft.2017.07.002

[15] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. Advances in engineering software, 69, 46-61. DOI: https://doi.org/10.1016/j.advengsoft.2013.12.007