International Journal of Innovative
Computing, Information and Control
Volume **15**, Number **3**, June **2019**

ICIC International ©2019 ISSN 1349-4198
pp. **1049–1066**

APPLICATION OF IMPROVED WHALE OPTIMIZATION ALGORITHM IN MULTI-RESOURCE ALLOCATION

Zhao Wang[1,2], Haojiang Deng[1], Xiaoyong Zhu[1,*] and Linlin Hu[1]

[1]National Network New Media Engineering Research Center
Institute of Acoustics, Chinese Academy of Sciences
No. 21, North 4th Ring Road, Haidian District, Beijing 100190, P. R. China
{ wangzhao; denghj; hull }@dsp.ac.cn; *Corresponding author: zhuxy@dsp.ac.cn

[2]School of Electronic, Electrical and Communication Engineering
University of Chinese Academy of Sciences
No. 19(A), Yuquan Road, Shijingshan District, Beijing 100049, P. R. China

Received June 2018; revised October 2018

Abstract. *In this paper, we propose an Improved Whale Optimization Algorithm (I-WOA) to improve the performance of WOA in three aspects. First, nonlinearly changed convergence factor is introduced to preferably adjust exploration and exploitation process. Then, inspired by the inertia weight factor of Particle Swarm Optimization (PSO), we add new inertia weight factor to further enhance the exploitation and exploration ability of WOA. Finally, random variation of current optimal individual is conducted during exploitation iterations to reduce the possibility of falling into local optimum. The IWOA is benchmarked on 29 well-known test functions and the results are verified by comparing IWOA with basic WOA, Grey Wolf Optimization (GWO) and Particle Swarm Optimization (PSO). We also apply the proposed IWOA to multi-resource allocation problem in resource-constrained embedded system. The results demonstrate that the proposed IWOA performs much better than WOA, GWO or PSO on most test functions and provides best performance in multi-resource allocation problem.*

Keywords: Improved whale optimization algorithm, Nonlinear convergence factor, Inertia weight factor, Random variation, Multi-resource allocation

1. **Introduction.** Meta-heuristic optimization algorithms are very popular Swarm Intelligence (SI) optimization algorithms in recent years and have been used in many fields such as machine learning, engineering and environment modeling [1]. It has the advantages of simplicity, flexibility, derivation independency and escaping from local optimum. Meta-heuristic algorithms create randomly initialized population and improve the population during iterations to search for global optimum in search space. The typical search process of optimization algorithm can be divided into two phases: exploration and exploitation [1-3]. In the exploration phase, search agents investigate the search space as widely as possible to obtain the promising region. While in the exploitation phase, search is carried out in the local region obtained by exploration phase to find the global optimum solution. It is very important for an algorithm to strike a proper balance between exploration and exploitation to avoid local optimum and find global optimum solution quickly. Some of the most popular meta-heuristic algorithms are Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Artificial Fish Swarm Algorithm (AFSA) and Grey Wolf Optimization (GWO).

Whale Optimization Algorithm (WOA) is a new meta-heuristic optimization algorithm proposed by Seyedali Mirjalili and Andrew Lewis in 2016. WOA mimics the hunting

DOI: 10.24507/ijicic.15.03.1049

behavior of humpback whales called bubble-net feeding method. Humpback whales prefer to hunt school of krill or small fishes close to the surface by creating distinctive bubbles along a circle or '9'-shaped path [1,4]. Some studies have been proposed to improve the performance of basic WOA and apply it into different fields [4-10]. CWOA is proposed in [5] which blends WOA with chaos theory and applies CWOA to solving transient stability constrained OPF problem. The authors in [6] propose another CWOA to improve the diversity and egocentricity of search agents and apply the proposed CWOA to optimizing the Elman neural network. LWOA is proposed in [7] to improve the performance of WOA based on Lévy flight trajectory and is applied to solving infinite impulse response model identification. To improve the exploration and exploitation ability as well as reduce the possibility of falling into local optimum of WOA, we optimize WOA in three aspects and apply the improved WOA to multi-resource allocation problem. The main contributions of this work are described as follows.

- Introducing nonlinearly changed convergence factor to preferably adjust the exploration and exploitation process of WOA. The linearly changed convergence factor limits the exploration and exploitation ability of WOA, which can be improved by using nonlinearly changed convergence factor.
- Adding a new inertia weight factor to adjust the influence of current best search agent on the movements of other search agents. Consequently, the exploration and exploitation ability and convergence speed of WOA are greatly improved.
- Conducting random variation of current best search agent during exploitation process. In the exploitation process of WOA, all the search agents move towards the best search agent which may be a local optimum. To reduce the possibility of falling into local optimum, we conduct a certain number of variations during each iteration of exploitation process.
- Testing IWOA with 29 benchmark functions and comparing it with other well-known meta-heuristic algorithms. Four types of benchmark functions are employed to test the performance of algorithms from different perspectives, including exploration ability, exploitation ability, ability to escape from local minima and convergence speed.
- Applying IWOA to multi-resource allocation problem to evaluate the ability of solving engineering problem. Multi-resource allocation is a general engineering problem and various kinds of meta-heuristics have been proposed to solve it. IWOA and compared well-known meta-heuristics were benchmarked with system utility model to valuate the performance of solving multi-resource problem.

The rest of this paper is organized as follows. Section 2 outlines the basic WOA. Section 3 presents the Improved WOA (IWOA). Experimental results and performance analysis of the proposed IWOA are provided in Section 4. Section 5 applies the proposed IWOA to solving multi-resource allocation problem. Finally, in Section 6, conclusions and possible future works are given.

2. Basic Whale Optimization Algorithm.

The basic WOA includes three parts: encircling prey, bubble-net attacking method and search for prey. In this section, we will briefly describe the three parts.

2.1. Encircling prey.

WOA algorithm assumes that the current best search agent is the target prey or is close to the optimum. The other search agents will update their positions towards the best search agent according to the following equations:

$$D = |C \cdot X^*(t) - X(t)| \tag{1}$$

$$X(t+1) = X^*(t) - A \cdot D \tag{2}$$

where t is current iteration, and A and C are coefficient vectors. $|\ |$ is absolute value and $\cdot$ is an element-by-element multiplication. X is position vector and X^* is position vector of best solution obtained so far. X^* will be updated in each iteration if there is a better solution.

The vectors A and C are calculated as follows:

$$A = 2ar - a \tag{3}$$

$$C = 2r \tag{4}$$

where r is a random value in $[0, 1]$. a is a convergence factor that is linearly decreased from 2 to 0 during the iteration according to the following equation:

$$a = 2 - t \cdot 2/t_{\max} \tag{5}$$

where t is current iteration and $t_{\max}$ is the total iteration number.

2.2. **Bubble-net attacking method (exploitation phase).** Two approaching methods are designed in WOA to mathematically model the bubble-net behavior of humpback whales.

- Shrinking encircling mechanism: This behavior is achieved by decreasing the value of a in Equation (3). A is a random value in the interval $[-a, a]$, where a is decreased from 2 to 0 during the iteration. When the value of A is in the interval $[-1, 1]$, the new position of a search agent will be defined anywhere in between the original position of the search agent and the position of current best search agent.
- Spiral updating position: The helix-shaped movement of humpback whales is modeled by the following equation:

$$X(t+1) = D' \cdot e^{bl} \cdot \cos(2\pi l) + X^*(t) \tag{6}$$

where $D' = |X^*(t) - X(t)|$ and indicates the distance of the i^{th} whale to the prey (best solution obtained so far), b is a constant for defining the shape of the logarithmic spiral, l is a random number in $[-1, 1]$, and $\cdot$ is an element-by-element multiplication [1,4].

To model the two simultaneous approaching behaviors during the bubble-net attacking method, there is a probability of 50% to choose between either the shrinking encircling mechanism or the spiral model to update the position of whales. The mathematical model is shown as follows:

$$X(t+1) = \begin{cases} X^*(t) - A \cdot D & \text{if } p < 0.5 \\ D' \cdot e^{bl} \cdot \cos(2\pi l) + X^*(t) & \text{if } p \geq 0.5 \end{cases} \tag{7}$$

where p is a random number in $[0, 1]$. In addition to the bubble-net attacking method, the humpback whales also search for prey randomly, which is described as follows.

2.3. **Search for prey (exploration phase).** Humpback whales search randomly according to the position of each other. This behavior is also achieved by decreasing the value of factor a in Equation (3). When the value of A is greater than 1 or less than -1, the search agent will move far away from the reference whale. Different from the exploitation phase, a search agent updates its position according to a randomly chosen search agent rather than the current best search agent. This mechanism will allow the WOA algorithm to perform a global search. The mathematical model is as follows:

$$D = |C \cdot X_{rand} - X| \tag{8}$$

$$X(t+1) = X_{rand} - A \cdot D \tag{9}$$

where X_{rand} is a random position vector (a random whale) chosen from the current population [1,4].

The WOA algorithm starts with a set of randomly initialized solutions. At each iteration, the current best solution is selected when $|A| < 1$, while a random search agent is chosen when $|A| > 1$ for updating the position of the search agents. Depending on the value of p, WOA switches between either a spiral or circular movement. Finally, the WOA algorithm is terminated by the satisfaction of an end criterion. In WOA, the exploration and exploitation ability and balance between them mostly depend on convergence factor a. However, the linear change of a limits the exploration and exploitation ability and cannot reflect the real search process of WOA, causing low solution precision and slow convergence speed when facing complex global searching problem. What is more, all the search agents move towards best individual in exploitation process, which may cause local optimum.

3. Improved Whale Optimization Algorithm (IWOA).

The performance of basic WOA is improved in three aspects: nonlinear convergence factor, inertia weight factor and random variation of best search agent. The detailed introduction of the three aspects and the pseudo code of IWOA will be discussed in the following three subsections.

3.1. Nonlinear convergence factor.

As we can know from current research concerning WOA and GWO, the linearly changed convergence factor a cannot reflect the real optimizing process of the algorithm and limits the exploration and exploitation ability. Nonlinearly changed convergence factor is a usual and effective improvement method [11-14]. Hence, we propose the following nonlinearly changed convergence factor:

$$a(t) = \frac{2 \times (1 - t/t_{\max})^2}{(1 - \mu \times t/t_{\max})^3} \tag{10}$$

where t is current iteration and $t_{\max}$ indicates the total iteration number. μ is the adjustment coefficient and the value is in the interval $[15, 35]$.

3.2. Inertia weight factor.

To further enhance exploration and exploitation ability and accelerate convergence speed, we introduce a new inertia weight factor to WOA inspired by PSO algorithm. In PSO, inertia weight factor can adjust the global and local search abilities. The global search ability of PSO is better when inertia weight is big while the local search ability is better when inertia weight is small. In WOA, the movement of a search agent is determined by the referenced search agent. Inertia weight factor ω is introduced to tune the influence of referenced search agent on the movement of search agent. After introducing the inertia weight factor, the exploration and exploitation ability and convergence speed of WOA are adjusted by convergence factor a and inertia weight factor ω together. The position updated method with inertia weight factor is represented by the following equations:

$$X(t+1) = \begin{cases} \omega X^*(t) - A \cdot D & \text{if } p < 0.5,\ |A| < 1 \\ \omega X_{rand}(t) - A \cdot D & \text{if } p < 0.5,\ |A| \geq 1 \\ D' \cdot e^{bl} \cdot \cos(2\pi l) + \omega X^*(t) & \text{if } p \geq 0.5 \end{cases} \tag{11}$$

where t is current iteration, and A and C are coefficient vectors. $D' = |X^*(t) - X(t)|$ and D is calculated as Equation (1) or Equation (8). $|\ |$ is absolute value, $\cdot$ is an element-by-element multiplication. X is position vector and X^* is position vector of best solution obtained so far. ω is inertia weight factor and is calculated as follows:

$$\omega = \alpha \times rand() \tag{12}$$

where α is a number in the interval $[0.5, 2.5]$.

3.3. **Random variation of best search agent.** In the exploitation process of WOA, all the search agents move towards the current best search agent. If the current best solution is a local optimum, the WOA will fall into local optimum. To reduce the probability of local optimum, we propose random variation of the current best search agent. Supposing the current best search agent is $X_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, we choose an element x_k ($k = 1, 2, \ldots, d$) from X_i and replace it with a random number in $[l_i, u_i]$. The newly generated search agent is $X_i' = (x_{i1}', x_{i2}', \ldots, x_{id}')$. The mathematical model is shown as follows:

$$X_i' = \begin{cases} l_i + \lambda \times (u_i - l_i), & i = k \\ X_i, & \text{otherwise} \end{cases} \tag{13}$$

where l_i and u_i are the lower bound and upper bound of x_i respectively, and λ is a random value in $[0, 1]$. However, a single time of variation of best search agent may result in the search agent jumping out the current region that includes the global best solution and reaching a local better solution. To reduce the possibility of this phenomenon, the above random variation operation is carried out for N times during each exploitation iteration.

Above all, the pseudo code of IWOA algorithm is presented in Figure 1. In IWOA, the exploration and exploitation process is divided by A and which kind of approaching behavior is adopted in exploitation process is determined by p. If the value of p is larger than 0.5, the search agent will perform spiral movement towards current best search agent, which is shown in line 14 of Figure 1. When the value of p is less than 0.5 and

(1) Initialize the whales population X_i ($i = 1, 2, \ldots, n$)
(2) Calculate the fitness of each search agent, X^* = the best search agent
(3) **while** ($t <$ maximum iteration number)
(4) **for** each search agent
(5) Update a, A, C, l, and p
(6) **if1** ($p < 0.5$)
(7) **if2** ($|A| < 1$)
(8) **Update the position of the current search agent by Equation (11)**
(9) **else if2** ($|A| \geq 1$)
(10) Select a random search agent (X_{rand})
(11) **Update the position of the current search agent by Equation (11)**
(12) **end if2**
(13) **else if1** ($p \geq 0.5$)
(14) **Update the position of the current search by Equation (11)**
(15) **end if1**
(16) **end for**
(17) Check if any search agent goes beyond the search space and amend it
(18) Calculate the fitness of each search agent, Update X^* if there is a better solution
(19) **if3** ($|A| < 1$)
(20) **Random variation of current best search agent for N times by Equation (13)**
(21) **end if3**
(22) $t = t + 1$
(23) **end while**
(24) return X^*

FIGURE 1. Pseudo code of IWOA algorithm

the absolute value of A is less than 1, the search agent will perform shrinking encircling movement towards current best search agent. Otherwise, the search agent will conduct search behavior by selecting a random search agent and moving towards it. The two movements are shown in lines 8 and 11. Line 20 shows that the random variation of current best search agent is conducted for N times if it is exploitation process.

4. **Results on Benchmark Experiment.** The numerical efficiency of the proposed I-WOA algorithm is benchmarked on 29 test functions [15,16], which can be divided into four groups: unimodal, multi-modal, fixed-dimension multimodal and composite functions. Different types of functions benchmark the performance of the algorithm from different perspectives. These functions are listed in Tables 1-4 where Dim indicates dimension of the functions, Range is the boundary of the function's search space, and $f_{\min}$ is the optimum.

Table 1. Unimodal benchmark functions

Function	Dim	Range	$f_{\min}$
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]$	0
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	30	$[-100, 100]$	0
$F_4(x) = \max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$	30	$[-100, 100]$	0
$F_5(x) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	30	$[-30, 30]$	0
$F_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30	$[-100, 100]$	0
$F_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	30	$[-1.28, 1.28]$	0

Table 2. Multimodal benchmark functions

Function	Dim	Range	$f_{\min}$
$F_8(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert} \right)$	30	$[-500, 500]$	-418.9829×5
$F_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	$[-5.12, 5.12]$	0
$F_{10}(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2} \right)$ $- \exp\left(\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i) \right) + 20 + e$	30	$[-32, 32]$	0
$F_{11}(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	30	$[-600, 600]$	0
$F_{12}(x) = \frac{\pi}{n} \left\{ 10 \sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right\}$ $+ \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	$[-50, 50]$	0
$F_{13}(x) = 0.1 \left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{n} (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] \right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	$[-50, 50]$	0

TABLE 3. Fixed-dimension multimodal benchmark functions

Function	Dim	Range	$f_{\min}$
$F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j+\sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$	2	$[-65, 65]$	0
$F_{15}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_i^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right]^2$	4	$[-5, 5]$	0.00030
$F_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]$	-1.0316
$F_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	$[-5, 5]$	0.398
$F_{18}(x) = \left[1 + (x_1 + x_2 + 1)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right] \times \left[30 + (2x_1 - 3x_2)^2 \times \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right]$	2	$[-2, 2]$	3
$F_{19}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$	3	$[1, 3]$	-3.86
$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$	6	$[0, 1]$	-3.32
$F_{21}(x) = -\sum_{i=1}^{5} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	-10.1532
$F_{22}(x) = -\sum_{i=1}^{7} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	-10.4028
$F_{23}(x) = -\sum_{i=1}^{10} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	-10.5363

To verify the results, IWOA is compared to basic WOA, GWO and PSO. The values of μ and α are set to 25 and 0.5 respectively, which will be justified in the following Subsection 4.5. The value of N is set to 20. Each algorithm runs 30 times on each test function and the collected statistical results (average and standard deviation of the best solution) are shown in Tables 5, 7, 9, 11. For unimodal, multimodal and fixed-dimension multimodal functions, each of them is solved by 30 search agents over 1000 iterations. Each composite function is solved by 30 search agents over 100 iterations. Apart from average and standard deviation, p-value of Wilcoxon ranksum test [17] is conducted to determine the significance level of two algorithms and the results are listed in Tables 6, 8, 10, 12. If a p-value is less than 0.05, it means that the difference between two compared algorithms is statistically significant.

4.1. **Evaluation of exploitation capability.** Functions F_1-F_7 are unimodal since they have only one global optimum. These functions allow to evaluate the exploitation capability of the algorithm. It can be seen from Table 5 that IWOA achieves best performance in F_1-F_5, F_7 and second best performance in F_6. Besides, the p-values in Table 6 are much less than 0.05 and it means that IWOA performs statistically much better than the compared algorithms. The above test results demonstrate that IWOA provides very good exploitation capability.

4.2. **Evaluation of exploration capability.** In contrast to umimodal functions, multimodal functions (F_8-F_{23}) have many local optima whose number increases exponentially with the number of variables. Therefore, this kind of test functions is very suitable to

TABLE 4. Composite benchmark functions

Function	Dim	Range	$f_{\min}$
$F_{24}(CF1)$	30	$[-5,5]$	0
$f_1, f_2, f_3, \ldots, f_{10}$ = *Sphere Function*,			
$[\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{10}] = [1, 1, 1, \ldots, 1]$			
$[\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{10}] = [5/100, 5/100, 5/100, \ldots, 5/100]$			
$F_{25}(CF2)$	30	$[-5,5]$	0
$f_1, f_2, f_3, \ldots, f_{10}$ = *Griewank's Function*,			
$[\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{10}] = [1, 1, 1, \ldots, 1]$			
$[\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{10}] = [5/100, 5/100, 5/100, \ldots, 5/100]$			
$F_{26}(CF3)$	30	$[-5,5]$	0
$f_1, f_2, f_3, \ldots, f_{10}$ = *Griewank's Function*			
$[\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{10}] = [1, 1, 1, \ldots, 1]$,			
$[\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{10}] = [1, 1, 1, \ldots, 1]$			
$F_{27}(CF4)$	30	$[-5,5]$	0
f_1, f_2 = *Ackley's Function*, f_3, f_4 = *Rastrigin's Function*,			
f_5, f_6 = *Weierstrass Function*, f_7, f_8 = *Griewank's Function*,			
f_9, f_{10} = *Sphere Function*, $[\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{10}] = [1, 1, 1, \ldots, 1]$			
$[\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{10}] = [5/32, 5/32, 1, 1, 5/0.5, 5/0.5, 5/100,$ $5/100, 5/100, 5/100]$			
$F_{28}(CF5)$	30	$[-5,5]$	0
f_1, f_2 = *Rastrigin's Function*, f_3, f_4 = *Weierstrass Function*,			
f_5, f_6 = *Griewank's Function*, f_7, f_8 = *Ackley's Function*,			
f_9, f_{10} = *Sphere Function*, $[\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{10}] = [1, 1, 1, \ldots, 1]$			
$[\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{10}] = [1/5, 1/5, 5/0.5, 5/0.5, 5/100,$ $5/100, 5/32, 5/32, 5/100, 5/100]$			
$F_{29}(CF6)$	30	$[-5,5]$	0
f_1, f_2 = *Rastrigin's Function*, f_3, f_4 = *Weierstrass Function*,			
f_5, f_6 = *Griewank's Function*, f_7, f_8 = *Ackley's Function*,			
f_9, f_{10} = *Sphere Function*			
$[\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{10}] = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]$			
$[\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{10}] = [0.1 * 1/5, 0.2 * 1/5, 0.3 * 5/0.5, 0.4 * 5/0.5,$ $0.5 * 5/100, 0.6 * 5/100, 0.7 * 5/32,$ $0.8 * 5/32, 0.9 * 5/100, 1 * 5/100]$			

evaluate the exploration capability of an algorithm. According to the results in Tables 7 and 9, IWOA outperforms all other compared algorithms on all the multimodal test functions except for F_{16}, F_{17}, F_{22} and F_{23}. The p-values presented in Tables 8 and 10 also certify that IWOA performs significantly better than other compared algorithms. It is obvious that IWOA also exhibits very excellent exploration capability.

4.3. **Ability to escape from local minima.** Composite functions are very challenging test beds for meta-heuristic algorithms because it tests the exploitation and exploration ability simultaneously. Only a proper balance is struck between the exploration and exploitation phases can local optima be avoided. The results in Tables 11 and 12 show that IWOA outperforms other algorithms on F_{24}, F_{29} and provides very competitive performance on other composite functions. The above results prove that the IWOA can provide very competitive ability to escape from local minima.

TABLE 5. Results of unimodal benchmark functions

F	Type	IWOA	WOA	GWO	PSO
F_1	Ave	**0**	1.9041e-151	4.8457e-59	1.0958e-08
	Std	**0**	1.0366e-150	1.4651e-58	2.1943e-08
F_2	Ave	**0**	1.1382e-102	1.0347e-34	7.0003
	Std	**0**	5.2393e-102	1.0982e-34	8.3667
F_3	Ave	**0**	22019.7193	1.4007e-15	16.7928
	Std	**0**	10741.5251	3.0218e-15	8.4393
F_4	Ave	**0**	41.3522	1.7273e-14	0.63892
	Std	**0**	31.7993	2.061e-14	0.15003
F_5	Ave	**0.99072**	27.2429	26.8548	160.7873
	Std	5.0979	**0.51479**	0.85161	545.3538
F_6	Ave	0.029631	0.088295	0.65025	**1.2409e-07**
	Std	0.03281	0.12346	0.33988	**5.8755e-07**
F_7	Ave	**3.6978e-05**	0.0014232	0.00078793	3.277
	Std	**3.0105e-05**	0.0016038	0.00040596	5.1537

TABLE 6. p-values of the Wilcoxon ranksum test over unimodal benchmark functions

F	IWOA	WOA	GWO	PSO
F_1	N/A	1.2118e-12	1.2118e-12	1.2118e-12
F_2	N/A	1.2118e-12	1.2118e-12	1.2118e-12
F_3	N/A	1.2118e-12	1.2118e-12	1.2118e-12
F_4	N/A	1.2118e-12	1.2118e-12	1.2118e-12
F_5	N/A	5.0723e-10	4.1997e-10	8.9934e-11
F_6	N/A	0.029205	3.0199e-11	3.0199e-11
F_7	N/A	1.4643e-10	3.0199e-11	3.0199e-11

TABLE 7. Results of multi-modal benchmark functions

F	Type	IWOA	WOA	GWO	PSO
F_8	Ave	**−12568.425**	−11424.1706	−6196.377	−6192.0758
	Std	**0.45296**	1568.8643	650.5042	1530.2941
F_9	Ave	**0**	**0**	0.15873	106.7903
	Std	**0**	**0**	0.86938	28.4414
F_{10}	Ave	**8.8818e-16**	4.0856e-15	1.7112e-14	0.038557
	Std	**0**	2.158e-15	3.8108e-15	0.21089
F_{11}	Ave	**0**	0.0019151	0.0025294	0.0068944
	Std	**0**	0.010489	0.005314	0.0087035
F_{12}	Ave	**9.8882e-05**	0.0075511	0.038067	0.0069113
	Std	**0.00010377**	0.0088261	0.019881	0.026302
F_{13}	Ave	**0.001424**	0.17788	0.51169	0.0032962
	Std	**0.0016209**	0.11249	0.18207	0.0051211

4.4. **Analysis of convergence behavior.** In this subsection the convergence behavior of IWOA is investigated. According to Bergh and Engelbrecht [18], there should be abrupt changes in the movement of search agents over the initial steps of optimization to explore the search space extensively, and then the changes should be reduced to focus on exploitation at the end of optimization. This guarantees a population-based algorithm

TABLE 8. p-values of the Wilcoxon ranksum test over multi-modal benchmark functions

F	IWOA	WOA	GWO	PSO
F_8	N/A	1.1567e-07	3.0199e-11	3.0199e-11
F_9	N/A	NaN	0.0013247	1.2118e-12
F_{10}	N/A	2.742e-09	3.7782e-13	1.2118e-12
F_{11}	N/A	0.33371	0.011035	1.2118e-12
F_{12}	N/A	3.3384e-11	3.0199e-11	8.4848e-09
F_{13}	N/A	3.0199e-11	3.0199e-11	0.007959

TABLE 9. Results of fixed-dimension multimodal benchmark functions

F	Type	IWOA	WOA	GWO	PSO
F_{14}	Ave	**0.998**	2.5357	5.1739	3.6607
	Std	**9.3074e-11**	3.0119	4.3923	2.4925
F_{15}	Ave	**0.00037713**	0.0005287	0.003681	0.0037375
	Std	**8.0891e-05**	0.00014271	0.00759	0.0068748
F_{16}	Ave	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**
	Std	3.9212e-06	6.1097e-11	5.181e-09	**6.6486e-16**
F_{17}	Ave	**0.39789**	**0.39789**	**0.39789**	**0.39789**
	Std	6.6218e-07	2.4908e-06	3.7745e-07	**0**
F_{18}	Ave	**3**	**3**	5.7	5.7
	Std	**1.9809e-05**	2.7488e-05	14.7885	14.7885
F_{19}	Ave	**−3.8628**	−3.8604	−3.8614	−3.8617
	Std	**2.5611e-07**	0.0028611	0.0027758	0.002725
F_{20}	Ave	**−3.2744**	−3.2524	−3.251	−3.2374
	Std	**0.059241**	0.083037	0.081519	0.096601
F_{21}	Ave	**−9.8131**	−9.2207	−9.3107	−8.8907
	Std	**1.2934**	2.1533	1.9151	2.3641
F_{22}	Ave	−9.339	−8.3607	**−9.8737**	−9.1099
	Std	2.162	2.7558	**1.6134**	2.6826
F_{23}	Ave	−9.9903	−8.7696	**−10.536**	−10.1699
	Std	1.6486	2.7791	**0.00019883**	1.3585

TABLE 10. p-values of the Wilcoxon ranksum test over fixed-dimension multimodal benchmark functions

F	IWOA	WOA	GWO	PSO
F_{14}	N/A	0.00015846	3.2555e-07	0.00039091
F_{15}	N/A	0.00016813	0.00010407	7.3445e-10
F_{16}	N/A	3.0199e-11	3.6897e-11	2.3638e-12
F_{17}	N/A	0.19073	0.16687	1.2118e-12
F_{18}	N/A	0.00026806	0.65204	1.4507e-10
F_{19}	N/A	3.0199e-11	3.8202e-10	4.2293e-07
F_{20}	N/A	0.00028389	0.0014423	0.59676
F_{21}	N/A	2.5721e-07	5.6073e-05	0.00017163
F_{22}	N/A	0.00037704	0.63088	7.1797e-05
F_{23}	N/A	1.0907e-05	0.19073	2.2841e-08

TABLE 11. Results of composite benchmark functions

F	Type	IWOA	WOA	GWO	PSO
F_{24}	Ave	**63.0787**	330.9793	161.2208	298.247
	Std	126.9148	**89.2588**	142.7551	141.3892
F_{25}	Ave	554.0999	541.5759	**369.4023**	373.0556
	Std	286.0643	138.8732	139.3397	**125.6001**
F_{26}	Ave	609.0276	1023.5434	**474.6884**	625.768
	Std	285.5999	167.7353	121.7745	**108.8543**
F_{27}	Ave	890.9206	908.0536	800.0873	**767.9273**
	Std	**49.7298**	56.327	141.9259	153.9118
F_{28}	Ave	639.5027	575.1635	**127.6304**	323.7544
	Std	389.3627	286.6679	**113.3831**	158.1828
F_{29}	Ave	**900**	**900**	900.0005	929.3696
	Std	**0**	2.7426e-12	0.00044274	20.5434

TABLE 12. p-values of the Wilcoxon ranksum test over composite benchmark functions

F	IWOA	WOA	GWO	PSO
F_{24}	N/A	7.1186e-09	2.4327e-05	1.0666e-07
F_{25}	N/A	1	0.046085	0.0034018
F_{26}	N/A	5.3824e-08	0.27413	0.77171
F_{27}	N/A	2.4759e-09	0.12592	0.00053813
F_{28}	N/A	0.47275	0.0018645	0.0029259
F_{29}	N/A	1.6024e-10	1.2118e-12	1.2118e-12

eventually converges to a point in the search space. In order to observe the convergence behavior of IWOA, convergence curves of the IWOA, WOA, GWO and PSO are provided in Figure 2 for some of the benchmark functions. The average best-so-far indicates the average of the best solution obtained so far in each iteration over 30 runs.

As is shown in Figure 2, IWOA provides best solution and convergence rate on most of benchmark functions and provides second best performance on some functions. There are three different convergence behaviors when IWOA optimizes the test functions. The first convergence curve accords with Bergh and Engelbrecht [18]. The slope of convergence curve is large in initial iterations and gradually becomes small at the end of optimization. This behavior is evident in F_5, F_6, F_{15}, F_{18}. We notice that IWOA performs better than basic WOA and other two algorithms on these functions. This is due to the nonlinear change of convergence factor a which preferably adjusts exploration and exploitation process. The second convergence curve directly searches the theoretical best solution in the first half of iterations and this behavior is evident in F_1, F_9. This superior search ability is mainly due to the introduction of inertia weight ω which further enhances the exploration and exploitation ability. The third convergence behavior shows abrupt changes in the second half of iterations and this behavior is evident in F_7, F_{12}, F_{24}. This benefits from the Random Variation Mechanism of current best search agent during exploitation iterations.

To sum up, the results verify the high performance of IWOA in solving different kinds of benchmark functions compared to well-known meta-heuristics. In the following sections, the performance of IWOA is further verified on multi-resource allocation problem.

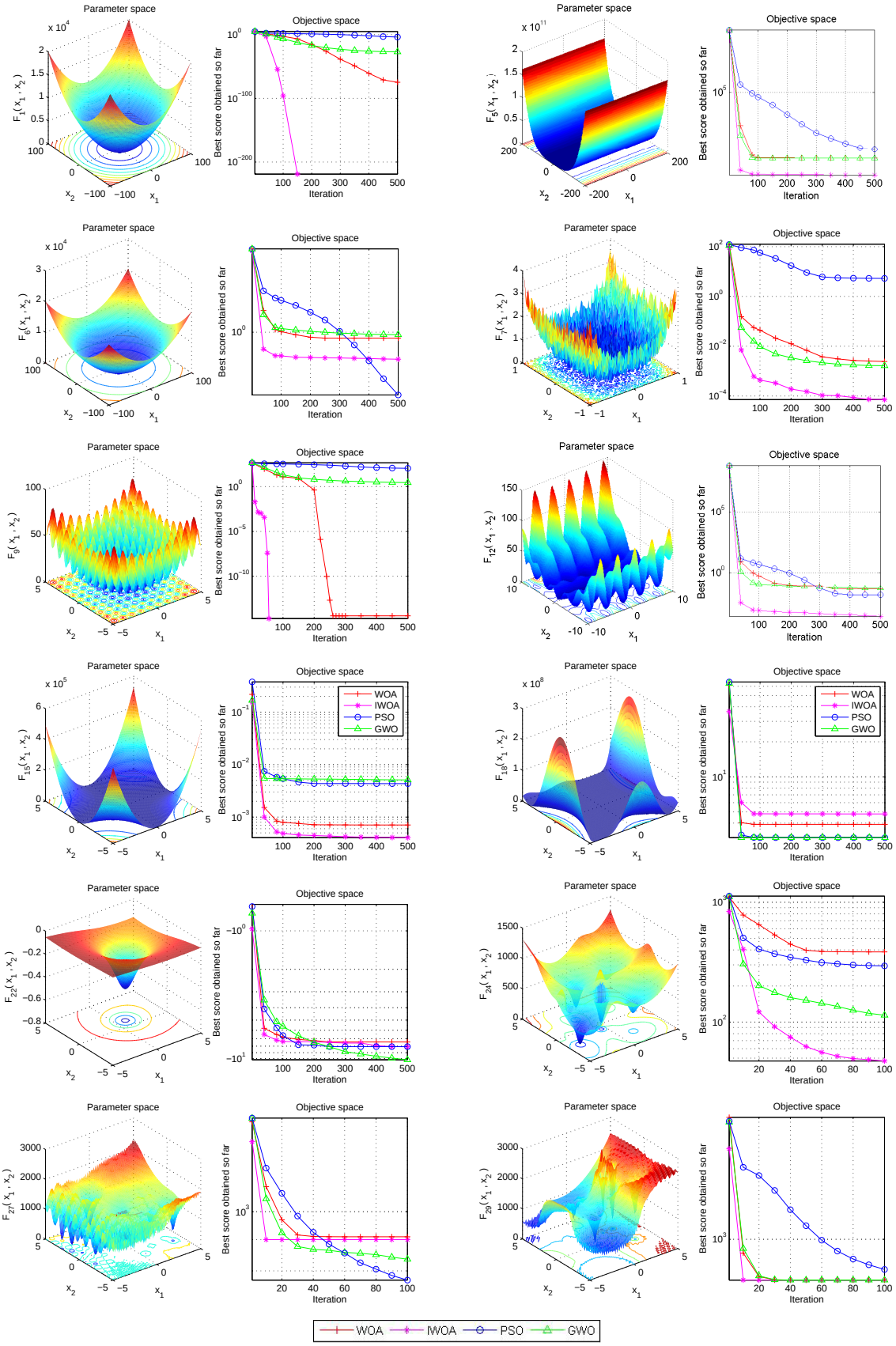

FIGURE 2. Comparison of convergence curves in some of the benchmark functions

4.5. **Experiments on the setting of values μ and α.** In this subsection, the settings of values μ and α in Equation (10) and Equation (12) are evaluated. Considering that there are two types of values to be evaluated, one of the values changes while the other one stays unchanged to verify the impact on algorithm performance. In the experiments of setting value μ, μ is set to 15, 20, 25, 30, 35 and α is set to 1.5. In the experiments of setting value α, α is set to 0.5, 1, 1.5, 2, 2.5 and μ is set to 25. Each setting is benchmarked 30 times on the test functions with 30 search agents over 100 iterations. Some of the results are shown in Figure 3 and Figure 4. As we can see from Figure 3, IWOA converges faster on unimodal benchmark functions when the value of α is smaller and the performance is competitive on other types of test functions. IWOA converges slower on unimodal benchmark functions when the value of α is 2.5 and performances are unstable on other types of test functions. Figure 4 shows that the bigger the value of μ is, the faster IWOA converges to optimal value on unimodal benchmark functions. However,

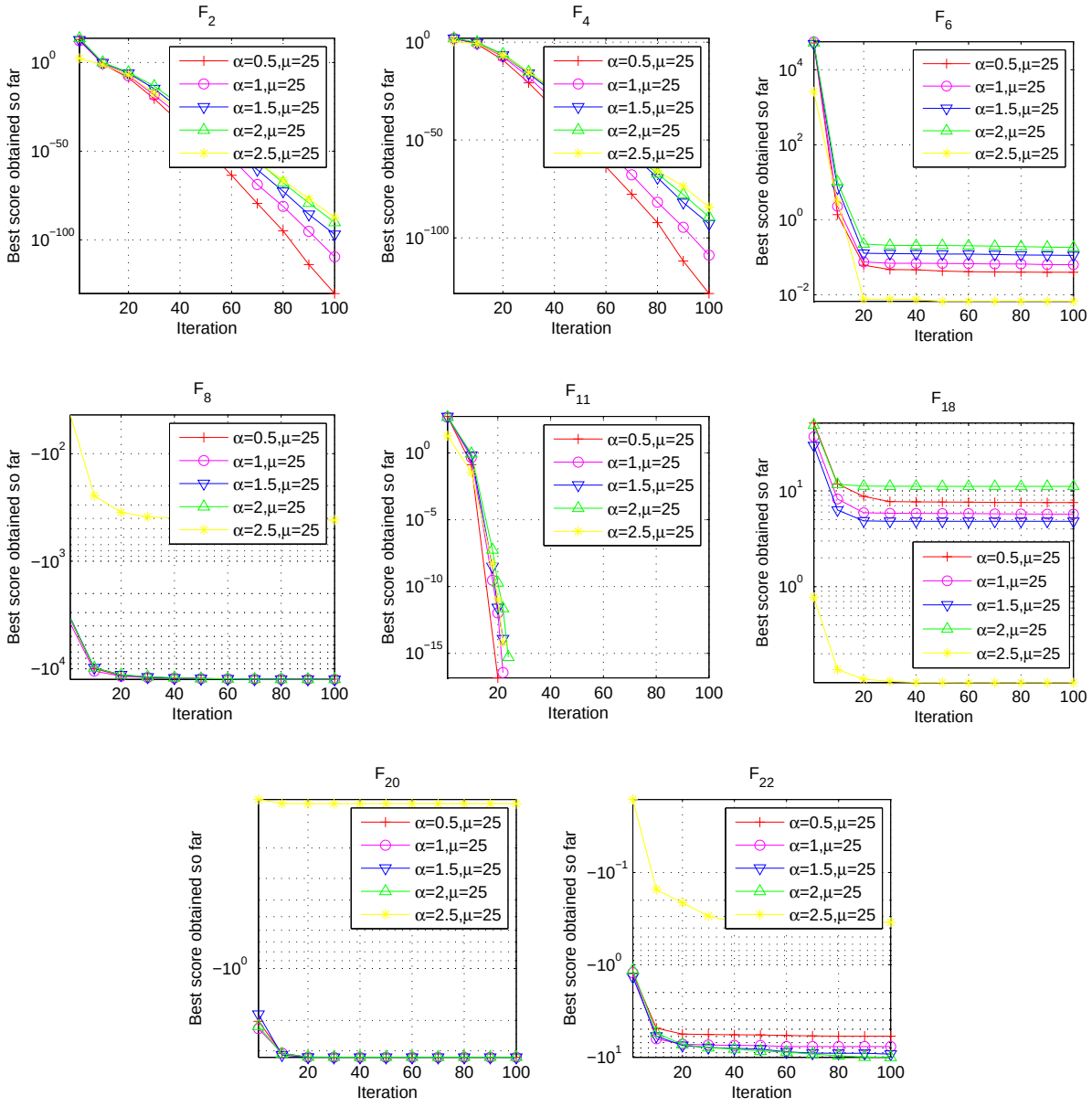

FIGURE 3. Experiments on the setting of value α

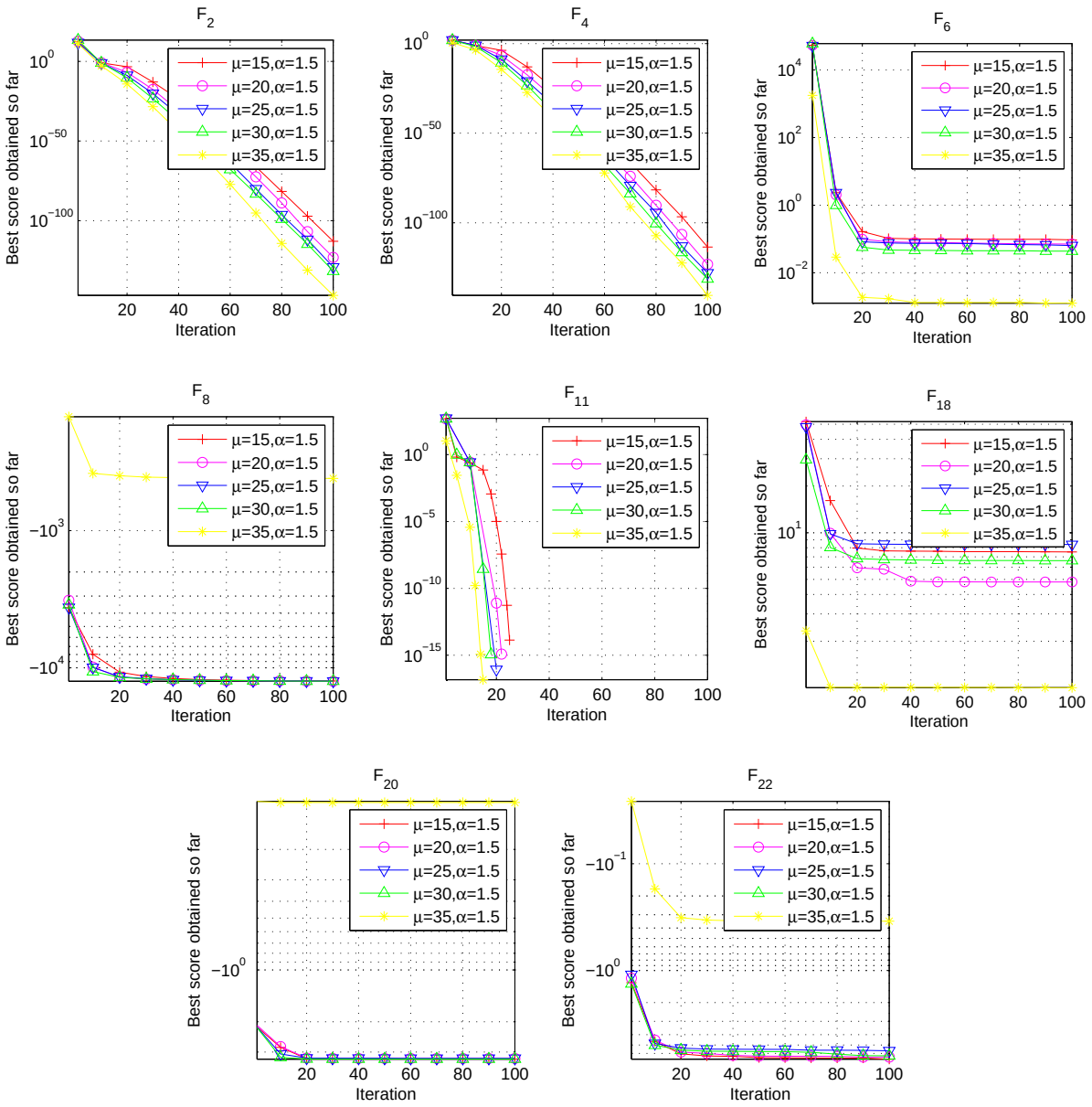

FIGURE 4. Experiments on the setting of value μ

the performance tends to become unstable on other types of test functions. A middle value is more acceptable. Hence, we set the value of α and μ to 0.5 and 25 respectively.

5. **IWOA for Multi-Resource Allocation.** Multi-resource allocation is a general engineering problem that exists in many fields such as wireless network [19-22], embedded system [23,24], real-time system [25], cloud computing [26-28], mobile edge computing [29] and data center [30,31]. In multi-resource allocation problem, many tasks compete for multiple kinds of limited system resources. Each task has a minimum requirement of each type of resource. The general solution to multi-resource allocation problem is allocating multiple types of resources among multiple tasks to fulfil a certain objective. There are three kinds of objectives in existing multi-resource allocation researches. First, maximize the overall system utility [32,33]. Second, achieve fairness of resource allocation between multiple tasks [30,34]. Third, trade off between system utility and fairness [23,27]. Many

different methods such as neural network [34], mixed integer programming [35], auction approach [20,36] and meta-heuristics algorithm are adopted to solve multi-resource allocation problem. Meta-heuristic algorithm is one of very important methods and various kinds of meta-heuristic algorithms [22,28,32,37-39] have been used to solve multi-resource allocation problem. Meta-heuristic algorithms have advantages in optimization [2,3,25]. What is more, when the number of resources and tasks increases, many traditional methods used in multi-resource allocation become complicated and the time complexity tends to become unacceptable [32]. Hence, we apply the proposed IWOA algorithm to solve multi-resource allocation problem and the objective is to maximize the overall system utility.

5.1. **Multi-resource allocation model.** The multi-resource allocation problem is described as follows. (1) Set of n tasks $\pi = \{\pi_1, \pi_2, \ldots, \pi_n\}$; (2) Set of m shared resource $R = \{R_1, R_2, \ldots, R_m\}$; (3) Suppose that the resource R_i acquired by π_j is R_{ij} $(1 \le i \le m,\ 1 \le j \le n)$, then $R_{i1} + R_{i2} + \cdots + R_{in} = \sum_{j=1}^{n} R_{ij} \le R_i,\ 1 \le i \le m$; (4) The resource demand of R_i by π_j has a minimum value $R_{ij}^{\min}$ and a maximum value $R_{ij}^{\max}$. $R_{ij}^{\min}$ must be satisfied to reach the lowest QoS. We can improve QoS by adding the resource allocation R_{ij} until R_{ij} reaches $R_{ij}^{\max}$, when the QoS will not improve even if we continue to add R_{ij}; (5) The utility function [30,32] of π_j is defined as follows: $q_j = \lambda_j f_j(R_{1j}, R_{2j}, \ldots, R_{mj})$, where λ_j is a value in $[0, 1]$, which indicates the weight parameter of task π_j. $q_j\left(R_{1j}^{\min}, R_{2j}^{\min}, \ldots, R_{mj}^{\min}\right) = q_{j\min}$, $q_j\left(R_{1j}^{\max}, R_{2j}^{\max}, \ldots, R_{mj}^{\max}\right) = q_{j\max}$; (6) The utility function of different tasks may be different from each other. To have a uniform measure standard of the utility of different tasks, we normalize the utility function as follows:

$$Q_j(R_{1j}, \ldots, R_{mj}) = \begin{cases} 0, & R_{ij} = R_{ij}^{\min} \\ \dfrac{q_j - q_{j\min}}{q_{j\max} - q_{j\min}}, & R_{ij}^{\min} < R_{ij} < R_{ij}^{\max} \\ 1, & R_{ij} = R_{ij}^{\max} \end{cases} \tag{14}$$

where $1 \le i \le m$, $1 \le j \le n$, $0 \le Q_j \le 1$.

5.2. **System utility maximization model.** In this section we give the system utility maximization model with n tasks and m resources. In a system with task set $\{\pi_1, \pi_2, \ldots, \pi_n\}$ and resource set $\{R_1, R_2, \ldots, R_m\}$, the utility of task π_j is Q_j $(R_{1j}, R_{2j}, \ldots, R_{mj})$. We need to search for an $m \times n$ resource allocation matrix U_{ij} to maximize the utility of all tasks. The system utility maximization model can be formulated as follows:

$$\max \sum_{j=1}^{n} Q_j(R_{1j}, R_{2j}, \ldots, R_{mj}) \tag{15}$$

subject to:

$$\sum_{j=1}^{n} R_{ij} \le R_i, \quad 1 \le i \le m \tag{16}$$

$$R_{ij}^{\min} < R_{ij} < R_{ij}^{\max}, \quad 1 \le i \le m \tag{17}$$

In addition, Q_j in Equation (15) satisfies the normalization in Equation (14).

5.3. **Results of IWOA on multi-resource allocation problem.** In the experiments, we set the number of resource types m to 4 and the number of parallelized tasks n to 10. We give 4 types of tasks as in [32]:

(1) $Q_j = \left(\sum_{i=1}^{4} R_{ij} - \sum_{i=1}^{4} R_{ij}^{\min}\right) \Big/ \left(\sum_{i=1}^{4} R_{ij}^{\max} - \sum_{i=1}^{4} R_{ij}^{\min}\right)$

(2) $Q_j = \left(\prod_{i=1}^{4} R_{ij} - \prod_{i=1}^{4} R_{ij}^{\min}\right) \Big/ \left(\prod_{i=1}^{4} R_{ij}^{\max} - \prod_{i=1}^{4} R_{ij}^{\min}\right)$

(3) $Q_j = \sin\left(\frac{\pi}{2} \times \frac{\sum_{i=1}^{4} R_{ij} - \sum_{i=1}^{4} R_{ij}^{\min}}{\sum_{i=1}^{4} R_{ij}^{\max} - \sum_{i=1}^{4} R_{ij}^{\min}}\right)$

(4) $Q_j = 1 - \sin\left(\frac{\pi}{2} \times \frac{\prod_{i=1}^{4} R_{ij}^{\max} - \prod_{i=1}^{4} R_{ij}}{\prod_{i=1}^{4} R_{ij}^{\max} - \prod_{i=1}^{4} R_{ij}^{\min}}\right)$

The total available resources of each type are 1, and the resource requirements of 10 tasks are shown in Table 13 [32].

TABLE 13. Task type and resource requirement

Task	Minimum resource requirement	Maximum resource requirement	Task type
1	(0.02, 0.01, 0.02, 0.01)	(0.35, 0.4, 0.2, 0.3)	4
2	(0.01, 0.01, 0.01, 0.01)	(0.3, 0.25, 0.22, 0.25)	1
3	(0.02, 0.03, 0.03, 0.01)	(0.25, 0.3, 0.35, 0.2)	4
4	(0.01, 0.01, 0.02, 0.03)	(0.2, 0.3, 0.3, 0.3)	2
5	(0.02, 0.03, 0.01, 0.03)	(0.32, 0.3, 0.2, 0.2)	1
6	(0.03, 0.02, 0.03, 0.01)	(0.28, 0.22, 0.38, 0.19)	1
7	(0.01, 0.01, 0.02, 0.02)	(0.19, 0.18, 0.23, 0.26)	3
8	(0.01, 0.01, 0.01, 0.01)	(0.32, 0.28, 0.19, 0.18)	3
9	(0.02, 0.03, 0.01, 0.02)	(0.38, 0.3, 0.32, 0.29)	2
10	(0.01, 0.02, 0.01, 0.02)	(0.3, 0.4, 0.5, 0.35)	2

To verify the effectiveness of IWOA on multi-resource allocation problem, WOA, GWO and PSO algorithms are employed to solve multi-resource allocation problem. Each algorithm runs 30 times, and the collected statistical results (average and standard deviation of the best solution) are reported in Table 14. The problem is solved using 30 search agents over 1000 iterations. As is shown in Table 14, the proposed IWOA algorithm outperforms all other compared algorithms and is more stable. The p-values show that the difference between IWOA and compared algorithm is statistically significant.

TABLE 14. Result on multi-resource allocation problem

Algorithm	Ave	Std	p-value
IWOA	4.0004	4.1829e-05	N/A
WOA	3.6676	0.15068	3.0199e-11
GWO	3.9924	0.0051296	3.0199e-11
PSO	3.8466	0.19378	8.4848e-09

6. **Conclusions.** This paper proposed an Improved Whale Optimization Algorithm (IWOA). A nonlinear convergence factor a was adopted and a random inertia weight control parameter ω was introduced. To avoid local optimum, we introduced random variation of current best search agent during exploitation iterations. The performance of IWOA was tested on 29 benchmark functions and compared to basic WOA, GWO and PSO algorithms. The p-values of Wilcoxon statistical tests were also conducted to compare the performance of algorithms. The proposed IWOA was finally applied to multi-resource

allocation problem. The results show that IWOA provides highly competitive performances and outperforms other compared algorithms on most of the benchmark functions and provides best performance in multi-resource allocation problem. For future work, we intend to further improve the performance of IWOA, especially on composite functions. Besides, the random variation mechanism of best search agent can be well-designed to further improve local minima avoidance ability of IWOA.

Acknowledgment. This work is partially supported by Pioneering Action Initiative of Chinese Academy of Sciences (shuaixianxingdong jihua) (Project No. Y654101601). The authors would like to thank the anonymous referees for their comments and suggestions. Furthermore, the authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] H. Faris, I. Aljarah, M. A. Al-Betar et al., Grey wolf optimizer: A review of recent variants and applications, *Neural Computing & Applications*, no.22, pp.1-23, 2017.
[2] S. Mirjalili and A. Lewis, The whale optimization algorithm, *Advances in Engineering Software*, vol.95, pp.51-67, 2016.
[3] S. Mirjalili, S. M. Mirjalili and A. Lewis, Grey wolf optimizer, *Advances in Engineering Software*, vol.69, no.3, pp.46-61, 2014.
[4] H. Hu, Y. Bai and T. Xu, A whale optimization algorithm with inertia weight, *WSEAS Trans. Comput.*, vol.15, pp.319-326, 2016.
[5] D. Prasad, A. Mukherjee, G. Shankar et al., Application of chaotic whale optimisation algorithm for transient stability constrained optimal power flow, *IET Science Measurement & Technology*, vol.11, no.8, pp.1002-1013, 2017.
[6] W. Z. Sun and J. S. Wang, Elman neural network soft-sensor model of conversion velocity in polymerization process optimized by chaos whale optimization algorithm, *IEEE Access*, vol.5, pp.13062-13076, 2017.
[7] Y. Ling, Y. Zhou and Q. Luo, Lévy flight trajectory-based whale optimization algorithm for global optimization, *IEEE Access*, vol.5, pp.6168-6186, 2017.
[8] A. Saha and L. C. Saikia, Combined application of redox flow battery and DC link in restructured AGC system in the presence of WTS and DSTS in distributed generation unit, *IET Generation Transmission & Distribution*, vol.12, no.9, pp.2072-2085, 2018.
[9] A. H. Gandomi and A. R. Kashani, Construction cost minimization of shallow foundation using recent swarm intelligence techniques, *IEEE Trans. Industrial Informatics*, 2017.
[10] P. H. Venkatrao and S. S. Damodar, HWFusion: Holoentropy and SP-whale optimisation-based fusion model for magnetic resonance imaging multimodal image fusion, *IET Image Processing*, vol.12, no.4, pp.572-581, 2018.
[11] W. Long, S. Cai, J. Jiao et al., Improved whale optimization algorithm for large scale optimization problems, *Systems Engineering – Theory & Practice*, 2017.
[12] M. H. Zhong and W. Long, Whale optimization algorithm based on stochastic adjustment control parameter, *Science Technology & Engineering*, 2017.
[13] N. Mittal, U. Singh and B. S. Sohi, Modified grey wolf optimizer for global engineering optimization, *Applied Computational Intelligence and Soft Computing*, vol.2016, 2016.
[14] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, 1992.
[15] M. Molga and C. Smutnicki, *Test Functions for Optimization Needs*, 2005.
[16] X. S. Yang, Test problems in optimization, *Mathematics*, vol.2, no.2, pp.63-86, 2010.
[17] J. Derrac, S. García, D. Molina et al., A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, *Swarm & Evolutionary Computation*, vol.1, no.1, pp.3-18, 2011.
[18] F. van den Bergh and A. P. Engelbrecht, A study of particle swarm optimization particle trajectories, *Information Sciences*, vol.176, no.8, pp.937-971, 2006.
[19] C. Lee, An optimal resource allocation scheme for D2D communications underlaying 5G-based cellular networks, *ICIC Express Letters*, vol.12, no.7, pp.731-736, 2018.

[20] Y. Zhang, C. Lee, D. Niyato et al., Auction approaches for resource allocation in wireless systems: A survey, *IEEE Communications Surveys & Tutorials*, vol.15, no.3, pp.1020-1041, 2013.
[21] Z. Qi, L. J. Zhao and J. Li, Multi-dimension resource allocation algorithm based on QoS guarantee in dense WLAN, *Journal on Communications*, 2014.
[22] N. Yu, Z. Song, H. Du et al., Multi-resource allocation in cloud radio access networks, *IEEE International Conference on Communications*, pp.1-6, 2017.
[23] S. I. Park, V. Raghunathan and M. B. Srivastava, Energy efficiency and fairness tradeoffs in multi-resource, multi-tasking embedded systems, *International Symposium on Low Power Electronics and Design*, pp.469-474, 2003.
[24] Y. Tahir, S. Yang, U. Adeel et al., Symbiot: Congestion-driven multi-resource fairness for multi-user sensor networks, *IEEE International Conference on High Performance Computing & Communications*, 2015.
[25] D. Juedes, F. Drews, L. Welch et al., Heuristic resource allocation algorithms for maximizing allowable workload in dynamic, distributed real-time systems, *International Parallel and Distributed Processing Symposium*, p.117, 2004.
[26] S. M. Parikh, A survey on cloud computing resource allocation techniques, *Nirma University International Conference on Engineering*, pp.1-5, 2013.
[27] L. Zhao, M. Du and L. Chen, A new multi-resource allocation mechanism: A tradeoff between fairness and efficiency in cloud computing, *China Communications*, vol.15, no.3, pp.57-77, 2018.
[28] W. Wang, B. Liang and B. Li, Multi-resource fair allocation in heterogeneous cloud computing systems, *IEEE Trans. Parallel & Distributed Systems*, vol.26, no.10, pp.2822-2835, 2015.
[29] J. Guo, Z. Song, Y. Cui et al., Energy-efficient resource allocation for multi-user mobile edge computing, *IEEE Global Communications Conference*, pp.1-7, 2018.
[30] P. Poullie, T. Bocek and B. Stiller, A survey of the state-of-the-art in fair multi-resource allocations for data centers, *IEEE Trans. Network & Service Management*, 2017.
[31] K. Shen, X. Zheng, Y. Song et al., Fair multi-node multi-resource allocation and task scheduling in datacenter, *Cloud Computing Congress*, pp.59-63, 2013.
[32] Z. Huan, H. Ni, L. Hu et al., AAFSA-RA: A multi-resource allocation method based on an advanced artificial fish swarm algorithm, *Journal of Xi'an Jiaotong University*, vol.48, no.10, pp.120-125, 2014.
[33] J. Chen, H. Ni and P. Sun, Pricing mechanism based multi-resource allocation for multimedia system, *Journal of Xi'an Jiaotong University*, vol.46, no.6, pp.98-103, 2012.
[34] J. Lin, H. Ni, S. Peng et al., Adaptive resource allocation based on neural network PID control, *Journal of Xi'an Jiaotong University*, vol.47, no.4, pp.112-117+136, 2013.
[35] R. Rajkumar, C. Lee, J. P. Lehoczky et al., Practical solutions for QoS-based resource allocation problems, *Proc. of Real-Time Systems Symposium*, pp.296-306, 1998.
[36] C. Xu, X. Zeng and Z. Guo, CARA: A multi-resource allocation mechanism for smart TV system based on combinatorial auction, *Journal of Xi'an Jiaotong University*, vol.47, no.10, pp.25-30, 2013.
[37] Z. Wu, L. Zhang, Y. Wang et al., Optimization for multi-resource allocation and leveling based on a self-adaptive ant colony algorithm, *International Conference on Computational Intelligence and Security*, pp.47-51, 2008.
[38] J. Khamse-Ashari, I. Lambadaris, G. Kesidis et al., An efficient and fair multi-resource allocation mechanism for heterogeneous servers, *IEEE Trans. Parallel & Distributed Systems*, 2017.
[39] S. Divyalakshmi and D. Sivakumar, Multi-resource allocation to the job using bee optimization algorithm, *International Conference on Communications and Signal Processing*, pp.1333-1337, 2014.