Article

Quasi-Reflective Chaotic Mutant Whale Swarm Optimization Fused with Operators of Fish Aggregating Device

Shoubao Su [1,2,*], Chao He [1,2] and Liukai Xu [1,3]

[1] Jiangsu Key Laboratory of Data Science and Smart Software, Jinling Institute of Technology, Nanjing 211169, China; 209070019@just.edu.cn (C.H.); 1220045319@njupt.edu.cn (L.X.)
[2] School of Computer, Jiangsu University of Science and Technology, Zhenjiang 212003, China
[3] School of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
[*] Correspondence: showbo@jit.edu.cn

Abstract: To improve the performance of the whale optimization algorithm and further enhance the search accuracy, while increasing the convergence speed, a quasi-reflective chaotic mutant whale swarm optimization, namely QNWOA, is proposed, fused with an operator of Fish Aggregating Devices (FADs) in this paper. Firstly, the swarm diversity is increased by using logistic chaotic mapping. Secondly, a quasi-reflective learning mechanism is introduced to improve the convergence speed of the algorithm. Then, the FADs vortex effect and wavelet variation of the marine predator algorithm (MPA) are introduced in the search phase to enhance the stability of the algorithm in the early and late stages and the ability to escape from the local optimum by broking the symmetry of iterative routes. Finally, a combination of linearly decreasing and nonlinear segmentation convergence factors is proposed to balance the local and global search capabilities of the algorithm. Nine benchmark functions are selected for the simulation, and after comparing with other algorithms, the results show that the convergence speed and solution accuracy of the proposed algorithm are promising in this study.

Keywords: whale optimization algorithm (WOA); logistic chaos; quasi-reflection-based learning; Fish Aggregating Devices (FADs); chaotic mapping

check for updates

Citation: Su, S.; He, C.; Xu, L. Quasi-Reflective Chaotic Mutant Whale Swarm Optimization Fused with Operators of Fish Aggregating Device. *Symmetry* **2022**, *14*, 829. https://doi.org/10.3390/sym14040829

Academic Editors: Deming Lei and Ming Li

Received: 7 March 2022
Accepted: 13 April 2022
Published: 15 April 2022

1. Introduction

The whale optimization algorithm (WOA) is a newly proposed swarm intelligence optimization algorithm, developed in recent years. Due to its simple algorithm structure, few operators required for operation, fast convergence of iterations, and easy implementation, it has received wide attention and applications in various disciplines, such as power systems [1], data mining [2], training artificial neural networks (ANN) [3], network optimization [4] robot path planning [5], aero-engine optimization [6], and construction engineering [7]. According to existing studies, WOA has the advantage of strong global search capability, but occasionally falls into local optimality and cannot perform global searches comprehensively [8].

To address the above drawbacks and to improve the performance of the algorithm, on the one hand, domestic and foreign researchers have proposed many relevant improved WOA algorithms by using many strategies, such as changing operators [9–12], fusing chaotic mappings [13–15], and mixing other algorithms with the WOA algorithm [16–19]. For example, a Cauchy mutator was introduced into WOA to vary the individual movement step of whales through the Cauchy inverse cumulative distribution function method [9]; the combination of lens imaging backward learning and optimal worst backward learning strategies were used to improve the quality of swarm individuals [10]. Both the convergence factor and inertia weights were modified by introducing nonlinear strategies to balance the ability of the algorithm between the global exploration and local search, and accelerate the convergence speed of the algorithm [11]. Through differential evolution and

Levy mutation, the algorithm was expanded the range of each search and enhanced the global search capability [12]. On the other hand, by introducing Iterative mapping and nonlinear fitting into the algorithm, the initial diversity of the swarm is ensured, while balancing the global search ability and local search of the algorithm to some extent [13]. By introducing Gauss chaos mapping [14] and Singer chaos mapping [15] into the algorithm, it is further demonstrated that chaos mapping has a good improvement effect on the WOA algorithm. In addition, mixing other algorithms is a common strategy in WOA improvement, and combining local search strategies, it can effectively reduce the occurrence of WOA algorithms falling into local optimum situations. For example, by mixing with algorithms, such as slime mould algorithm (SMA) [16], social group optimization (SGO) [17], teaching-learning-based optimization (TLBO) [18], particle swarm optimization (PSO) [19,20], bat algorithm (BA) [21], and grey wolf optimizer (GWO) [22], it not only reduces the occurrence of falling into local optima, but also solves the problems of insufficient search capability and low efficiency of WOA when high-dimensional problems exist, which is of high use for the performance improvement of WOA algorithms.

The data of the above-mentioned improved WOA algorithm show that, although the performance of some of the test functions has been improved, the computational power of some of the test functions decreases too, and there are still problems, such as easily falling into local optimum and not finding the optimal solution. Therefore, based on previous research, we propose a quasi-reflective chaotic mutant whale swarm optimization fused with FADs (QNWOA) to further improve the convergence speed and optimization accuracy of the algorithm.

2. Whale Optimization Algorithm

The Whale Optimization Algorithm (WOA) is a new heuristic optimization algorithm that mimics the hunting behavior of humpback whales and is, thus, proposed. It is easy to implement and robust compared to other swarm intelligence algorithms, and the algorithm requires fewer control parameters. In nature, humpback whales have a special hunting method, and this hunting method is called the bubble-net hunting strategy. In this strategy, the humpback whale dives tens of meters and then creates a spiral bubble around its prey to gather them before swimming to the surface to feed. Interestingly, the symmetrical route of whale hunting forms a spiral, as shown in Figure 1, and the original whale optimization algorithm was proposed based on the following text.

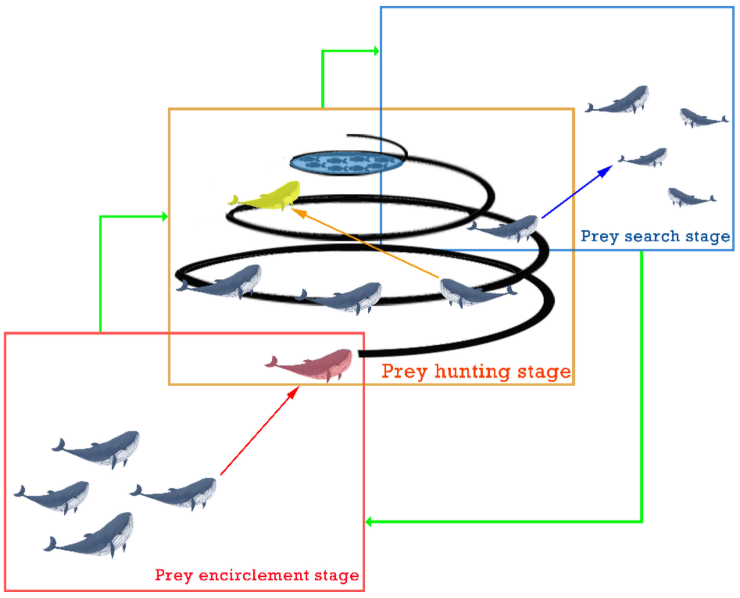


Figure 1. WOA three stages and conversion schematic.

(1) Prey encirclement stage

In this stage, the position of each whale is considered as a solution, and the whale closest to the prey is considered as the optimal solution. The other whale individuals complete an "encirclement", such as the movement towards the prey based on the position of the best individual in the current iteration. In other words, the whales do not currently know the exact location of the prey and simply move in the direction of the current optimal whale, thus, surrounding the prey. To describe this behavior, an equation is written below.

$$D = |CX(\mathrm{t}) - X(t)| \tag{1}$$

$$X(t+1) = X(\mathrm{t}) - AD \tag{2}$$

where t indicates the current number of iterations; D represents the step size between the current individual and the optimal individual; A and C are coefficients; X (t) is the position vector of the optimal whale in the current iteration; X (t) indicates the current position vector of the whale to be moved; A and C can be obtained from the following equations:

$$A = 2ar_1 - a \tag{3}$$

$$C = 2\mathrm{r}_2 \tag{4}$$

$$a = 2 - \frac{2t}{T} \tag{5}$$

where r_1 and r_2 are random numbers between (0, 1); the value of a decreases linearly from 0 to 2; t denotes the number of current iterations; T denotes the maximum number of iterations.

(2) Prey hunting stage

During this stage, the whale dives into the sea by swimming upward in a spiral position and spitting out bubbles of varying sizes. The bubbles rise to sea level and form a web of bubbles to gather the prey together. In the algorithm, the distance between the individual and the current optimal whale position is firstly calculated, and secondly, the whale is simulated to swim along a spiral trajectory towards the prey position. Based on the behavior of humpback whales, the equation for their spiral swim towards prey is as follows:

$$X(t+1) = X(t) + D_p e^{bl} \cos(2\pi l) \tag{6}$$

where D_P = ∣ X (t) − $X(t)$∣ denotes the distance vector from the individual whale to the current best whale; b is a constant used to define the shape of the helix, usually taking the value 1, and l is a random number between $(-1, 1)$.

As the whale swims in a spiral towards its prey, it also needs to shrink its encirclement. Therefore, to describe this model of synchronous behavior, a random number P between (0, 1) is introduced. Assume that there is a probability of P to choose the contracting encirclement mechanism and a probability of $1 - P$ to choose spiral predation to update the whale's position with the following equation:

$$X(t+1) = \begin{cases} X(t) - AD, \mathrm{r} < P, & (7) \\ X(t) + D_p e^{bl} \cos(2\pi l), \mathrm{r} \geq P & (8) \end{cases}$$

In the hunting stage, the algorithm sets the value of a to decrease linearly as it approaches the prey, so that the fluctuation of A also decreases with a. During the iterative process, the value of a decreases linearly from 2 to 0, A is a random value within $[-a, a]$, and the algorithm sets that when ∣A∣ ≤ 1, the whale attacks the prey, i.e., enters the prey encirclement stage and prey hunting stage.

(3) Prey search stage

In this stage, to enhance the diversity of the swarm, the algorithm randomly selects an individual whale in the current iteration and updates the positions of other whales based on the randomly selected whale position, forcing the whales to deviate from the prey and, thus, find a more suitable prey; at the same time, the search capability of the algorithm is enhanced to enable the WOA algorithm to perform a global search. The algorithm sets the whale to perform the prey search stage when $|A| \geq 1$, which is formulated as follows:

$$D = |CX_{rand} - X(t)| \tag{9}$$

$$X(t+1) = X_{rand} - AD \tag{10}$$

where X_{rand} is a randomly selected whale position vector.

3. Improved Whale Optimization Algorithm

3.1. Quasi-Reflection-Based Learning

Ewees et al. [23] proposed a new concept of quasi-reflective-based learning (QRBL) based on opposition-based learning (OBL) and quasi-opposition-based learning (QOBL). The main idea is to determine the location of the quasi-reflection solution by comparing the current solution with the middle value of the upper and lower bounds of the algorithm, and it is known through experiments that QRBL is closer to the optimal solution compared to OBL and QOBL [23] and the equation to generate the quasi-reflective solution $X^{qr}(t)$ is as follows:

$$M = \frac{(lb + ub)}{2} \tag{11}$$

$$X^{\mathrm{qr}}(t) = \begin{cases} M + (X(t) - M) \times rand(0,1); X(t) > M & (12) \\ X(t) + (M - X(t)) \times rand(0,1); X(t) < M & (13) \end{cases}$$

3.2. Logistic Chaos Mapping

Chaotic mapping is a nonlinear theory with the characteristics of non-linearity, universality, traversal, and randomness, which can traverse all states in a certain range without repetition, according to its characteristics, and can help generate new solutions and increase swarm diversity in intelligent algorithm optimization; thus, it is widely used. Although WOA has a good convergence speed, there is still room for improvement in the optimization-seeking capability. Therefore, to improve the performance of the algorithm, this paper introduces logistic chaotic mapping in the WOA algorithm, inspired by the chaotic Jaya algorithm of Jian et al. [24]. Its equation is given as:

$$X(t+1) = \beta \cdot X(t) \cdot (1 - X(t)) \tag{14}$$

It has been shown that the chaotic sequence has better pseudo-random properties when $\beta \in [3.57, 4]$.

3.3. Introduction of Mutation Operations for Vortex Effects in FADs

To enhance the ability of the algorithm to escape from the local optimum later, the vortex effect of FADs in the marine predator algorithm (MPA) [25] is introduced into the search stage of the algorithm. In the MPA algorithm, iterations in the later stage cause individuals to obtain longer steps to find another prey distribution environment, and this strategy enables MPA to overcome the early convergence problem and escape from local extremes in the process of finding the best vortex effect, known as FADs. By introducing the FADs eddy current effect and improving it, Equation (10) is changed to Equations (16) and (17), as follows:

$$CF = (1 - \frac{t}{T})^{(2 \times \frac{t}{T})} \tag{15}$$

$$X(t+1) = \begin{cases} X(t) + CF \times x_{rand1},\ rand(0,1) < \text{FADs} & (16) \\ X(t) + (\text{FADs} \times (1-r) + r) \times (x_{rand1} - x_{rand2}), rand(0,1) > \text{FADs} & (17) \end{cases}$$

where CF is the adaptive parameter controlling the movement step of the predator; FADs are the influence probability, taken as 0.2; r is the random number within [0, 1]; x_{rand1} and x_{rand2} are the randomly selected individual whale position vectors, respectively.

On this basis, to limit the FADs to perform the search blindly in the early stage, which affects the convergence of the algorithm, this paper only enables the FADs in the late stage of the algorithm iteration and strengthens the upfront capability of the algorithm by introducing the variation operation. Inspired by the wavelet variation particle swarm algorithm proposed by Su et al. [26], wavelet variation is introduced in the preliminary search stage, as follows:

$$w(\varphi) = e^{\frac{-\varphi^2}{2}} \cos(5\varphi) \tag{18}$$

Then Equation (10) changes to:

$$X(t) = \begin{cases} x_i + w(\varphi) \times (ub - X(t)), w(\varphi) > 0 & (19) \\ x_i + w(\varphi) \times (X(t) - lb), w(\varphi) \le 0 & (20) \end{cases}$$

In this paper, Morlet wavelet is chosen as the wavelet basis, w (φ) is the wavelet function value, φ is the random value within [−2.5, 2.5], and the mutation probability α is 0.1.

3.4. Convergence Factors for Linear Differential Decrement and Nonlinear Segmentation

In the original WOA algorithm, the search relies on A to switch between global and local through linear decrement. However, due to the disadvantage of the linear decreasing, the algorithm tends to fall into local optimum due to insufficient global search capability in the early stage, and it fails to find the optimal solution in the later stage due to insufficient local exploitation capability. Thus, from Equation (5), a convergence strategy that integrates linear differential decrement and nonlinear decrement is proposed, as follows:

$$a = \begin{cases} 2 - (\frac{2}{T^2}) \times t^2, t \le \frac{T}{2} & (21) \\ \mu \times t^{\lambda} + \rho, t > \frac{T}{2} & (22) \end{cases}$$

where T denotes the maximum number of iterations, t is the current generation, and μ, λ, and ρ are constants. By introducing the convergence factor of nonlinear segmentation, the algorithm maintains good global searchability in the early stage and accelerates the convergence speed in the later stage, which can effectively balance the local search and global search of the algorithm.

3.5. Algorithm Flowchart and Pseudo-Code

The flow and pseudo-code of the proposed algorithm are as shown in Figure 2 and Algorithm 1 respectively.

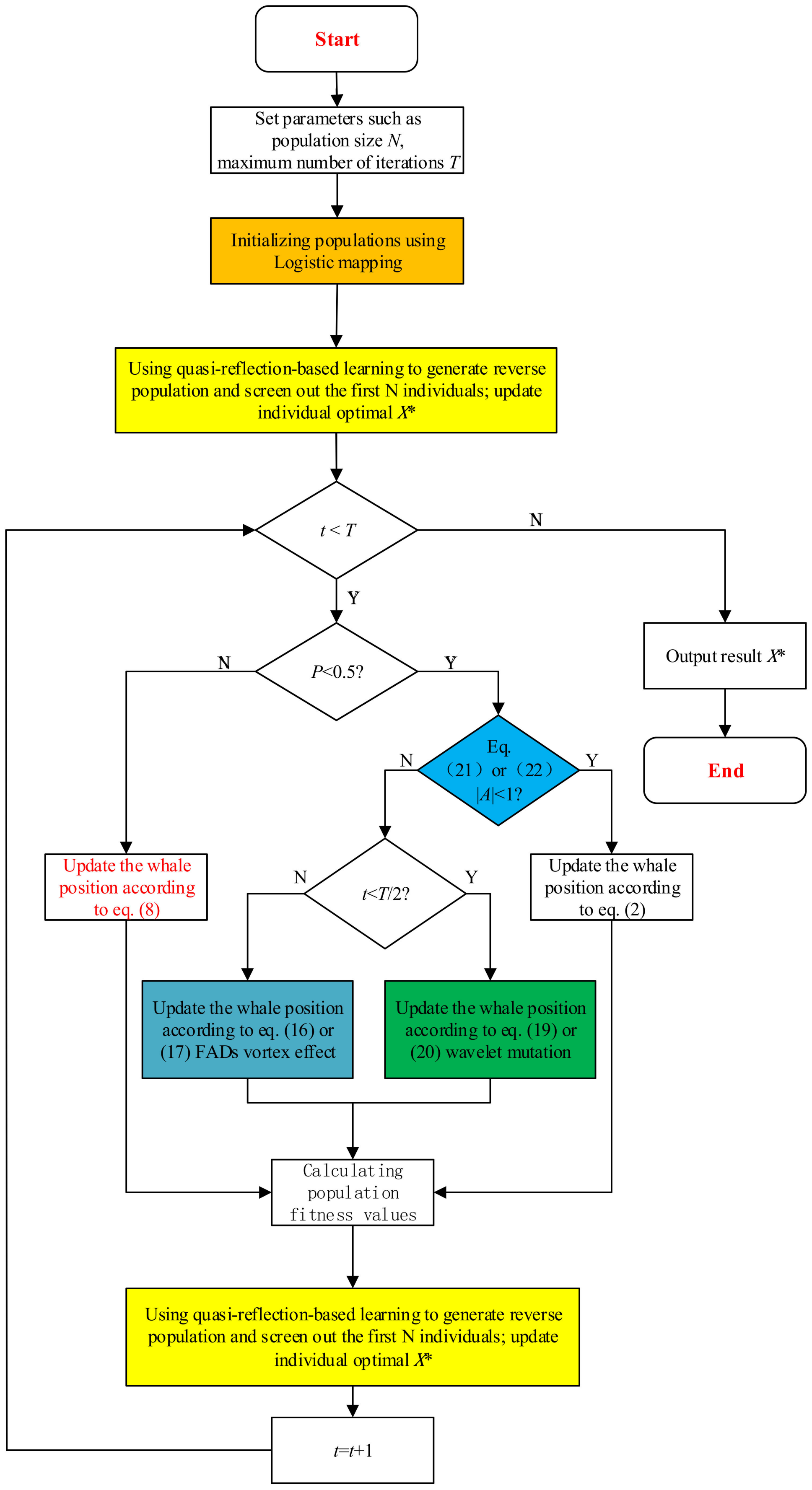

Start
Set parameters such as population size N, maximum number of iterations T
Initializing populations using Logistic mapping
Using quasi-reflection-based learning to generate reverse population and screen out the first N individuals; update individual optimal X*
t < T
N
Y
Output result X*
End
P<0.5?
N
Y
Eq. (21) or (22) |A|<1?
N
Y
Update the whale position according to eq. (8)
t<T/2?
N
Y
Update the whale position according to eq. (2)
Update the whale position according to eq. (16) or (17) FADs vortex effect
Update the whale position according to eq. (19) or (20) wavelet mutation
Calculating population fitness values
Using quasi-reflection-based learning to generate reverse population and screen out the first N individuals; update individual optimal X*
t=t+1

Figure 2. QNWOA algorithm flowchart.

Algorithm 1 the proposed algorithm (QNWOA)

```
1: Input: The swarm size $N$, maximum number of iteration $T$.
2: Output: Initialize the random swarm $X_i(i = 1, 2 \ldots, N)$;
Generating species group $N^*$ according to logistic chaos and calculate fitness
Quasi—reflex learning generates inverse swarm $N^{qr}$ and calculate fitness
The first $N$ are selected in new swarm and the optimal individual $X^*$ was updated
3: While (t < T) do
4: Update $l$, $P$, $r$, variation rate $r_1$, and influence probability FADs;
Update a, A, C by Equation (21) or (22)
5: if ($P < 0.5$)
6:  if (|A| $\geq$ 1)
7:   if (t < T/2)
8:    if (t < $r_1$)
9:     Update $X_i$ location by Equation (19) or (20).
10:     else Update $X_i$ location by Equation (10).
11:    end if
12:     else if (t $\geq$ T/2) Update $X_i$ location by Equation (16) or (17).
13:    end if
14:  else if (|A| < 1)
15:    Update $X_i$ location by Equation (2).
16:  end if
17: else if ($P \geq 0.5$)
18: Update $X_i$ location by Equation (8).
19: end if
20: Calculate the N individual fitness value of the current swarm;
21: Quasi—reflex learning generates inverse swarm $N^{qr}$ and calculate fitness.
22: The first $N$ was selected in new swarm and the optimal individual $X^*$ was updated.
23: End while
24: Return $X^*$
```

4. Experimental Results and Analysis

4.1. Experimental Design

To test the performance of the improved algorithm, the experiment selects nine benchmark functions that are more representative for testing, as shown in Table 1: F1–F3 for unimodal (UM) function; F4–F7 for multi-modal (MM) function; F8-F9 for composition (CM) function. From the 2D forms of nine benchmarks as shown in Figure 3a–i, it can be seen that the UM function has only one optimal solution and is often used to test the algorithm's ability to find the optimal solution. Since the MM function has multiple optima, it is difficult to find globally optimal solutions for these functions, but it can be used to test the algorithm's ability to explore and jump out of local optima by broking the symmetry of iterative routes. The CM function also has a large number of optimal solutions, but it can be used to test the stability of the algorithm, since the optimal solution is easily found due to the low dimensionality. The tests are done in an experimental environment of CPU Inter(R) Core (TM) i7-8700, RAM 16 GB, and MATLAB.

Table 1. Test Functions.

No.	Formula	Dim	Range	Optimal Value
F1	$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
F2	$f_2(x) = \max_i\{\lvert x_i\rvert, 1 \leq i \leq n\}$	30	$[-100, 100]$	0
F3	$f_3(x) = \sum_{i=1}^{n} x_i^4 + random(0,1)$	30	$[-1.28, 1.28]$	0

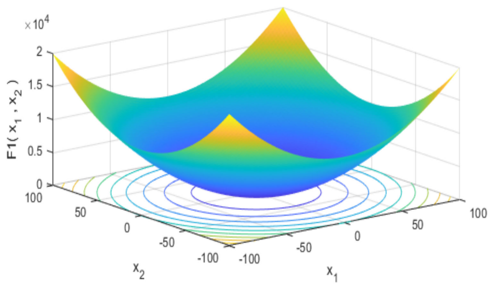
Table 1. *Cont.*

No.	Formula	Dim	Range	Optimal Value
F4	$f_4(x) = \sum_{i=1}^{n} [x_i{}^2 - 10\cos(2\pi x_i) + 10]$	30	$[-5.12, 5.12]$	0
F5	$f_5(x) = -20\exp[-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)}] + 20 + e$	30	$[-32, 32]$	0
F6	$f_6(x) = 0.1\left\{\sin(3\pi x_i) + \sum_{i=1}^{n-1}(x_i-1)^2[1+\sin(3\pi x_{i+1})] + (x_n-1)^2[1+\sin^2(2\pi x_{i+1})]\right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4) u(x, a, k, m) = \begin{cases} k(x_i-a)^m & x_i > a \\ 0 & -a \le x_i \le a \\ k(-x_i-a)^m & x_i < -a \end{cases}$	30	$[-50, 50]$	0
F7	$f_7(x) = 0.1(\sin^2(3\pi x) + \sum_{i=1}^{n-1}(x_i-1)^2 [1+\sin^2(3\pi x_i+1)] + (x_n-1)^2 (1+\sin^2(2\pi x_n)) + \sum_{i=1}^{n} u(x_i, 5, 100, 4))$	30	$[-50, 50]$	0
F8	$f_8(x) = \sum_{i=1}^{11} [a_i - \frac{x_1(b_i{}^2+b_i x_2)}{b_i{}^2+b_i x_3+x_4}]^2$	4	$[-5, 5]$	0.0003
F9	$f_9(x) = -\sum_{i=1}^{5} [(X-a_i)(X-a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.1532

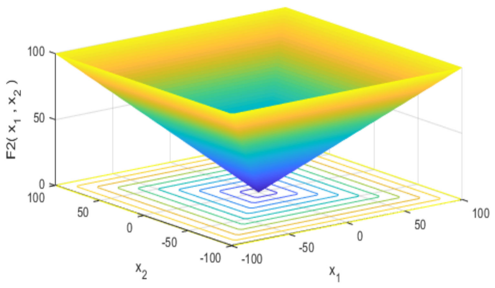
On the other hand, QNWOA is compared and experimented with the currently existing WOA algorithms and improved WOA algorithms: WOA (traditional whale optimization algorithm), OWOA (whale optimization algorithm based on reverse learning), EWOA (Whale optimization algorithm introducing nonlinear time-varying adaptive weights and differential mutation perturbation factors), WOAWC (Whale optimization algorithm based on adaptive weight and Cauchy mutation); there are also three more representative swarm intelligence algorithms: PSO (particle swarm optimization algorithm), SOA (seagull optimization algorithm), and GWO (gray wolf optimization algorithm).

4.2. Analysis of Experimental Test Results

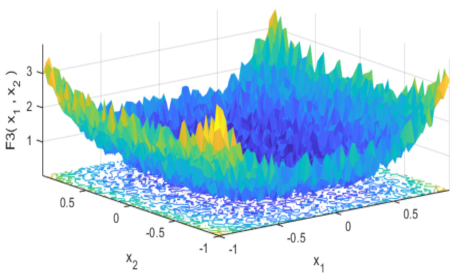
In our experimental tests, the algorithm parameters are set as shown in Table 2, and all algorithms run 30 times repeatedly on each function to record and count the optimal value (Best), mean value (Mean) and standard deviation (SD) run results, as shown in Tables 3 and 4. By analyzing the experimental statistical results, it can be seen from the data in Table 3, combining with the lefts of Figures 4–6, that both QNWOA and WOAWC converge to the minimum value in the UM function F1, but the convergence speed of QNWOA is much higher than several other algorithms, and QNWOA outperforms other algorithms, in terms of convergence speed, optimal value, mean and standard deviation in F2 and F3. Further, it can be seen from Table 4 that QNWOA outperforms several other swarm intelligence algorithms in F1–F3, in terms of the mean and standard deviation of optimal values. In summary, QNWOA in the UM function has a higher search capability compared to several other algorithms.



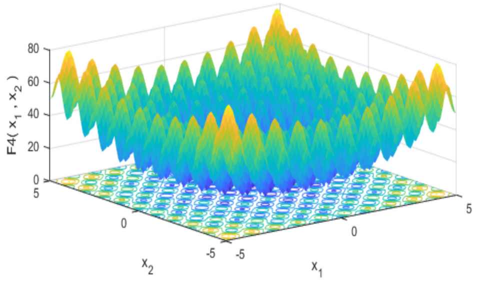
(**a**) F1



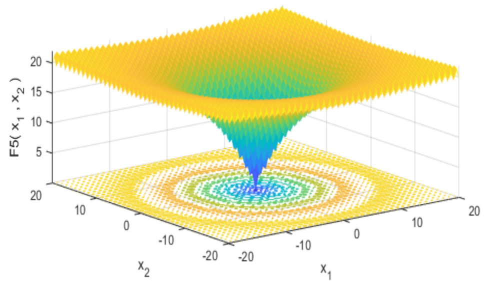
(**b**) F2



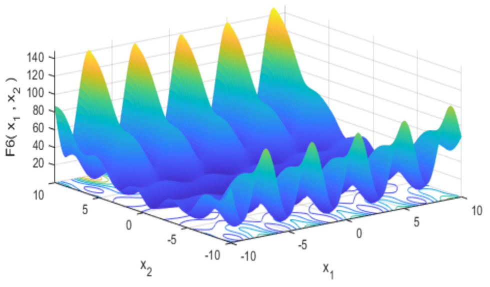
(**c**) F3



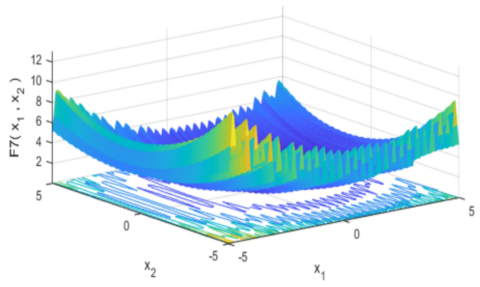
(**d**) F4



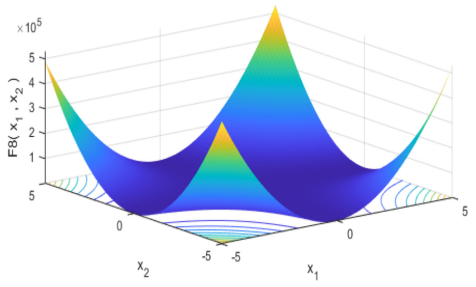
(**e**) F5



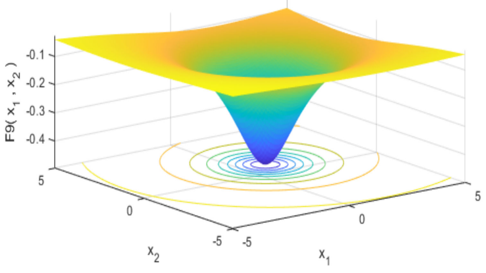
(**f**) F6



(**g**) F7



(**h**) F8



(**i**) F9

Figure 3. Demonstration of functions.

Table 2. Algorithm parameter settings.

Maximum number of iterations T	500
Number of swarms N	30
Number of runs	30
QNWOA Mutation rate α	0.1
μ	46,850,000
λ	−3.103
ρ	−0.1976
β	4

Table 3. Function experiment result 1.

No.		**Algorithm**				
		WOA	**OWOA**	**EWOA**	**WOAWC**	**QNWOA**
F1	Best	9.9137×10^{-86}	1.3067×10^{-244}	2.7715×10^{-124}	0	0
	Mean	4.6344×10^{-73}	1.4789×10^{-169}	2.1114×10^{-91}	0	0
	SD	2.5241×10^{-72}	0	1.1564×10^{-90}	0	0
F2	Best	1.7473	0.22051	2.5734×10^{-24}	3.4581×10^{-204}	0
	Mean	47.3422	76.0267	3.2752×10^{-15}	1.9033×10^{-182}	0
	SD	28.2177	17.6122	1.0942×10^{-14}	0	0
F3	Best	0.0001083	8.9531×10^{-6}	9.9806×10^{-5}	8.8107×10^{-6}	3.3573×10^{-6}
	Mean	0.0034336	0.00035576	0.0051922	0.00013845	6.9352×10^{-5}
	SD	0.0032397	0.000539	0.0057851	8.5724×10^{-5}	6.7198×10^{-5}
F4	Best	0	0	0	0	0
	Mean	0	0	15.7555	0	0
	SD	0	0	41.2958	0	0
F5	Best	8.8818×10^{-16}	8.8818×10^{-16}	8.8818×10^{-16}	8.8818×10^{-16}	8.8818×10^{-16}
	Mean	3.9672×10^{-15}	3.0198×10^{-15}	3.3751×10^{-15}	8.8818×10^{-16}	8.8818×10^{-16}
	SD	2.5945×10^{-15}	2.2079×10^{-15}	1.6559×10^{-15}	0	0
F6	Best	0.0084862	0.011174	0.061662	0.0034545	1.1773×10^{-20}
	Mean	0.036696	0.0357	0.36672	0.011456	0.0086561
	SD	0.044194	0.019212	0.28237	0.025047	0.020263
F7	Best	0.10767	0.24004	0.6611	0.071237	1.1285×10^{-18}
	Mean	0.51226	0.83157	1.3322	0.13458	0.00049852
	SD	0.30266	0.29615	0.34059	0.04074	0.0010646
F8	Best	0.00030762	0.00030827	0.00031749	0.00030871	0.00030756
	Mean	0.00063911	0.00070656	0.0019246	0.00049698	0.00033584
	SD	0.00047147	0.00044626	0.0041292	0.00019152	6.4589×10^{-5}
F9	Best	−10.1515	−10.1532	−10.1532	−10.1511	−10.1532
	Mean	−8.3563	−10.1527	−8.0397	−8.7354	−10.1532
	SD	2.5867	0.00072348	2.8988	2.2751	5.4746×10^{-5}

Table 4. Function experiment result 2.

No.		Algorithm				
		WOA	**SOA**	**PSO**	**GWO**	**QNWOA**
F1	Best	1.1605×10^{-85}	5.6494×10^{-15}	6.2393×10^{-5}	1.2139×10^{-29}	0
	Mean	5.56×10^{-73}	3.4144×10^{-12}	0.00043896	7.8984×10^{-28}	0
	SD	2.2092×10^{-72}	8.3955×10^{-12}	0.00034267	1.7267×10^{-27}	0
F2	Best	3.9921	5.4184×10^{-5}	0.2385	4.2405×10^{-8}	0
	Mean	48.4683	0.0038117	0.35914	7.0301×10^{-7}	0
	SD	28.5727	0.0057043	0.11779	6.3454×10^{-7}	0
F3	Best	5.6855×10^{-5}	0.00036337	0.067248	0.00061349	3.4638×10^{-6}
	Mean	0.0028777	0.003037	0.11449	0.0017964	7.3138×10^{-5}
	SD	0.0028102	0.0022382	0.042168	0.00086993	8.1288×10^{-5}
F4	Best	0	3.979×10^{-13}	16.9228	0	0
	Mean	0	1.0798	47.642	3.0001	0
	SD	0	3.3776	13.9832	4.178	0
F5	Best	8.8818×10^{-16}	19.9581	0.0049207	6.839×10^{-14}	8.8818×10^{-16}
	Mean	3.9672×10^{-15}	19.9609	0.012974	1.0167×10^{-13}	8.8818×10^{-16}
	SD	2.5945×10^{-15}	0.0012602	0.0056487	1.8079×10^{-14}	0
F6	Best	0.0072029	0.15078	1.4541×10^{-5}	0.019941	5.1833×10^{-24}
	Mean	0.020625	0.31326	0.65821	0.043437	0.0045909
	SD	0.012054	0.14127	0.14127	0.017322	0.012509
F7	Best	0.17038	1.7962	1.0661×10^{-5}	0.10076	4.9255×10^{-18}
	Mean	0.5566	2.0746	0.0019401	0.62787	0.0010112
	SD	0.19806	0.16968	0.004153	0.24273	0.0044074
F8	Best	0.00030808	0.00031441	0.00030749	0.0003075	0.00030755
	Mean	0.00069231	0.0011808	0.0029492	0.0038868	0.00031558
	SD	0.00051019	0.00020275	0.010759	0.007502	8.4307×10^{-6}
F9	Best	−10.152	−10.1283	−10.1532	−10.1531	−10.1532
	Mean	−8.202	−4.0799	−4.8943	−8.9708	−10.1531
	SD	2.8621	4.6169	3.0995	2.1763	8.9997×10^{-5}

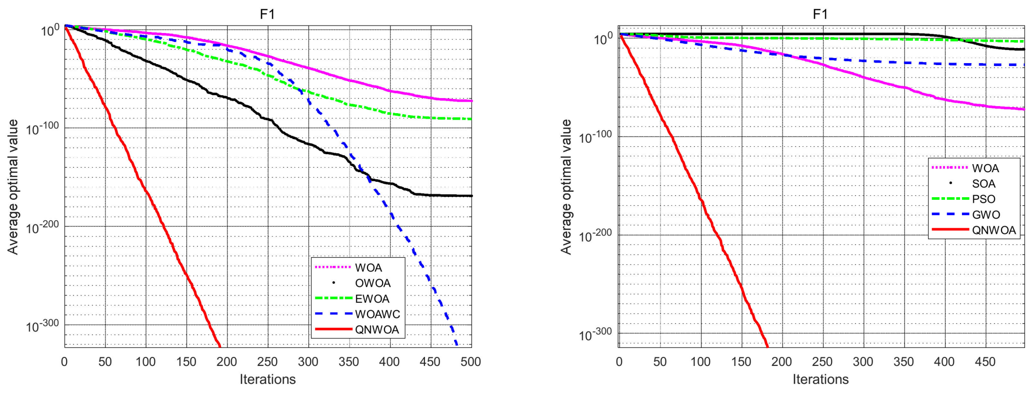


Figure 4. F1 convergence comparison chart.

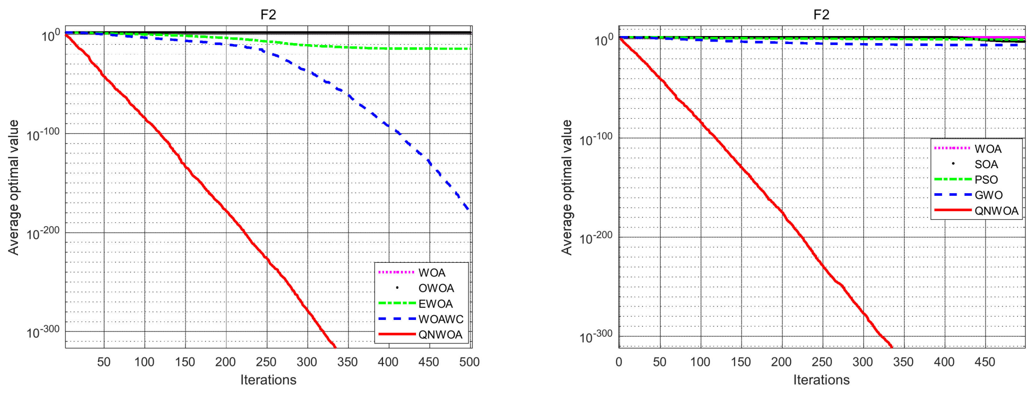


Figure 5. F2 convergence comparison chart.

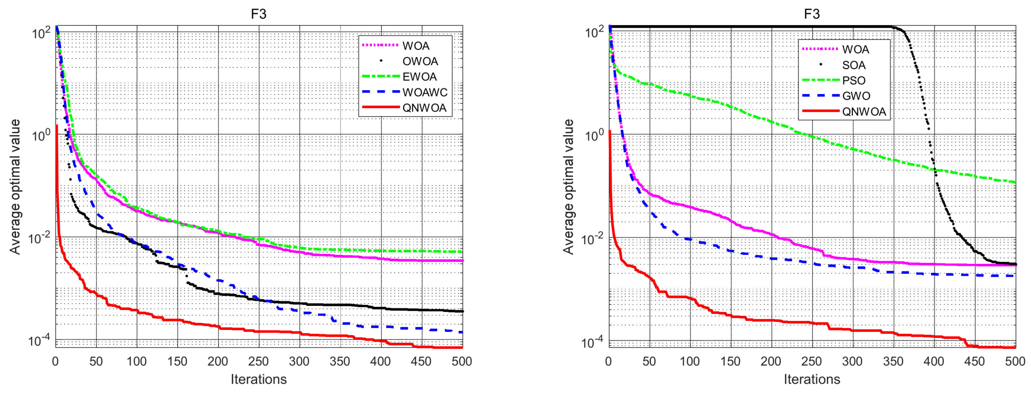


Figure 6. F3 convergence comparison chart.

From the data in Table 3, combined with Figures 7–9, we can see that QNWOA and WOAWC in the CM functions, F4 and F5, have significantly better optimal values, mean values, and standard deviations compared with other improved WOA algorithms, while QNWOA is much faster than WOAWC in terms of convergence speed. The improvement in the optimal value is especially large. From Table 4, it can be seen that the QNWOA proposed in this paper outperforms the other three group intelligence algorithms compared in all data, especially in terms of optimal values. It proves that the improved QNWOA effectively avoids the local optimum in the CM function and has a better global capability.

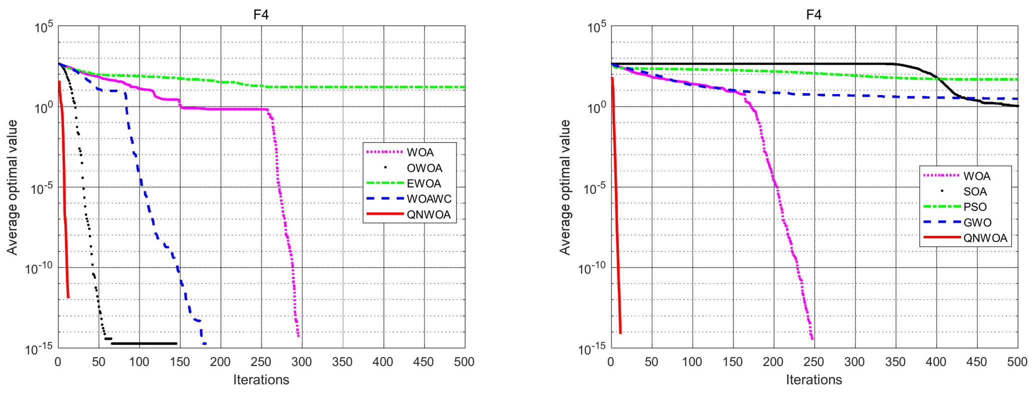


Figure 7. F4 convergence comparison chart.

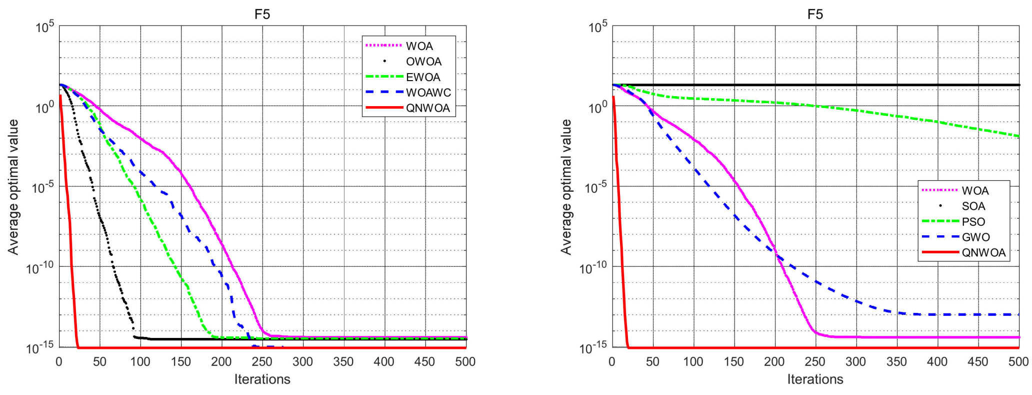


Figure 8. F5 convergence comparison chart.

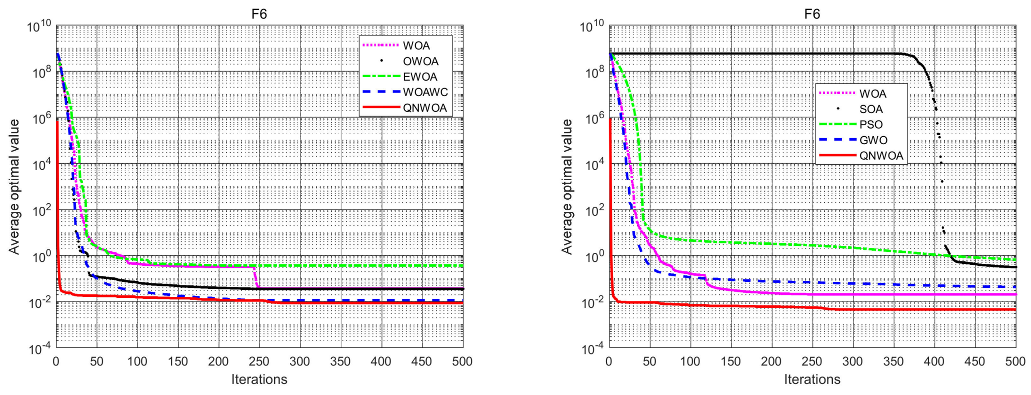


Figure 9. F6 convergence comparison chart.

From the data in Tables 3 and 4, combined with Figures 10–12, we can see that QNWOA in CM functions F7 and F8 has a certain degree of improvement in the optimal value and mean value, compared with the other six algorithms, and the standard deviation has significantly reduced. It has shown that the improved algorithm not only performs better, but also has better stability in the CM functions.

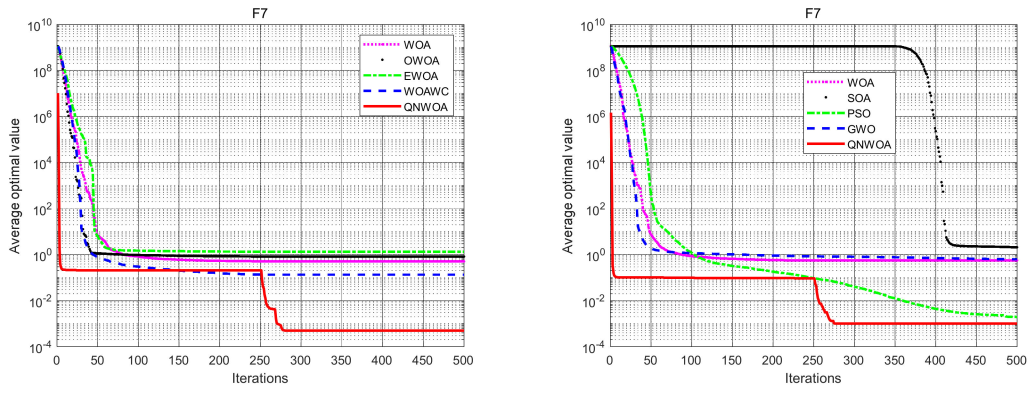


Figure 10. F7 convergence comparison chart.

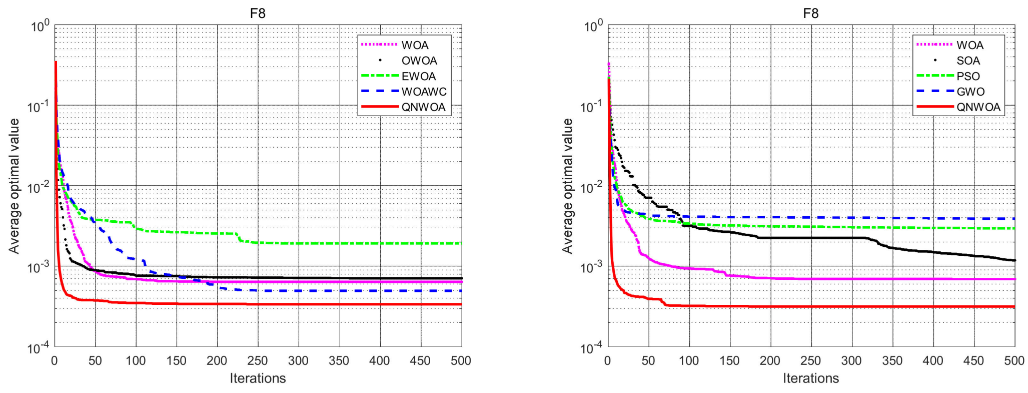


Figure 11. F8 convergence comparison chart.

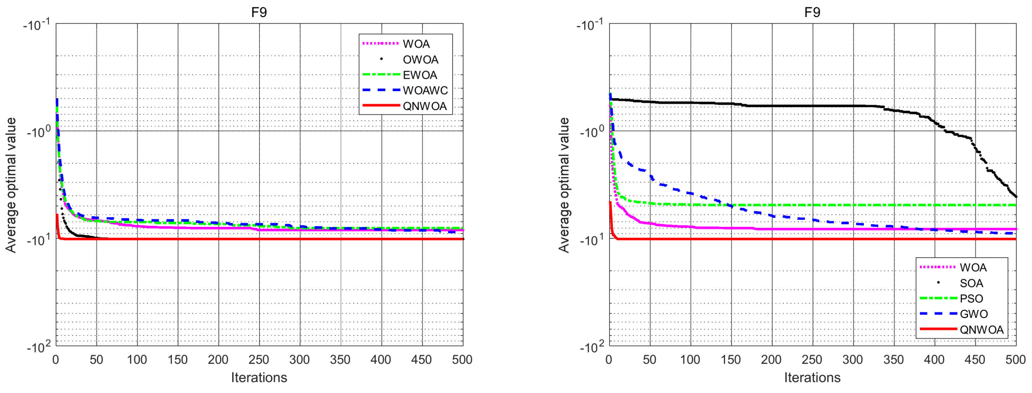


Figure 12. F9 convergence comparison chart.

In general, the improved QNWOA in this paper has significantly improved the convergence speed, convergence accuracy, and stability compared with other algorithms.

5. Conclusions

In this paper, by introducing logistic chaotic mapping and a quasi-reflection-based learning mechanism, we proposed a whale swarm optimization algorithm, namely QNWOA, that fused with the FADs vortex effect and wavelet mutation of marine predator algorithm (MPA) in the search phase. In the algorithm, a nonlinear segmentation convergence factor is used to effectively improve the convergence speed and the search accuracy of the WOA algorithm, according to the experimental results. However, the proposed algorithm fluctuates a lot when the FADs vortex effect is enabled, and the next work should consider how to efficiently coordinate the FADs vortex effect for iterations without affecting the convergence of the algorithm, which require further study in the future.

Author Contributions: Methodology, S.S. and C.H.; project administration, S.S.; supervision, S.S.; investigation, C.H.; software, C.H. and L.X.; investigation, L.X.; software, C.H. and L.X.; writing—review and editing, S.S. and L.X. All authors have read and agreed to the published version of the manuscript.

Funding: The work was partially supported by the Scientific Research Foundations and the Virtual Experimental Class Projects of Jingling Institute of Technology (nos. JIT-rcyj-201505 and D2020005), the National Natural Science Foundation of China (nos. 61375121 and 41801303), the Major Project of Natural Scientific Research for Universities in Jiangsu Province (no.21KJA120001), and sponsored by the funds from Jiangsu Provincial Sci-Tech Innovation Team of Swarm Computing and Smart Software, led by S.S. (corresponding author).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kumar, R.; Singha, R.; Ashfaqa, H.; Singh, S.K.; Badoni, M. Power system stability enhancement by damping and control of Sub-synchronous torsional oscillations using Whale optimization algorithm based Type-2 wind turbines. *ISA Trans.* **2021**, *108*, 240–256. [CrossRef] [PubMed]
2. Mafarja, M.M.; Mirjalili, S. Hybrid whale optimization algorithm with simulated annealing for feature selection. *Neurocomputing* **2017**, *260*, 302–312. [CrossRef]
3. Alameer Zakaria Mohamed, A.E.; Ewees, A.A.; Ye, H.; Jianhua, Z. Forecasting gold price fluctuations using improved multilayer perceptron neural network and whale optimization algorithm. *Resour. Policy* **2019**, *61*, 250–260. [CrossRef]
4. Deepa, R.; Revathi, V. Enhancing Whale Optimization Algorithm with Levy Flight for coverage optimization in wireless sensor networks. *Comput. Electr. Eng.* **2021**, *94*, 107359. [CrossRef]
5. Petrović, M.Z.; Miljković, Z.; Jokić, A. A novel methodology for optimal single mobile robot scheduling using whale optimization algorithm. *Appl. Soft Comput.* **2019**, *81*, 105520. [CrossRef]
6. Huang, X.; Wang, R.; Zhao, X.; Hu, K. Aero-engine performance optimization based on whale optimization algorithm. In Proceedings of the 2017 36th Chinese Control Conference (CCC), Dalian, China, 26–28 July 2017; pp. 11437–11441.
7. Aala, K.; Vamsi, K.R.; Venkata, L.N. A combinatorial social group whale optimization algorithm for numerical and engineering optimization problems. *Appl. Soft Comput.* **2021**, *99*, 106903. [CrossRef]
8. Zhou, Y.; Ling, Y.; Luo, Q. Lévy Flight Trajectory-Based Whale Optimization Algorithm for Global Optimization. *IEEE Access* **2017**, *5*, 6168–6186.
9. Guo, Z.Z.; Wang, P.; Ma, Y.F.; Qi, W.; Chang-qing, G. Whale algorithm based on adaptive weight and Cauchy mutation. *Microelectron. Comput.* **2017**, *34*, 20–24.
10. Chakraborty, S.; Sharma, S.; Saha, A.K.; Chakraborty, S. SHADE–WOA: A metaheuristic algorithm for global optimization. *Appl. Soft Comput.* **2021**, *113*, 107866. [CrossRef]
11. Xu, H.; Zhang, M.D.; Wang, Y.R.; Song, T.T.; Fan, Y. Hybrid strategy to improve whale optimization algorithm. *Comput. Eng. Des.* **2020**, *41*, 12–19.
12. Liu, M.; Yao, X.; Li, Y. Hybrid whale optimization algorithm enhanced with Lévy flight and differential evolution for job shop scheduling problems. *Appl. Soft Comput.* **2020**, *87*, 105954. [CrossRef]
13. Li, S.Y.; Ju, C.X.; Ding, H.Q. Whale optimization algorithm based on iterative mapping and nonlinear fitting. *J. Chongqing Univ.* 2021. Available online: https://kns.cnki.net/kcms/detail/50.1044.n.20210812.1732.002.html (accessed on 10 January 2022).
14. Zhao, Y.; Peng, Z.R. Whale Optimization Algorithm with Chaotic Maps and Its Application in Finite Element Model Updating. *J. Lanzhou Jiao Tong Univ.* **2021**, *40*, 39–46. (In Chinese)
15. Oliva, D.; El Aziz, M.A.; Hassanien, A.E. Parameter estimation of photovoltaic cells using an improved chaotic whale optimization algorithm. *Appl. Energy* **2017**, *200*, 141–154. [CrossRef]
16. Abdel-Basset, M.; Chang, V.; Mohamed, R. HSMA_WOA: A hybrid novel Slime mould algorithm with whale optimization algorithm for tackling the image segmentation problem of chest X-ray images. *Appl. Soft Comput.* **2020**, *95*, 106642. [CrossRef]
17. Su, S.B.; Zhao, W.; Wang, C.S. Parallel swarm intelligent motion planning with energy-balanced for multirobot in obstacles environment. *Wirel Commun. Mob. Com.* **2021**, *32*, 8902328.
18. Vijaya Lakshmi, A.; Mohanaiah, P. WOA-TLBO: Whale optimization algorithm with Teaching- learning-based optimization for global optimization and facial emotion recognition. *Appl. Soft Comput.* **2021**, *110*, 107623. [CrossRef]
19. Ul Hassan, N.; Bangyal, W.H.; Ali Khan, M.S.; Nisar, K.; Ag. Ibrahim, A.A.; Rawat, D.B. Improved Opposition-Based Particle Swarm Optimization Algorithm for Global Optimization. *Symmetry* **2021**, *13*, 2280. [CrossRef]
20. Pervaiz, S.; Bangyal, W.H.; Ashraf, A.; Kashif, N.; Reazul, H.M.; Ag. Asri, A.I.; Chowdhry, B.S.; Waqas, R.; Joel, J.P.C.R.; Richard, E.; et al. Comparative research directions of population initialization techniques using pso algorithm. *Intell. Autom. Soft Comput.* **2022**, *32*, 1427–1444. [CrossRef]
21. Bangyal, W.H.; Hameed, A.; Ahmad, J.; Nisar, K.; Haque, M.R.; Asri, A.; Ibrahim, A.; Rodrigues, J.J.P.C.; Khan, M.A.; Rawat, D.B.; et al. New Modified Controlled Bat Algorithm for Numerical Optimization Problem. *Comput. Mater. Contin.* **2021**, *70*, 2241–2259. [CrossRef]
22. Raja, M.; Dhanasekaran, S.; Vasudevan, V. Opposition Based Joint Grey Wolf-Whale Optimization Algorithm Based Attribute Based Encryption in Secure Wireless Communication. *Wirel. Pers. Commun.* **2021**, *121*, 1–21. [CrossRef]
23. Ewees, A.A.; Abd Elaziz, M.; Houssein, E.H. Improved Grasshopper Optimization Algorithm using Opposition-based Learning. *Expert Syst. Appl.* **2018**, *112*, 156–172. [CrossRef]
24. Jian, X.Z.; Weng, Z.Y. A logistic chaotic JAYA algorithm for parameters identification of photovoltaic cell and module models. *Optik* **2020**, *203*, 164041. [CrossRef]
25. Faramarzi, A.; Heidarinejad, M.; Mirjalili, S.; Gandomi, A.H. Marine Predators Algorithm: A nature-inspired meta heuristic. *Expert Syst. Appl.* **2020**, *152*, 113377. [CrossRef]
26. Su, S.B.; Guo, H.F.; Tian, H.M.; Wu, G. A novel pattern clustering algorithm based on particle swarm optimization joint adaptive wavelet neural network model. *Mobile Netw. Appl.* **2017**, *22*, 692–701. [CrossRef]