

Article

Improved Whale Optimization Algorithm Based on Fusion Gravity Balance

Chengtian Ouyang ¹, Yongkang Gong ¹, Donglin Zhu ²  and Changjun Zhou ^{2,*}¹ School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou 341000, China² College of Mathematics and Computer Science, Zhejiang Normal University, Jinhua 321000, China

* Correspondence: zhouchangjun@zjnu.edu.cn

Abstract: In order to improve the shortcomings of the whale optimization algorithm (WOA) in dealing with optimization problems, and further improve the accuracy and stability of the WOA, we propose an enhanced regenerative whale optimization algorithm based on gravity balance (GWOA). In the initial stage, the nonlinear time-varying factor and inertia weight strategy are introduced to change the foraging trajectory and exploration range, which improves the search efficiency and diversity. In the random walk stage and the encircling stage, the excellent solutions are protected by the gravitational balance strategy to ensure the high quality of solution. In order to prevent the algorithm from rapidly converging to the local extreme value and failing to jump out, a regeneration mechanism is introduced to help the whale population escape from the local optimal value, and to help the whale population find a better solution within the search interval through reasonable position updating. Compared with six algorithms on 16 benchmark functions, the contribution values of each strategy and Wilcoxon rank sum test show that GWOA performs well in 30-dimensional and 100-dimensional test functions and in practical applications. In general, GWOA has better optimization ability. In each algorithm contribution experiment, compared with the WOA, the indexes of the strategies added in each stage were improved. Finally, GWOA is applied to robot path planning and three classical engineering problems, and the stability and applicability of GWOA are verified.

Keywords: nonlinear time-varying factor; inertia weight; gravity balance; regeneration mechanism

Citation: Ouyang, C.; Gong, Y.; Zhu, D.; Zhou, C. Improved Whale Optimization Algorithm Based on Fusion Gravity Balance. *Axioms* **2023**, *12*, 664. <https://doi.org/10.3390/axioms12070664>

Academic Editors: Rodrigo Olivares and Ricardo Soto

Received: 17 May 2023

Revised: 27 June 2023

Accepted: 28 June 2023

Published: 4 July 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Optimization problems can be found in industrial manufacturing [1], engineering optimization [2,3], image processing [4], path planning [5,6], aerospace [7], medical [8], construction project risk aversion [9] and other research fields. However, with the emergence of more complex optimization problems, traditional optimization schemes such as the gradient descent method and the Newton method cannot achieve high efficiency standards in terms of time cost and solution accuracy. Therefore, through exploration and research, researchers have proposed a number of excellent swarm intelligence optimization algorithms according to the habits of natural organisms, which provides the research direction for the majority of scholars. A novel Particle Swarm Optimization (PSO) algorithm was proposed by Eberhart R et al. [10] in 2002. In 1991, Dorigo M designed and proposed the Ant colony Optimizing (ACO) algorithm, which was inspired by the pheromones produced during ant communication [11]. In 2014, Seyedali Mirjalili et al. [12] proposed a Grey Wolf Optimizer (GWO), with reasonable logic and simple parameters inspired by the hunting method of wolves. Fathollahi-Fard et al. [13] developed a simple and intelligent single-solution Social Engineering Optimizer (SEO) in 2018. Xue et al. [14] proposed a Sparrow Search Algorithm (SSA) with simple parameters and population classification based on the characteristics of the division of labor of sparrows seeking food in 2020. Compared with traditional optimization methods, swarm intelligence algorithms have advantages such as simple

parameters, clear logic and unique mechanisms, which have been explored and studied by many scholars.

Seyedali Mirjalili et al. [15] designed and proposed a novel metaheuristic optimization algorithm named Whale Optimization Algorithm (WOA) by observing the behavior of humpback whales in three stages. This algorithm has clear logic, simple structure, fast convergence and strong global search and local search performance. According to the “no free lunch” theory [16], an algorithm can not achieve good results in all optimization problems. At present, the WOA has gradually shown its advantages in solving many complex optimization problems. However, the WOA is easily affected by the local optimal value in the iterative process and cannot jump out of the local extreme value. In the optimization process, due to the fast convergence speed and low solving accuracy, the WOA cannot achieve satisfactory results in the face of many complex optimization problems.

In order to improve the shortcomings of the WOA mentioned above, many scholars have conducted researches on WOA variants to make them have a stronger ability to solve optimization problems. Xiong et al. [17] put forward two new prey search strategies, which effectively balanced local search and global exploration, better solved the problems of stagnation and premature convergence and obtained obvious advantages in solving the parameter extraction problem of photovoltaic models. Li et al. [18] introduced nonlinear adjustment parameters and Gaussian perturbation operators, and achieved good results in both convergence rate and optimization accuracy, as well as good application results in the location problem of the critical sliding surface of a soil slope. Shahraki et al. [19] proposed a unique pooling mechanism and combined it with three effective search strategies to propose a binary improved whale optimization algorithm, which effectively solved the problem of COVID-19 medical detection. Yang et al. [20] introduced adaptive nonlinear inertial weights into the WOA to control the convergence rate, and adopted the finite mutation mechanism to improve the convergence efficiency, showing superior performance in economic load scheduling problems. Shen et al. [21] showed that, combined with the multi-population evolution strategy, WOAs have a unique global search ability and excellent ability to jump out of local optima, and have excellent performance in various engineering design problems. Guo et al. [22] proposed that the wavelet mutation strategy and social learning principle have greatly improved the performance of global searching and jumping out of local optima, and achieved good application results in water resource demand prediction. Oliva et al. [23] used chaotic mapping to greatly improve the ability to search for optimal solutions, and achieved excellent results in parameter estimation of solar cells. Ning et al. [24] added the Gaussian variational method and the non-fixed penalty function method to the initial population and convergence factors, which played a prominent role in global search ability, stability, convergence rate, convergence accuracy and robustness, and also had a good effect on more complex constrained optimization problems. Zhang et al. [25] improved the algorithm performance to a great extent by using nonlinear adaptive weights and gold sine operators, which has fast global convergence and can avoid falling into local optimality. Yang et al. [26] adopted a multi-strategy approach integrating chaotic mapping, dynamic convergence factors, Levi flight mechanisms and evolutionary population dynamics mechanisms, which has significant advantages and effectiveness in dealing with global optimization problems. Li et al. [27] used chaotic mapping and tournament selection strategies to enrich the diversity and randomness of the population in the initialization stage, which made it difficult to fall into local optimality and significantly improved the accuracy of the algorithm. Jiang et al. [28] used scheduling rules, nonlinear convergence factors and mutation operations to effectively improve the quality of the initial solution, balance the exploration and development ability of the algorithm and avoid stagnation, which has been proven to effectively solve the energy-saving scheduling problem. Although the above scholars put forward relatively complete and constructive attempts, a larger search scope was not realized in the exploration stage. In the local search stage, there is still insufficient exploration of the surrounding area, which leads to insufficient precision or local optima. In addition, there was still a lack of a protection

mechanism for better solutions in the random walk stage, resulting in insufficient solving accuracy and slow convergence rate. To this end, this study makes a new exploration and attempts to solve the above problems by proposing an enhanced regenerative whale optimization algorithm based on gravity balance (GWOA). The main work is as follows:

- Using improved nonlinear time-varying factors and inertia weights;
- By combining the idea of gravitation with the random walk strategy and shrink and surround strategy in the whale optimization algorithm, a new position update model is proposed;
- Designing a novel strategy for rebirth;
- Testing the improved algorithm on the benchmark test function, cec2013, robot path planning and three classical engineering optimization problems, and finally analyzing related experiments.

The article has a clear logical structure. Section 1 introduces the research background in the field of optimization algorithm of recent years. Section 2 briefly introduces the details of the WOA. Section 3 introduces the improved strategy according to the search characteristics of different stages, and analyzes the time complexity of GWOA. Section 4 verifies the feasibility of the GWOA, which combines search strategies of different stages, and tests the GWOA and six other algorithms on 16 benchmark test functions. The experimental results are recorded in a table, and the experimental data are compared and analyzed to verify the overall performance of the GWOA. Then, a contribution value experiment is carried out to verify the contribution value of related strategies. Finally, the uniqueness of the GWOA is verified with the Wilcoxon rank sum test. Section 5 applies the GWOA to robot path planning and three classical engineering optimization problems to further verify the feasibility and effectiveness of the improved algorithm. Section 6 analyzes and summarizes the work of this paper, and makes a brief plan and outlook for the next step.

2. Whale Optimization Algorithm

A paper by Seyedali Mirjalili et al [15]. proposed an intelligent optimization algorithm with clear logic and a complete structure after studying the complete predation behavior of humpback whales. The predation behavior of the WOA mainly includes the following three aspects: random walk search, that is, whales randomly search for prey in the surrounding environment in the ocean; surrounding prey stage, where whales spiral around their prey from bottom to top; and bubble net predation, where whales engage in bubble net hunting to kill food as they spiral up and surround the prey.

2.1. Random Walk Search

In the optimization process of the WOA algorithm, each whale carries out different position updates according to different stages, and seeks prey through position changes in the optimization process; the prey position is the optimal location. At this stage, the purpose of the whale is relatively random and the exact location of the prey is not clear, so it is necessary to conduct a large-scale search in the search space, and slowly determine the population of whales to determine the final prey location. The location update formula of the random walk stage of the WOA algorithm is controlled by parameter A . When $A > 1$ and $p \leq 0.5$, the WOA will randomly search for an individual whale for a location update based on whale population location information. Its position update formula is as follows [15]:

$$X = (X_1, X_1, X_1, \dots, X_d) \quad (1)$$

$$D = CX_{rand} - X_t \quad (2)$$

$$X_{t+1} = X_{rand} - AD \quad (3)$$

$$A = Ca - a \quad (4)$$

$$C = 2r \quad (5)$$

X_{rand} is the random individual, X_t is the individual under t iterations, X_{t+1} is the updated value, a is a series of decreasing numbers from 2 to 0, A and C are the representation coefficients, and r is a random number of (0,1).

2.2. Surrounding Prey Stage

When the WOA reaches the stage of surrounding prey, its position update will no longer be a random walk search, but rather a position update towards the current optimal solution. According to the previous stage, the whale can roughly grasp the general position of the prey, and can know the position of the individual whale closest to the target position, so as to use the position of the nearest whale as a reference to achieve the purpose of approaching the target position. At this stage, when parameter $A < 1$ and $p < 0.5$, the optimal location of the WOA population is close to or just at the target position. Therefore, excellent individuals of the WOA population serve as the driving force for progress. Other individuals of the WOA population update their positions through the current optimal individual position, forming a spiraling encircling state and gradually approaching the target position until it is reached. The formula of position update in this stage is as follows [15]:

$$X_{t+1} = X_{best(t)} - A|C X_{best(t)} - X_t| \quad (6)$$

$$D = CX_{best} - X_t \quad (7)$$

2.3. Bubble Net Predation

During the WOA iteration process, the algorithm continues to perform position transformations based on the optimal whale. At this stage, the optimal whale determines the final location of the target, and drives the prey through the whale's unique spiral and bursts a series of bubbles, finally driving the gathered prey into the whale to eat. The hunt is now complete. The model is analyzed as follows: When parameter $A < 1$ and probability $p \geq 0.5$, the WOA meets the hunting requirements and uses a bubble net strategy to hunt prey. This phase can be modeled as follows [15]:

$$X_{t+1} = D_{b-t} e^{bl} \cos(2\pi l) + X_{best}, p \geq 0.5 \quad (8)$$

b is a constant and l is a random number $(-1,1)$.

The WOA flowchart is shown in Figure 1:

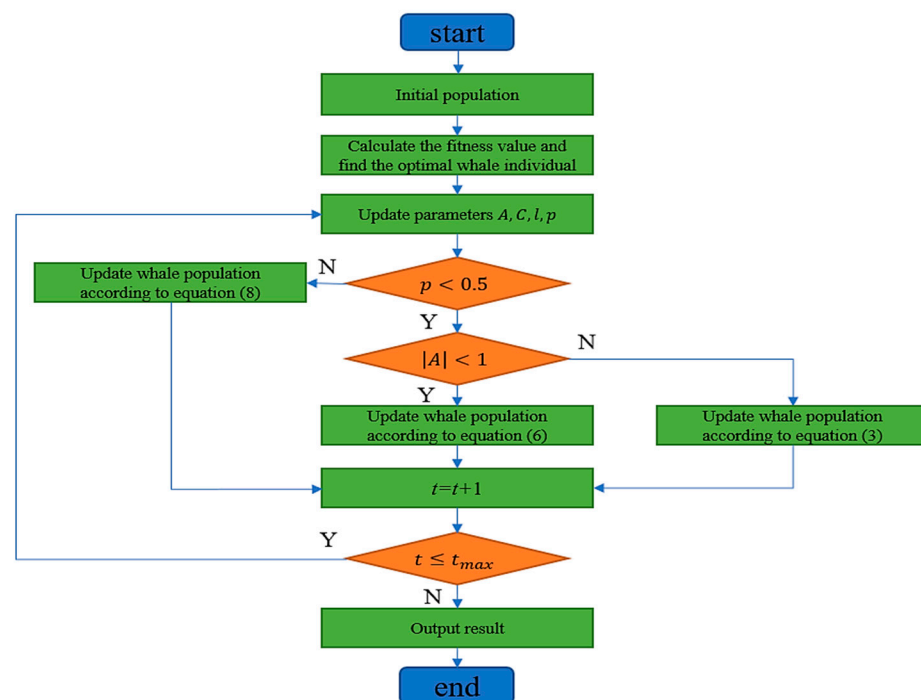


Figure 1. Flowchart of the WOA.

3. Improvement Strategies

In the WOA with multi-strategy integration, the WOA itself has three phases. Therefore, it is worth trying out which strategy to use at different stages of the WOA. Therefore, in this paper, nonlinear time-varying factors and inertia weight strategies are introduced in the random search phase of the WOA to make the population more diverse. In the wandering and encircling phases of the WOA, the gravitational balancing strategy is added to help the poor individuals get closer to the good ones faster. At the end of the three stages, a respawn mechanism is added to help individual whales jump out of the local optimum. The strategy proposed above is an attempt to improve the WOA, and is described in detail below.

3.1. Nonlinear Time-Varying Factors and Inertia Weights

Parameter A has important significance during the optimization period, and the whale optimization algorithm adjusts it through the step size factor a . In the literature, A is linearly decreasing. The magnitude of A decreases significantly in the early exploration phase, and the population relies on a longer step size to search for surrounding food. Later in the iteration, the magnitude of A , due to its own large step descent value, appears to be missing when it is necessary to mine hidden optimal solutions around the optimal solution. Therefore, inspired by the dynamic convergence factor in reference [20], this article proposes an improved nonlinear time-varying factor to improve a . This is used to compensate for the insufficient exploration ability in the early exploration phase of the the WOA and enhance its later development. The formula for the nonlinear time-varying factors is as follows:

$$\begin{cases} a = 1 + \sin\left(\frac{pi}{2} - pi \times \frac{t}{Maxiter}\right) & t < Maxiter/2 \\ a = 1 - \sin\left(pi \times \frac{t-250}{Maxiter}\right) & t \geq Maxiter/2 \end{cases} \quad (9)$$

Moreover, the adaptive weighting strategy is added to give weight to the population, which delays the convergence rate of the GWOA, enhances its search ability, and makes the search area more thorough. The weight growth will be slowed down by the increase in the number of iterations. At the beginning of the iteration, the weight is small and has

little effect on the position and exploration ability of the whale. However, with the gradual increase in the number of iterations, the weight gradually increases, and the position of the optimal solution becomes more attractive to the population, which makes the population close to the optimal value by degrees. The search ability, exploitation ability and solution accuracy of the GWOA are improved. The formula is described as follows:

$$w = pi * \tan(pi/4 \times (t/Maxiter)) \quad (10)$$

$$X_{t+1} = w \times X_{rand} - AD \quad (11)$$

$$X_{t+1} = w \times X_{best} - AD \quad (12)$$

X_{rand} is a random whale individual position under iteration t , and X_{best} is the location of the best individual whale under iteration t .

3.2. Gravitational Equilibrium Strategy

Because the adaptive weight strategy delays the convergence rate of the algorithm in the iteration period, and considering the randomness of the whale population in the random walk stage of the whale optimization algorithm, there are two methods. First, in the random walk stage, individuals with poor fitness move closer to excellent individuals. Secondly, the individuals with good fitness value are displaced by those with poor fitness value. Whale populations need excellent individuals to help the population capture the final location of their prey, encouraging the first situation above in the way of location renewal. Some of the best whales are therefore affected by the randomness of the random walk. We aimed to reduce the influence of the random walk strategy on excellent individuals and make up for the slow convergence caused by inertia weight strategy. Therefore, inspired by the Virtual Force algorithm (VFA) proposed by Howard A et al. [29], this study proposes a gravity balance strategy to solve the problem of the influence of inferior solutions on excellent individuals in the random walk stage, and at the same time make up for the slow convergence of the inertia weight strategy. The concept of this strategy comes from physics. Celestial bodies have a mutual attraction; this state of mutual attraction exists even between two atoms have interaction, in which case it is defined as quantum entanglement. The concept of gravitational equilibrium is that two objects that are attracted to each other have a hypothetical equilibrium point on their line such that the point exerts the same attraction on both A and B. The model diagram is shown in Figure 2. Therefore, under the condition that the positions of the two celestial bodies remain unchanged, excluding the influence of other objects on the two reference objects, there are three conditions for the equilibrium position of the two celestial bodies:

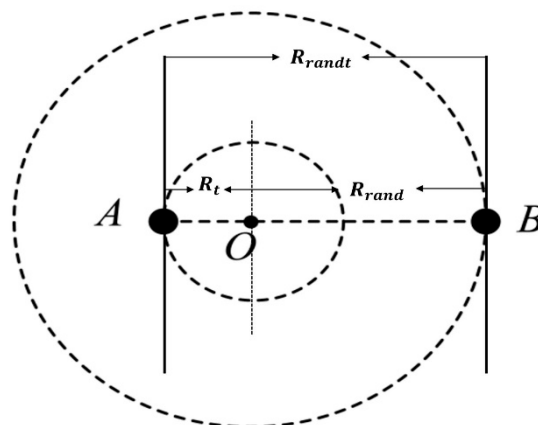


Figure 2. Conceptual diagram of gravitational equilibrium.

- If the mass of celestial body A is greater than that of celestial body B , the equilibrium point is on the line between the two celestial bodies and is close to celestial body B ;
- If the mass of object A is less than that of object B , the equilibrium point is on the line between the two bodies and is close to object A ;
- If the mass of object A is equal to that of object B , the equilibrium point is at the midpoint of the line between the two bodies;
- Its position expression is:

$$\frac{GM_t M_o}{R_{to}^2} = \frac{GM_{rand} M_o}{R_{rand}^2} \quad (13)$$

$$R_{to} = \frac{L_{trand}}{\sqrt{M_t/M_{rand}} + 1} \quad (14)$$

$$R_{to} + R_{rand} = L_{trand} \quad (15)$$

If M_t is equal to M_{rand} , the formula is:

$$R_{to} = R_{rand} = \frac{L_{trand}}{2} \quad (16)$$

M is the fitness value of t whale individuals in the number of iterations, L_{trand} is the distance between two whale individuals, and G is the gravitational constant.

Finally, the final position updating formula is combined with Formulas (11) and (12) to obtain the new Formulas (17) and (18). Considering that the random walk only finds the individual whale population for displacement, according to the gravitational balance strategy, it can be known that the R_{to} of the excellent solution is small when it moves to the inferior solution, ensuring the quality of the excellent solution itself. When the disadvantageous solution moves to the superior solution, the distance of R_{rand} is long, which is also helpful for the position update of the disadvantageous solution. The construction of the strategy model shows a reasonable protection for the excellent solution during the process of the random walk. At the same time, the inferior solution retains the original location update characteristics and increases the process of the inferior solution to develop new excellent solutions around the excellent solution.

$$X_{t+1} = w \times X_{rand} - AL_{trand} / (\sqrt{M_t/M_{rand}} + 1) \quad (17)$$

$$X_{t+1} = w \times X_{best} - AL_{tbest} / (\sqrt{M_t/M_{best}} + 1) \quad (18)$$

3.3. Regeneration Mechanism

Like other optimization algorithms, it is easy for the WOA to be caught in local extremum stagnation during the iterative process, which affects the search accuracy. This study proposes a regeneration mechanism to help the WOA jump out of local stagnation stasis and find better solutions, inspired by the benefit for hundreds of millions of organisms that arises after whale death. As the saying goes, when a whale falls, everything is born. When a whale dies, it creates a micro-ecosystem around it, and the whale becomes an energy source for a small biosphere that benefits the creatures in it. When the WOA falls into stagnation in the late iteration period, the current optimal value does not reach the ideal state and no position update occurs, and the whale enters the near-death state by default. When the number of Nd population exceeds the near-death warning value Ng , no position change is made at the same time, and it is considered that the whale needs to give up its current position and fall into a local optimal state. At this time, it needs to assign a new position to replace the current position of an individual whale. It can continue to find

more suitable points around the local optimal solution, and realize a better solution in the sub-optimal selection. Its location is updated as follows:

$$X_{i,j}^{t+1} = (ub - lb) \times rand + lb \quad (19)$$

3.4. Algorithm Flow

- Step 1: Initialize the population size N , problem dimension dim , maximum number of iterations T_{max} , and calculate the initial fitness values of each whale and record them;
- Step 2: Update parameters a and w according to Formulas (9) and (10), and update A , C , L and p at the same time;
- Step 3: If $p < 0.5$ & $|A| \geq 1$, refer to Formula (17) to carry out the random walk strategy. If not, skip to Step (4);
- Step 4: If $p < 0.5$ & $|A| < 1$, carry out the strategy of encircling the prey according to Formula (18). If conditions (3) and (4) are not met, skip to step (5);
- Step 5: If $p \geq 0.5$ & $|A| < 1$, update the location of the globally optimal whale according to spiral Formula (8);
- Step 6: Determine whether the regeneration strategy position update mechanism is satisfied. If yes, update the whale position according to Equation (19);
- Step 7: $t = t + 1$, judge whether the loop condition is over, meet the end condition, output the global optimal location and solution, otherwise return to step 2.

The pseudo-code of the GWOA is as follows in Algorithm 1:

Algorithm 1. GWOA

1. The population size N of the algorithm is initialized, the fitness value function problem dimension Dim , T_{max} , FES_{max} .
 2. $t = 0$
 3. $FES = 0$
 4. **While** ($t \leq T_{max}$ and $FES \leq FES_{max}$) **do**
 5. $t = t + 1$
 6. **for** $i \rightarrow N$ **do**
 7. Calculate the initial population fitness value, record the current population fitness value, Find the whale individual with the best fitness X_{best} . The individual fitness value $FES + 1$ is calculated once
 8. **end for**
 9. **for** $j \rightarrow N$ **do**
 10. according to Formulas (9) and (10), Update parameters a , w , A , C , L and p
 11. **If** ($p < 0.5$ & $|A| \geq 1$)
 12. Finding a random individual in the whale population
 13. Updating whale position with random walk strategy according to Formula (17)
 14. **Else if** ($p < 0.5$ & $|A| < 1$)
 15. Find an optimal individual in the whale population
 16. Use the Formula (18) to surround the prey, update the whale position
 17. **Else**
 18. Updating the optimal whale location based on the spiral Formula (8)
 19. **end if**
 20. **end for**
 21. **for** $i \rightarrow N$ **do**
 22. The machine will determine whether the position update mechanism of the regeneration strategy is met, if it is met, a new whale individual will be generated based on the Formula (19) to replace the position of the endangered whale
 23. **end for**
 24. **End while**
 25. **end**
 26. **output:** optimal solution
-

3.5. Time Complexity Analysis

Time complexity is one of the important indices for measuring optimization efficiency, performance and computation cost. To prove that the GWOA does not increase the time complexity of the WOA, this paper analyzes both the GWOA and WOA. In the WOA process, the population size is M , the time consumed by the initial individuals is $O(t_1)$, and the time consumed by the objective function of the Dim dimension problem is $O(D)$. Therefore, the time complexity to enter the initial population stage is $O(M \times (t_1))$. Enter the cycle and calculate the fitness value of the individual population as $O(M \times (D))$. As can be seen from Equation (3), the time at which the solution is generated in the random walk stage is $O(t_2)$. As can be seen from Equation (6), the time at which a new solution is generated according to the optimal solution in the shrinkage inclusion stage is $O(t_3)$. From Equation (8), it can be seen that, in the hunting stage ended by the last step, the time at which the solution is generated is $O(t_4)$. Because t_2 , t_3 and t_4 are parallel structures, the total time complexity of each phase of the WOA is $O(M \times (t_1)) + O(M \times (D + t))$. Assuming that the maximum number of iterations is T , the final time complexity of the WOA is $O(M \times (t_1)) + O(M \times (D + t_2 + t_3 + t_4)) = O(T \times M \times (t + D))$. At present, the PSO time complexity analysis methods are as follows: the time to initialize the population M and calculate the fitness value is $O(M \times t_1)$, the time to calculate the speed V is t_2 when entering the cycle, and the time complexity of the adopted problem is $O(D)$ when the population individual updates t_3 . Therefore, the time taken at the end of a cycle is $O(t_1) + O((t_2 + t_3 + D) \times M)$, and the number of iterations assumed is T . Then, the final time complexity is $O(M \times t_1) + O(T \times M \times (D + t_2 + t_3)) = O(T \times M \times (D + t))$.

Although different strategies are added to the GWOA in different stages, the nonlinear factors, inertia weights and gravitational balance strategies are the same as for the WOA in stochastic and enveloping stages. Although the respawning mechanism produces new solutions, only very few of them need to be replaced. It is assumed that the time of the generation of the regeneration stage solution is $o(t_5)$. Therefore, the total time complexity of the GWOA is $O(M \times (t_1)) + O(T \times M \times (D + t_2 + t_3 + t_4 + t_5)) = O(T \times M \times (t + D))$. Therefore, It can be concluded that when all parameters are the same, the time complexity of the GWOA is the same as that of the WOA and PSO, and the improvement of the GWOA will not increase the time complexity of the algorithm.

4. Experiment Tests and Analysis

In order to verify the optimization performance of the algorithm after adding the above strategies, 16 classical benchmark test functions were first selected to test the performance of GWOA, and six algorithms were selected to conduct 30-dimensional and 100-dimensional comparison experiments. The mean and standard deviation values were used to measure the comprehensive performance of GWOA. At the same time, the convergence graphs and interval box graphs of each algorithm in 30 dimensions are listed to help prove the convergence effect and stability of GWOA. Next, through the contribution experiment of each algorithm, we verify the improvement of the three strategies added to the original algorithm. Finally, the Wilcoxon rank sum test and Friedman test were used to help verify the difference between algorithms. The comprehensive experiment and analysis results will be presented below.

The following table shows 16 common benchmark function expressions, dimensions, ranges of variable values and theoretical optimal values in the range of variable values. The function F1–F7 is a unimodal function, F8–F11 is a multimodal function, and F12–F16 is a test function with fixed dimension. Detailed test function information is shown in Table 1:

Table 1. The details of benchmark functions.

Function	Dim	Range	Optimum Value
$F_1(x) = \sum_{i=1}^d x_i^2$	30/ 100	$[-100, 100]$	0
$F_2(x) = \sum_{i=1}^d x_i + \prod_{i=1}^d x_i $	30/ 100	$[-10, 10]$	0
$F_3(x) = \sum_{i=1}^d \left(\sum_{j=1}^d x_j \right)^2$	30/ 100	$[-100, 100]$	0
$F_4(x) = \max_i \{ x_i , 1 \leq i \leq d\}$	30/ 100	$[-100, 100]$	0
$F_5(x) = \sum_{i=1}^d [100(x_{i+1} - x_i^2)^2 - (x_i - 1)^2]$	30/ 100	$[-10, 10]$	0
$F_6(x) = \sum_{i=1}^d ([x_i + 0.5])^2$	30/ 100	$[-30, 30]$	0
$F_7(x) = \sum_{i=1}^d ix_i^4 + \text{random}[0, 1)$	30/ 100	$[-1.28, 1.28]$	0
$F_8(x) = \sum_{i=1}^d -x_i \sin(\sqrt{ x_i })$	30/ 100	$[-500, 500]$	-498.5 n
$F_9(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{d} \sum_{i=1}^d x_i^2}\right) - \exp\left(\frac{1}{d} \sum_{i=1}^d \cos(2\pi x_i)\right) + 20 + e$	30/ 100	$[-32, 32]$	0
$F_{10}(x) = \frac{\pi}{d} \left\{ 10 \sin(\pi y_i) + \sum_{i=1}^{d-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1}) + (y_d - 1)^2] \right\}$ $+ \sum_{i=1}^d u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_{i+1}}{4}, u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30/ 100	$[-50, 50]$	0
$F_{11}(x) = 0.1 \{ \sin^2(3\pi x_1) + \sum_{i=1}^d (x_i - 1)^2 [1 + \sin^2(3\pi x_1 + 1)] + (x_n - 1)^2 [1 + \sin^2 \sin^2(2\pi x_n)] \} + \sum_{i=1}^d u(x_i, 10, 100, 4)$	30/ 100	$[0, \pi]$	0
$F_{12}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^2 (x_i - a_{ij})} \right)^{-1}$	2	$[-65, 65]$	1
$F_{13}(x) = \left[1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2) \right] \times [30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	$[-2, 2]$	3
$F_{14}(x) = -\sum_{i=1}^4 [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.1532
$F_{15}(x) = -\sum_{i=1}^7 [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.4028
$F_{16}(x) = -\sum_{i=1}^{10} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	$[0, 10]$	-10.5363

Simulation experiments are carried out on the MATLAB2020b platform, and experiments were run to solve problems in 30 and 100 dimensions of 16 test functions. The optimization results of the GWOA are compared with the WOA [15], DEWGA [30], GWO [12], PSO [10], MPA [31] and HHO [32]. During the experiments, in order to ensure the rationality of the experimental results of each algorithm, the control variable method

were adopted. the GWOA and some comparison algorithms are not suitable for comparison based on the unfairness of the same number of iterations [33], so a more reasonable evaluation method based on FEs was chosen. Therefore, we set the overall size of all algorithms to 30 and evaluated 50,000 times. The parameter settings related to each algorithm are shown in Table 2:

Table 2. The parameters of each algorithm.

Algorithm	Parameter Setting
WOA	$N = 30, a = 2 - t \times (2/t_{max})$
DECWOA	$N = 30, t_{max} = 500, w = 0.5 + \exp(-f_{fit}(x)/u)^t$
GWO	$N = 30, a = 2 - t \times (2/t_{max})$
PSO	$N = 30, w_{max} = 0.9, w_{min} = 0.4, c_1 = c_2 = 2.0$
MPA	$N = 30, P = 0.5, P_f = 0.2$
HHO	$N = 30$
GWOA	$N = 30, w = \pi \times \tan(\pi/4 \times (t/Maxiter))$

The simulation environment of this experiment was as follows: the simulation experiment software was MATLAB R2020b, the operating system was Windows 11, the main frequency was 3.20 ghz, and 16 GB memory was used. The results are sufficient to avoid the influence of experimental randomness and chance. The GWOA and comparison algorithms ran each benchmark function 30 times independently on two dimensions, and the records are shown in Tables 3 and 4. In this study, the mean value and standard deviation are used to evaluate the performance of the above algorithms. The average value represents the average optimization ability of the algorithm on the objective function, and the smaller the standard deviation, the better the stability of the optimization algorithm in the current benchmark test function. Finally, under the same benchmark function, the optimal data are marked in bold.

Table 3. Experimental results of each optimization algorithm (30 D/50,000 fes).

Function	Algorithm	Mean	Std	Best	Worst	Rank
F1	DECWOA	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0	1
	GWO	1.109×10^{-100}	1.760×10^{-100}	5.883×10^{-105}	6.970×10^{-100}	4
	HHO	4.980×10^{-55}	1.847×10^{-54}	2.140×10^{-66}	9.961×10^{-54}	5
	MPA	6.532×10^{-43}	2.324×10^{-42}	1.581×10^{-45}	1.305×10^{-41}	6
	PSO	9.451×10^{-10}	1.336×10^{-9}	4.020×10^{-11}	5.412×10^{-9}	7
	WOA	1.159×10^{-201}	0.000×10^0	1.382×10^{-247}	3.478×10^{-200}	3
	GWOA	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0	1
F2	DECWOA	1.038×10^{-281}	0.000×10^0	4.690×10^{-289}	1.510×10^{-280}	2
	GWO	1.802×10^{-58}	3.299×10^{-58}	3.987×10^{-60}	1.723×10^{-57}	4
	HHO	4.471×10^{-28}	1.831×10^{-27}	5.553×10^{-36}	1.016×10^{-26}	6
	MPA	1.798×10^{-24}	2.762×10^{-24}	6.237×10^{-28}	1.364×10^{-23}	5
	PSO	1.067×10^1	9.286×10^0	8.323×10^{-7}	3.000×10^1	7
	WOA	1.766×10^{-172}	0.000×10^0	1.665×10^{-187}	5.258×10^{-171}	3
	GWOA	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0	1
F3	DECWOA	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0	1
	GWO	1.241×10^{-26}	4.816×10^{-26}	8.402×10^{-35}	2.395×10^{-25}	4
	HHO	2.361×10^{-39}	1.040×10^{-38}	1.402×10^{-59}	5.788×10^{-38}	3
	MPA	3.677×10^{-11}	1.532×10^{-10}	5.647×10^{-21}	8.471×10^{-10}	5
	PSO	1.925×10^1	9.203×10^0	9.355×10^0	5.321×10^1	6
	WOA	8.803×10^3	7.316×10^3	3.430×10^2	2.726×10^4	7
	GWOA	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0	1

Table 3. Cont.

Function	Algorithm	Mean	Std	Best	Worst	Rank
F4	DECWOA	3.418×10^{-239}	0.000×10^0	8.379×10^{-252}	7.898×10^{-238}	2
	GWO	1.583×10^{-24}	2.883×10^{-24}	1.276×10^{-26}	1.374×10^{-23}	4
	HHO	2.839×10^{-27}	1.032×10^{-26}	6.385×10^{-34}	5.218×10^{-26}	3
	MPA	9.160×10^{-17}	7.741×10^{-17}	1.150×10^{-17}	3.151×10^{-16}	5
	PSO	9.300×10^{-1}	2.608×10^{-1}	4.862×10^{-1}	1.577×10^0	6
	WOA	3.432×10^1	3.190×10^1	7.631×10^{-3}	8.751×10^1	7
	GWOA	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0	1
F5	DECWOA	7.587×10^{-2}	1.097×10^{-1}	2.745×10^{-5}	4.682×10^{-1}	3
	GWO	2.652×10^1	7.037×10^{-01}	2.527×10^1	2.796×10^1	5
	HHO	3.959×10^{-2}	5.344×10^{-2}	2.327×10^{-1}	2.135×10^{-4}	2
	MPA	2.558×10^1	6.968×10^{-1}	2.450×10^1	2.748×10^1	4
	PSO	1.785×10^2	5.416×10^2	5.711×10^0	3.034×10^3	7
	WOA	2.655×10^1	6.984×10^{-1}	2.595×10^1	2.872×10^1	6
	GWOA	5.542×10^{-3}	1.031×10^{-2}	1.539×10^{-5}	4.511×10^{-2}	1
F6	DECWOA	1.047×10^{-2}	2.142×10^{-2}	2.228×10^{-5}	1.200×10^{-1}	5
	GWO	5.726×10^{-1}	3.101×10^{-1}	5.436×10^{-6}	1.004×10^0	7
	HHO	8.034×10^{-4}	1.490×10^{-3}	6.621×10^{-8}	6.556×10^{-3}	3
	MPA	6.638×10^{-2}	2.771×10^{-2}	2.557×10^{-2}	1.342×10^{-1}	6
	PSO	1.174×10^{-9}	1.511×10^{-9}	9.635×10^{-12}	6.856×10^{-9}	1
	WOA	7.657×10^{-3}	1.318×10^{-2}	1.914×10^{-3}	7.494×10^{-2}	4
	GWOA	2.213×10^{-4}	2.728×10^{-4}	5.203×10^{-8}	8.539×10^{-4}	2
F7	DECWOA	8.528×10^{-5}	9.926×10^{-5}	6.381×10^{-6}	4.341×10^{-4}	2
	GWO	5.375×10^{-4}	2.586×10^{-4}	1.258×10^{-4}	1.184×10^{-3}	4
	HHO	3.872×10^{-4}	3.588×10^{-4}	5.965×10^{-6}	1.845×10^{-3}	3
	MPA	7.656×10^{-4}	3.933×10^{-4}	2.328×10^{-4}	1.766×10^{-3}	5
	PSO	5.598×10^0	5.362×10^0	2.693×10^{-2}	1.882×10^1	7
	WOA	2.162×10^{-3}	1.150×10^{-3}	1.952×10^{-5}	4.614×10^{-3}	6
	GWOA	2.105×10^{-5}	1.736×10^{-5}	6.791×10^{-7}	6.585×10^{-5}	1
F8	DECWOA	-1.093×10^4	1.556×10^3	-1.256×10^4	-7.419×10^3	4
	GWO	-6.043×10^3	7.729×10^2	-7.901×10^3	-4.666×10^3	7
	HHO	-1.255×10^4	8.996×10^1	-1.257×10^4	-1.207×10^4	2
	MPA	-8.633×10^3	5.135×10^2	-9.670×10^3	-7.842×10^3	5
	PSO	-7.004×10^3	6.751×10^2	-8.304×10^3	-5.265×10^3	6
	WOA	-1.122×10^4	1.609×10^3	-1.257×10^4	-8.155×10^3	3
	GWOA	-1.257×10^4	5.298×10^{-3}	-1.257×10^4	-1.257×10^4	1
F9	DECWOA	1.717×10^{-15}	1.503×10^{-15}	8.882×10^{-16}	4.441×10^{-15}	3
	GWO	1.072×10^{-14}	3.393×10^{-15}	4.441×10^{-15}	1.510×10^{-14}	6
	HHO	8.882×10^{-16}	0.000×10^0	8.882×10^{-16}	8.882×10^{-16}	2
	MPA	4.086×10^{-15}	1.066×10^{-15}	8.882×10^{-16}	4.441×10^{-15}	5
	PSO	4.697×10^{-05}	5.923×10^{-05}	3.723×10^{-06}	2.387×10^{-04}	7
	WOA	3.612×10^{-15}	2.371×10^{-15}	8.882×10^{-16}	7.994×10^{-15}	4
	GWOA	8.882×10^{-16}	0.000×10^0	8.882×10^{-16}	8.882×10^{-16}	1
F10	DECWOA	1.761×10^{-1}	1.223×10^{-1}	4.751×10^{-2}	4.687×10^{-1}	7
	GWO	3.382×10^{-2}	1.810×10^{-2}	6.540×10^{-3}	7.931×10^{-2}	6
	HHO	1.337×10^{-2}	1.514×10^{-2}	3.056×10^{-6}	6.628×10^{-2}	5
	MPA	1.654×10^{-3}	1.139×10^{-3}	3.977×10^{-4}	5.364×10^{-3}	2
	PSO	6.911×10^{-3}	2.586×10^{-2}	1.771×10^{-13}	1.037×10^{-01}	4
	WOA	2.197×10^{-3}	3.159×10^{-3}	2.711×10^{-4}	1.392×10^{-2}	3
	GWOA	6.194×10^{-6}	1.370×10^{-5}	6.630×10^{-10}	7.023×10^{-5}	1

Table 3. Cont.

Function	Algorithm	Mean	Std	Best	Worst	Rank
F11	DECWOA	9.795×10^{-1}	2.554×10^{-1}	2.336×10^{-1}	1.450×10^0	7
	GWO	4.945×10^{-1}	2.342×10^{-1}	9.743×10^{-2}	1.120×10^0	6
	HHO	4.249×10^{-4}	5.236×10^{-4}	5.594×10^{-11}	1.954×10^{-3}	2
	MPA	3.575×10^{-2}	2.176×10^{-2}	5.093×10^{-3}	9.639×10^{-2}	4
	PSO	3.296×10^{-3}	5.035×10^{-3}	1.775×10^{-11}	1.099×10^{-2}	3
	WOA	1.061×10^{-1}	1.075×10^{-1}	8.354×10^{-3}	3.793×10^{-1}	5
	GWOA	6.110×10^{-5}	4.245×10^{-5}	3.741×10^{-8}	2.120×10^{-4}	1
F12	DECWOA	9.980×10^{-1}	0.000×10^0	9.980×10^{-1}	9.980×10^{-1}	1
	GWO	3.874×10^0	3.724×10^0	9.980×10^{-1}	1.267×10^1	7
	HHO	2.119×10^0	1.646×10^0	9.980×10^{-1}	5.929×10^0	5
	MPA	9.980×10^{-1}	0.000×10^0	9.980×10^{-1}	9.980×10^{-1}	1
	PSO	3.463×10^0	2.478×10^0	9.980×10^{-1}	1.076×10^1	6
	WOA	2.079×10^0	2.451×10^0	9.980×10^{-1}	1.076×10^1	4
	GWOA	9.980×10^{-1}	0.000×10^0	9.980×10^{-1}	9.980×10^{-1}	1
F13	DECWOA	3.000×10^0	2.153×10^{-4}	3.000×10^0	3.001×10^0	6
	GWO	3.000×10^0	3.260×10^{-6}	3.000×10^0	3.000×10^0	3
	HHO	3.000×10^0	1.271×10^{-5}	3.000×10^0	3.000×10^0	4
	MPA	3.000×10^0	1.240×10^{-15}	3.000×10^0	3.000×10^0	1
	PSO	3.000×10^0	1.347×10^{-15}	3.000×10^0	3.000×10^0	5
	WOA	3.000×10^0	1.236×10^{-5}	3.000×10^0	3.000×10^0	4
	GWOA	3.000×10^0	3.507×10^{-3}	3.000×10^0	3.015×10^0	7
F14	DECWOA	-5.146×10^0	2.326×10^0	-9.879×10^0	-2.435×10^0	7
	GWO	-9.646×10^0	1.520×10^0	-1.015×10^1	-5.055×10^0	3
	HHO	-5.739×10^0	1.614×10^0	-1.011×10^1	-4.924×10^0	6
	MPA	-1.015×10^1	5.299×10^{-15}	-1.015×10^1	-1.015×10^1	1
	PSO	-6.708×10^0	2.936×10^0	-1.015×10^1	-2.630×10^0	5
	WOA	-9.303×10^0	1.900×10^0	-1.015×10^1	-5.055×10^0	4
	GWOA	-1.015×10^1	6.019×10^{-05}	-1.015×10^1	-1.015×10^1	2
F15	DECWOA	-5.217×10^0	2.150×10^0	-1.002×10^1	-2.325×10^0	7
	GWO	-1.040×10^1	9.969×10^{-5}	-1.040×10^1	-1.040×10^1	3
	HHO	-5.237×10^0	9.564×10^{-1}	-1.038×10^1	-4.937×10^0	6
	MPA	-1.040×10^1	1.376×10^{-15}	-1.040×10^1	-1.040×10^1	1
	PSO	-8.738×10^0	2.575×10^0	-1.040×10^1	-2.766×10^0	5
	WOA	-8.830×10^0	2.658×10^0	-1.040×10^1	-2.766×10^0	4
	GWOA	-1.040×10^1	5.405×10^{-5}	-1.040×10^1	-1.040×10^1	2
F16	DECWOA	-6.657×10^0	2.796×10^0	-1.053×10^1	-3.362×10^0	6
	GWO	-1.036×10^1	9.707×10^{-1}	-1.054×10^1	-5.128×10^0	3
	HHO	-5.180×10^0	1.011×10^0	-9.945×10^0	-2.382×10^0	7
	MPA	-1.054×10^1	1.020×10^{-14}	-1.054×10^1	-1.054×10^1	1
	PSO	-9.637×10^0	2.012×10^0	-5.128×10^0	-1.054×10^1	4
	WOA	-9.589×10^0	2.121×10^0	-1.054×10^1	-3.835×10^0	5
	GWOA	-1.054×10^1	8.819×10^{-5}	-1.054×10^1	-1.054×10^1	2

Table 4. Experimental results of each optimization algorithm (100 D/50,000 fes).

Function	Algorithm	Mean	Std	Best	Worst	Rank
F1	DECWOA	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0	1
	GWO	4.767×10^{-52}	6.152×10^{-52}	1.652×10^{-53}	2.762×10^{-51}	4
	HHO	4.289×10^{-51}	2.253×10^{-50}	1.536×10^{-63}	1.256×10^{-49}	5
	MPA	7.552×10^{-37}	2.216×10^{-36}	1.013×10^{-39}	1.063×10^{-35}	6
	PSO	4.476×10^0	2.353×10^0	1.966×10^0	1.466×10^1	7
	WOA	8.745×10^{-249}	0.000×10^0	3.406×10^{-281}	2.367×10^{-247}	3
	GWOA	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0	1

Table 4. Cont.

Function	Algorithm	Mean	Std	Best	Worst	Rank
F2	DECWOA	2.121×10^{-280}	0.000×10^0	1.711×10^{-288}	5.598×10^{-279}	2
	GWO	3.760×10^{-31}	2.429×10^{-31}	6.812×10^{-32}	9.339×10^{-31}	4
	HHO	5.974×10^{-27}	2.200×10^{-26}	1.261×10^{-32}	9.778×10^{-26}	5
	MPA	1.433×10^{-21}	2.219×10^{-21}	9.374×10^{-23}	1.165×10^{-20}	6
	PSO	1.330×10^2	3.058×10^1	7.062×10^1	2.022×10^2	7
	WOA	6.285×10^{-173}	0.000×10^0	7.623×10^{-186}	9.895×10^{-172}	3
	GWOA	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0	1
F3	DECWOA	3.284×10^{-300}	0.000×10^0	0.000×10^0	9.850×10^{-299}	2
	GWO	8.879×10^{-2}	2.704×10^{-1}	5.568×10^{-8}	1.099×10^0	5
	HHO	2.418×10^{-22}	6.512×10^{-22}	5.585×10^{-59}	3.628×10^{-21}	3
	MPA	3.422×10^{-3}	4.402×10^{-3}	3.841×10^{-8}	1.647×10^{-2}	4
	PSO	1.402×10^4	3.849×10^3	8.340×10^3	3.029×10^4	6
	WOA	6.921×10^5	1.518×10^5	3.080×10^5	1.010×10^6	7
	GWOA	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0	1
F4	DECWOA	4.888×10^{-238}	0.000×10^0	2.822×10^{-250}	1.264×10^{-236}	2
	GWO	1.798×10^{-6}	4.767×10^{-6}	5.242×10^{-10}	1.988×10^{-5}	5
	HHO	1.672×10^{-26}	1.034×10^{-25}	6.348×10^{-33}	4.837×10^{-25}	3
	MPA	6.422×10^{-14}	4.993×10^{-14}	1.003×10^{-14}	2.731×10^{-13}	4
	PSO	1.047×10^1	1.067×10^0	7.704×10^0	2.631×10^1	6
	WOA	7.355×10^1	2.539×10^1	1.574×10^1	9.677×10^1	7
	GWOA	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0	1
F5	DECWOA	7.731×10^{-2}	1.680×10^{-1}	9.121×10^{-6}	8.441×10^{-1}	2
	GWO	9.757×10^1	7.703×10^{-1}	9.608×10^1	9.844×10^1	5
	HHO	1.032×10^{-1}	1.659×10^{-1}	3.784×10^{-6}	8.328×10^{-1}	3
	MPA	9.762×10^1	6.992×10^{-1}	9.609×10^1	9.848×10^1	6
	PSO	1.089×10^4	1.644×10^4	2.919×10^3	9.666×10^4	7
	WOA	9.713×10^1	5.503×10^{-1}	9.641×10^1	9.818×10^1	4
	GWOA	6.354×10^{-3}	8.316×10^{-3}	8.213×10^{-5}	3.236×10^{-2}	1
F6	DECWOA	7.564×10^0	2.046×10^0	2.046×10^0	1.548×10^1	4
	GWO	9.311×10^0	9.871×10^{-1}	7.506×10^0	1.151×10^1	6
	HHO	1.600×10^{-3}	1.563×10^{-3}	2.680×10^{-6}	5.413×10^{-3}	2
	MPA	8.759×10^0	1.125×10^0	6.888×10^0	1.110×10^1	5
	PSO	1.271×10^1	4.203×10^0	1.641×10^0	1.835×10^1	7
	WOA	6.353×10^{-1}	2.295×10^{-1}	2.935×10^{-1}	1.179×10^0	3
	GWOA	6.107×10^{-4}	7.569×10^{-4}	2.193×10^{-3}	9.506×10^{-7}	1
F7	DECWOA	1.074×10^{-4}	1.202×10^{-4}	3.821×10^{-6}	5.318×10^{-4}	2
	GWO	1.367×10^{-3}	6.785×10^{-4}	4.672×10^{-4}	3.082×10^{-3}	6
	HHO	3.580×10^{-4}	3.410×10^{-4}	1.990×10^{-5}	1.467×10^{-3}	3
	MPA	1.241×10^{-3}	5.527×10^{-4}	3.081×10^{-4}	2.415×10^{-3}	5
	PSO	2.465×10^2	1.131×10^2	5.989×10^1	6.061×10^2	7
	WOA	1.162×10^{-3}	1.860×10^{-3}	8.025×10^{-5}	9.878×10^{-3}	4
	GWOA	8.503×10^{-5}	5.969×10^{-5}	2.406×10^{-6}	3.279×10^{-4}	1
F8	DECWOA	-3.594×10^4	7.935×10^3	-4.190×10^4	-1.530×10^4	3
	GWO	-1.674×10^4	2.410×10^3	-2.007×10^4	-6.180×10^3	7
	HHO	-4.151×10^4	1.693×10^3	-4.190×10^4	-3.254×10^4	2
	MPA	-2.054×10^4	1.297×10^3	-2.354×10^4	-1.800×10^4	6
	PSO	-2.122×10^4	1.871×10^3	-2.488×10^4	-1.633×10^4	5
	WOA	-3.504×10^4	6.199×10^3	-4.190×10^4	-2.708×10^4	4
	GWOA	-4.190×10^4	2.467×10^{-2}	-4.190×10^4	-4.190×10^4	1

Table 4. Cont.

Function	Algorithm	Mean	Std	Best	Worst	Rank
F9	DECWOA	1.480×10^{-15}	1.324×10^{-15}	4.441×10^{-15}	8.882×10^{-16}	3
	GWO	3.393×10^{-14}	4.118×10^{-15}	2.576×10^{-14}	3.997×10^{-14}	6
	HHO	8.882×10^{-16}	0.000×10^0	8.882×10^{-16}	8.882×10^{-16}	1
	MPA	4.441×10^{-15}	0.000×10^0	4.441×10^{-15}	4.441×10^{-15}	5
	PSO	3.117×10^0	1.421×10^0	2.134×10^0	1.054×10^1	7
	WOA	3.730×10^{-15}	2.321×10^{-15}	8.882×10^{-16}	7.994×10^{-15}	4
	GWOA	8.882×10^{-16}	0.000×10^0	8.882×10^{-16}	8.882×10^{-16}	1
F10	DECWOA	1.361×10^{-1}	4.666×10^{-2}	7.870×10^{-2}	2.776×10^{-1}	5
	GWO	2.441×10^{-1}	6.136×10^{-2}	1.489×10^{-1}	4.127×10^{-1}	6
	HHO	1.065×10^{-5}	1.547×10^{-5}	3.754×10^{-8}	6.632×10^{-5}	2
	MPA	1.256×10^{-1}	3.040×10^{-2}	8.727×10^{-2}	2.134×10^{-1}	4
	PSO	2.223×10^0	8.594×10^{-1}	8.134×10^{-1}	4.035×10^0	7
	WOA	7.379×10^{-3}	1.122×10^{-2}	2.556×10^{-3}	6.692×10^{-2}	3
	GWOA	2.117×10^{-6}	2.512×10^{-6}	4.615×10^{-9}	1.084×10^{-5}	1
F11	DECWOA	3.791×10^0	1.362×10^0	1.841×10^0	7.195×10^0	4
	GWO	6.154×10^0	3.653×10^{-1}	5.562×10^0	6.847×10^0	5
	HHO	2.725×10^{-4}	2.417×10^{-4}	4.121×10^{-7}	9.645×10^{-4}	2
	MPA	8.861×10^0	1.438×10^0	4.728×10^0	9.704×10^0	6
	PSO	5.850×10^1	1.642×10^1	2.893×10^1	8.736×10^1	7
	WOA	1.016×10^0	4.636×10^{-1}	3.253×10^{-1}	2.274×10^0	3
	GWOA	5.125×10^{-5}	7.072×10^{-5}	3.475×10^{-9}	2.589×10^{-4}	1

From Table 3, we can see that the average ranking of GWOA is 1.625, ranking first among the comparison algorithms. Among the seven single-peak functions F1–F7, the average and standard deviation of six functions are better for the GWOA than the comparison algorithm, and GWOA can find the optimal value in F1 and F3 functions, but the accuracy of the F6 function is not as good as that of PSO. For the multimodal function F8–F11, GWOA ranks the first among the four multimodal functions, the average accuracy of the solution of F10 is ahead of the comparison algorithm, and the average accuracy of the solution of F8 is closest to the theoretical optimal value, thus proving the advantage of the GWOA in the multimodal function. The GWOA also shows excellent optimization performance on complex fixed-dimensional functions. Only the F12 function ranks first, but the average values of the F14, F15 and F16 fixed dimensional functions is infinitely close to the theoretical optimal value, only slightly lower than the first MPA algorithm in standard deviation. However, it can also be noticed that the F13 function ranks last; there are still some shortcomings. Combined with the data of fixed dimension function, GWOA is better than the WOA algorithm in terms of accuracy and stability of solution. Therefore, from the data analysis of the three types of test functions, it can be judged that the overall performance of the GWAO algorithm is the best, and it can show good optimization ability for more functions.

In order to verify the stability of GWOA, the interval box diagram was drawn. The results of 30 experiments on 16 functions of each algorithm are analyzed. As can be seen from the box diagram in Figure 3, the box diagram of GWOA is lower and narrower. Therefore, it can be concluded that GWOA stability is good on the test function.

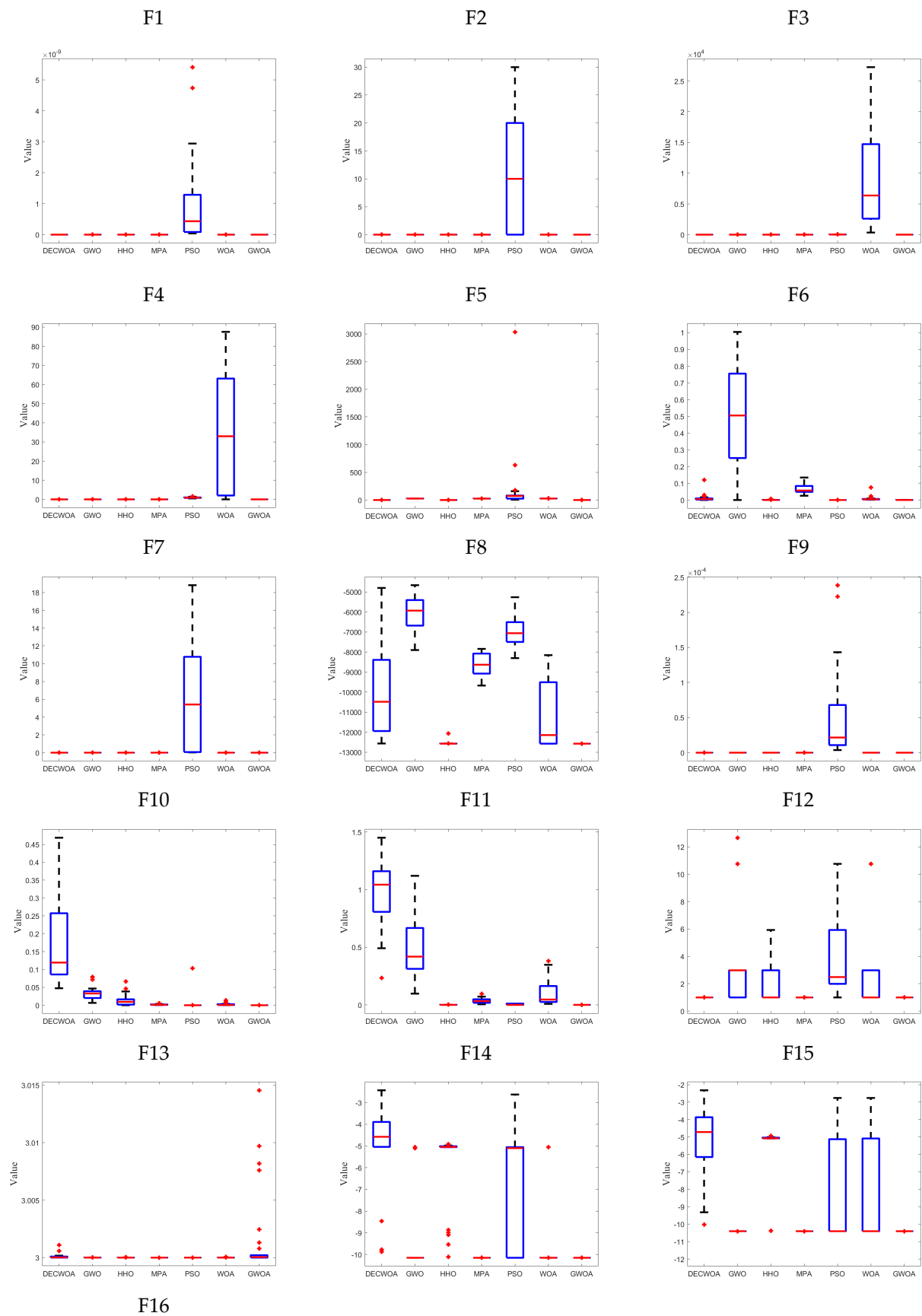


Figure 3. Cont.

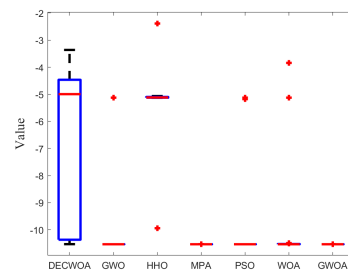


Figure 3. The box plots of each algorithm on the test functions.

In the experimental data shown in Table 4 (100-dimensional), the comprehensive rank of the GWOA is 1, which can find the optimal solution on the test functions F1, F2, F3, F4 and F8, and its accuracy is better than that of the comparison algorithm on the functions F5, F6, F9, F10 and F11. Finally, the GWOA has little difference in comparison with its own 30-dimensional data, and its stability is good. At the same time, the GWOA also shows some advantages in solving high latitude problems.

According to the analysis of convergence accuracy in Figure 4 (30-dimensional), the improved whale optimization algorithm shows certain optimization effects in terms of optimization accuracy and convergence speed. The optimal solution is found in the seven functions F1, F2, F3, F4, F8, F14, F15 and F16. The convergence accuracy of GWOA in F5 is not as high as that of PSO. However, in F6, F7, F9, F10, F11 and F12 functions, when the convergence speed of each algorithm is similar, the GWOA has better stability and more accurate solutions compared to the original whale optimization algorithm when the convergence speed of each algorithm is similar. The convergence diagrams of each benchmark test function are shown in Figure 4:

Finally, in order to verify the role of the three strategies proposed in this article in the optimization process, this article combines the basic Whale Optimization Algorithm (WOA), Whale Optimization Algorithm with nonlinear time-varying factors (WOA-1), Whale Optimization Algorithm with a combination of nonlinear time-varying factors and inertial weights (WOA-2), and Whale Optimization Algorithm with a combination of the WOA-2 and gravity balance strategy (WOA-3). Experimental comparisons and analyses were conducted between the Whale Optimization Algorithm (WOA-4), which combines the WOA-2 with the Rebirth Strategy, and the Improved Whale Optimization Algorithm, which combines all strategies. Experiment-related parameters were kept the same, the population was 30, the fitness function problem dimension was 30, and there was a calculated maximum of 15,000 assessments. The experimental data and evaluation criteria are shown in Table 5. The optimal indicator data are displayed in bold. The number of optimal indicators for each algorithm were counted.

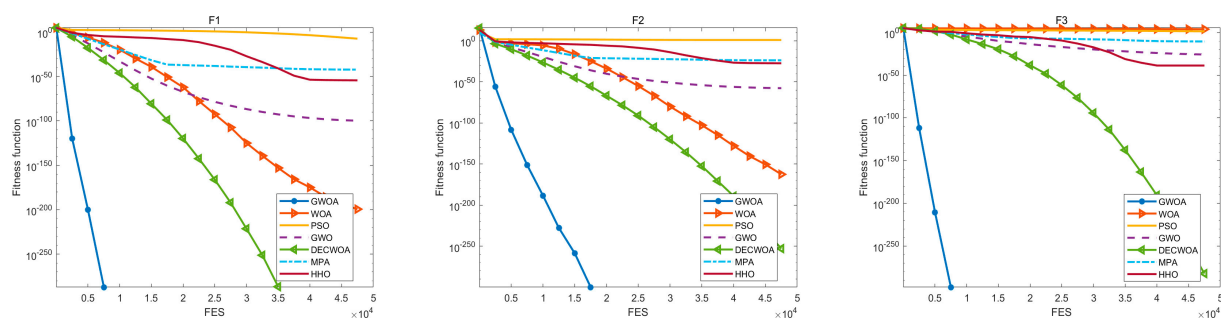


Figure 4. Cont.

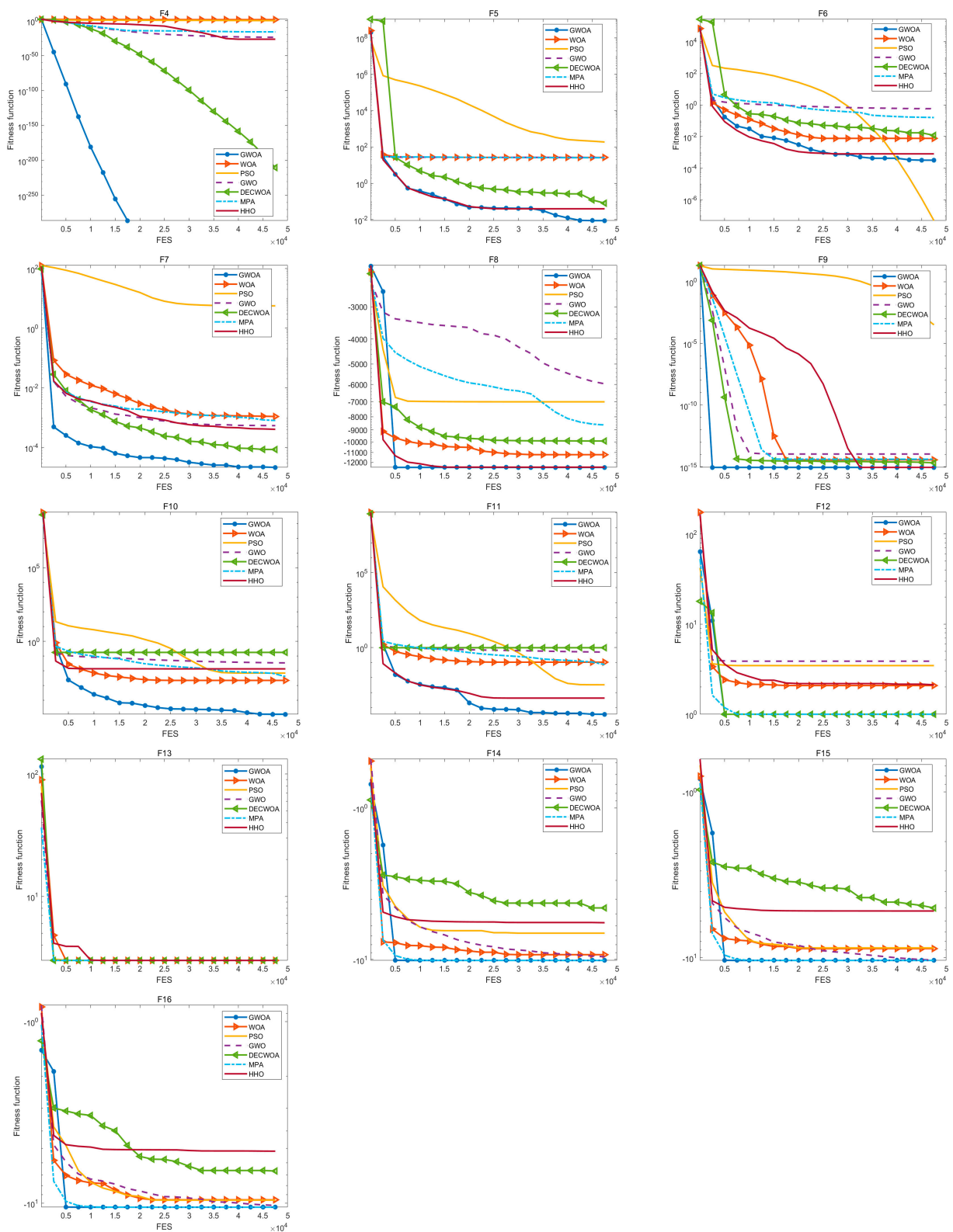


Figure 4. Convergence curves on 16 benchmark functions.

Table 5. Test results of each strategy.

Function	Index	WOA	WOA-1	WOA-2	WOA-3	WOA-4	GWOA
F1	Mean	3.046×10^{-71}	1.797×10^{-104}	8.514×10^{-248}	0.000×10^0	0.000×10^0	0.000×10^0
	Std	1.639×10^{-70}	6.244×10^{-104}	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0
	Best	5.624×10^{-86}	5.817×10^{-115}	1.234×10^{-274}	0.000×10^0	0.000×10^0	0.000×10^0
	Worst	9.135×10^{-70}	2.846×10^{-103}	1.252×10^{-246}	0.000×10^0	0.000×10^0	0.000×10^0
F2	Mean	1.810×10^{-51}	2.917×10^{-67}	6.176×10^{-136}	0.000×10^0	5.771×10^{-223}	2.616×10^{-227}
	Std	4.928×10^{-51}	1.224×10^{-66}	1.830×10^{-135}	0.000×10^0	0.000×10^0	0.000×10^0
	Best	3.145×10^{-57}	7.767×10^{-75}	5.150×10^{-150}	0.000×10^0	5.765×10^{-255}	1.203×10^{-270}
	Worst	2.668×10^{-50}	5.625×10^{-66}	8.052×10^{-135}	0.000×10^0	1.076×10^{-221}	7.847×10^{-226}
F3	Mean	4.390×10^4	2.893×10^4	7.764×10^{-179}	0.000×10^0	0.000×10^0	0.000×10^0
	Std	1.312×10^4	1.488×10^4	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0
	Best	1.659×10^4	1.385×10^3	3.409×10^{-217}	0.000×10^0	0.000×10^0	0.000×10^0
	Worst	7.255×10^4	5.585×10^4	1.553×10^{-177}	0.000×10^0	0.000×10^0	0.000×10^0
F4	Mean	4.861×10^1	8.027×10^1	3.220×10^{-108}	0.000×10^0	5.683×10^{-230}	1.586×10^{-233}
	Std	2.601×10^1	1.516×10^1	1.403×10^{-107}	0.000×10^0	0.000×10^0	0.000×10^0
	Best	5.627×10^0	3.321×10^1	4.010×10^{-120}	0.000×10^0	1.100×10^{-246}	4.056×10^{-277}
	Worst	8.861×10^1	9.271×10^1	6.438×10^{-107}	0.000×10^0	1.108×10^{-228}	4.130×10^{-232}
F5	Mean	2.803×10^1	2.879×10^1	2.877×10^1	2.877×10^1	4.630×10^{-2}	5.944×10^{-3}
	Std	3.640×10^{-1}	2.571×10^{-2}	2.166×10^{-2}	2.244×10^{-2}	6.285×10^{-2}	6.671×10^{-3}
	Best	2.749×10^1	2.874×10^1	2.873×10^1	2.871×10^1	7.328×10^{-6}	2.331×10^{-7}
	Worst	2.876×10^1	2.885×10^1	2.880×10^1	2.882×10^1	2.307×10^{-1}	2.815×10^{-2}
F6	Mean	3.828×10^{-1}	2.477×10^0	7.878×10^{-1}	6.386×10^{-1}	4.702×10^{-3}	2.553×10^{-3}
	Std	2.616×10^{-1}	3.087×10^{-1}	3.985×10^{-1}	2.817×10^{-1}	2.611×10^{-3}	3.258×10^{-3}
	Best	8.880×10^{-2}	1.941×10^0	1.840×10^{-1}	7.272×10^{-2}	1.170×10^{-4}	9.584×10^{-6}
	Worst	1.169×10^0	2.965×10^0	1.765×10^0	9.993×10^{-1}	7.972×10^{-3}	1.484×10^{-2}
F7	Mean	4.958×10^{-3}	2.235×10^{-3}	7.719×10^{-5}	6.720×10^{-5}	3.636×10^{-4}	2.713×10^{-5}
	Std	6.545×10^{-3}	1.988×10^{-3}	5.770×10^{-5}	5.441×10^{-5}	3.005×10^{-4}	2.563×10^{-5}
	Best	4.967×10^{-5}	1.018×10^{-4}	8.626×10^{-6}	7.830×10^{-8}	3.213×10^{-5}	1.628×10^{-6}
	Worst	3.094×10^{-2}	6.387×10^{-3}	1.787×10^{-4}	1.910×10^{-4}	9.762×10^{-4}	8.928×10^{-5}
F8	Mean	-1.044×10^4	-6.838×10^3	-1.183×10^4	-1.257×10^4	-1.257×10^4	-1.257×10^4
	Std	1.745×10^3	1.988×10^2	7.802×10^2	1.192×10^{-1}	9.278×10^{-2}	1.932×10^{-1}
	Best	-1.257×10^4	-1.184×10^4	-1.257×10^4	-1.257×10^4	-1.257×10^4	-1.257×10^4
	Worst	-6.624×10^3	-3.409×10^3	-1.033×10^4	-1.257×10^4	-1.257×10^4	-1.257×10^4
F9	Mean	2.901×10^{-15}	3.908×10^{-15}	8.882×10^{-16}	8.882×10^{-16}	8.882×10^{-16}	8.882×10^{-16}
	Std	2.543×10^{-15}	2.814×10^{-15}	0.000×10^0	0.000×10^0	0.000×10^0	0.000×10^0
	Best	8.882×10^{-16}	8.882×10^{-16}	8.882×10^{-16}	8.882×10^{-16}	8.882×10^{-16}	8.882×10^{-16}
	Worst	7.994×10^{-15}	7.994×10^{-15}	8.882×10^{-16}	8.882×10^{-16}	8.882×10^{-16}	8.882×10^{-16}
F10	Mean	2.246×10^{-2}	3.384×10^{-1}	1.155×10^{-1}	3.522×10^{-2}	3.009×10^{-5}	4.793×10^{-6}
	Std	1.458×10^{-2}	2.090×10^{-1}	8.382×10^{-2}	2.210×10^{-2}	2.730×10^{-5}	6.325×10^{-6}
	Best	5.130×10^{-3}	9.040×10^{-2}	1.430×10^{-2}	1.202×10^{-2}	9.999×10^{-7}	7.593×10^{-8}
	Worst	5.779×10^{-2}	6.623×10^{-1}	3.012×10^{-1}	8.957×10^{-2}	9.420×10^{-5}	2.805×10^{-5}
F11	Mean	6.729×10^{-1}	1.581×10^0	5.807×10^{-1}	4.446×10^{-1}	2.019×10^{-4}	8.990×10^{-5}
	Std	2.858×10^{-1}	5.461×10^{-1}	3.515×10^{-1}	2.270×10^{-1}	2.366×10^{-4}	1.910×10^{-4}
	Best	1.035×10^{-1}	6.989×10^{-1}	9.693×10^{-2}	7.115×10^{-2}	1.752×10^{-6}	1.175×10^{-7}
	Worst	1.254×10^0	2.590×10^0	1.336×10^0	9.332×10^{-1}	7.285×10^{-4}	9.061×10^{-4}
F12	Mean	2.278×10^0	7.246×10^0	2.256×10^0	2.977×10^0	9.980×10^{-1}	9.980×10^{-1}
	Std	2.448×10^0	4.901×10^0	7.676×10^{-1}	4.936×10^{-1}	0.000×10^0	0.000×10^0
	Best	9.980×10^{-1}	9.983×10^{-1}	9.980×10^{-1}	9.980×10^{-1}	9.980×10^{-1}	9.980×10^{-1}
	Worst	1.076×10^1	1.550×10^1	3.093×10^0	3.835×10^0	9.980×10^{-1}	9.980×10^{-1}

Table 5. Cont.

Function	Index	WOA	WOA-1	WOA-2	WOA-3	WOA-4	GWOA
F13	Mean	3.000×10^0	3.000×10^0	3.000×10^0	3.000×10^0	3.569×10^0	3.014×10^0
	Std	8.684×10^{-5}	8.360×10^{-3}	1.175×10^{-3}	1.326×10^{-4}	7.881×10^{-1}	5.108×10^{-2}
	Best	3.000×10^0	3.000×10^0	3.000×10^0	3.000×10^0	3.000×10^0	3.000×10^0
	Worst	3.000×10^0	3.038×10^0	3.004×10^0	3.000×10^0	5.842×10^0	3.275×10^0
F14	Mean	-8.278×10^0	-7.556×10^0	-9.327×10^0	-9.599×10^0	-1.015×10^1	-1.015×10^1
	Std	2.681×10^0	1.364×10^0	9.704×10^{-1}	6.138×10^{-1}	1.130×10^{-5}	5.371×10^{-5}
	Best	-1.015×10^1	-1.040×10^1	-1.040×10^1	-1.015×10^1	-1.015×10^1	-1.015×10^1
	Worst	-2.627×10^0	-4.287×10^0	-6.550×10^0	-8.635×10^0	-1.015×10^1	-1.015×10^1
F15	Mean	-7.213×10^0	-6.511×10^1	-9.591×10^1	-9.885×10^0	-1.040×10^1	-1.040×10^1
	Std	3.015×10^0	1.996×10^0	1.091×10^0	2.506×10^0	1.078×10^{-5}	8.550×10^{-5}
	Best	-1.040×10^1	-1.040×10^1	-1.040×10^1	-1.974×10^1	-1.040×10^1	-1.040×10^1
	Worst	-2.766×10^0	-4.187×10^0	-7.032×10^0	-5.053×10^0	-1.040×10^1	-1.040×10^1
F16	Mean	-7.214×10^0	-7.377×10^0	-10.09×10^1	-9.949×10^0	-1.054×10^1	-1.054×10^1
	Std	3.206×10^0	2.069×10^0	7.744×10^0	7.981×10^{-1}	9.72×10^{-5}	9.409×10^{-5}
	Best	-1.054×10^1	-1.054×10^1	-1.054×10^1	-1.054×10^1	-1.054×10^1	-1.054×10^1
	Worst	-1.859×10^0	-3.405×10^0	-7.107×10^0	-8.468×10^0	-1.054×10^1	-1.054×10^1
Optimal number of indicators		1	1	2	7	8	13

From the analysis in Table 5, it can be seen that the GWOA performs well in all 16 test functions, performs average among 14 functions, and slightly worse in the F2, F4, and F13 functions. However, the GWOA can achieve higher accuracy in F2 and F4 functions. Through comparative analysis of WOA-1, WOA-2, and the WOA, it is found that WOA-1, with only the addition of nonlinear time-varying factor strategy, has better accuracy and variance in solutions than the WOA only in F1, F2 and F7 functions. WOA-2, with the addition of nonlinear time-varying factor and inertia weight, has more F2, F3, F4, F7, F8, F9, F11, F14, F15, and F16 functions than WOA-1's leading functions, indicating that the improvement strategy indeed has good effects; However, the WOA algorithm with inertia weight strategy is not good enough in terms of convergence speed. In the analysis of WOA-3 and WOA, it was found that WOA-3 has an additional lead in the F12 function ahead of WOA-2, but there is a significant improvement in the accuracy and standard deviation of the lead function solution compared to WOA-2 and the WOA. Although good results were obtained, there is still a problem of falling into local optima for F5 or other complex functions. At the same time, we can know that the full weight has a delaying effect on the convergence rate of the algorithm, and also that the accuracy of the algorithm is improved. This proves that the gravity balance strategy added on top of WOA-2 improves the algorithm's development ability and stability compared with the WOA. WOA-4 shows that only F13 is slightly worse than the WOA on the basis of WOA-2, but the indicators under the other 15 functions are far better than for the WOA. From this, it can be seen that WOA-2, which integrates the regeneration mechanism, the exploration and development ability of the algorithm is retained, and the ability of the algorithm to escape local optima is also improved, which improves the accuracy of understanding and enhances the stability of the algorithm.

The average and standard deviation indicators cannot comprehensively evaluate the overall performance of the GWOA. Therefore, by incorporating hypothesis-testing experiments, the Wilcoxon rank sum test has the ability to evaluate algorithm similarity. This article uses the above method to validate the GWOA and determine the differences between the GWOA and the six mentioned algorithms. The significance level was set to 5%, and the test results are shown in Table 6. Those with high algorithm similarity are represented by NaN, while those that cannot be compared are replaced with "-". The final

data analysis showed significant differences between the GWOA and various algorithms, further verifying the uniqueness of the GWOA.

Table 6. Wilcoxon rank sum test results.

Algorithm	DECWOA	WOA	GWO	PSO	HHO	MPA
F1	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}
F2	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}
F3	4.56×10^{-11}	1.21×10^{-12}	1.21×10^{-12}	2.36×10^{-12}	1.21×10^{-12}	1.21×10^{-12}
F4	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}
F5	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	6.76×10^{-5}	3.01×10^{-11}
F6	3.01×10^{-11}	3.01×10^{-11}	5.07×10^{-10}	2.31×10^{-6}	2.83×10^{-4}	3.15×10^{-10}
F7	2.12×10^{-4}	1.15×10^{-7}	7.77×10^{-9}	3.01×10^{-11}	2.68×10^{-4}	5.53×10^{-8}
F8	2.12×10^{-4}	3.68×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	5.57×10^{-10}	3.01×10^{-11}
F9	4.29×10^{-10}	7.46×10^{-7}	1.10×10^{-12}	1.21×10^{-12}	5.34×10^{-6}	1.19×10^{-12}
F10	2.12×10^{-4}	3.01×10^{-11}	3.01×10^{-11}	2.38×10^{-8}	6.51×10^{-9}	3.01×10^{-11}
F11	3.01×10^{-11}	3.01×10^{-11}	2.15×10^{-6}	1.59×10^{-3}	2.19×10^{-7}	3.68×10^{-11}
F12	3.01×10^{-11}	1.10×10^{-3}	9.03×10^{-4}	6.67×10^{-5}	2.19×10^{-7}	1.36×10^{-11}
F13	9.21×10^{-5}	4.80×10^{-7}	6.73×10^{-9}	2.66×10^{-11}	4.19×10^{-10}	2.57×10^{-11}
F14	3.33×10^{-11}	2.87×10^{-10}	3.52×10^{-7}	1.88×10^{-3}	3.01×10^{-11}	3.01×10^{-11}
F15	3.01×10^{-11}	6.06×10^{-11}	5.46×10^{-6}	3.54×10^{-4}	3.01×10^{-11}	3.01×10^{-11}
F16	3.01×10^{-11}	7.38×10^{-11}	2.00×10^{-6}	3.47×10^{-4}	3.01×10^{-11}	3.01×10^{-11}

Finally, in order to verify the ability of the algorithm for global functions, we used the Friedman test of the average value of 30 simulation results obtained by seven algorithms [34], and the results were recorded in Table 7. As can be seen, GWOA ranks first, with 1.8125 in 30 dimensions and 1.0909 in 100 dimensions. Meanwhile, by consulting the table [35], for $k = 7$, the number of problems N is 16. When $\alpha = 0.05$, $Q_\alpha = 2.949$. The critical value $CD = 2.252$ is calculated according to $CD = Q_\alpha \sqrt{k(k+1)/6N}$. The post-test CD line is the ranking plus the critical value $CD = 4.0645$. Therefore, it can be concluded that the GWOA is significantly superior to DECWOA, GWO, PSO and the WOA in 30 dimensions. In 100 dimensions, there are significant differences between GWO, MPA, PSO and the WOA.

Table 7. Friedman statistical results.

30—dim							
Algorithm	DECWOA	GWO	HHO	MPA	PSO	WOA	GWOA
Rank	4.1875	4.6875	3.9063	3.7188	5.1250	4.5625	1.8125
100—dim							
Algorithm	DECWOA	GWO	HHO	MPA	PSO	WOA	GWOA
Rank	2.7727	5.3636	2.8636	5.1818	6.6364	4.0909	1.0909

5. Engineering Application Experiment Based on the GWOA

The GWOA shows good performance on the test functions. However, the ultimate purpose of the algorithm is to solve the actual optimization problem. Compared with the test function, the optimization problem in the real world has certain constraints and is more complex, which is more challenging for the GWOA. Meanwhile, the optimization

problem in the real world is also used as an evaluation index to measure the comprehensive performance of the algorithm.

5.1. Robot Path Planning Experiment

There have been many previous experiments using swarm intelligence algorithms and robot path planning [36,37]. Path planning refers to a new technology for robots to independently achieve non-artificial navigation. In experimental environments with obstacles, based on evaluation criteria such as distance, time, energy consumption, and inflection points, the optimal collision-free path is found. This article selects the shortest path and inflection point as evaluation indicators. Therefore, on this basis, this article uses the GWOA to optimize the robot path-planning project, with the aim of verifying the applicability and feasibility of the GWOA. Therefore, the grid method [38] was used to simulate the robot's walking path. A set of data was used to construct an abstract two-dimensional or three-dimensional experimental environment, with 0 and 1 representing the feasible and obstacle areas. The environmental information stored in each grid of the same type is the same, and the robot can plan its walking path in an area with a value of 0. The robot has eight options for moving in the current node direction; this is called octree search. The search direction diagram is shown in Figure 5. White represents possible routes and black represents obstacles. The orange arrow represents the direction of travel.

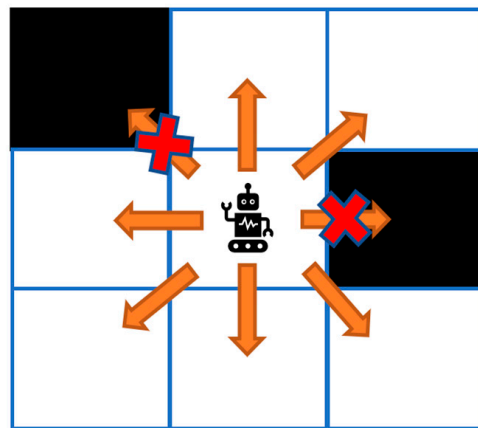


Figure 5. Octree search diagram.

Its model is:

$$f(x_i) = \sum_{j=1}^{D-1} \sqrt{(x_{j+1} - x_j)^2 + (y_{j+1} - y_j)^2} \quad (20)$$

j is the dimension of the current whale. $x_j = a \times [\text{mod}(j, y) - a/2]$, $y_j = a \times [x + a/2 - \text{ceil}(j/x)]$. The experimental environment is a two-dimensional grid map with x rows and y columns, where a is the pixel edge length of the grid map's 1×1 grid, and $\text{ceil}(n)$ is the smallest integer greater than or equal to n .

In order to verify the feasibility and applicability of the improved GWOA, the improved GWOA was compared with the WOA, DECWOA and PSO algorithms. The four algorithms were set with the same parameters, the total number was 30, the maximum number of iterations was 100 and the same plane grid map was used. The optimal path planning image of each algorithm is shown in Figure 6, where the red dot represents the beginning and the green dot represents the end. The performance indicators of each algorithm are shown in Table 8:

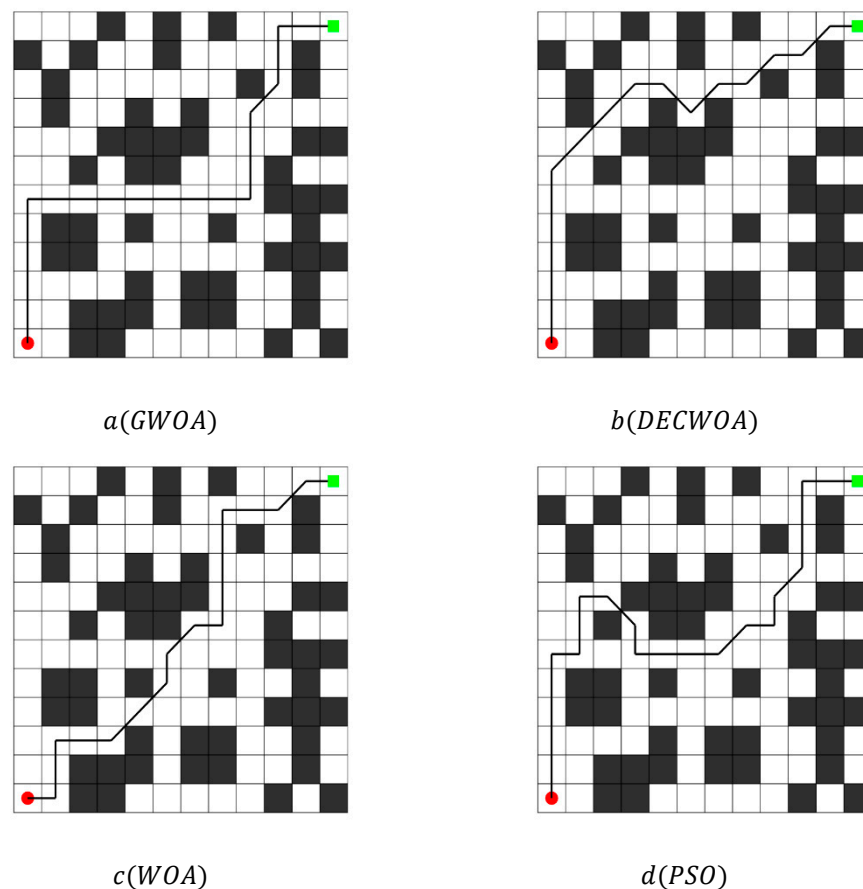


Figure 6. Optimal path planning diagram of each calculation.

Table 8. Performance indicators of each algorithm.

Map Size	Index	GWOA	DECWOA	WOA	PSO
12 × 12	Shortest	15.5563	15.5563	18.3848	21.2132
	Worse	21.2132	24.0416	26.8701	29.6985
	Average	16.4402	18.0312	21.6552	24.6604
	Inflection points	7.3750	8.2188	9.4063	11.0625
Rank		1	2	3	4

In order to reduce the impact of chance among algorithms, 30 simulation tests were carried out for each test algorithm, and the optimal value, worst value, average value and number of inflection points of the path were recorded. According to the analysis of Table 8, the four indexes of the GWOA have the best effect, followed by DECWOA. Obviously, the GWOA is better than the WOA and PSO in terms of search ability and stability. The number of inflection points represents that the robot needs to spend more energy on path planning; the GWOA has the least number of inflection points, which proves the GWOA's optimization ability and better robustness.

5.2. Engineering Optimization

Engineering optimization problems have been studied by a large number of scholars. Therefore, in order to improve the efficiency of solving engineering optimization problems and reduce the cost of optimization, many scholars have tried various methods, such as the traditional gradient descent method and Newton method to solve engineering problems, and optimize and solve more complex engineering problems according to swarm intelligence algorithms, so as to improve efficiency and accuracy. In this paper, the improved whale optimization algorithm is applied to three classical engineering optimization prob-

lems through simulation experiments. The simulation platform was MATLAB 2020b. Each algorithm was independently run 30 times in the corresponding engineering problems, and the optimal value was obtained and recorded in Tables 9–11.

Table 9. Optimal results of welding beam engineering problems.

Algorithm	x_1	x_2	x_3	x_4	$f(x)$	Rank
GWOA	0.17362529	4.2017926	9.54738792	0.205705	1.724852323	2
WOA	0.20234729	3.5835421	9.03836035	0.387305	1.735021793	7
DECWOA	0.20572964	3.4704887	9.03662391	0.205738	1.724852309	1
GWO	0.20565067	3.4722534	9.03682293	0.205740	1.725003904	3
PSO	0.21125253	3.4656346	8.76782271	0.218537	1.755888165	6
HHO	0.20523806	3.4809689	9.03733442	0.205739	1.725132082	5
MPA	0.20572094	3.4715304	9.03703657	0.205728	1.725018576	4

Table 10. Optimization results of pressure vessel problems.

Algorithm	x_1	x_2	x_3	x_4	$f(x)$	Rank
GWOA	0.78662974	0.4011552	40.6259534	195.7791	5953.705854	1
WOA	0.77880150	0.4566316	40.3196388	199.9997	6061.269412	5
DECWOA	0.83717191	0.4480266	43.2230683	163.1819	6021.244209	2
GWO	0.93334071	0.4543938	47.6220701	118.4064	6044.889025	4
PSO	0.78547815	0.4378789	40.5516213	196.7954	6062.606402	6
HHO	0.83574701	0.4039123	42.1031221	176.5787	6107.918295	7
MPA	0.78668547	0.4236772	40.5305527	197.0843	6032.512874	3

Table 11. Tensile and compression spring engineering design results.

Algorithm	x_1	x_2	x_3	$f(x)$	Rank
GWOA	0.05143415	0.3506162	11.65585649	0.012665251	3
WOA	0.06327765	0.7048988	3.285910365	0.012686305	7
DECWOA	0.05171397	0.3573276	11.25330213	0.012665244	2
GWO	0.05605708	0.4711903	6.775892624	0.012667203	5
PSO	0.05534713	0.4512736	7.329839403	0.012667690	6
HHO	0.05742952	0.5111917	5.845519837	0.012666156	4
MPA	0.05420961	0.4204409	8.341112809	0.012665243	1

5.2.1. Subsubsection

This project optimization problem was proposed by Coello [39], and has been solved through research. The problem is that the beam is subject to vertical gravity, creating a beam balance effect in the groove, which ultimately minimizes the cost of welding the beam in the end. The engineering problem is subject to seven constraints, including the pressure of the wall on the beam, the disturbance, the welding of the head and the geometry. The variable x_1 is the thickness of the welded beam, x_2 is the height of the welded beam, the length is x_3 and the thickness of the steel bar is x_4 . The engineering drawing comes from the literature [3], the engineering model is shown in Figure 7, and its objective function can be expressed as [3]:

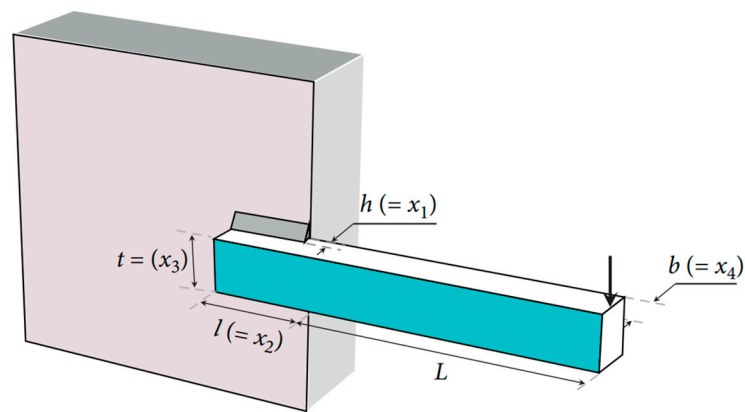


Figure 7. Schematic diagram of welded beam structure [3].

Minimize:

$$f(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2) \quad (21)$$

Objective:

$$g_1(x) = \tau(x) - \tau_{max} \leq 0 \quad (22)$$

$$g_2(x) = \sigma(x) - \sigma_{max} \leq 0$$

$$g_3(x) = \delta(x) - \delta_{max} \leq 0$$

$$g_4(x) = x_1 - x_4 \leq 0$$

$$g_5(x) = p - p_c(x) \leq 0$$

$$g_6(x) = 0.125 - x_1 \leq 0$$

$$g_7(x) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2) - 5 \leq 0$$

$$\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}$$

$$\tau' = \frac{P}{\sqrt{2}x_1x_2}$$

$$\tau'' = \frac{MR}{J}$$

$$M = P\left(L + \frac{x_2}{2}\right)$$

$$R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}$$

$$J = 2\sqrt{2}x_1x_2R^2$$

$$\sigma(\vec{x}) = \frac{6PL}{x_4x_3^2}$$

$$\delta(\vec{x}) = \frac{6PL^3}{Ex_4x_3^2}$$

$$P_c(\vec{x}) = \frac{4.013E\sqrt{x_3^2x_4^6/36}}{L^2} \left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)$$

$$P = 6000lb$$

$$L = 14in$$

$$\delta_{max} = 0.25in$$

$$E = 30 \times 10^6 psi$$

$$G = 12 \times 10^6 psi$$

$$\tau_{max} = 13600 psi$$

$$\sigma_{max} = 30000 psi$$

Range of variables:

$$0.1 \leq x_1 \quad (23)$$

$$x_4 \leq 2$$

$$0.1 \leq x_2$$

$$x_3 \leq 10$$

5.2.2. Pressure Vessel Design

The pressure vessel engineering model, which consists of a cylindrical vessel and two hemispherical heads, is shown below [3]. The engineering problem is to minimize the cost of materials, molding and welding. The design problem of the pressure vessel is restricted by four constraints; including the thickness of the cylindrical shell $T_s(x_1)$, the thickness of the cover $T_h(x_2)$, the inner diameter of the head, the cylindrical shell $R(x_3)$ and the section length of the cylindrical shell $L(x_4)$. Note that x_1 and x_2 must be multiples of 0.0625 inches, and that the other two variables, x_3 and x_4 , are continuous. The engineering model is shown in Figure 8. Its objective function can be expressed as [3]:

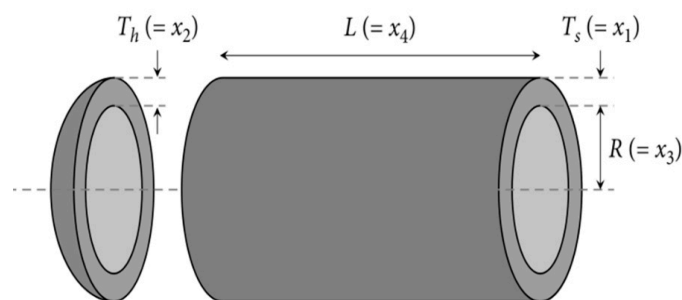


Figure 8. Schematic diagram of pressure vessel structure [3].

Minimize:

$$f(x) = 0.6224x_1x_2x_3 + 1.7781x_2x_3^2 + 3.1661x_4x_1^2 + 19.84x_3x_1^2 \quad (24)$$

Objective:

$$\begin{aligned} g_1(x) &= -x_1 + 0.0193x_3 \leq 0 \\ g_2(x) &= -x_2 + 0.00954x_3 \leq 0 \\ g_3(x) &= -\pi x_3^2 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0 \\ g_4(x) &= x_4 - 240 \leq 0 \end{aligned} \quad (25)$$

Range of variables:

$$\begin{aligned} x_1, x_2 &\in \{1 \times 0.0625, 2 \times 0.0625, 3 \times 0.0625, \dots, 1600 \times 0.0625\} \\ 10 &\leq x_3 \\ x_4 &\leq 200 \end{aligned} \quad (26)$$

5.2.3. Tensile and Compression Spring Engineering Design

The purpose of tensile compression spring design engineering [40] is to minimize the weight of the tensile spring under a variety of constraints. The design problem is mainly affected by the winding degree of the tensile spring, the unit area of the tensile spring section under pressure (shear stress), the tensile length, the average diameter of the original coil thickness of the tensile spring and the average diameter of the original coil D . Therefore, the relative stress variable is set as the spring thickness x_2 , the average diameter of the section surface of the original spring coil is x_1 and the number of stretching spring coils is x_3 . Its engineering drawing is taken from the literature [3]. The engineering design problem model is shown in Figure 9 [3]:

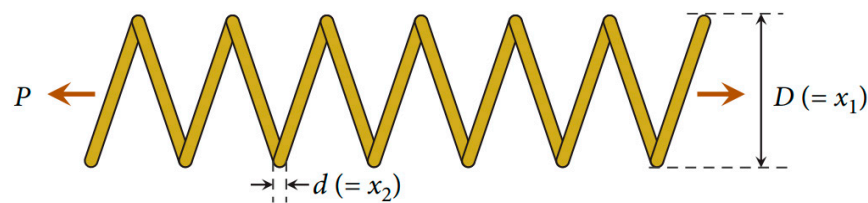


Figure 9. Tension/compression spring diagram [3].

Minimize:

$$f(x) = (x_3 + 2)x_2x_1^2 \quad (27)$$

Objective:

$$\begin{aligned} g_1(x) &= 1 - \frac{x_2^3x_3}{71785x_1^4} \\ g_2(x) &= \frac{4x_2^2 - x_1x_2}{12566(x_1^3x_2 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \\ g_3(x) &= 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0 \\ g_4(x) &= \frac{x_1 + x_2}{1.5} - 1 \leq 0 \end{aligned} \quad (28)$$

The experimental results of the welding beam engineering design problem in Table 8 are second only to DECWOA, and the value is only 10×10^{-8} , with a very small gap. In Table 9, the results of the pressure vessel design experiment rank first, and the optimal value obtained by the improved algorithm is the smallest among the comparison algorithms; that is, the cost is the smallest. In Table 10 showing the tensile and compression spring engineering design problems, it ranks the third, but it has the same accuracy as the first

two, and the difference in value is only 10×10^{-7} , but the difference in magnitude is small. To sum up, the improved whale optimization algorithm has a certain competitiveness and universal applicability in engineering applications, and also has good performance in terms of optimization accuracy and optimization effect.

6. Conclusions

In this study, a whale optimization algorithm with improved gravity balance (GWOA) is proposed. The GWOA aims to improve the development and exploration ability in the optimization process, and make up for the fact that the WOA is prone to become trapped in local extremum stagnation in the late iterations. The GWOA was tested on 16 benchmark functions, and the experimental results were compared with six intelligent algorithms. Finally, the GWOA was applied to robot path planning and three classical engineering optimization problems, and experimental simulation results and test comparison data were obtained. The final analysis results are as follows:

- The GWOA has better convergence accuracy and optimization results compared to the comparison algorithms, and can jump out of the local extremum when the original whale optimization algorithm falls into a local optimal function. Among the 16 functions tested in 30 dimensions, it ranked first for 12; among the 11 functions tested in 100 dimensions, it ranked first for 10.
- The GWOA has good convergence performance, incorporating nonlinear time-varying factors and inertia weights to balance the development and exploration capabilities of the WOA. To enhance the selection of the best among the population, a gravitational balance strategy was added to protect the excellent solution while increasing the trend of inferior solutions approaching the excellent solution; finally, considering the contribution of whale death to the population, a rebirth mechanism was added to help the whale algorithm jump out of local stagnation.
- The contribution experiment shows that, when the balance strategy and regeneration mechanism are added under the strategy of nonlinear time-varying factors and inertia weights, the whale optimization algorithm with balance strategy ranks first on seven functions, and the whale optimization algorithm with regeneration mechanism ranks first on eight functions. There is a performance improvement compared to the original whale algorithm; the GWOA can achieve a lead in 13 functions, but there are still shortcomings in three functions that need improvement.
- In the robot path planning experiment, the comprehensive data ranked first, and in the three classic engineering problems, the ranking was 1, 2, and 3, respectively. The overall solution accuracy was slightly different from the optimal algorithm.
- Overall, the GWOA has a good optimization performance and good robustness, and exhibits a certain uniqueness in rank sum testing. However, its optimization performance is not good enough in some functions (such as the F13 function), and its convergence speed is not fast enough in the F2, F4, F5, and F6 functions.

The nonlinear time-varying factor and inertial weight strategy were introduced to balance the development and exploration ability of the whale optimization algorithm. A novel gravity balance strategy was proposed in the whale search and encircling phases, which can effectively avoid the excellent solution from the inferior one in the random walk process, thus improving the quality of the solution. Adding a respawning mechanism at the end of the iteration helps the algorithm jump out of the local optimal. Next, the improved whale optimization algorithm was compared on 16 benchmark test functions and six comparison algorithms, and it is concluded that the overall performance of the improved GWOA is the best among the comparison algorithms. Finally, the improved algorithm was applied to robot path planning to verify the feasibility and applicability of the algorithm. The results show that the improved algorithm has a clear path and the least cost. It can be seen that the introduction of multiple strategies improves the accuracy and stability of the basic whale optimization algorithm.

However, in some functions, the GWOA is slightly less accurate than the comparison algorithm. Therefore, in the next stage, the aim will be to comprehensively improve the accuracy of the solution, and more complex applications will be sought as the evaluation criteria of the algorithm in the next stage.

Author Contributions: Conceptualization, C.O. and Y.G.; methodology, C.O. and Y.G.; software, Y.G.; validation, Y.G. and D.Z.; formal analysis, Y.G. and D.Z.; investigation, Y.G. and D.Z.; resources, C.O. and D.Z.; data curation, Y.G. and D.Z.; writing original draft preparation, Y.G.; writing review and editing, C.O. and D.Z.; visualization, Y.G. and D.Z.; supervision, C.O. and C.Z.; project administration, C.Z.; funding acquisition, C.O. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China (Grant No.62272418 and 62002046).

Data Availability Statement: All data for this study are available from the corresponding author.

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

References

- Wang, R.; Feng, Y. Evaluation research on green degree of equipment manufacturing industry based on improved particle swarm optimization algorithm. *Chaos Solitons Fractals Interdiscip. J. Nonlinear Sci. Nonequilibrium Complex Phenom.* **2020**, *131*, 109502. [\[CrossRef\]](#)
- Abualigah, L.; Elaziz, M.A.; Khasawneh, A.M.; Alshinwan, M.; Ibrahim, R.A.; Al-Qaness, M.A.; Mirjalili, S.; Sumari, P.; Gandomi, A.H. Meta-heuristic optimization algorithms for solving real-world mechanical engineering design problems: A comprehensive survey, applications, comparative analysis, and results. *Neural Comput. Appl.* **2022**; *prepublish.* [\[CrossRef\]](#)
- Bayzidi, H.; Talatahari, S.; Saraee, M.; Lamarche, C.P. Social Network Search for Solving Engineering Optimization Problems. *Comput. Intell. Neurosci.* **2021**, *2021*, 32. [\[CrossRef\]](#)
- Du, Y.; Yang, N. Analysis of image processing algorithm based on bionic intelligent optimization. *Clust. Comput.* **2018**, *22*, 3505–3512. [\[CrossRef\]](#)
- Chengtian, O.; Yaxian, Q.; Donglin, Z. Adaptive Spiral Flying Sparrow Search Algorithm. *Sci. Program.* **2021**, *2021*. [\[CrossRef\]](#)
- Rashid, A.S.; Mohamed, O.; Khalek, S.A.; Sayed, A.E.; Faihan, A.M. Optimal path planning for drones based on swarm intelligence algorithm. *Neural Comput. Appl.* **2022**, *34*, 10133–10155.
- Zhu, D.; Huang, Z.; Liao, S.; Zhou, C.; Yan, S.; Chen, G. Improved Bare Bones Particle Swarm Optimization for DNA Sequence Design. *IEEE Trans. NanoBioscience* **2022**, 603–613. [\[CrossRef\]](#)
- Engy, E.-S.; Sallam, K.M.; Chakraborty, R.K.; Abohamy, A.A. A clustering based Swarm Intelligence optimization technique for the Internet of Medical Things. *Expert Syst. Appl.* **2021**, *173*, 114648.
- Mojgan, S.; FathollahiFard, A.M.; Kamyar, K.; Maziar, Y.; Mohammad, S. Selecting Appropriate Risk Response Strategies Considering Utility Function and Budget Constraints: A Case Study of a Construction Company in Iran. *Buildings* **2022**, *12*, 98.
- Eberhart, R.; Kennedy, J. A new optimizer using particle swarm theory. In Proceedings of the Mhs95 Sixth International Symposium on Micro Machine & Human Science, Nagoya, Japan, 4–6 October 1995.
- Dorigo, M. The ant system: An autocatalytic optimizing process. In Proceedings of the First European Conference on Artificial Life, Paris, France, 11–13 December 1991.
- Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [\[CrossRef\]](#)
- Fathollahi-Fard, A.M.; Hajiaghahi-Keshteli, M.; Tavakkoli-Moghaddam, R. The Social Engineering Optimizer (SEO). *Eng. Appl. Artif. Intell.* **2018**, *72*, 267–293. [\[CrossRef\]](#)
- Xue, J.; Shen, B. A novel swarm intelligence optimization approach: Sparrow search algorithm. *Syst. Sci. Control. Eng.* **2020**, *8*, 22–34. [\[CrossRef\]](#)
- Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [\[CrossRef\]](#)
- Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **1997**, *1*, 67–82. [\[CrossRef\]](#)
- Xiong, G.; Zhang, J.; Shi, D.; He, Y. Parameter extraction of solar photovoltaic models using an improved whale optimization algorithm. *Energy Convers. Manag.* **2018**, *174*, 388–405. [\[CrossRef\]](#)
- Li, S.H.; Luo, X.H.; Wu, L.Z. An improved whale optimization algorithm for locating critical slip surface of slopes. *Adv. Eng. Softw.* **2021**, *157*, 103009. [\[CrossRef\]](#)
- Nadimi-Shahraki, M.H.; Zamani, H.; Mirjalili, S. Enhanced whale optimization algorithm for medical feature selection: A COVID-19 case study. *Comput. Biol. Med.* **2022**, *148*, 105858. [\[CrossRef\]](#)
- Yang, K.; Yang, K. Improved Whale Algorithm for Economic Load Dispatch Problem in Hydropower Plants and Comprehensive Performance Evaluation. *Water Resour. Manag.* **2022**, *36*, 5823–5838. [\[CrossRef\]](#)
- Shen, Y.; Zhang, C.; Gharehchopogh, F.S.; Mirjalili, S. An improved whale optimization algorithm based on multi-population evolution for global optimization and engineering design problems. *Expert Syst. Appl.* **2023**, *215*, 119269. [\[CrossRef\]](#)

22. Guo, W.; Liu, T.; Dai, F.; Xu, P. An improved whale optimization algorithm for forecasting water resources demand. *Appl. Soft Comput.* **2020**, *86*, 105925. [\[CrossRef\]](#)
23. Oliva, D.; Abd El Aziz, M.; Hassanien, A.E. Parameter estimation of photovoltaic cells using an improved chaotic whale optimization algorithm. *Appl. Energy* **2017**, *200*, 141–154. [\[CrossRef\]](#)
24. Ning, G.Y.; Cao, D.Q. Improved whale optimization algorithm for solving constrained optimization problems. *Discret. Dyn. Nat. Soc.* **2021**, *2021*, 1–13. [\[CrossRef\]](#)
25. Zhang, J.; Wang, J.S. Improved whale optimization algorithm based on nonlinear adaptive weight and golden sine operator. *IEEE Access* **2020**, *8*, 77013–77048. [\[CrossRef\]](#)
26. Yang, W.; Xia, K.; Fan, S.; Wang, L.; Li, T.; Zhang, J.; Feng, Y. A multi-strategy Whale optimization algorithm and its application. *Eng. Appl. Artif. Intell.* **2022**, *108*, 104558. [\[CrossRef\]](#)
27. Li, Y.; Han, M.; Guo, Q. Modified whale optimization algorithm based on tent chaotic mapping and its application in structural optimization. *KSCE J. Civ. Eng.* **2020**, *24*, 3703–3713. [\[CrossRef\]](#)
28. Jiang, T.; Zhang, C.; Zhu, H.; Gu, J.; Deng, G. Energy-efficient scheduling for a job shop using an improved whale optimization algorithm. *Mathematics* **2018**, *6*, 220. [\[CrossRef\]](#)
29. Howard, A.; Mataric, M.; Sukhatme, G.S. Mobile sensor net-work potential field: A distributed scalable solution to the area deployment using coverage problem. *Distrib. Auton. Robot. Syst.* **2002**, 299–308. [\[CrossRef\]](#)
30. Liu, L.; Zhang, R. Multistrategy Improved Whale Optimization Algorithm and Its Application. *Comput. Intell. Neurosci.* **2022**, 2022. [\[CrossRef\]](#) [\[PubMed\]](#)
31. Faramarzi, A.; Heidarinejad, M.; Mirjalili, S.; Gandomi, A.H. Marine predators algorithm: A nature-inspired metaheuristic. *Expert Syst. Appl.* **2020**, *152*, 113377. [\[CrossRef\]](#)
32. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H. Harris hawks optimization: Algorithm and applications. *Future Gener. Comput. Syst.* **2019**, *97*, 849–872. [\[CrossRef\]](#)
33. Ravber, M.; Liu, S.H.; Mernik, M.; Črepinšek, M. Maximum number of generations as a stopping criterion considered harmful. *Appl. Soft Comput.* **2022**, *128*, 109478. [\[CrossRef\]](#)
34. Demšar, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.
35. Veček, N.; Črepinšek, M.; Mernik, M. On the influence of the number of algorithms, problems, and independent runs in the comparison of evolutionary algorithms. *Appl. Soft Comput.* **2017**, *54*, 23–45. [\[CrossRef\]](#)
36. Akka, K.; Khaber, F. Mobile robot path planning using an improved ant colony optimization. *Int. J. Adv. Robot. Syst.* **2018**, *15*, 1729881418774673. [\[CrossRef\]](#)
37. Qiuyun, T.; Hongyan, S.; Hengwei, G.; Ping, W. Improved particle swarm optimization algorithm for AGV path planning. *IEEE Access* **2021**, *9*, 33522–33531. [\[CrossRef\]](#)
38. Zhang, H.; Zhuang, Q.; Li, G. Robot Path Planning Method Based on Indoor Spacetime Grid Model. *Remote Sens.* **2022**, *14*, 2357. [\[CrossRef\]](#)
39. Coello, C.A.C. Use of a self-adaptive penalty approach for engineering optimization problems. *Comput. Ind.* **2000**, *41*, 113–127. [\[CrossRef\]](#)
40. *Introduction to Optimum Design*; Elsevier Inc.: Amsterdam, The Netherlands, 2012.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.