Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

An optimized grey wolf optimizer based on a mutation operator and eliminating-reconstructing mechanism and its application*#

Xiao-qing ZHANG†‡[1,2], Zheng-feng MING[1]

([1]*School of Mechano-Electronic Engineering, Xidian University, Xi'an 710071, China*)

([2]*School of Physics and Electronic Engineering, Xianyang Normal University, Xianyang 712000, China*)

†E-mail: 249140543@qq.com

Received Sept. 14, 2016; Revision accepted Dec. 18, 2016; Crosschecked Nov. 26, 2017

Abstract: Due to its simplicity and ease of use, the standard grey wolf optimizer (GWO) is attracting much attention. However, due to its imperfect search structure and possible risk of being trapped in local optima, its application has been limited. To perfect the performance of the algorithm, an optimized GWO is proposed based on a mutation operator and eliminating-reconstructing mechanism (MR-GWO). By analyzing GWO, it is found that it conducts search with only three leading wolves at the core, and balances the exploration and exploitation abilities by adjusting only the parameter $\boldsymbol{a}$, which means the wolves lose some diversity to some extent. Therefore, a mutation operator is introduced to facilitate better searching wolves, and an eliminating-reconstructing mechanism is used for the poor search wolves, which not only effectively expands the stochastic search, but also accelerates its convergence, and these two operations complement each other well. To verify its validity, MR-GWO is applied to the global optimization experiment of 13 standard continuous functions and a radial basis function (RBF) network approximation experiment. Through a comparison with other algorithms, it is proven that MR-GWO has a strong advantage.

Key words: Swarm intelligence; Grey wolf optimizer; Optimization; Radial basis function network
https://doi.org/10.1631/FITEE.1601555 **CLC number:** TP273+.1

1 Introduction

Optimal control is an important branch of control theory, the key to which is the optimization algorithm design. Given social and economic developments, traditional optimization algorithms cannot realistically provide the optimal control needed for many complex practical problems, which is the motivation behind the development of advanced intelligent optimization algorithms (Oftadeh *et al*., 2010). A popular research direction is biological swarm intelligence optimization algorithms (Karaboga, 2005; Li and Huang, 2016). Classical biological swarm intelligence algorithms include mainly genetic algorithms (GAs), evolutionary algorithms (EAs), particle swarm optimization (PSO), ant colony optimization (ACO), and so on (Gao *et al*., 2012). In recent years, some new algorithms have been proposed (Nabil, 2016; Thamaraiselvi and Santhi, 2016), such as the dolphin swarm algorithm (DSA) (Wu *et al*., 2016) and the grey wolf optimizer (GWO) (Mirjalili *et al*., 2014). The GWO algorithm can be applied to all walks of life immediately because it is easy to understand (Chaman-Motlagh, 2015; Komaki and Kayvanfar, 2015; Korayem *et al*., 2015; Mahdad and Srairi, 2015; Mirjalili, 2015; Zhang and Zhou, 2015; Kamboj *et al*., 2016; Medjahed *et al*., 2016; Mirjalili *et al*., 2016; Zhang *et al*., 2016), especially in solving problems in electric

‡ Corresponding author

* Project supported by the National High-Tech R&D Program (863) of China (No. 2015AA7041003), the Scientific Research Plan Projects of Shanxi Education Department (No. 17JK0825), and the Scientific Research Plan Projects of Xianyang Normal University (No. 15XSYK036)

Electronic supplementary materials: The online version of this article (https://doi.org/10.1631/FITEE.1601555) contains supplementary materials, which are available to authorized users

ORCID: Xiao-qing ZHANG, http://orcid.org/0000-0001-8939-2570

power systems (Sharma and Saikia, 2015; Hadidian-Moghaddam *et al.*, 2016). These problems include maximum power point tracking (Mohanty *et al.*, 2016), reactive power optimization scheduling (Sulaiman *et al.*, 2015), and a wide range of power system stabilizers when considering a time delay (Shakarami and Davoudkhani, 2016).

In the GWO algorithm, the hunting behavior of a grey wolf is simulated in the optimization process, and the so-called prey is the best optimization result.

2 GWO and literature review

2.1 GWO algorithm

The grey wolf, also called a 'timber wolf' or a 'western wolf', belongs to the Canidae family, living in the desolate places of Europe and North America. A grey wolf pack generally consists of 5–11 wolves forming a family, which usually involves one or two adult wolves, three to six teen wolves, and one to three young wolves. In special cases, a large pack is composed of two or three wolf families. The grey wolf is the upper member of the food chain. They are excellent hunters, and have a very strict social hierarchy. Moreover, they attach great importance to their pack organization and management, which can be seen from their manager, who is picked out usually not for its strongest fighting capacity, but for its best management and organizational ability. Grey wolves often like to hunt in groups under the leadership of their manager. GWO was proposed to imitate their hunting behavior in groups.

In the standard GWO algorithm, the grey wolves are divided into four levels: the highest are the alpha (α) wolves, the second are beta (β) wolves, the third are delta (δ) wolves, and the last are omega (ω) wolves. The α, β, and δ wolves can be regarded as the leaders, and there is always only one wolf in each category. The ω wolves, a dozen or more, undertake the major search tasks. Here, the optimization procedure means the process of capturing prey, and when the prey is captured, the optimal value is found. The prey is presumed to stay at the position of the α wolf. Thus, it is seen that finding the best position of the α wolf means the optimal value is found. The specific steps are as follows:

Step 1: initialization. Establish the initial positions of α, β, δ, and ω wolves, and calculate the initial solution α_score, β_score, and δ_score for the α, β, and δ wolves, respectively.

Step 2: social hierarchy. Calculate the solution for every ω wolf, and then compare the results with the α_score, β_score, and δ_score, respectively. If it is superior, update the corresponding α, β, or δ wolf with this ω wolf.

Step 3: encircling and hunting. Perform encircling and hunting under the leader of the α, β, and δ wolves by all ω wolves. All ω wolves move around the α, β, and δ wolves. The specific calculation can be represented as

$$\boldsymbol{D}_\alpha = \left|\boldsymbol{C}_1 \cdot \boldsymbol{X}_\alpha - \boldsymbol{X}(t)\right|, \tag{1}$$

$$\boldsymbol{D}_\beta = \left|\boldsymbol{C}_2 \cdot \boldsymbol{X}_\beta - \boldsymbol{X}(t)\right|, \tag{2}$$

$$\boldsymbol{D}_\delta = \left|\boldsymbol{C}_3 \cdot \boldsymbol{X}_\delta - \boldsymbol{X}(t)\right|, \tag{3}$$

$$\boldsymbol{X}_1(t+1) = \boldsymbol{X}_\alpha - \boldsymbol{A}_1 \cdot \boldsymbol{D}_\alpha, \tag{4}$$

$$\boldsymbol{X}_2(t+1) = \boldsymbol{X}_\beta - \boldsymbol{A}_2 \cdot \boldsymbol{D}_\beta, \tag{5}$$

$$\boldsymbol{X}_3(t+1) = \boldsymbol{X}_\delta - \boldsymbol{A}_3 \cdot \boldsymbol{D}_\delta, \tag{6}$$

$$\boldsymbol{X}(t+1) = \frac{\boldsymbol{X}_1(t+1) + \boldsymbol{X}_2(t+1) + \boldsymbol{X}_3(t+1)}{3}, \tag{7}$$

where t indicates the current number of iterations, $\boldsymbol{X}$ is the current position vector of the ω wolf, $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, and $\boldsymbol{X}_\delta$ are the position vectors of the α, β, and δ wolves, respectively. $\boldsymbol{D}$ is the distance between the ω wolf and the α, β, or δ wolf. The computation formulas of coefficient matrices $\boldsymbol{A}$ and $\boldsymbol{C}$ are $\boldsymbol{A}=2\boldsymbol{a}\cdot\boldsymbol{r}_1-\boldsymbol{a}$ and $\boldsymbol{C}=2\boldsymbol{r}_2$, respectively, where $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are two random vectors in [0, 1], $\boldsymbol{a}$ is a variable to make the calculation of $\boldsymbol{A}$ convenient, and it can be thought as the weight value of the distance $\boldsymbol{D}$. The element of $\boldsymbol{a}$ decreases from 2 to 0 with the increase of the number of iterations, and thus the variable range of $\boldsymbol{A}$ is obtained as [−1, 1].

Step 4: determining whether to complete the calculation. If the search accuracy meets the requirements, or the number of iterations reaches the maximum, the calculation ends. Otherwise, return to the second step until the ending condition is met.

2.2 Literature review

GWO is a meta-heuristic algorithm, and as such, it is usually important to balance the abilities of exploration and exploitation (Gao, 2013). GWO is also a swarm intelligence algorithm to imitate the grey

wolves' hunting behavior by parallel iteration (Han *et al*., 2015), which would accelerate the convergence. From the above section, the α wolf is used to memorize the best solution until the end of the iteration, and the search information of the previous iteration has been retained in the process of optimization.

There is only one parameter to be adjusted in the iterative process for GWO, that is, $\boldsymbol{a}$. The exploration ability and exploitation ability are balanced mainly by adjusting the parameter $\boldsymbol{a}$, which makes the algorithm structure simple and easy to implement, but meanwhile brings some challenges. In early iterations, when the value of $\boldsymbol{a}$ is larger, the exploration ability of GWO is stronger. In later iterations, with $\boldsymbol{a}$ gradually decreasing, the exploitation ability is strengthened and the exploration ability is weakened. Thus, for the whole iteration process, both of the two phases have been developed. However, since exploration is reduced in later iterations, the risk of being trapped in local optima will be increased.

Moreover, in GWO, the search behavior occurs mainly around the three leader grey wolves α, β, and δ, and if the three leader grey wolves are all in local optima area, it will be hard for the algorithm to jump out of the local optimum, and actually this problem has been exposed in the optimization results (Mirjalili *et al*., 2014). Therefore, many excellent variations of GWO have been developed (Saremi *et al*., 2015; Emary *et al*., 2016; Emary and Zawbaa, 2016; Kamboj, 2016; Lu *et al*., 2016; Yao *et al*., 2016).

Evolutionary population dynamics (EPD) has been applied to improve GWO (GWO-EPD) (Saremi *et al*., 2015), in which the searching wolves with poorer performances are found first, and then the algorithm reinitializes them around the α, β, and δ wolves or in the whole feasible scope. Here, GWO is improved to some extent, whereas with careful analysis it could be found that the step size of the reinitialized operator around α, β, and δ wolves is too large, which would be of little help to the exploration ability. The step sizes of other reinitialized operators in the whole feasible scope are also large, which may limit the diversity of the reinitialization and bring in an excessively strong exploration ability, thus easily plunging the system into local optima.

Emary *et al*. (2016) have been proposed two kinds of chaos GWOs. The idea is to apply a chaotic function to replace the parameter $\boldsymbol{a}$. Great care is taken with the exploration ability of GWO in later iterations with decreasing $\boldsymbol{a}$, and thus they planned to apply the chaos to replace the change in $\boldsymbol{a}$, and the algorithm was denoted as a CGWO. The exploration ability in later iterations is improved to a certain extent since chaos is stochastic and ergodic. However, irregularity and sensitivity to the initial chaos condition would make the algorithm oscillating and even unstable over the entire iteration cycle (including early and later iterations), thus leading to failure in finding the optimal result.

Two binary GWO algorithms (bGWO1 and bGWO2) have been proposed mainly for feature selection in the literature (Emary *et al*., 2016). In bGWO1, the special bitwise operators are given to Eqs. (4)–(6) to transfer the state value of each dimension to a binary 0 or 1, and then the updated value of Eq. (7) is bitwise determined by a simple stochastic crossover strategy. In bGWO2, the special bitwise operator is only given to Eq. (7), which is simpler than bGWO1. For both, since the result of each bit, 0 or 1, is determined according to a random probability, the variation is implied in the calculation process, which could increase the diversity of the wolves and is reflected more obviously and simply in bGWO2. The binary GWO algorithms are very convenient for the optimization of some specific problems, which can be described clearly only with a binary 0 or 1. However, for the design of the two algorithms, only the conditions for converting the state value of each dimension into a simple integer value of 0 or 1 are considered. When the situation becomes complicated where the decimal would be needed to describe the problem, the binary GWO algorithms offer little help.

An improved GWO algorithm (IGWO) has also been proposed (Yao *et al*., 2016), inspired by the PSO algorithm, where Eq. (7) is first multiplied by a learning factor, and then an addend is added that could reflect the distance between the grey wolf and the current best α wolf. In fact, it can be seen from Eq. (1) that the distance is considered in GWO to a certain extent; therefore, IGWO is not sufficient for improving GWO completely. The hybrid PSO-GWO algorithm (Kamboj, 2016) is another improved GWO affected by PSO, which takes the best position as the search agent's current position by imitating the PSO. Since position movement is always influenced by the two best values, its performance with respect to

diversity is limited.

Although there is another discrete variant of GWO (Lu *et al*., 2016), it still does not generate the best optimization effect.

Therefore, for better applications of GWO and a comprehensive improvement of its exploration and exploitation abilities, an improved GWO algorithm is proposed based on the mutation operator and eliminating-restructuring mechanism (MR-GWO). All the ideas for mutation and eliminating-reconstructing come from the concepts of biological evolution. Therefore, MR-GWO can also be considered as a hybrid GWO based on biological evolution. Note that the idea of biological evolution is related to concepts similar to, but different from, evolutionary programming (EP), evolution strategies (ES), and differential evolution (DE).

To verify the effectiveness of MR-GWO, GWO variants (chaos GWO and the binary GWO) and MR-GWO are compared in the experiments.

3 Improved GWO (MR-GWO)

This section introduces the mutation operator and eliminating-reconstructing mechanism to GWO to obtain MR-GWO, to enhance the exploration and exploitation abilities.

3.1 Eliminating-restructuring mechanism

The first step is to sort the ω wolves, keep the better R_num ones, and then eliminate the inferior ones, which will be reconstructed following Eq. (8) or (9). Eq. (8) denotes the new ω wolves, which will be produced randomly in the whole feasible domain, and Eq. (9) means the new ones, which will be produced around the α wolf.

$$x_i^k(t+1)=x_{\min}+r_2(x_{\max}-x_{\min}), \tag{8}$$

$$x_i^k(t+1)=x_\alpha^k(t)+\eta r_1(x_{\max}-x_{\min}). \tag{9}$$

$x_i^k(t+1)$ is the kth status value of the ith ω wolf in the next iteration, and $x_\alpha^k(t)$ is the kth status value of the α wolf. The parameters r_1 and r_2 are random numbers in (0, 1), and η is a closing factor to reflect the degree to which the ω wolf is close to the α wolf. Parameter η affects the exploitation of the system, especially exploitation of the problem with unimodal characteristics.

3.2 Mutation operator for excellent search wolves

The differential mutation operator is introduced to accelerate the algorithm to find the global optimum. The mutation probability means that the mutation will occur only for some excellent ω wolves. Some other excellent ω wolves will keep their original statuses, which will be helpful for the whole team to maintain its basic exploration ability. The specific mutation algorithm is given as follows:

$$x_i^k(t+1)=\begin{cases}x_i^k(t)+F\cdot(x_j^k(t)-x_l^k(t)+x_\alpha^k(t)-x_n^k(t)),\\ \qquad \text{rand}_i>P_\text{m},\ j\neq l\neq n\neq i,\\ x_i^k(t),\quad \text{rand}_i\le P_\text{m}.\end{cases} \tag{10}$$

$x_i^k(t)$ and $x_i^k(t+1)$ are the kth status values of the ith ω wolf in the current iteration and the next iteration, respectively. $x_\alpha^k(t)$, $x_j^k(t)$, $x_l^k(t)$, and $x_n^k(t)$ are the kth status values of the α wolf, the jth, lth, and nth ω wolves, respectively. F is the amplification factor, which is generally varying in the scope (0, 2), rand_i is a random number in (0, 1), and P_m is the mutation probability.

The second step is to use a greedy selection operator (Storn and Price, 1997) to compare the mutated offspring with its parent. If better-mutated, the offspring will be preserved; otherwise, the parent is kept unchanged.

3.3 Improved GWO algorithms

Definition 1 MR-GWO is defined as an optimized GWO algorithm based on an eliminating-reconstructing mechanism and the mutation operator.

To illustrate how the eliminating-reconstructing mechanism and the mutation operator complement each other in function, a derivative algorithm is defined.

Definition 2 R-GWO is another improved GWO algorithm based on only the eliminating-reconstructing mechanism, which is a derivative algorithm of MR-GWO.

The detailed frame structure of the MR-GWO algorithm is shown in Algorithm 1, and it is necessary to remove the mutation step (step 14) to obtain the frame structure of R-GWO.

The out-of-bound process is performed as follows:

$$x_i^k(t)=\begin{cases}x_{\max}, & x_i^k(t)>x_{\max},\\ x_{\min}, & x_i^k(t)<x_{\min}.\end{cases} \tag{11}$$

$x_i^k(t)$ is the kth status value of the ith ω wolf in the current iteration, $x_{\max}$ is the upper limit of the status variable, and $x_{\min}$ is the lower limit of the status variable.

Algorithm 1 Frame structure of MR-GWO

```
Initialize the positions of all wolves, parameters η, F, P_m, and so on, set iteration=l (the number of iterations), and calculate the solutions of α, β, and δ wolves;
Check the positions of all wolves one by one;
if the position is out of the bound
  Execute the process as given in Eq. (11);
else
  Calculate the solutions of ω wolves;
  if the solution of ω wolf is better than that of α, β, or δ wolf
    Update the position of α, β, or δ wolf with the better ω wolf;
  else
    Keep the position of α, β, or δ wolf unchanged;
  end if
end if
Update the positions of ω wolves following Eqs. (1)–(7);
Calculate and sort the solutions of ω wolves, and do mutation for the better R_num ω wolves following Eq. (10);
Eliminate the worse ω wolves and reconstruct them following Eq. (8) or (9);
if iteration≥Max_iteration or the ending condition is met
  Finish the calculation;
else
  Return to step 2;
end if
```

4 Global optimization experiments

Thirteen benchmark functions are shown in Tables 1 and 2 as the models of the global optimization experiments. Seven functions in Table 1 are unimodal problems, and the other six functions in Table 2 are multimodal problems. Expressions of benchmark functions are given in Table S1, and graphs of benchmark functions in Fig. S1 in the supplementary materials.

During these experiments we have compared MR-GWO, R-GWO, GWO, PSO, EA, and several GWO variants (CGWO1, CGWO2, bGWO1, bGWO2). DE is selected for EA, whose mutation operator is obtained using the random vector difference method (Venske *et al*., 2016) as follows:

$$x_i^k(t+1)=x_i^k(t)+f\cdot(x_j^k(t)-x_m^k(t)), \tag{12}$$

where j and m are two randomly selected individuals different from each other and different from the current one. The amplification factor f generally has its value from [0, 2].

In the experiments, the maximum number of iterations is 500 for each algorithm, and each experiment is repeated 30 times. All the parameters are listed in Table 3.

Table 1 Unimodal benchmark functions

No.	Name	Dim	Range	$f_{\min}$
1	Sphere	30	[−100, 100]	0
2	Schwefel 2.22	30	[−100, 100]	0
3	Schwefel 1.2	30	[−100, 100]	0
4	Schwefel 2.21	30	[−100, 100]	0
5	Rosebrock	30	[−30, 30]	0
6	Step	30	[−100, 100]	0
7	Quartic	30	[−1.28, 1.28]	0

Dim: dimensionality

Table 2 Multimodal benchmark functions

No.	Name	Dim	Range	$f_{\min}$
8	Schwefel 2.26	30	[−500, 500]	−418.9829×Dim
9	Rastrigin	30	[−5.12, 5.12]	0
10	Shifted Ackley	30	[−32, 32]	0
11	Griewank	30	[−600, 600]	0
12	Penalized1	30	[−50, 50]	0
13	Penalized2	30	[−50, 50]	0

Dim: dimensionality

4.1 Experimental results

4.1.1 Results of unimodal problems

The unimodal function optimization results are shown in Table 4. If the absolute error is less than or equal to 1, the optimization procedure is deemed successful; otherwise, it is unsuccessful.

From Table 4, we can see that MR-GWO can find the theoretical optimal value of 0 for functions 1, 2, 3, and 4 with no failure. Five out of seven functions for MR-GWO give the best results, showing that its

Table 3 Parameters and functions of five algorithms

Algorithm	Parameter	Value
MR-GWO	Mutation probability	0.9
	Closing factor	10^{-5}
CGWO1	Chaos function	$y_k=ay_k^2\sin(\pi y_k)$, $y_k\in(0, 1)$
CGWO2	Chaos function	$y_k=\mu(7.86y_k-23.31y_k^2+28.75y_k^3-13.3y_k^4)$, $y_k\in(0, 1)$, $\mu=0.9+0.18\text{rand}$
PSO	Maximum velocity	6 m/s
	Inertia weight	$\omega\in[0.2, 0.9]$ gradually decreased with the increase of the iteration number
	Acceleration coefficient	$\eta_1=\eta_2=2$
EA	Crossover probability	0.9
	Amplification factor	$f=0.5$

optimization ability is better than the others. The second score should be given to bGWO2. Comparing MR-GWO with GWO, we can clearly see that MR-GWO has better results. In addition, although the results of R-GWO are not as good as those of MR-GWO, its results are better than those of PSO and EA for six functions, and superior to those of GWO for functions 3, 5, 6, and 7. R-GWO has a relatively good stability for all the seven functions. R-GWO is not the best in the group experiments, but it can prove that the improved design is reasonable. The results of R-GWO are inferior to those of MR-GWO, showing that the eliminating-reconstructing operator and the differential mutation operator are dissimilar, and they complement each other in function.

Fig. 1 shows the iterative trajectories of each optimization algorithm for the seven unimodal functions. MR-GWO trajectories can be hardly seen for functions 1 to 4, since its optimal value is obtained at a high convergence speed. This means MR-GWO can converge to the vicinity of the best theoretical value of 0 under a few iterations. The iterative trajectories of bGWO2 show the same results as those of MR-GWO for functions 1, 2, and 3.

4.1.2 Results of multimodal problems

Table 5 shows the results for the multimodal functions. For functions 8, 9, and 11, MR-GWO performs very well, as its results are the best. The very competitive GWO variant, bGWO2, achieves the best score for functions 9, 10, 11, and 13. The optimization success rates of MR-GWO for all functions are 100%, the same as R-GWO, but for other algorithms, including bGWO2, the success rates are not always 100%. Although there are so many local minima in the high dimensional (30 dimensions) experiments, the results for MR-GWO and R-GWO are still close to the ideal value. These good results show the effectiveness of the two improved GWOs.

Fig. 2 shows the iterative optimization trajectories of six multimodal functions. For functions 9 and 11, it is difficult to find MR-GWO's trajectories curves, meaning that MR-GWO has a better convergence performance, similar to bGWO2. From the curves of functions 9, 10, 11, and 13, bGWO2 could be regarded as a very strong competitor to MR-GWO. Whatsoever, almost all the results of MR-GWO and R-GWO are better than those of GWO, in terms of the convergence speed and the final results. Thus, it is proved that the direction of the improved algorithm is correct.

4.1.3 Average error analysis

The average MAE of each algorithm for two groups of functions is shown in Tables 6 and 7. According to the average MAE, MR-GWO ranks first in the two groups of experiments.

Considering whether the average MAE from Tables 6 and 7 can fully describe the advantages and disadvantages of each algorithm, the Wilcoxon test is conducted to statistically analyze all the function errors of the algorithms.

4.1.4 Wilcoxon test

In this section, the Wilcoxon test is performed. Specifically, in the Wilcoxon test, we match each vector involving 30 errors obtained from each algorithm in 30 repeated experiments for each function with the zero vector. In the test, first the algorithm error vector is assumed to be equal to the zero vector. The h results are shown in Table 8, and the average p results are shown in Fig. 3.

Table 4 Results of seven unimodal benchmark functions

No.	Algorithm	Average	Best	Worst	std	MAE	s-rate	
	GWO	3.57×10^{-33}	3.98×10^{-35}	4.74×10^{-32}	8.84×10^{-33}	6.31×10^{-33}	100%	
	R-GWO	2.05×10^{-29}	1.41×10^{-31}	1.18×10^{-28}	2.90×10^{-29}	2.05×10^{-29}	100%	
	MR-GWO	0	0	0	0	0	100%	+
	PSO	1.04×10^{-5}	4.42×10^{-7}	8.18×10^{-5}	1.68×10^{-5}	1.04×10^{-5}	100%	
1	EA	4.94	1.432 94	9.769 393	2.061 803	4.94	0	
	CGWO1	11 120.07	2.30×10^{-19}	25 972.33	10 347.68	11 120.07	43.33%	
	CGWO2	0.119 317	2.94×10^{-18}	3.579 520	0.653 528	0.119 317	96.67%	
	bGWO1	6.566 667	2	12	2.144 493	6.566 667	0	
	bGWO2	0	0	0	0	0	100%	+
	GWO	5.73×10^{-20}	6.09×10^{-21}	1.62×10^{-19}	3.75×10^{-20}	9.41×10^{-20}	100%	
	R-GWO	1.94×10^{-17}	3.13×10^{-18}	6.24×10^{-17}	1.41×10^{-17}	1.94×10^{-17}	100%	
	MR-GWO	0	0	0	0	0	100%	+
	PSO	0.005 51	0.000 772	0.027 696	0.005 740	0.005 51	100%	
2	EA	2.70	1.305 983	5.947 446	1.123 394	2.70	0	
	CGWO1	4.90×10^{9}	8.96×10^{-11}	1.41×10^{11}	2 574 953	4.90×10^{9}	33.33%	
	CGWO2	31.151 28	2.74×10^{-10}	934.4760	170.6108	31.151 28	96.67%	
	bGWO1	5.733 333	2	9	1.595 971	5.733 333	0	
	bGWO2	0	0	0	0	0	100%	+
	GWO	6.99×10^{-8}	9.14×10^{-11}	7.94×10^{-7}	1.95×10^{-7}	6.99×10^{-8}	100%	
	R-GWO	1.52×10^{-10}	8.58×10^{-15}	2.31×10^{-9}	4.71×10^{-10}	1.52×10^{-10}	100%	
	MR-GWO	0	0	0	0	0	100%	+
	PSO	45.177 35	14.877 41	93.839 77	19.326 17	45.2	0	
3	EA	3 277.105	5652.626	1885.093	938.7532	3.28×10^{3}	0	
	CGWO1	18 633.62	2.38×10^{-7}	77 393.01	26 339.98	18 633.622	60%	
	CGWO2	0.015 793	2.37×10^{-9}	0.471 593	0.086 088	0.015 793 5	100%	
	bGWO1	432.6667	76	986	196.5464	432.666 67	0	
	bGWO2	0	0	0	0	0	100%	+
	GWO	1.97×10^{-8}	2.87×10^{-9}	6.26×10^{-8}	1.68×10^{-8}	1.97×10^{-8}	100%	
	R-GWO	0.009 871	0.000 624	0.052 086	0.010 705	0.009 87	100%	
	MR-GWO	0	0	0	0	0	100%	+
	PSO	0.864 623	0.545 123	1.157 536	0.160 402	0.865	76.7%	
4	EA	11.450 50	7.043 886	16.730 55	2.429 164	11.5	0	
	CGWO1	37.769 29	0.001 527	88.112 54	30.763 96	37.769 29	36.67%	
	CGWO2	0.0147 65	0.000 867	0.096 083	0.0242 46	0.014 765	100%	
	bGWO1	1	1	1	0	1	100%	
	bGWO2	1	1	1	0	1	100%	
	GWO	26.618 69	25.413 39	28.540 41	0.666 911	26.6	0	
	R-GWO	0.120 767	2.04×10^{-5}	0.547 932	0.154 256	0.121	100%	
	MR-GWO	0.029 322	6.65×10^{-6}	0.139 783	0.037 445	0.0293	100%	
	PSO	59.507 65	17.840 74	160.0221	41.197 85	59.5	0	
5	EA	394.0020	158.4372	832.6271	162.0948	394	0	
	CGWO1	1.21×10^{7}	26.505 32	7.67×10^{7}	1.91×10^{7}	1.21×10^{7}	0	
	CGWO2	28.608 29	26.065 97	29.677 66	0.767 756	28.608 29	0	
	bGWO1	0	0	0	0	0	100%	+
	bGWO2	17.833 33	0	103	38.287 55	17.833 33	80%	
	GWO	0.377 588	3.46×10^{-5}	0.754 195	1.236 827	0.378	100%	
	R-GWO	0.000 118	1.79×10^{-6}	0.000 276	6.38×10^{-5}	0.000 118	100%	
	MR-GWO	0.000 118	5.64×10^{-5}	0.000 315	5.61×10^{-5}	0.000 118	100%	
	PSO	1.21×10^{-5}	4.43×10^{-7}	0.000 120	2.27×10^{-5}	1.21×10^{-5}	100%	+
6	EA	4.803 558	1.706 722	10.692 74	2.111 010	4.80	0	
	CGWO1	13 023.97	2.833 594	31 023.06	12 099.49	13 023.97	0	
	CGWO2	3.678 516	2.616 965	4.799 469	0.582 690	3.678 516	0	
	bGWO1	21.9	15.5	29.5	4.148 950	21.9	0	
	bGWO2	7.5	7.5	7.5	0	7.5	0	
	GWO	0.001 122	0.000 324	0.002 303	0.000 552	0.001 12	100%	
	R-GWO	0.001 087	2.05×10^{-5}	0.003 223	0.001 060	0.001 09	100%	
	MR-GWO	0.000 163	5.98×10^{-6}	0.000 522	0.000 162	0.000 163	100%	
	PSO	0.124 744	0.045 526	0.330 937	0.053 441	0.125	100%	
7	EA	0.074 642	0.040 863	0.116 465	0.017 930	0.0746	100%	
	CGWO1	7.347 431	0.003 713	40.361 47	11.249 28	7.347 431	53.33%	
	CGWO2	0.023 778	0.004 268	0.470 359	0.084 420	0.023 778	100%	
	bGWO1	2.000 055	1.07×10^{-6}	20.000 05	4.510 526	2.000 055	63.33%	
	bGWO2	6.20×10^{-5}	6.82×10^{-7}	0.000 194	5.40×10^{-5}	6.20×10^{-5}	100%	+

Average: average values; Best: best values; Worst: worst values; std: standard error; MAE: mean absolute error; s-rate: success rate; +: the best algorithm

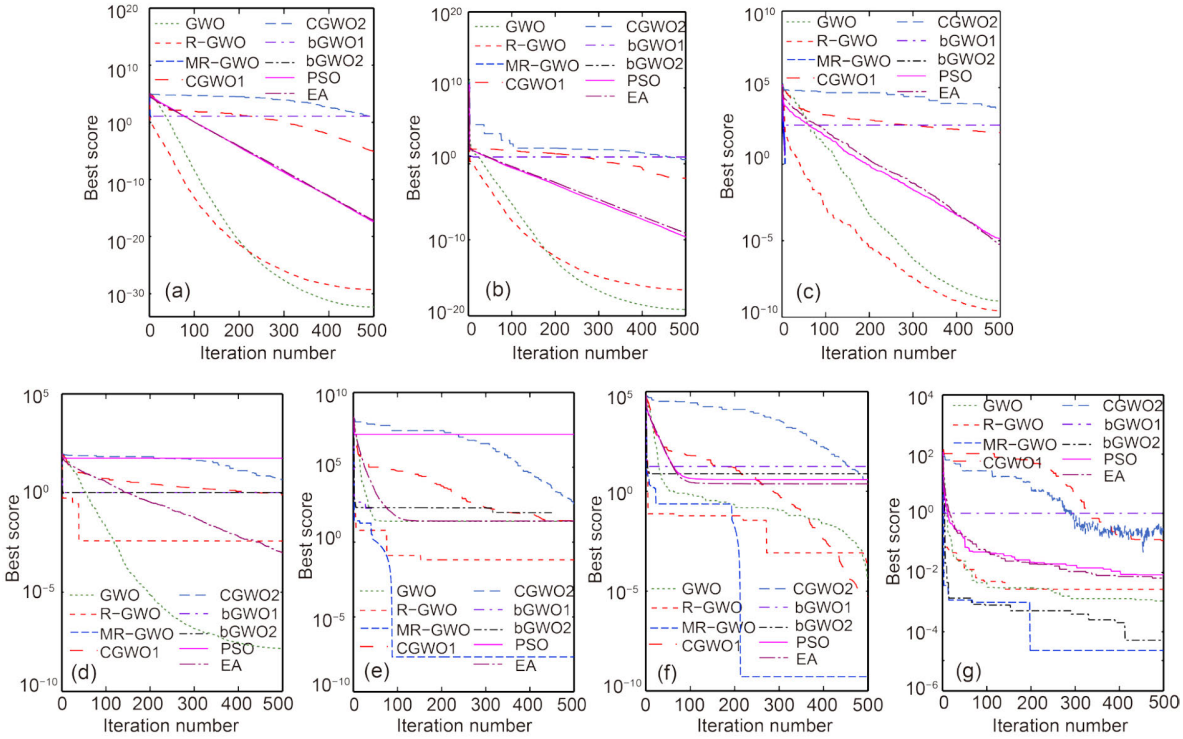

Fig. 1 Iterative trajectories in the unimodal benchmark function experiments with functions 1 (a), 2 (b), 3 (c), 4 (d), 5 (e), 6 (f), and 7 (g)

In Table 8, h=0 means the total difference between the experimental error and the zero vector is not significant, and h=1 means that the total difference is significant. MR-GWO has the most 0 values, indicating a greater likelihood that its error result is close to 0, and thus is the best algorithm. bGWO2 comes in the second, and GWO takes the third place. The same conclusion can be obtained from Fig. 3.

The p value returns the probability that the error vector is the same as a zero vector, $p\in[0, 1]$. When p is close to 0, the original hypothesis will be challenged; when p is close to 1, there is a higher probability that the original hypothesis is right. From Fig. 3, the average p value of MR-GWO is the largest, showing that the probability of its error vector being equal to a zero vector is the greatest, and its optimization effect is the best. bGWO2 still comes in the second, and GWO the third.

4.2 Discussion

GWO has many advantages, which has been proven by many researchers, but it also has some shortcomings. GWO uses only parameter $\boldsymbol{a}$ to balance the exploration and exploitation abilities, and it is hard to ensure that the two abilities remain superb throughout the optimization process. Specifically, originally $\boldsymbol{a}$=2, $|\boldsymbol{A}|>1$, and GWO has a great exploration ability. Then with the number of iterations increasing, $\boldsymbol{a}$ gradually decreases to zero, accompanied by $|\boldsymbol{A}|<1$, and the exploitation ability is enhanced. Meanwhile, its exploration ability is greatly weakened, which makes it hard for GWO to escape from local optima.

Why can MR-GWO achieve a greater performance than GWO by adding the eliminating-restructuring mechanism for search wolves with poor performances and the mutation operator for search wolves with excellent performances? A detailed analysis is given in the following.

Note that GWO has a better exploration ability in early iterations, and MR-GWO does not change the structure or any element of GWO with the exception of the addition of two operators. Therefore, MR-GWO has a better exploration ability in early iterations, which ensures that MR-GWO reaches one of two good situations in later iterations, the neighborhood of the global optimal value or local optima. Two later iteration situations are analyzed in detail:

Table 5 Results of six multimodal benchmark functions

No.	Algorithm	Average	Best	Worst	std	MAE	s-rate	
	GWO	−6221.62	−8005.14	−3110.76	888.0604	6347.864	0	
	R-GWO	−12 569.4	−12 569.5	−12 569.3	0.055 172	0.0375	100%	
	MR-GWO	−12 569.5	−12 569.5	−12 569.4	0.020 994	0.016 851	100%	+
	PSO	−5627.79	−7515.18	−3055.65	1293.242	6941.695	0	
8	EA	−4707.81	−5470.14	−4201.57	286.7858	7861.678	0	
	CGWO1	−3888.89	−6703.10	−1861.52	1698.756	8680.601	0	
	CGWO2	−5159.59	−7038.17	−3879.39	773.0654	7409.899	0	
	bGWO1	−2332.29	−3688.89	−1659.02	452.6846	10 237.20	0	
	bGWO2	−2370.36	−3227.82	−1310.34	465.8631	10 199.13	0	
	GWO	1.885 759	0	9.822 066	2.872 974	1.885 759	63.3%	
	R-GWO	1.14×10^{-14}	0	5.68×10^{-14}	2.31×10^{-14}	1.14×10^{-14}	100%	
	MR-GWO	0	0	0	0	0	100%	+
	PSO	51.003 94	29.8617	72.644 90	13.060 56	51.003 94	0	
9	EA	229.8300	195.3829	257.8129	11.590 54	229.83	0	
	CGWO1	179.8396	73.182 53	389.4503	109.7950	179.8396	0	
	CGWO2	95.361 80	60.352 12	188.1213	30.108 43	95.361 80	0	
	bGWO1	4.4	2	8	1.631 585	4.4	0	
	bGWO2	0	0	0	0	0	100%	+
	GWO	4.39×10^{-14}	3.64×10^{-14}	5.77×10^{-14}	4.50×10^{-15}	4.39×10^{-14}	100%	
	R-GWO	6.32×10^{-14}	4.35×10^{-14}	8.97×10^{-14}	1.15×10^{-14}	6.32×10^{-14}	100%	
	MR-GWO	8.88×10^{-16}	8.88×10^{-16}	2.66×10^{-14}	1.0×10^{-31}	8.88×10^{-16}	100%	
	PSO	0.025 019	0.000 361	0.645 844	0.117 298	0.025 019	100%	
10	EA	1.560 736	0.799 847	2.461 708	0.422 073	1.560 736	10%	
	CGWO1	10.308 14	3.611 600	19.035 68	7.196 844	10.308 14	0	
	CGWO2	3.618 457	3.461 055	3.774 014	0.067 593	3.618 457	0	
	bGWO1	1.784 382	1.225 741	2.281 199	0.277 313	1.784 382	0	
	bGWO2	8.88×10^{-16}	8.88×10^{-16}	8.88×10^{-16}	0	8.88×10^{-16}	100%	+
	GWO	0.002 748	0	0.014 104	0.005 109	0.002 748	100%	
	R-GWO	0.000 330	0	0.009 911	0.001 809	0.000 33	100%	
	MR-GWO	0	0	0	0	0	100%	+
	PSO	0.007 220	2.77×10^{-8}	0.039 300	0.009 596 2	0.007 22	100%	
11	EA	1.040 429	0.951 632	1.114 229	0.030 278 6	1.040 429	10%	
	CGWO1	74.783 96	0	263.6219	97.643 635	74.783 96	60%	
	CGWO2	1.858 030	0	55.554 48	10.141 644	1.858 030	96.67%	
	bGWO1	0.323 426	0.129 898	0.658 175	0.152 413 0	0.323 426	100%	
	bGWO2	0	0	0	0	0	100%	+
	GWO	0.028 439	7.33×10^{-6}	0.071 211	0.016 955 4	0.028 439	100%	
	R-GWO	6.50×10^{-6}	2.22×10^{-18}	2.74×10^{-5}	7.507×10^{-6}	6.5×10^{-6}	100%	
	MR-GWO	7.81×10^{-6}	2.16×10^{-8}	0.000 234	6.09×10^{-6}	7.81×10^{-6}	100%	
	PSO	0.003 455	7.09×10^{-9}	0.103 671	0.018 927 7	0.003 456	100%	+
12	EA	0.809 538	0.118 142	2.072 280	0.515 1333	0.809 539	66.7%	
	CGWO1	9.04×10^{6}	3.232 734	7.58×10^{7}	1.800×10^{7}	9.04×10^{6}	0	
	CGWO2	6.872 274	3.425 248	12.498 60	2.535 725	6.872 274	0	
	bGWO1	2.796 256	2.172 935	3.717 551	0.434729	2.796 256	0	
	bGWO2	1.668 971	1.668 971	1.668 971	0	1.668 971	0	
	GWO	0.430 387	6.35×10^{-6}	0.851 413	0.209 407	0.440 036	100%	
	R-GWO	9.75×10^{-5}	7.32×10^{-7}	0.000 579	0.000 125	0.000 143	100%	
	MR-GWO	0.000 159	2.3×10^{-21}	0.000 357	0.000 121	0.000 125	100%	
	PSO	0.002 567	1.69×10^{-7}	0.010 99	0.004 725	0.001 497	100%	
13	EA	3.879 96	1.035 73	11.249 15	2.639 906	3.834 828	16.7%	
	CGWO1	7.52×10^{7}	1.695 07	$3.04\text{E}\times10^{8}$	8.17×10^{7}	7.52×10^{7}	0	
	CGWO2	2.903 902	1.564 47	4.026 368	0.656 289	2.903 902	0	
	bGWO1	1.35×10^{-32}	1.35×10^{-32}	1.35×10^{-32}	0	1.35×10^{-32}	100%	+
	bGWO2	1.35×10^{-32}	1.35×10^{-32}	1.35×10^{-32}	0	1.35×10^{-32}	100%	+

Average: average values; Best: best values; Worst: worst values; std: standard error; MAE: mean absolute error; s-rate: success rate; +: the best algorithm

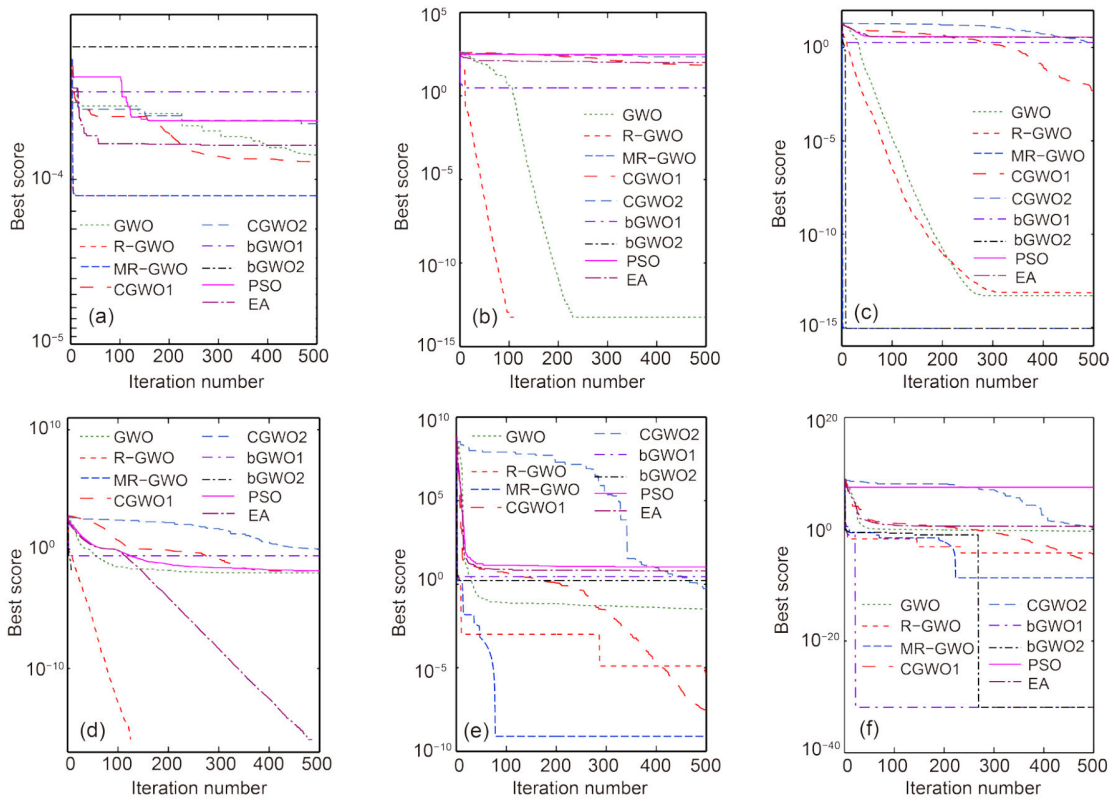

Fig. 2 Iterative trajectories in the multimodal benchmark function experiments with functions 8 (a), 9 (b), 10 (c), 11 (d), 12 (e), and 13 (f)

Table 6 Error results of the unimodal group of functions

Algorithm	Average mean absolute error	Rank
GWO	3.856 772	4
R-GWO	0.018 835	2
MR-GWO	0.004 229	1
PSO	15.097 13	6
EA	527.8688	8
CGWO1	7.02×10^{8}	9
CGWO2	9.087 390	5
bGWO1	67.123 82	7
bGWO2	3.761 914	3

Table 7 Error results of the multimodal group of functions

Algorithm	Average mean absolute error	Rank
GWO	1058.37	3
R-GWO	0.006 33	2
MR-GWO	0.002 83	1
PSO	1165.456	4
EA	1349.792	6
CGWO1	1.40×10^{7}	9
CGWO2	1253.419	5
bGWO1	1707.751	8
bGWO2	1700.133	7

1. The α wolf is located in the neighborhood of the global optimal value. If the optimal procedure has reached the neighborhood of the global optimal value, signifying that the α wolf is in this neighborhood, it is needed to strengthen the neighborhood search for the α wolf, which implies that more wolves moving closer to the neighborhood of the α wolf is better. For GWO, the search can be done only in a circular field (Fig. 4) noted with the R radius. It cannot be ensured that an α wolf is included. Thus, the task of moving as close as possible to the global optimal value appears to be a bit difficult. However, if Eq. (9) is applied to restructure the new wolves, the case for MR-GWO will be significantly different, where the value of parameter η could be designed to obtain the new wolves very close to the α wolf. The smaller η is, the closer it is to the α wolf. In Eq. (9), r_2 is a random variable, which makes the restructured wolves (black

dots in Fig. 4) uniformly distributed around the neighborhood of the α wolf (the blue dot in Fig. 4). Therefore, the neighborhood search for the α wolf is strengthened, and thus not only the search speed is increased, but the exploitation ability is promoted.

Table 8 The *h* results in the Wilcoxon test

Function No.	*h*								
	GWO	R	MR	C1	C2	b1	b2	PSO	EA
1	1	1	0	1	1	1	0	1	1
2	1	1	0	1	1	1	0	1	1
3	1	1	0	1	1	1	0	1	1
4	1	1	0	1	1	1	1	1	1
5	1	1	1	1	1	0	1	1	1
6	0	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1	1
9	1	0	0	1	1	1	0	1	1
10	1	1	1	1	1	1	1	1	1
11	1	1	0	1	1	1	0	1	1
12	1	1	1	1	1	1	1	1	1
13	0	1	1	1	1	1	1	1	1

R: R-GWO; MR: MR-GWO; C1: CGWO1; C2: CGWO2; b1: bGWO1; b2: bGWO2

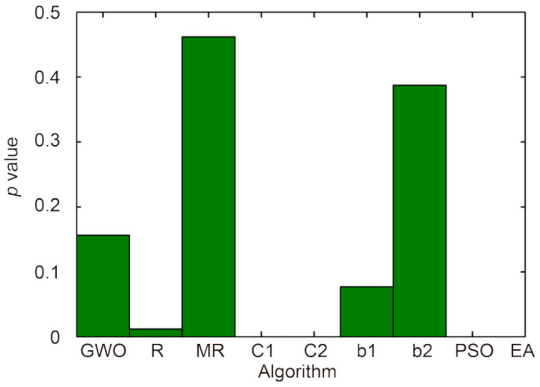

Fig. 3 The average *p* values of each algorithm
R: R-GWO; MR: MR-GWO; C1: CGWO1; C2: CGWO2; b1: bGWO1; b2: bGWO2

Moreover, according to Storn and Price (1997), the variation behavior of excellent search wolves in Eq. (10) involves the optimal information from the best wolf (the α wolf) in the current iteration, which will accelerate the search in the neighborhood of the α wolf. In this sense, the mutation operator can help improve the neighborhood search performance of the α wolf. Moreover, Eq. (10) takes advantage of the information of two random individual grey wolves, and therefore the risk of being trapped in local optima is reduced.

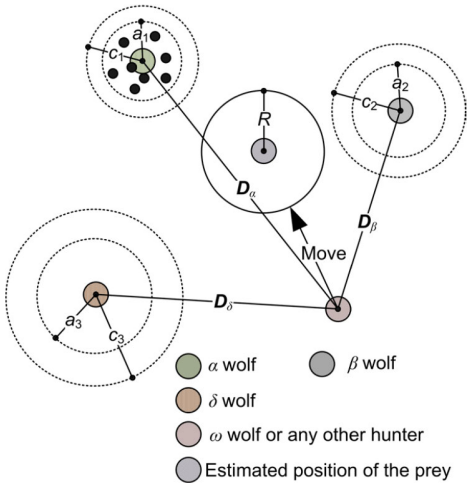

Fig. 4 Position updating in gray wolf optimizer
c_1, c_2, c_3: weights of the α, β, and δ wolves, respectively; $\boldsymbol{D}_\alpha$, $\boldsymbol{D}_\beta$, $\boldsymbol{D}_\delta$: distances between the ω wolf and the α, β, and δ wolves, respectively; a_1, a_2, a_3: weights of $\boldsymbol{D}_\alpha$, $\boldsymbol{D}_\beta$, and $\boldsymbol{D}_\delta$, respectively. References to color refer to the online version of this paper

Here, notice that η may bring some difficulties if it is too large or too small. For example, when the α wolf is in a relatively large neighborhood of the global optimal value, the exploitation ability of the α wolf's neighborhood could be strengthened, and it may reduce the speed of the ω wolf to get close to the α wolf. If it is too small, it would not help quickly find the optimal value, and would even reduce the convergence speed. Therefore, a good η value is very important for some practical problems.

2. The α wolf is located in the neighborhood of local optima. When the α wolf is located in the neighborhood of local optima, the local optima cannot be mistaken for the prey, or it will be trapped. To avoid being trapped in local optima, the method usually adopted is to increase the diversity of individuals in swarm intelligence algorithms. If the α wolf has reached the neighborhood of one of the local optima, GWO increases the diversity of wolves mainly by adjusting the value of $\boldsymbol{a}$ to change $\boldsymbol{A}$ to avoid falling into local optima. Its prey-attacking behavior is shown in Fig. 5. When $|\boldsymbol{A}|<1$ (Fig. 5a), the wolf will attack the prey, which will not occur if the α wolf is located in the neighborhood of local optima, or it will

jump into local optima. When $|\boldsymbol{A}|>1$ (Fig. 5b), the wolf will be forced to get away from the prey to increase the diversity of wolves to avoid falling into local optima. However, there is a circular boundary, which is not good for the diversity of the wolves across the entire feasible scope, meaning that the diversity is not comprehensive.

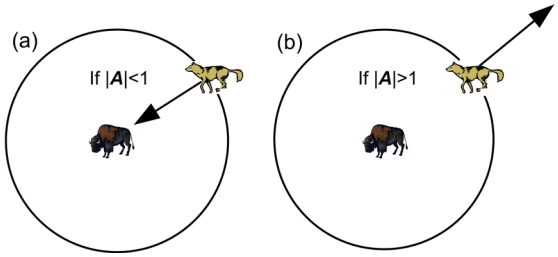

Fig. 5 Attacking prey (a) and searching for prey (b)

As for MR-GWO, the operator in Eq. (8) guarantees that the reconstructed wolves are distributed across the entire feasible region to greatly improve the diversity, and Eq. (9) increases the diversity around the α wolf, greatly enhancing the exploration ability. Therefore, no matter whether the α wolf is far from the actual optimal value or not, it will have a high chance of avoiding falling into local optima, offering a higher probability of achieving the global optimum. In addition, Eq. (10) can introduce some diverse individuals. Therefore, it is easier for MR-GWO to jump out of local optimization than GWO.

Although both the eliminating-restructuring mechanism and mutation operator can improve the diversity of wolves to change the performance of MR-GWO, the functions of the two operators are not identical. The restructuring mechanism (Eq. (8)) produces mainly the wolves close to the α wolf, being more conducive to its exploitation phase. Eq. (9) is used mainly to produce wolves within the whole feasible region. However, the wolves produced from the mutation operator (Eq. (10)) are not as good as those from the restructuring operator (Eq. (8)) close to the α wolf, nor are they the same as those from the restructuring operator (Eq. (9)) distributed in the whole feasible region. They mainly ensure that the algorithm maintains a strong search ability. Therefore, the restructuring mechanism and mutation operator are not the same, but they complement each other.

The performances of MR-GWO are summarized as follows:

1. It has a quick convergence and stable property.
2. It has a good capability for resisting local optima.
3. It has strong exploration and exploitation abilities throughout the iterations.

4.3 Analysis of parameter influence

4.3.1 Mutation probability P_m

In MR-GWO, P_m plays an important role in balancing exploration and exploitation abilities. In this section, three functions are adopted to study the effect of P_m (Fig. 6).

In Fig. 6, for functions 10 and 13, the best score can be given to P_m=0.95. For function 6, the best is P_m=1 and the second is P_m=0.95. Therefore, P_m=0.95 can be a good choice. For different problems, the influences of P_m are not always the same. From the overall statistical estimate of the experiments, a P_m in the range of 0.7 to 1 is a good choice in general.

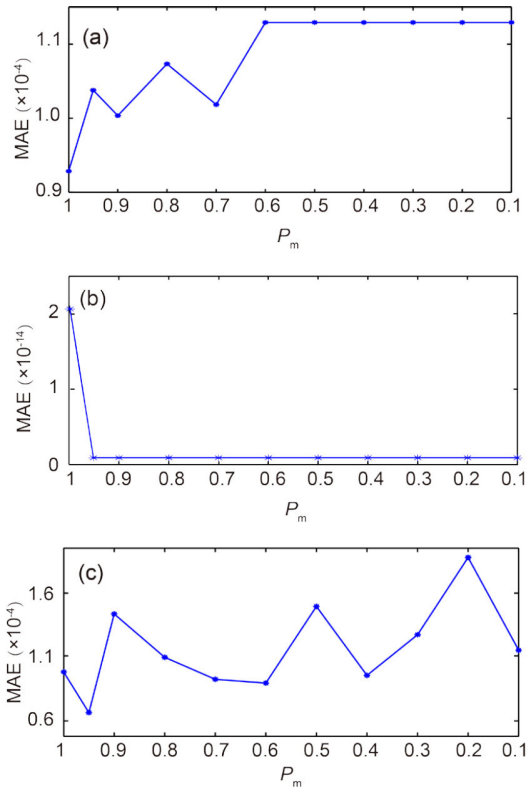

Fig. 6 The effect of mutation probability P_m on functions 6 (a), 10 (b), and 13 (c) (MAE: mean absolute error)

4.3.2 Closing factor η

The closing factor η reflects the distance between a new ω wolf and the α wolf. The smaller η is, the closer the wolf is to the α wolf, and the stronger the MR-GWO exploitation ability is. A greater η means a stronger exploration ability. The research results are shown in Fig. 7. The best value seems to be η=10^{-3} for functions 5 and 13. Usually η=10^{-3} is a good alternative, but it is not always the best. When a high accuracy is required, a better and different η is required for each problem.

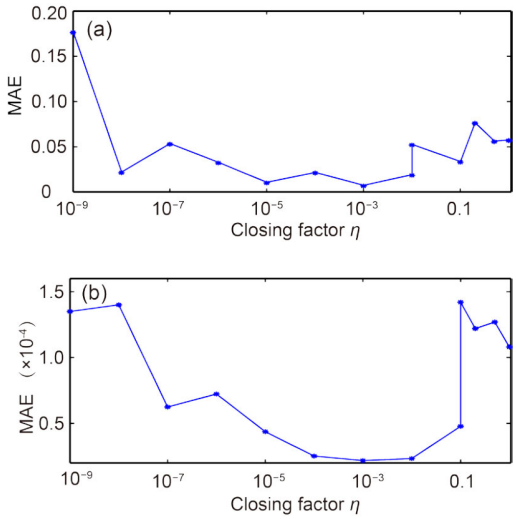

Fig. 7 The effect of closing factor η on functions 5 (a) and 13 (b) (MAE: mean absolute error)

5 MR-GWO application in an RBF network approximation

The radial basic function (RBF) neural network, a kind of three-layer forward network, has the advantage of fast learning speed and the ability to avoid local minima. Therefore, it is often used in real-time control systems. If the RBF network is used for an approximation experiment, the training of the parameters for the Gaussian basis function and the weights of the network is the key, because these parameters have a great influence on the performance of the system. In this experiment, the MR-GWO algorithm is adopted to optimize the parameters of the Gaussian basis function and the weights of the network. A discrete system is selected as the object:

$$y(k)=u(k)^3+\frac{y(k-1)}{1+y(k-1)^2}. \tag{13}$$

$y(k)$ is the output of the system, $u(k)$ is the input of the system, k and k–1 mean the current sampling time and the last sampling time, respectively. The optimization structure of RBF network approximation based on MR-GWO is shown in Fig. 8.

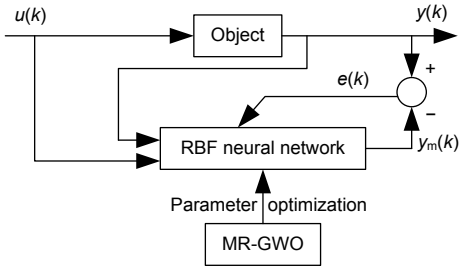

Fig. 8 Structure of RBF network approximation based on MR-GWO

The error in network approximation is

$$E(t)=(y(t)-y_{\mathrm{m}}(t))^2/2,\quad t=kT, \tag{14}$$

where T is the sampling time, $E(t)$ is the approaching error, and $y_{\mathrm{m}}(t)$ is the output of RBF neural network.

The network structure (Liu, 2014) is 2-5-1, and two inputs are $u(k)$ and $y(k)$ (Fig. 8), $u(k)$=sin(kT), T=0.001 s. The weights will be adjusted according to the gradient descent method:

$$\begin{aligned}\omega_j(t)&=\omega_j(t-1)+\Delta\omega_j(t)+\alpha(\omega_j(t-1)-\omega_j(t-2))\\&=\omega_j(t-1)-\rho\frac{\partial E}{\partial\omega_j}+\alpha(\omega_j(t-1)-\omega_j(t-2)).\end{aligned} \tag{15}$$

α=0.05, ρ=0.15, and the initial network weights are random values from (0, 1). The simulation results are shown in Figs. 9 and 10. The results of output y

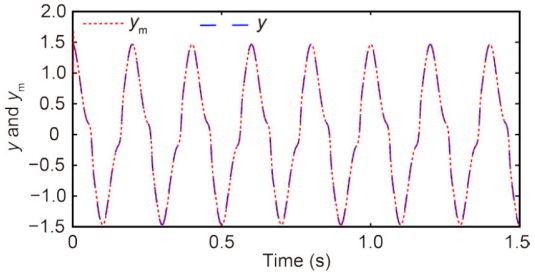

Fig. 9 Results of RBF network approximation based on MR-GWO

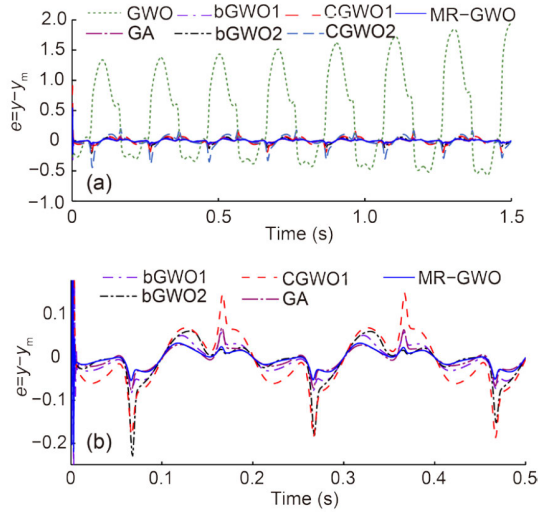

Fig. 10 Errors of RBF network approximation

and optimal approximation output y_m for MR-GWO are shown in Fig. 9, and y_m is essentially coincident with y. The error for each algorithm is shown in Fig. 10. The approximation errors of seven optimization algorithms, namely GWO, GA, bGWO1, bGWO2, CGWO1, CGWO2, and MR-GWO, are given (Fig. 10a), and it is obvious that the performances of GWO and CGWO2 are poorer than those of the other algorithms. Apart from these two algorithms, the enlarged error curves of the others are shown in Fig. 10b, and it can be seen that the error of MR-GWO is relatively small.

6 Conclusions

MR-GWO is put forward for the standard GWO, and the eliminating-reconstructing mechanism and differential mutation operator are introduced to improve the optimization capability. The entire study can be considered mainly from four aspects:

1. The eliminating-reconstructing operator and differential mutation operator are designed according to an analysis of the GWO structure.

2. The MR-GWO algorithm is designed and its frame structure is given.

3. MR-GWO is applied in a global optimization experiment and the RBF network approximation experiment is conducted to verify its effectiveness.

4. A detailed analysis of the performance of MR-GWO is conducted.

In addition, the influences of the mutation probability and closing factor are analyzed, and their general scopes are clarified.

According to the analysis of the entire research, it is found that not only in the global optimization experiment, but also in the RBF network approximation experiment, all results of the improved algorithm, MR-GWO, are better, and the improved algorithm is not easily trapped in local optima. This fully confirms that MR-GWO has stronger abilities in dealing with not only unimodal problems but also multimodal problems, and even some parameter optimization problems.

References

Chaman-Motlagh, A., 2015. Superdefect photonic crystal filter optimization using grey wolf optimizer. *IEEE Photon. Technol. Lett.*, **27**(22):2355-2358. https://doi.org/10.1109/LPT.2015.2464332

Emary, E., Zawbaa, H.M., 2016. Impact of chaos functions on modern swarm optimizers. *PLoS ONE*, **11**(7):e0158738. https://doi.org/10.1371/journal.pone.0158738

Emary, E., Zawbaa, H.M., Hassanien, A.E., 2016. Binary grey wolf optimization approaches for feature selection. *Neurocomputing*, **172**:371-381. https://doi.org/10.1016/j.neucom.2015.06.083

Gao, W.F., 2013. Artificial Bee Colony Algorithm and Its Applications. PhD Thesis, Xidian University, Xi'an, China (in Chinese).

Gao, W.F., Liu, S.Y., Huang, L.L., 2012. Particle swarm optimization with chaotic opposition-based population initialization and stochastic search technique. *Commun. Nonl. Sci. Numer. Simul.*, **17**(11):4316-4327. https://doi.org/10.1016/j.cnsns.2012.03.015

Hadidian-Moghaddam, M.J., Arabi-Nowdeh, S., Bigdeli, M., 2016. Optimal sizing of a stand-alone hybrid photovoltaic/wind system using new grey wolf optimizer considering reliability. *J. Renew. Sustain. Energy*, **8**:035903. https://doi.org/10.1063/1.4950945

Han, Z.M., Lin, Z.Y., Fu, M.Y., *et al*., 2015. Distributed coordination in multi-agent systems: a graph Laplacian perspective. *Front. Inform. Technol. Electron. Eng*., **16**(6):429-448. https://doi.org/10.1631/FITEE.1500118

Kamboj, V.K., 2016. A novel hybrid PSO-GWO approach for unit commitment problem. *Neur. Comput. Appl.*, **27**(6): 1643-1655. https://doi.org/10.1007/s00521-015-1962-4

Kamboj, V.K., Bath, S.K., Dhillon, J.S., 2016. Solution of non-convex economic load dispatch problem using grey wolf optimizer. *Neur. Comput. Appl.*, **27**(5):1301-1316. https://doi.org/10.1007/s00521-015-1934-8

Karaboga, D., 2005. An Idea Based on Honey Bee Swarm for Numerical Optimization. Technical Report No. TR06, Erciyes University, Kayseri, Turkey.

Komaki, G.M., Kayvanfar, V., 2015. Grey wolf optimizer

algorithm for the two-stage assembly flow shop scheduling problem with release time. *J. Comput. Sci.*, **8**:109-120. https://doi.org/10.1016/j.jocs.2015.03.011

Korayem, L., Khorsid, M., Kassem, S.S., 2015. Using grey wolf algorithm to solve the capacitated vehicle routing problem. *IOP Conf. Ser. Mater. Sci. Eng.*, **83**:012014. https://doi.org/10.1088/1757-899X/83/1/012014

Li, Z.C., Huang, X.L., 2016. Glowworm swarm optimization and its application to blind signal separation. *Math. Probl. Eng.*, **2016**:5481602. https://doi.org/10.1155/2016/5481602

Liu, J.K., 2014. Intelligent Control (3rd Ed.). Publishing House of Electronics Industry, Beijing, China, p.132-140 (in Chinese).

Lu, C., Xiao, S.Q., Li, X.Y., *et al*., 2016. An effective multi-objective discrete grey wolf optimizer for a real-world scheduling problem in welding production. *Adv. Eng. Softw.*, **99**:161-176. https://doi.org/10.1016/j.advengsoft.2016.06.004

Mahdad, B., Srairi, K., 2015. Blackout risk prevention in a smart grid based flexible optimal strategy using grey wolf-pattern search algorithms. *Energy Conv. Manag.*, **98**:411-429. https://doi.org/10.1016/j.enconman.2015.04.005

Medjahed, S.A., Saadi, T.A., Benyettou, A., *et al*., 2016. Gray wolf optimizer for hyperspectral band selection. *Appl. Soft Comput.*, **40**:178-186. https://doi.org/10.1016/j.asoc.2015.09.045

Mirjalili, S., 2015. How effective is the grey wolf optimizer in training multi-layer perceptrons. *Appl. Intell.*, **43**(1):150-161. https://doi.org/10.1007/s10489-014-0645-7

Mirjalili, S., Mirjalili, S.M., Lewis, A., 2014. Grey wolf optimizer. *Adv. Eng. Softw.*, **69**:46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

Mirjalili, S., Saremi, S., Mirjalili, S.M., *et al*., 2016. Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization. *Expert Syst. Appl.*, **47**:106-119. https://doi.org/10.1016/j.eswa.2015.10.039

Mohanty, S., Subudhi, B., Ray, P.K., 2016. A new MPPT design using grey wolf optimization technique for photovoltaic system under partial shading conditions. *IEEE Trans. Sustain. Energy*, **7**(1):181-188. https://doi.org/10.1109/TSTE.2015.2482120

Nabil, E., 2016. A modified flower pollination algorithm for global optimization. *Expert Syst. Appl.*, **57**:192-203. https://doi.org/10.1016/j.eswa.2016.03.047

Oftadeh, R., Mahjoob, M.J., Shariatpanahi, M., 2010. A novel meta-heuristic optimization algorithm inspired by group hunting of animals: hunting search. *Comput. Math. Appl.*, **60**(7):2087-2098. https://doi.org/10.1016/j.camwa.2010.07.049

Saremi, S., Mirjalili, S.Z., Mirjalili, S.M., 2015. Evolutionary population dynamics and grey wolf optimizer. *Neur. Comput. Appl.*, **26**(5):1257-1263. https://doi.org/10.1007/s00521-014-1806-7

Shakarami, M.R., Davoudkhani, I.F., 2016. Wide-area power system stabilizer design based on grey wolf optimization algorithm considering the time delay. *Electr. Power Syst. Res.*, **133**:149-159. https://doi.org/10.1016/j.epsr.2015.12.019

Sharma, Y., Saikia, L.C., 2015. Automatic generation control of a multi-area ST-Thermal power system using grey wolf optimizer algorithm based classical controllers. *Int. J. Electr. Power Energy Syst.*, **73**:853-862. https://doi.org/10.1016/j.ijepes.2015.06.005

Storn, R., Price, K., 1997. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.*, **11**(4):341-359. https://doi.org/10.1023/A:1008202821328

Sulaiman, M.H., Mustaffa, Z., Mohamed, M.R., *et al*., 2015. Using the gray wolf optimizer for solving optimal reactive power dispatch problem. *Appl. Soft Comput.*, **32**:286-292. https://doi.org/10.1016/j.asoc.2015.03.041

Thamaraiselvi, A., Santhi, R., 2016. A new approach for optimization of real life transportation problem in neutrosophic environment. *Math. Probl. Eng.*, **2016**:5950747. https://doi.org/10.1155/2016/5950747

Venske, S.M., Gonçalves, R.A., Benelli, E.M., *et al*., 2016. ADEMO/D: an adaptive differential evolution for protein structure prediction problem. *Expert Syst. Appl.*, **56**:209-226. https://doi.org/10.1016/j.eswa.2016.03.009

Wu, T.Q., Yao, M., Yang, J.H., 2016. Dolphin swarm algorithm. *Front. Inform. Technol. Electron. Eng.*, **17**(8):717-729. https://doi.org/10.1631/FITEE.1500287

Yao, P., Wang, H.L., Ji, H.X., 2016. Multi-UAVs tracking target in urban environment by model predictive control and improved grey wolf optimizer. *Aerosp. Sci. Technol.*, **55**:131-143. https://doi.org/10.1016/j.ast.2016.05.016

Zhang, S., Zhou, Y.Q., 2015. Grey wolf optimizer based on Powell local optimization method for clustering analysis. *Discr. Dynam. Nat. Soc.*, **2015**:481360. https://doi.org/10.1155/2015/481360

Zhang, S., Zhou, Y.Q., Li, Z.M., *et al*., 2016. Grey wolf optimizer for unmanned combat aerial vehicle path planning. *Adv. Eng. Softw.*, **99**:121-136. https://doi.org/10.1016/j.advengsoft.2016.05.015

List of supplementary materials

Table S1 Benchmark functions

Fig. S1 Graphs of benchmark functions (Dim=2)