An Improved Grey Wolf Optimizer Based on Differential Evolution and Elimination Mechanism

Jie-Sheng Wang¹,² & Shu-Xia Li¹

The grey wolf optimizer (GWO) is a novel type of swarm intelligence optimization algorithm. An improved grey wolf optimizer (IGWO) with evolution and elimination mechanism was proposed so as to achieve the proper compromise between exploration and exploitation, further accelerate the convergence and increase the optimization accuracy of GWO. The biological evolution and the “survival of the fittest” (SOF) principle of biological updating of nature are added to the basic wolf algorithm. The differential evolution (DE) is adopted as the evolutionary pattern of wolves. The wolf pack is updated according to the SOF principle so as to make the algorithm not fall into the local optimum. That is, after each iteration of the algorithm sort the fitness value that corresponds to each wolf by ascending order, and then eliminate R wolves with worst fitness value, meanwhile randomly generate wolves equal to the number of eliminated wolves. Finally, 12 typical benchmark functions are used to carry out simulation experiments with GWO with differential evolution (DGWO), GWO algorithm with SOF mechanism (SGWO), IGWO, DE algorithm, particle swarm algorithm (PSO), artificial bee colony (ABC) algorithm and cuckoo search (CS) algorithm. Experimental results show that IGWO obtains the better convergence velocity and optimization accuracy.

The swarm intelligence algorithms are proposed to mimic the swarm intelligence behavior of biological in nature, which has become a hot of cross-discipline and research field in recent years. The appearance of swarm intelligent optimization algorithm provides the fast and reliable methods for finding solutions on many complex problems². Because the swarm intelligence algorithm have characteristics of self-organization, parallel, distributive, flexibility and robustness, now they have been very widespread used in many cases, such as electric power system, communication network, system identification and parameter estimation, robot control, transportation and other practical engineering problems³⁴. Therefore, the research on the swarm intelligence optimization algorithms has an important academic value and practical significance.

At present, a variety of swarm intelligence optimization algorithms have been proposed by simulating the biotic population and evolution process in nature, such as particle swarm optimization (PSO) algorithm, shuffled frog leaping algorithm (SFLA), artificial bee colony (ABC) algorithm, ant colony optimization (ACO) algorithm, biogeography-based optimization (BBO) algorithm, and cuckoo search (CS) algorithm. Particle Swarm Optimization (PSO) algorithm put forward by Kennedy and Eberhart to mimic the foraging behavior of birds and fish flock⁵, but the convergence velocity and searching accuracy of PSO algorithm are unsatisfactory to some extent. Shuffled Frog-Leaping Algorithm (SFLA) put forward by Eusuff in 2003 is a novel swarm intelligent cooperative searching strategy based on the natural mematics⁶,⁷. On the one hand, individuals exchange information in the global searching process, and its search precision is high. On the other hand, SFLA has the disadvantage of slow convergence velocity and easy to falling into the local optimum. Artificial Bee Colony (ABC) Algorithm put forward by Karaboga in 2005 to mimics the finding food source behavior of bees⁸. In order to mimic the social behavior of the ant colony, Dorigo et al. Proposed the an novel Ant Colony Optimization (ACO) Algorithm in 2006⁹. But their disadvantages are the slow convergence speed and easy to premature. Biogeography-Based Optimization (BBO) algorithm was put forward by Simon in 2008¹⁰, whose idea is based on the geographical distribution principle in the biogeography. Cuckoo Search (CS) Algorithm was proposed by Yang and Deb in 2009 based on the cuckoo’s parasitic reproduction mechanism and Levy flights searching strategy¹¹,¹², whose advantage

¹School of Electronic and Information Engineering, University of Science & Technology Liaoning, Anshan, 114044, China. ²National Financial Security and System Equipment Engineering Research Center, University of Science & Technology Liaoning, Anshan, 114044, China. Correspondence and requests for materials should be addressed to J.-S.W. (email: wang_jiesheng@126.com)
is that CS algorithm is not easy to fall into the local optimum compared with other intelligent algorithms and has less parameters, and whose disadvantage is that the adding of Levy Flight search mechanism leads to strong leap in the process of search, thus, its local search is not careful.

The common shortcoming of these algorithms is that each swarm intelligence algorithm has problem in different degrees that the convergence velocity is slow, the optimization precision is low, and easy to fall into the local optimum. The key reason cause these shortcoming is that whether an algorithm is able to achieve the proper compromise between exploration and exploitation in its each searching phase or not. Exploration and exploration are contradictory. Exploration reflects the ability of the algorithm to search for new space, while exploration reflects the refining ability of the algorithm. These two criteria are generally used to evaluate stochastic optimization algorithms. Exploration is refers to that a particle leave the original search path in a certain extent and search towards a new direction, which reflects the ability of exploiting unknown regions. Exploitation is refers to that a particle continue to search more carefully on the original trajectory in a certain extent, which can insure the wolf make a detailed search to the region that have been explored. Too small exploration can cause a premature

Figure 1. 2D position vectors and possible next locations.

Figure 2. Position updating of IGWO.
convergence and falling into a local optimum, however, too small exploitation will make the algorithm converge too slowly.

The grey wolf optimizer (GWO) as a novel swarm intelligent optimization algorithm was put forward by Seyedali Mirjalili etc in 2014, which mainly mimics wolf leadership hierarchy and hunting mechanism in nature. Seyedali and Mirjalili etc has proved that the optimization performance of standard GWO is superior to that of PSO, GSA, DE and FEP algorithm. Due to the wolves algorithm with the advantages of simple in principle, fast seeking speed, high search precision, and easy to realize, it is more easily combined with the practical engineering problems. Therefore, GWO has high theoretical research value. But GWO is as a new biological intelligence algorithm, the research about it is just at the initial phase, so research and development of the theory are still not perfect. In order to make the algorithm plays a more superior performance, further exploration and research is needed.

Many swarm intelligence algorithms are mimic the hunting and searching behaviors of some animals. However, GWO simulates internal leadership hierarchy of wolves, thus, in the searching process the position of best solution can be comprehensively assessed by three solutions. But for other swarm intelligence algorithms, the best solution is searched only leaded by a single solution. So GWO can greatly decrease the probability of premature and falling into the local optimum. So as to achieve the proper compromise between exploration and exploitation, an improved GWO with evolution and elimination mechanism is proposed. The biological evolution and the SOF principle of biological updating of nature are added to the basic wolf algorithm. In order to verify the performance of the improved GWO, 12 typical benchmark functions are adopted to carry out simulation experiments, meanwhile, experimental results are compared with PSO algorithm, ABC algorithm and CS algorithm. The experimental results show that the improved grey wolf optimizer (IGWO) obtains the better convergence velocity and optimization accuracy.

The paper is organized as follows. In section 2, the grey wolf optimizer is introduced. A grey wolf optimizer with evolution and SOF mechanism is presented in section 3. In section 4, the simulation experiments are carried out and the simulation results are analyzed in details. Finally, the conclusion illustrates the last part.

Grey Wolf Optimizer
The grey wolf optimizer is a novel heuristic swarm intelligent optimization algorithm proposed by Seyedali Mirjalili et al. in 2014. The wolf as top predators in the food chain, has a strong ability to capture prey. Wolves generally like social life and in the interior of the wolves exists a rigid social hierarchy.

In order to mimic wolves internal leadership hierarchy, the wolves is divided into four types of wolf: alpha, beta, delta and omega, where the best individual, second best individual and third best individual are recorded as alpha, beta, and delta, and the rest of the individuals are considered as omega. In the GWO, the hunting (optimization) is guided by alpha, beta, and delta. They guide other wolves ($W$) tend to the best area in searching space. In iterative searching process, the possible position of prey is assessed by three wolves alpha, beta, and delta. In optimization process, the locations of wolves are updated based on Eqs (1) and (2).

$$D = | \bar{C} \cdot \bar{X}_p(t) - \bar{X}(t) |$$  \hspace{1cm} (1)

$$\bar{X}(t + 1) = \bar{X}_p(t) - \bar{A} \cdot D$$  \hspace{1cm} (2)

where, $t$ represents the $t$-th iteration, $\bar{A}$ and $\bar{C}$ are coefficient vector, $\bar{X}_p$ is the position vector of prey, $\bar{X}$ represents the wolf position. The vector $\bar{A}$ and $\bar{C}$ can be expressed by:
where, the coefficient $\vec{a}$ linearly decreases from 2 to 0 with the increasing of iteration number, $\vec{r}_1$ and $\vec{r}_2$ are random vector located in the scope $[0, 1]$.

Principle of the position updating rules described in Eqs (1) and (2) are shown in Fig. 1. It can be seen from Fig. 1 the wolf at the position $(X, Y)$ can relocate itself position around the prey according to above updating formulas. Although Fig. 1 only shows 7 positions that the wolf possible move to, by adjusting the random parameters C and A it can make the wolf to relocate itself to any position in the continuous space near prey. In the GWO, it always assumes that position of alpha, beta and delta is likely to be the prey (optimum) position. In the iteration searching process, the best individual, second best individual and third best individual obtained so far are respectively recorded as alpha, beta, and delta. However, other wolves who are regarded as omega relocate their locations according to the locations of alpha, beta, and delta. The following mathematical formulas are used to re-adjust positions of the wolf omega. The conceptual model that wolf update its position is shown in Fig. 2.

Figure 4. Flow chart of IGWO algorithm.

\[
\vec{A} = 2\vec{a} \cdot \vec{r}_1 - \vec{a}
\]

(3)

\[
\vec{C} = 2 \cdot \vec{r}_2
\]

(4)
where, \( \overrightarrow{X}_{\alpha} \), \( \overrightarrow{X}_{\beta} \) and \( \overrightarrow{X}_{\delta} \) are the position vector of alpha, beta, and delta, respectively. \( \overrightarrow{C}_1 \), \( \overrightarrow{C}_2 \), \( \overrightarrow{C}_3 \) are randomly generated vectors, \( \overrightarrow{X} \) represents the position vector of current individual. The Eqs (5), (6) and (7) respectively calculate the distances between the position of current individual and that of individual alpha, beta, and delta. So the final position vectors of the current individual are calculated by:

\[
\overrightarrow{X}_1 = \overrightarrow{X}_\alpha - \overrightarrow{A}_1 \cdot (D_1) \\
\overrightarrow{X}_2 = \overrightarrow{X}_\beta - \overrightarrow{A}_2 \cdot (D_2) \\
\overrightarrow{X}_3 = \overrightarrow{X}_\delta - \overrightarrow{A}_3 \cdot (D_3)
\]
where, $\overrightarrow{A_1}$, $\overrightarrow{A_2}$, $\overrightarrow{A_3}$ are randomly generated vectors, and $t$ represents the number of iterations.

In the plane, three points is able to determine a region. Thus, the scope of position of the prey can be determined by the best three wolves. The GWO that whose target solution is comprehensively assessed by three solutions, can greatly decrease the probability of trapping into the local extreme.

It can be seen from the above formula that Eqs (5–7) respectively define the step size of the omega tend to alpha, beta, and delta. The final positions of the omega wolves are defined by Eqs (8–11).

The exploration ability and exploitation ability have important influence on the searching performance of an algorithm. For the GWO, exploration is refers to a wolf leave the original search path in a certain extent and search towards a new direction, which reflects the wolf’s ability of exploiting unknown regions. Exploitation is refers to that a wolf continue to search more carefully on the original trajectory in a certain extent, which can insure the wolf make a detailed search to the region that have been explored. So how to make the algorithm achieve a proper compromise between exploration and exploitation is a question that worth research.

It can be observed that the two random and adaptive vectors $\overrightarrow{A}$ and $\overrightarrow{C}$ can be used to obtain a proper compromise between exploration ability and exploitation ability of the GWO. As is shown in Fig. 3, when $\overrightarrow{A}$ is greater than 1 and is less than $-1$, that is $|\overrightarrow{A}| > 1$, the wolf shows exploration ability. When the value of vector $\overrightarrow{C}$ is greater than 1, it can also enhance the exploration ability of the wolf. In contrast, when $|\overrightarrow{A}| < 1$ and $C < 1$ the
wolf’s exploitation capacity is enhanced. For increasing the exploitation ability of the wolf gradually, the vector $\vec{A}$ decreases linearly with the iterations number increasing. However, in the course of optimization the value of $\vec{C}$ is generated randomly, which can make exploration and exploitation of the wolf reach a equilibrium at any stage. Especially in the final stages of the iteration, it is able to avoid the algorithm from trapping into a local extreme. The pseudo codes of the GWO are described as follows.

1. Initialize the population $X_i (i = 1, 2, \ldots, n)$ of GWO
2. Initialize GWO parameters ($a, A, C$)
3. Calculate the individual fitness value in the population
4. Record the best individual, second best individual and third best individual as $\overrightarrow{X}_{\alpha}$, $\overrightarrow{X}_{\beta}$, and $\overrightarrow{X}_{\delta}$
5. While ($t < \text{maximum iteration number}$)
   1. For each individual
   2. Update the position of current individual by Eqs (5–11)
   3. End for
   4. Update $a$, $A$, $C$
5. Calculate the fitness value of all individual in the population
6. Update $\overrightarrow{X}_{\alpha}$, $\overrightarrow{X}_{\beta}$, $\overrightarrow{X}_{\delta}$
7. $t = t + 1$
8. End While
9. Return $\overrightarrow{X}_{\alpha}$
The GWO has strong exploration ability, which can avoid the algorithm falling into the local optimum. For the GWO, the proper compromise between exploration ability and exploitation ability is very simple to be achieved, so it can effectively solve many complicated problems.

**Improved Grey Wolf Optimizer (IGWO)**

For increasing the search performance of the GWO, an improved grey wolf optimizer (IGWO) is proposed. In the IGWO, the biological evolution and the SOF principle of biological updating of nature are added to the standard GWO. Due to the differential evolution algorithm having the advantages of simple principle, less algorithm parameters and easy implementation, differential evolution (DE) strategy is chose as the evolutionary pattern of wolves in this paper. The wolf pack is updated according to the SOF principle so as to make the algorithm not fall into the local optimum. That is, after each iteration of the algorithm sort the fitness value that corresponds to each wolves in this paper. The wolf pack is updated according to the SOF principle so as to make the algorithm not fall into the local optimum. That is, after each iteration of the algorithm sort the fitness value that corresponds to each

**Grey Wolf Optimizer with evolution operation.** In nature, organisms evolve from the low level to advanced level continually under the action of heredity, selection and mutation. Similarly, there are also a series of changes like heredity, selection and mutation in the searching process of the wolves. The evolution law of SOF makes wolves gradually strong. In the same way, for increasing the searching performance of the algorithm, the evolution operation is added to basic GWO. Based on the biological evolution of nature, many evolution methods have been developed, such as differential evolution (DE), quantum evolution and cooperative evolution, etc. For all the evolution methods, the differential evolution strategy has simple principle, less parameters and easy implementation and it has been extensively researched and applied. Therefore, the DE strategy is chose as the evolution method of GWO. The basic principle of DE operator is to adopt the difference among individuals to recombine the population and obtain intermediate individuals, and then get the next generation population through a competition between parent individual and offspring individual. The basic operations of DE include
three operations: mutation, crossover and selection. After the operation of evolution is added to GWO, the wolf’s position updating is shown as Fig. 2.

Mutation operation. The most prominent feature of differential evolution is mutation operation. When an individual is selected, two differences with weight are added to the individual to accomplish its variation. The basic variation ingredient of DE is the difference vector of the parents, and each vector contains two different individuals \((X_{r_1}^t, X_{r_2}^t)\) of parent (the \(t\)-th generation). The difference vector is defined as follows.

\[
D_{d_{12}} = X_{r_1}^t - X_{r_2}^t
\]

where, \(r_1\) and \(r_2\) express index number of two different individuals of the population. Thus the mutation operation can be described as:

\[
V_{i}^{t+1} = X_{r_1}^t + F \times (X_{r_2}^t - X_{r_3}^t)
\]

where, \(r_1, r_2\) and \(r_3\) are different integers in the scope \((1, 2, \ldots, n)\) from the current target vector index \(i\). \(F\) is the scaling factor to control the scaling of differential vector.

In order to produce an ideal variation factor, to ensure that wolves can evolve toward the direction that good for development of the wolves. So in this paper, chooses outstanding individuals of wolves as parents. After a large number of simulation experiments, beta and delta are chose as two parents, and then combined with the alpha wolf to form a variation factor. Therefore, the variation factor is designed as Eq. (14).
In order to make the algorithm has a high exploration ability in the early stage to avoid falling into local optimum, and has a high exploitation ability in the latter stage to increase the convergence speed, a dynamic scaling factor is employed. So scaling factor $F$ change from large to small according to the iteration number in Eq. (15).

$$V_i^{t+1} = X_i^t + F \times (X_i^t - X_j^t)$$  \hspace{1cm} (14)
| Function | IGWO | SGWO | DGWO | GWO | DE |
|----------|------|------|------|-----|----|
| F1       | 1.5561e−035 | 3.0615e−034 | 1.8785e−034 | 3.4355e−013 | 1.5008e+003 |
| Ave      | 9.5901e−034 | 5.6751e−032 | 1.1231e−032 | 1.4097e−012 | 1.8397e+003 |
| Worst    | 1.8708e−032 | 8.7514e−031 | 6.8749e−032 | 3.6831e−012 | 2.3330e+003 |
| Std      | 7.1710e−034 | 8.0479e−034 | 9.9774e−034 | 2.8268e−012 | 5.5463e+003 |
| F2       | 2.4553e−036 | 6.3490e−034 | 8.1124e−035 | 1.6794e−013 | 98.3138 |
| Ave      | 7.1368e−034 | 2.1103e−031 | 9.6300e−032 | 1.0588e−012 | 4.3298e+003 |
| Worst    | 5.9594e−033 | 2.9801e−030 | 1.2238e−032 | 4.7932e−012 | 1.2236e+004 |
| Std      | 1.6942e−034 | 3.2532e−034 | 3.9658e−034 | 3.1877e−012 | 1.4523e+004 |
| F3       | 23.6476 | 43.0528 | 25.3814 | 65.1383 | 3.5233e+005 |
| Ave      | 679.3675 | 1.6907e+003 | 690.0015 | 818.9336 | 4.1657e+005 |
| Worst    | 2.5006e+003 | 1.1074e+004 | 719.8134 | 5.5568e+003 | 4.7246e+005 |
| Std      | 1.0607e−009 | 1.4133e−008 | 1.2798e−007 | 3.0654e−005 | 3.4534e−004 |
| F4       | 0.0018 | 0.0044 | 0.0035 | 0.0607 | 86.3861 |
| Ave      | 0.0382 | 0.4416 | 0.1585 | 0.9562 | 90.0565 |
| Worst    | 0.6247 | 2.5503 | 1.6033 | 3.1841 | 92.5876 |
| Std      | 0.4072 | 0.4114 | 0.6602 | 0.6513 | 0.5346 |
| F5       | 95.8954 | 96.0948 | 97.0946 | 98.8501 | 1.4284e+006 |
| Ave      | 96.8157 | 97.8475 | 97.5528 | 98.0217 | 2.1571e+006 |
| Worst    | 98.5244 | 98.5851 | 98.4426 | 98.5207 | 3.7347e+006 |
| Std      | 0.6162 | 0.6385 | 0.6579 | 0.6708 | 1.4353e+005 |
| F6       | 7.4830 | 10.3930 | 8.7155 | 9.7480 | 1.2876e+003 |
| Ave      | 9.3215 | 12.0996 | 11.2379 | 12.1379 | 1.1978e+003 |
| Worst    | 12.6824 | 13.8231 | 11.7620 | 12.7860 | 2.3616e+003 |
| Std      | 0.7722 | 0.7882 | 0.9711 | 0.9744 | 1.2354e+003 |
| F7       | 3.5264e−004 | 0.0013 | 6.4848e−004 | 0.0026 | 2.2367 |
| Ave      | 0.0024 | 0.0089 | 0.0077 | 0.0106 | 3.4390 |
| Worst    | 0.0043 | 0.0581 | 0.0240 | 0.0430 | 7.1933 |
| Std      | 7.0880e−004 | 9.1433e−004 | 0.0010 | 0.0021 | 2.3451 |
| F8       | −8.2275e+004 | −8.6793e+003 | −1.9539e+004 | −1.9447e+004 | −1.8290e+004 |
| Ave      | −6.0161e+004 | −7.1528e+003 | −1.6209e+004 | −1.5641e+004 | −1.6567e+004 |
| Worst    | −5.0981e+003 | −6.3427e+003 | −6.4163e+003 | −5.8575e+003 | −1.5388e+004 |
| Std      | 672.4919 | 2.3324e+003 | 3.3000e+003 | 4.6299e+003 | 2.3453e+003 |
| F9       | 0 | 0 | 0 | 4.8431e−011 | 754.1063 |
| Ave      | 1.6643 | 5.7776 | 4.6137 | 9.4635 | 805.4216 |
| Worst    | 12.7963 | 14.7901 | 9.4707 | 30.0252 | 860.7482 |
| Std      | 2.6605 | 3.2557 | 6.3543 | 6.4071 | 764.3451 |
| F10      | 6.4837e−014 | 6.8390e−014 | 6.7837e−014 | 5.9873e−008 | 6.7720 |
| Ave      | 7.7153e−014 | 8.9943e−014 | 8.5771e−014 | 1.1140e−007 | 7.4253 |
| Worst    | 9.3259e−014 | 1.1102e−013 | 8.9153e−014 | 2.2161e−007 | 9.0925 |
| Std      | 6.0990e−015 | 1.0364e−014 | 9.9761e−015 | 6.2628e−008 | 8.3465 |
| F11      | 0 | 0 | 0 | 1.4655e−013 | 6.2536e−004 |
| Ave      | 0.0028 | 0.0057 | 0.0042 | 0.0065 | 0.0091 |
| Worst    | 0.0254 | 0.0702 | 0.0748 | 0.0270 | 0.0420 |
| Std      | 0.0076 | 0.0082 | 0.0090 | 0.0110 | 0.0432 |
| F12      | 0.1056 | 0.1706 | 0.2034 | 0.2154 | 6.4979e+005 |
| Ave      | 0.1551 | 0.3353 | 0.3103 | 0.3968 | 1.5242e+006 |
| Worst    | 0.3331 | 0.4631 | 0.4281 | 0.4306 | 3.1513e+006 |
| Std      | 0.0406 | 0.0427 | 0.0421 | 0.0612 | 2.4325e+006 |

Table 4. Numerical statistics results of D = 100.

\[
F = \frac{f_{\text{min}} + (f_{\text{max}} - f_{\text{min}}) \times \frac{\text{Max_iter} - (\text{iter} - 1)}{\text{Max_iter}}}{\text{iter}}
\]  

where, \(f_{\text{min}}\) and \(f_{\text{max}}\) are the minimum and maximum of the scaling factor, \(\text{Max_iter}\) is maximum iteration number, \(\text{iter}\) is the \(\text{iter}\)-th iteration number.
Crossover operation. For the target vector individual $X_i^t$ of the wolves, make it have a crossover operation with the variation vector $V_i^{t+1}$, and produce a test individual $U_i^{t+1}$. In order to guarantee the individual $X_i^t$ taking place a evolution, a random choice method is adopt to insure at least one bit of $U_i^{t+1}$ is contributed by $V_i^{t+1}$. For other bits of $U_i^{t+1}$, the crossover probability factor CR is used to decide which bit of $U_i^{t+1}$ is contributed by $V_i^{t+1}$, and which bit is contributed by $X_i^t$. Crossover operation is express as follows.

$$U_i^{t+1} = \begin{cases} 
V_i^{t+1} & \text{rand}(j) \leq CR \text{ or } j = \text{rand}(i) \\
X_i^t & \text{rand}(j) > CR \text{ and } j = \text{rand}(i) 
\end{cases} \quad j = 1, 2, \ldots, D$$

(16)

where, $\text{rand}(j) \in [0, 1]$ obeys the random-uniform distribution, $j$ is the $j$-th variable (gene), CR is crossover probability, and $\text{rand}(i) \in [1, 2, \ldots, D]$.

It can be known from the Eq. (16), if the CR is larger, $V_i^{t+1}$ is able to make more contribution to $U_i^{t+1}$. When $CR = 1$, $U_i^{t+1} = V_i^{t+1}$. If the CR is smaller, $X_i^t$ is able to make more contribution to $U_i^{t+1}$.

Selection operation. The “greedy choice” strategy is applied to the selection operation. After mutation operation and crossover operation generate the experiment individual $U_i^{t+1}$ and the compete it with $X_i^t$. It can be expressed as Eq. (17).
where, $f$ is the fitness function, $X_{i}^{t+1}$ is the individual of $t$-th generation. From $U_{i}^{t+1}$ and $X_{i}^{t}$ choose the individual with best fitness as an individual of $(t+1)$-generation, and replace the individual of the $t$-th generation.

In the early stages of the algorithm, difference of the population is large, so mutation operation makes the algorithm has strong exploration ability. In the later stages of the algorithm iteration, namely when the algorithm tends to converge, difference between individuals of the population is small, which makes the algorithm has a strong exploitation ability.

**SOF wolves updating mechanism.** SOF is a role of nature that formed in the process of the biological evolution. In nature, some vulnerable wolves will be eliminated because of uneven distribution of the prey, hunger, disease and other reasons. Meanwhile, new wolves will join to this wolf organization to enhance fighting capacity of the organization, which can insure the wolf organization survival well in the complicated world. The wolf pack is updated according to the SOF principle so as to make the algorithm not fall into the local optimum.

In the new algorithm, assume the number of wolves in the pack is fixed, and the strength of wolves is measured by fitness value. The higher the fitness, the better the solution. Therefore, after each iteration of the algorithm

$$X_{i}^{t+1} = \begin{cases} U_{i}^{t+1}, & f(U_{i}^{t+1}) < f(X_{i}^{t}) \\ X_{i}^{t}, & f(U_{i}^{t+1}) \geq f(X_{i}^{t}) \end{cases} \quad i = 1, 2, \cdots n$$

(17)
sort the fitness value that corresponds to each wolf in ascending order, and then eliminate \( R \) wolves with larger fitness value, meanwhile randomly generate new wolves that equal to the number of eliminated wolves. When \( R \) is large, the number of wolves that new generating is big, which will help to increase the diversity of wolves. But if the value of \( R \) is too large, the algorithm tends to be searching randomly, which will results in the convergence speed becoming slow. If the value of \( R \) is too small, it is not conducive to maintain the diversity of population, which results in the ability of exploring new solution space weakened. Therefore, in this paper, \( R \) is a random integer between \( n/(2 \times \varepsilon) \) and \( n/\varepsilon \).

| Table 5. Numerical statistics results of \( D = 30 \). |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| Function | IGWO | GWO | PSO | ABC | CS |
| F1 | Best: 3.92273e-069 | 2.0291e-029 | 5.1422 | 1.9075e-005 | 5.3943 |
| | Ave: 1.1783e-064 | 1.0402e-027 | 157.3150 | 7.3239e-004 | 17.2298 |
| | Worst: 1.2505e-062 | 6.8810e-027 | 1.3957e+003 | 0.0061 | 39.5997 |
| | Std: 5.5606e-063 | 1.4561e-027 | 618.8885 | 5.5855e-004 | 10.6054 |
| F2 | Best: 7.6878e-068 | 2.0613e-030 | 2.8795 | 6.2539e-006 | 0.0791 |
| | Ave: 3.2484e-064 | 6.5600e-029 | 108.5533 | 2.1544e-004 | 0.2208 |
| | Worst: 8.4385e-063 | 4.8634e-028 | 1.0320e-003 | 0.0019 | 0.9046 |
| | Std: 4.1566e-066 | 1.5256e-028 | 113.0001 | 1.0957e-004 | 0.1073 |
| F3 | Best: 1.7846e-015 | 2.6272e-008 | 2.6128e+003 | 19.2563 | 82.0828 |
| | Ave: 3.6220e-0010 | 1.4099e-005 | 5.8899e+003 | 214.0276 | 388.8103 |
| | Worst: 5.5878e-008 | 1.7721e-004 | 1.5150e+004 | 512.2673 | 892.5678 |
| | Std: 4.3091e-008 | 5.1567e-005 | 2.7730e+003 | 353.0218 | 753.0257 |
| F4 | Best: 4.0775e-014 | 8.7151e-008 | 10.5604 | 34.6791 | 8.2559 |
| | Ave: 4.2932e-013 | 1.0129e-006 | 19.4012 | 65.4533 | 11.5000 |
| | Worst: 7.1302e-011 | 7.7390e-006 | 30.8453 | 82.1336 | 16.9572 |
| | Std: 3.7329e-012 | 3.5107e-007 | 4.9307 | 7.6916 | 1.2514 |
| F5 | Best: 23.1787 | 25.7961 | 582.0313 | 40.4974 | 111.9810 |
| | Ave: 25.8373 | 28.0325 | 1.7004e+004 | 88.0783 | 600.0555 |
| | Worst: 28.5692 | 28.7800 | 8.3990e+004 | 502.2101 | 2.2452e+003 |
| | Std: 0.6997 | 0.9258 | 4.9416e+004 | 29.4371 | 274.0168 |
| F6 | Best: 7.5296e-005 | 8.0083 | 7.6943 | 3.4009 | 8.6597 |
| | Ave: 0.6583 | 0.8993 | 300.3189 | 8.3567 | 24.2355 |
| | Worst: 2.1169 | 1.5192 | 1.1916e+003 | 23.0030 | 75.3820 |
| | Std: 0.2917 | 0.4055 | 374.9472 | 3.2274e-004 | 12.7746 |
| F7 | Best: 1.2695e-004 | 4.5944e-004 | 0.1186 | 0.0319 | 0.0177 |
| | Ave: 7.8361e-004 | 0.0022 | 0.6129 | 0.0528 | 0.0621 |
| | Worst: 0.0017 | 0.0064 | 2.2257 | 0.8170 | 0.1222 |
| | Std: 3.7090e-004 | 8.4619e-004 | 0.3202 | 0.0026 | 0.0159 |
| F8 | Best: 3.9429e-004 | 6.6130e+003 | 4.1322e+003 | 1.1681e+003 | 6.6092e+003 |
| | Ave: 9.0105e+003 | 3.6813e+003 | 3.4422e+003 | 1.1294e+003 | 5.9192e+003 |
| | Worst: 3.3482e+003 | 2.9262e+003 | 2.9516e+003 | 1.0866e+003 | 5.2163e+003 |
| | Std: 1.7921e+003 | 9.580854 | 3.2960e+003 | 205.6144 | 351.8938 |
| F9 | Best: 0 | 1.1369e-013 | 46.3981 | 5.1504 | 53.0900 |
| | Ave: 1.0783 | 3.2143 | 92.6839 | 9.0702 | 73.5890 |
| | Worst: 12.1696 | 15.8356 | 152.7472 | 13.8016 | 103.0096 |
| | Std: 1.5038 | 4.8809 | 22.7852 | 2.5126 | 14.3049 |
| F10 | Best: 1.5099e-017 | 7.5495e-013 | 2.0863 | 1.6725 | 4.1810 |
| | Ave: 1.2204e-016 | 1.0048e-012 | 5.7135 | 5.0268 | 7.0560 |
| | Worst: 2.2204e-016 | 1.4655e-013 | 10.5829 | 11.2607 | 14.1411 |
| | Std: 4.3110e-015 | 1.4373e-014 | 2.4526 | 1.7819 | 2.6353 |
| F11 | Best: 0 | 0 | 1.1890 | 3.3792e-005 | 1.0869 |
| | Ave: 0.0016 | 0.0048 | 7.4790 | 0.0649 | 1.2128 |
| | Worst: 0.0047 | 0.0286 | 26.8061 | 0.1863 | 1.5050 |
| | Std: 0.0074 | 0.0114 | 3.9852 | 0.0431 | 0.1070 |
| F12 | Best: 0.0065 | 0.0188 | 1.4887 | 1.8107 | 1.9495 |
| | Ave: 0.0481 | 0.0594 | 73.9029 | 4.5941 | 3.6666 |
| | Worst: 0.1091 | 0.0819 | 2.0290e+003 | 6.8171 | 6.5449 |
| | Std: 0.0151 | 0.0419 | 4.4614 | 0.3128 | 0.4337 |
| Function | IGWO       | GWO         | PSO         | ABC         | CS          |
|----------|------------|-------------|-------------|-------------|-------------|
| F1       | Best 1.5561e-035 3.4355e-013 3.7127e+004 122.6512 3.4168e+003 | Ave 9.5901e-034 1.4097e-012 4.9247e+004 4.0245e+003 5.7296e+003 | Worst 1.8708e-032 3.6831e-012 6.5987e+004 1.0067e+004 8.6850e+003 | Std 7.1710e-034 2.8268e-012 8.9305e+003 1.8523e+003 912.9310 |
| F2       | Best 2.4533e-036 1.6794e-013 2.6127e+004 91.5939 1.3798e+003 | Ave 7.1368e-034 1.0558e-012 3.7423e+004 3.9368e+003 2.0391e+003 | Worst 5.9594e-033 4.7932e-012 5.6874e+004 1.1487e+004 3.0315e+003 | Std 1.6942e-034 3.1877e-012 7.9858e+003 2.1492e+003 514.5256 |
| F3       | Best 23.4747e-036 65.1383 1.1274e+005 2.9201e+004 3.0971e+005 | Ave 67.3675 81.9336 1.8855e+005 1.2577e+005 5.4593e+005 | Worst 2.5006e+003 5.5568e+003 3.9436e+005 8.1136e+005 7.4755e+005 | Std 1.0067e-009 3.0654e-005 2.6625e+03 1.6236e+004 1.7767e+004 |
| F4       | Best 0.0018 0.0607 43.1594 90.3059 22.7549 | Ave 0.0382 0.9562 54.6512 95.0132 29.3341 | Worst 0.6247 3.1841 66.4211 98.3757 39.2722 | Std 0.4072 0.6513 4.8263 1.6038 2.8819 |
| F5       | Best 95.8954 96.8501 2.6394e+007 8.4200e+003 4.2931e+005 | Ave 96.8157 98.0217 5.7355e+007 7.8186e+005 1.0015e+006 | Worst 98.5244 98.5207 1.3108e+008 9.2994e+006 2.4179e+006 | Std 0.6162 0.6708 1.3951e+07 1.9970e+005 3.2088e+005 |
| F6       | Best 7.4830 9.7480 2.1793e+004 210.6135 4.2828e+003 | Ave 9.3215 12.1379 3.4705e+004 4.4723e+003 5.7352e+003 | Worst 12.6824 12.7860 4.6567e+04 1.1812e+04 8.5043e+04 | Std 0.7722 0.9744 4.9192e+03 2.5176e+03 1.0449e+03 |
| F7       | Best 3.5264e-004 0.0026 28.9078 7.9176 0.4508 | Ave 0.0024 0.0066 92.8878 10.3561 0.7003 | Worst 0.0043 0.0130 236.6874 11.2352 1.0116 | Std 7.0880e-004 0.0021 47.0314 4.3578 0.1182 |
| F8       | Best −8.2275e+004 −1.9447e+004 −1.6506e+004 −1.6103e+004 −1.2130e+004 | Ave −6.0161e+004 −1.5614e+004 −5.5154e+003 −1.0941e+004 −1.0789e+004 | Worst −5.0981e+003 −5.5875e+003 −1.7081e+003 −2.7028e+004 −9.3055e+003 | Std 672.4919 4.6329e+03 4.4865e+03 582.5893 643.1670 |
| F9       | Best 0 4.8431e+011 722.3155 243.0193 350.2439 | Ave 1.6643 9.4635 877.8493 293.0193 428.6253 | Worst 12.7963 30.0252 1.0625e+03 342.8732 493.4137 | Std 2.6605 6.4071 86.9610 21.9504 38.0722 |
| F10      | Best 6.4837e−014 5.9873e−008 15.8294 8.2314 10.9187 | Ave 7.7153e−014 1.1140e−007 17.0826 11.2388 13.3370 | Worst 9.3259e−014 2.2161e−007 20.7326 15.7326 17.6444 | Std 6.0990e−015 6.2626e−008 0.6998 1.6529 1.9379 |
| F11      | Best 0 1.4655e−013 301.0689 2.2155 38.0069 | Ave 0.0028 0.0065 430.3317 29.6147 54.1621 | Worst 0.0234 0.0270 580.6552 76.0269 75.9006 | Std 0.0076 0.0110 52.2674 15.3837 11.3528 |
| F12      | Best 0.1056 0.2154 8.6436e+006 20.1133 20.6985 | Ave 0.1551 0.3968 3.2730e+007 1.3921e+004 6.5202e+003 | Worst 0.3331 0.4306 9.5779e+007 1.4818e+005 8.1864e+004 | Std 0.0406 0.0612 4.3908e+007 3.2646e+004 0.8511 |

Table 6. Numerical statistics results of D = 100.

\[ R = [n/e, n/0.75 \times e] \]  

where, \( n \) is the total number of wolves, \( e \) is scale factor of wolves updating.

The flow chart of the improved grey wolf optimizer (IGWO) is illustrated in Fig. 4. The main procedure steps are described as follows.

1. Initialize the grey wolf population. Randomly generated position of wolves \( X(i = 1, 2, \ldots n) \) Initialize parameters \( a, A \) and \( C \).
(2) Calculate fitness of each wolf, choose the first three best wolves and save them as alpha beta and Delta in turn.

(3) Update position. According to Eqs (5–11) update position of the other wolves, that is, update position of omega wolf.

(4) Evolution operation. Use alpha, beta, and delta to form variation factor according to the Eq. (14). After crossover and selection operation, select the individual with good fitness as the wolf of next generation. Choose the first three best wolves and save them as alpha beta and Delta in turn.

(5) Update wolves. Sort fitness values that correspond by wolves from small to large, eliminate R wolves with larger fitness value. Meanwhile, randomly generate new R wolves.

(6) Update parameter \(a\), \(A\) and \(C\).

(7) Judge whether the termination condition is satisfied, if satisfied, output the position and fitness value of alpha as the optimal solution. If not satisfied, return to step (2).

**Simulation Experiments and Results Analysis**

Before carrying out the simulation experiments to compare the performances of adopted optimization algorithm, twelve benchmark functions are selected, which are listed in Table 1. The experiment consists of two parts. For validating the performance of two improvements to the GWO, one part is that separately do experiments for the GWO with differential evolution (DGWO), GWO with SOF mechanism (SGWO) and IGWO. And meanwhile, compare results with DE algorithm. The second part is that do experiments to compare IGWO with other swarm
Figure 19. Convergence curves Function $F_8$.

Figure 20. Convergence curves Function $F_9$.

Figure 21. Convergence curves Function $F_{10}$. 
Three swarm intelligence algorithms (PSO algorithm, ABC algorithm and CS algorithm) are selected to carry out the simulation contrast experiments with the proposed IGWO so as to verify its superiority on the convergence velocity and searching precision. When dimension $D = 30$, the simulation convergence results on the adopted testing functions are shown in Figs 12–23. When $D = 100$, the simulation convergence results on the adopted testing functions are shown in Figs 24–27.
It can be seen from the simulation convergence curves, IGWO has better convergence velocity and searching precision than other three algorithms (ABC, CS and PSO). Especially for function F8, F9 and F12, compared with GWO, the IGWO make their convergence speed are improved obviously. It can be seen from their surface figure that the three functions are multimodal function. So there are a lot of local minimum values within the scope of search. For IGWO, the addition of differential evolution and the SOF mechanism can improving the weakness of easily falling into the local extreme and obtain smaller function value. Thus, the convergence speeds of the three functions are improved significantly. At the same time, it suggests that the IGWO has the superiority to jump out of the local optimal.

For further validating the searching accuracy, every optimization algorithm is run independently thirty times and the best, worst and average values are recorded for the adopted twelve testing functions under the 30-dimension and 100-dimension. The maximum iterations number is Max_iter = 500. The statistical results of D = 30 and D = 100 are shown in Tables 5 and 6 respectively. Seen from the numerical results listed in Tables 5 and 6, the IGWO proposed in this paper makes optimization accuracy of 12 typical functions have a certain improvement. And the optimization accuracy of IGWO is better than that of PSO, ABC, CS algorithm under the same dimension. For the F1 function, when its dimension D = 30, its average value is raised from e-27 to e-63, relative to GWO been increased by 36 orders of magnitude; and when D = 100, its average value is raised from e-12 to e-34, relative to GWO increased by 22 orders of magnitude. For function F2, F9 and F11, when D = 30 and D = 100, IGWO respectively search to their ideal minimum value 0. The optimization performance for other testing functions all has been improved.

Thus, the simulation experiments results, including convergence curves and statistics data, show that the proposed improved grey wolves optimizer has a better convergence rate and optimization performance. According
to the improvement above, three reasons can be summarized: Firstly, the adding of evolution operation is able to increase the diversity of wolves, that is to say to increase the solution diversity so as to make the algorithm jump out the local extreme. Secondly, the selection method of parents of DE and dynamic scaling factor $F$ can make the algorithm has a good exploration ability in the early search stage and has a good exploitation ability in the later search stage, therefore, both search precision and convergence speed are improved. In addition, the adding of SOF wolf updating mechanism can also decrease the probability of the algorithm falling into the local extreme.

**Conclusions**

For achieving the proper compromise between exploration and exploitation, further accelerate the convergence and increase the optimization accuracy of GWO, an improved grey wolf optimizer (IGWO) is proposed in this paper. The biological evolution and SOF principle in nature are added to the standard GWO. The simulation experiments are carried out by adopting twelve typical function optimization problems. The simulation results show the proposed IGWO has better convergence velocity and optimization performance than DE algorithm, PSO algorithm, ABC algorithm and CS algorithm. On the one hand, the adoption of evolution operation can increase the wolves diversity and make the algorithm has a good exploration ability in the early searching stage and has a good exploitation ability in the later search stage. On the other hand, the adoption of SOF wolf updating mechanism can decrease the probability of falling into the local optimum. In the future work, we will carry out similar hybridisation of other swarm intelligent optimization algorithms, such as dragonfly algorithm (DA), ant lion optimizer (ALO), multi-verse optimizer (MVO), coral reefs optimization (CRO) algorithm, etc.

**Data Availability**

There are no data available for this paper.
References

1. Derenc, J., García, S. & Molina, D. A practical tutorial on the use of non-parametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. J. Swarm and Evolutionary Computation. 1, 3–18 (2011).

2. Cui, Z. H. & Gao, X. Z. Theory and applications of swarm intelligence. J. Neural Computing and Applications. 21, 205–206 (2012).

3. Zhang, Z. et al. On swarm intelligence inspired self-organized networking: its bionic mechanisms, designing principles and optimization approaches. J. Communications Surveys & Tutorials IEEE. 16, 513–537 (2014).

4. Leboucher, C. et al. A swarm intelligence method combined to evolutionary game theory applied to the resources allocation problem. J. International Journal of Swarm Intelligence Research (IJISIR). 3, 20–38 (2012).

5. Parpinelli, R. S. & Lopes, H. S. New inspirations in swarm intelligence: a survey. J. International Journal of Bio-Inspired Computation. 3, 1–16 (2011).

6. Kennedy, J. & Eberhart R. Particle swarm optimization, in Neural Networks, 1995. Proceedings., IEEE International Conference on. pp. 1942–1948 (1995).

7. Ebrahimi, J., Hosseinian, S. H. & Gharahpetian, G. B. Unit commitment problem solution using shuffled frog leaping algorithm. J. Power Systems, IEEE Transactions on. 26, 573–581 (2011).

8. Fang, C. & Wang, L. An effective shuffled frog-leaping algorithm for resource-constrained project scheduling problem. J. Computers & Operations Research. 39, 890–901 (2012).

9. Karaboga, D. & Basturk, B. On the performance of artificial bee colony (ABC) algorithm. J. Applied soft computing. 8, 687–697 (2008).

10. Dorigo, M., Birattari, M. & Stützle, T. Ant colony optimization. IEEE Computational Intelligence Magazine. 1, 28–39 (2006).

11. Simon, D. Biogeography-based optimization. Evolutionary Computation, IEEE Transactions on. 12, 702–713 (2008).

12. Yang, X. S. & Deb, S. Cuckoo search via Lévy flights. A. Proceedings of World Congress on Nature & Biologically Inspired Computing. C. IEEE Publications. USA, 210–214 (2009).

13. Mirjalili, S., Wang, G.-G., dos, L. & Coelho, S. Binary optimization using hybrid particle swarm optimization and gravitational search algorithm. J. Neural Computing and Applications. 25, 1423–1435 (2014).

14. Crepţniţ, M., Liu, S. H. & Marjan, M. Exploration and exploitation in evolutionary algorithms: a survey. ACM Comput. Surv. 45, 1–33 (2013).

15. Mirjalili, S., Mirjalili, S. M. & Lewis, A. Grey wolf optimizer. J. Advances in Engineering Software. 69, 46–61 (2014).

16. Brambilla, M. et al. Swarm robotics: a review from the swarm engineering perspective. J. Swarm Intelligence. 7, 1–41 (2013).

17. Liao, J. et al. A neighbor decay cellular automata approach for simulating particle swarm evolution. J. International Journal of Geographical Information Science. 28, 720–738 (2014).

18. Das, S. & Suganthan, P. N. Differential evolution: a survey of the state-of-the-art. J. Evolutionary Computation, IEEE Transactions on. 15, 4–31 (2011).

19. Palmer, K. et al. Differential Evolution of Cognitive Impairment in Nondemented Older Persons: Results From the Kungsholmen Project. J. Psychiatry. 159, 436–442 (2014).

20. Agrawal, R. et al. Parallelization of industrial process control program based on the technique of differential evolution using multi-threading(C/)[Industrial Engineering and Engineering Management (IEEM), 2014 IEEE International Conference on. IEEE, 546–550 (2014).

21. Islam, S. M. et al. An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization. J. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on. 42, 482–500 (2012).

22. Zheng, Y. J. et al. A hybrid fireworks optimization method with differential evolution operators. J. Neurocomputing. 148, 75–82 (2015).

23. Zhang, P., Wei, Y. X. & Xue, H. Q. Heterogeneous Multiple Colonies Ant Colony Algorithm Based on Survival of Fittest Rules. J. Computer Engineering. 38, 182–185 (2012).

24. Byler, C. E. & Rich, J. N. Survival of the fittest: cancer stem cells in therapeutic resistance and angiogenesis. J. Journal of Clinical Oncology. 26, 2839–2845 (2008).

25. Whitley, D. An executable model of a simple genetic algorithm. J. Foundations of genetic algorithms. 2, 45–62 (2014).

26. Vidal, T. et al. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. J. Computers & Operations Research. 40, 475–489 (2013).

27. Kristian, B. & Frédéric, R. N. Survival of the Fittest in Cities: Urbanisation and Inequality. J. The Economic Journal. 124, 1371–1400 (2014).

28. Bak, P. & Sneppen, K. Punctuated equilibrium and criticality in a simple model of evolution. Physical Review Letters. 24, 4083–4086 (1993).

29. Boettcher, S. & Percus, A. Nature’s way of optimizing. Artificial Intelligence. 1–2, 275–286 (2000).

Acknowledgements

This work was supported by the Project by National Natural Science Foundation of China (Grant No. 21576127), the Basic Scientific Research Project of Institution of Higher Learning of Liaoning Province (Grant No. 2017FWDF10), and the Project by Liaoning Provincial Natural Science Foundation of China (Grant No. 20180550700).

Author Contributions

Jie-Sheng Wang participated in the concept, design, interpretation and commented on the manuscript. A substantial amount of Shu-Xia Li’s contribution to the data collection, analysis and algorithm simulation, draft writing, and critical revision of this paper was undertaken.

Additional Information

Competing Interests: The authors declare no competing interests.

Publisher’s note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/.

© The Author(s) 2019