Adaptive Gray Wolf Optimization Algorithm based on Gompertz Inertia Weight Strategy

Qiuhua Pan
Institute of Mathematical Sciences, Shanghaitech University, Shanghai 201210, China

***Abstract*—To solve the problems that the Gray Wolf Optimizer (GWO) convergence speed is not fast enough and the solution accuracy is not high enough, this paper proposes an Adaptive Gray Wolf Optimizer based on Gompertz inertia weighting strategy (GGWO). GGWO uses the characteristics of the Gompertz function to achieve nonlinear adjustment of the inertia weight, which better balances the speed of global search and accuracy of local search of the GWO algorithm. At the same time, the Gompertz function is used to realize the adaptive adjustment of the individual gray wolf's position and to better update the gray wolves' position according to the fitness values of different gray wolf individuals. Use 6 classic test functions to compare the performance of GGWO in optimization and 10 other classic or improved swarm intelligence algorithms. Results show that GGWO has better solution accuracy, stability, and faster convergence than all other 10 swarm intelligence algorithms.**

Keywords—Gray wolf optimization algorithm; inertia weight; adaptive; Gompertz function; swarm intelligence algorithm

I. Introduction

The world is full of unknowns, and there are many uncertain problems in the unknown world. There is a lot of uncertain information that needs to be processed. To represent and process uncertain information, many optimization problems arise. Many optimization algorithms have emerged as the times require. In the fields of applied mathematics and engineering, there are a large number of optimization problems, and the computational solutions of these problems are located in a complex solution space. Therefore, finding new optimization methods with fast computation speed and strong convergence ability is of great practical significance. With the development of digitization and informatization, more and more meta-heuristic algorithms are being applied in the fields of science and engineering. Metaheuristic algorithms have the characteristics of self-organization, compatibility, parallelism, holism, and coordination. Their working principle is to initialize a set of random solutions and then perform repeated feedback iterations to approach the expected goal. In this search mechanism, the algorithm only needs to know the objective function and search range and can obtain the target solution regardless of whether the search range is continuous and differentiable. Metaheuristic algorithms are mainly divided into three types: biological evolution, natural phenomena, and species' living habits. The swarm intelligence optimization algorithm can specifically represent unknown information. The swarm intelligence algorithm is a model with optimization as its task. The swarm intelligence algorithm is generally based on imitating the living methods and behavioral habits of natural organisms and can optimize problems and methods with special methods. The iterative process of swarm intelligence algorithms is generally based on feedback from individuals within the population, such as particle swarm optimization, where all individuals in the population refer to the position information of the globally optimal individual to move. The cuckoo algorithm simulates the behavioral habits of cuckoo chicks and updates position information based on Levy flight. The grey wolf optimization algorithm is based on the hunting behavior habits of the grey wolf population, and the position updates of all individuals in the population are based on the globally optimal positions of three grey wolves. Researchers have proposed many solutions and drawn many important conclusions and achievements in exploring the performance improvement of swarm intelligence algorithms.

In the expansive domain of optimization algorithms, methodologies that diligently seek optimal or near-optimal solutions have demonstrated their indispensable value across a spectrum of disciplines and fields. These algorithms are not merely instrumental in enhancing the efficiency and performance of systems, designs, or models but also play a pivotal role in decision-making, resource allocation, and quality improvement. In various domains, such as machine learning, engineering, economics, logistics, biology, and computer science, optimization algorithms facilitate optimal parameter adjustments, design, resource distribution, and experimental design, thereby crafting a robust platform for interdisciplinary communication and collaboration. While these instances merely skim the surface regarding the application of optimization algorithms across disciplines, their profound impact and extensive connections undeniably propel the continuous progression and development of scientific technology. Transitioning from the general landscape of optimization algorithms, the Gray Wolf Optimizer (GWO) warrants specific attention, presenting its unique methodology in the rich field of optimization.

Researchers have been inspired by long-term observations of the social interactions, lifestyles, and biological behaviors of organisms such as fish, ants, elephants, wolves, and bees in nature, and have developed a series of related optimization algorithms to solve many practical problems such as engineering optimization and power dispatch. The GWO algorithm is implemented by utilizing the intelligence of gray wolves and their collective hunting characteristics. The GWO algorithm simulates a group of wolves following a specific hierarchical pattern, with different categories of wolves (named alpha, beta, delta, and omega) playing different roles in the hunting mechanism to achieve search and hunting purposes.

II. Literature Review

GWO is a swarm intelligence optimization algorithm proposed by Mirjalili. This algorithm is derived from the hunting mechanics and leadership levels of gray wolves in the natural world. Mirjalili [1] used GWO to train the multi-layer perceptron (MLP). It's found that GWO has higher accuracy and is competitive in avoiding local optimality. Mohanty [2] completed the maximum power point tracking design of photovoltaic PV systems by GWO. Mirjalili [3] proposed the multi-objective gray wolf optimizer (MOGWO) in 2016, which is very effective in solving multi-objective optimization problems. Emary [4] proposed a binary version called bGWO for better feature selection. Heidari [5] integrated Levy flight and greedy algorithms to obtain a new LGWO algorithm to improve optimization performance. Faris [6] also discussed different versions of GWO optimization in detail, divided them into modified versions, hybrid versions, and parallel versions, and analyzed their role in the main application fields. Kohli [7] introduced chaos theory into GWO, which greatly improved the global convergence speed. Gupta [8] proposed an improved algorithm called RWGWO rooted in random walks to solve optimization problems in life. Nadimi [9] proposed an efficient gray wolf optimizer (I-GWO) based on the dimensional learning hunting (DLH) search strategy. Al [10] combined GWO with PSO to implement a new BPSOGWO binary algorithm to find the best feature subset.

GWO transcends its foundational application in mathematical function optimization, demonstrating utility across diverse domains. Altan [11] used GWO to optimize the intrinsic model function output and efficiently utilize wind energy. Zhao [12] used GWO to search feature sets to improve the diagnostic accuracy of patients with paraquat poisoning. Jayabarathi [13] also used GWO to solve economic dispatch problems. Sulaiman [14] applied GWO to solve the power deployment problem (ORPD) in the power system. Shariati [15] created a model combining a hybrid extreme learning machine with GWO to forecast the strength of concrete if replaced partially by cement. Jino [16] applied optimization algorithms like GWO in advanced image processing fields. Ramakrishnan [17] uses MRG-GWO for segmentation in an estimate of the CT brain tumor images. The GWO algorithm can help find optimal or suboptimal scheduling solutions, Jiang [18] used GWO to solve cases of scheduling Job Shop and Flexible Job Shop. Wei [19] improved SVM by using GWO and applied it in predicting the second major. Yang [20] used grouped GWO to optimize the parameters of wind turbines and improve the maximum power and obstacle-breaking capability.

GWO has more application scenarios including machine learning, image processing, and engineering design.

Within the machine learning sphere, SVM's (Support Vector Machines) potential is often bound by the intricacies of parameter optimization. Zhou's research [21] in 2021 elucidated this challenge by introducing two models. While both models aimed at earthquake forecasting, the latter, harnessing the capabilities of GWO, showcased commendable performance. This reinforces GWO's capability for effective exploration and exploitation in parameter optimization.

In the domain of image processing, particularly image segmentation, achieving a synergy of quality and efficiency remains pivotal. Khairuzzaman's work [22] in 2017 offers a paradigm in this regard. By leveraging GWO for multilevel thresholding, the research underscored GWO's adaptability in delivering quality segmentation with computational expediency.

Transitioning to engineering design, particularly in wind energy optimization, the nuances of turbine efficiency stand paramount. Yang's 2017 study [20] on the optimization of Maximum power points for wind turbines, specifically those operating on doubly-fed induction generators, provides a testament to GWO's efficacy. The introduction of Grouped GWO in this context exemplifies the optimizer's finesse in adaptive parameter adjustments for enhanced energy outputs.

Beyond its established domains, GWO has shown remarkable adaptability in addressing various complex optimization problems, encompassing both classical combinatorial issues and cutting-edge applications.

A quintessential example is the Traveling Salesman Problem (TSP). Given the complexity of determining the shortest possible route that visits each city exactly once and returns to the origin, Panwar's contribution is notable. In 2021, Panwar [23] employed a discrete GWO approach, paving the way for efficient solutions to symmetric TSP instances. This application not only accentuates GWO's versatility but also underscores its potential in combinatorial optimization.

Furthermore, in the domain of software engineering, predicting software defects based on metrics data is a crucial task. GWO's application in Software Defect Prediction (SDP) focuses on optimizing both feature selection and classifier parameters. One intriguing application by Kermadi [24] highlighted GWO's efficacy in designing an efficient photovoltaic array hybrid maximum power point tracker, specifically tailored for intricate local shading conditions in SDP.

Multi-objective optimization problems (MOP) present another challenging arena, where the goal is to find non-inferior solutions across multiple objectives. Wu's research in 2020 [25] exemplifies this by integrating GWO with other objective optimizations for wind speed forecasting. This innovative approach, leveraging GWO's capabilities, orchestrates harmony among various objectives, generating superior solutions.

It further reveals novel applications of GWO in electrical and control systems. Lakum [26] employed GWO for optimally placing and sizing active power filters in radial systems, especially amidst nonlinear-distributed generation. Arora's work [27] ventured into algorithmic hybridization, combining GWO with the Crow Search Algorithm (CSA) for enhanced function optimization and feature selection. Sun's research [28], on the other hand, utilized GWO for the intricate task of feedback control optimization in PM hub motors. Moreover, Eltamaly's study [29] stands out in harnessing GWO-FLC to track dynamic maximum power points (MPP) for PV systems under variable shading conditions. Jamal employs the Improved Grey Wolf Optimization (IGWO) algorithm to optimize overcurrent relay coordination, demonstrating its enhanced efficiency and reliability over conventional methods [30]. Telugu utilizes the Chaos-enhanced Grey Wolf Optimization Algorithm (CGWO) for designing a two-stage CMOS Differential Amplifier, achieving significant improvements in reducing circuit size and power dissipation compared to traditional optimization methods [31].

The diverse applications of GWO demonstrate its adaptability and robustness across various research arenas, addressing distinct challenges and expanding its applicability in optimization landscapes. Its multifaceted use across sectors showcases its versatility in delivering optimal solutions. However, current improvement ideas often blend multiple algorithms, facing issues like low accuracy, slow convergence speed, and poor stability. This article suggests a streamlined and efficient enhancement plan solely based on the GWO algorithm, aiming to tackle these issues.

Section III introduces the background and basic principles of the GWO algorithm; Section IV introduces the two strategies of inertia weight and adaptive weight respectively, and uses them for GWO position update, thereby obtaining an adaptive algorithm based on Gompertz inertia weight(GGWO); Section V uses simulation experiments to compare and analyze the convergence performance, convergence speed, stability and time complexity of 11 algorithms including GWO and GGWO from three dimensions and six test functions; finally Section VI summarizes the full text and looks forward to the diverse application scenarios of GGWO in the future.

III. Gray Wolf Optimization Algorithm

GWO simulates the hunting mechanism and leadership levels of gray wolves by dividing these wolves into four layers according to the characteristics of gray wolves. The first layer is the α layer, the leaders of the population, in charge of leading the rest wolves to hunt prey, which is interpreted as the optimal solution in the algorithm. The second layer is the β layer, responsible for assisting the α layer wolf pack, and this layer interprets the sub-optimal solution in GWO. The third layer is the δ layer, which should obey any orders or decisions made by the previous two α and β and they have to investigate more etc. The grading mechanism of gray wolf packs is not static. Some of the α and β with poor fitness will degenerate to δ. The fourth layer is called the ω layer, which updates its position according to the previous three α, β, or δ. Furthermore, these four layers of wolves α, β, δ, and ω cooperate in the hunting. There are in total three main stages of gray wolf hunting, and they are simulated specifically as surrounding prey, chasing prey, and attacking prey.

In the first stage of surrounding the prey, the gray wolf will update its position based on the position information of the prey and gradually surround the prey, as shown in Eq. (1):

$$\vec{X}(\mathrm{t}+1) = \overrightarrow{X_p}(\mathrm{t}) - \vec{A}\cdot\vec{D} \tag{1}$$

where X stands for the position vector of the gray wolf, t represents the number of iterations, X_p

is the position vector of the prey, and the parameters A and D as shown in Eq. (2), (3):

$$\vec{A} = 2\vec{a}\vec{r_1} - \vec{a} \tag{2}$$

$$\vec{D} = |\vec{C}\cdot\overrightarrow{X_p}(\mathrm{t}) - \vec{X}(\mathrm{t})| \tag{3}$$

Among them, $\vec{a}$ linearly decreases from 2 to 0 during iteration, $\vec{r}_1$ is a random vector on [0,1], and

the parameter C is as follows in Eq. (4):

$$\vec{C} = 2\vec{r_2} \tag{4}$$

where is a random vector.

While hunting prey, the behaviors of all the gray wolves are guided by α, β and δ gray wolves and these three layers of wolves may also cooperate with ω gray wolves in hunting. To better simulate and reproduce the hunting strategy of gray wolves, we suppose that α, β, and δ have a better idea of the potential position of prey. Via sorting the fitness values of all the wolves, the best three layers of wolves are chosen as α, β, and δ respectively. The specific gray wolf location update steps are as follows:

First, the corresponding $\vec{D}_\beta$ and $\vec{D}_\delta$ are as shown in Eq. (5):

$$\vec{D}_\alpha = |\vec{C_1}\cdot\overrightarrow{X_\alpha} - \vec{X}| \quad \vec{D}_\beta = |\vec{C_2}\cdot\overrightarrow{X_\beta} - \vec{X}| \quad \vec{D}_\delta = |\vec{C_3}\cdot\overrightarrow{X_\delta} - \vec{X}| \tag{5}$$

Then, solve the position vector $\vec{X}(\mathrm{t}+1)$ of the current gray wolf in the next iteration, as shown in Eq. (6) and (7):

$$\overrightarrow{X_1} = \overrightarrow{X_\alpha} - \vec{A_1}\cdot\overrightarrow{D_\alpha} \quad \overrightarrow{X_2} = \overrightarrow{X_\beta} - \vec{A_2}\cdot\overrightarrow{D_\beta} \quad \overrightarrow{X_3} = \overrightarrow{X_\delta} - \vec{A_3}\cdot\overrightarrow{D_\delta} \tag{6}$$

$$\vec{X}(\mathrm{t}+1) = \frac{\overrightarrow{X_1}+\overrightarrow{X_2}+\overrightarrow{X_3}}{3} \tag{7}$$

In summary, this algorithm outlines a structured process for GWO, iterating through defined steps to ascertain optimal solutions by emulating the hierarchical and behavioral dynamics of gray wolves.

IV. Adaptive Gray Wolf Optimization Algorithm based on Gompertz Inertia Weight Strategy

A. Gompertz Inertia Weight Strategy

The Gompertz function [32] is monotonic, and its function expression is as shown in Eq. (8):

$$y = -e^{-e^{-x}} + 1 \tag{8}$$

Draw the graph of the Gompertz function as shown in Fig. 1:

As shown in Fig. 1, the Gompertz function curve is characterized by slow growth in the initial and final stages and rapid growth in the middle section. The image of the

Gompertz function tends to decrease as the abscissa increases, which is related to the iterative process of swarm intelligence algorithms. In the early stages of swarm intelligence algorithm iteration, the population is prone to falling into local optima, so it is necessary to give a larger step size initially. Giving a larger inertia weight can increase the step size of individual gray wolf movements, thereby helping the population to better conduct global search. As the algorithm iterates, the needs of individual populations gradually shift from global optimization to local optimization. In the later stages of the algorithm, some individuals need to strive to explore global optima within a small range, so giving a smaller step size in the later stages of the algorithm can help the algorithm perform local optimization. The Gompertz function has this feature, as its value decreases as the abscissa increases, which helps our algorithm balance global optimization and local optimization. This article uses it to improve the inertia weight of the GWO. The Gompertz inertia weight ω used in this article is as follows in Eq. (9):

$$w = -e^{-e^{-\frac{t}{M}}} + 1 \tag{9}$$

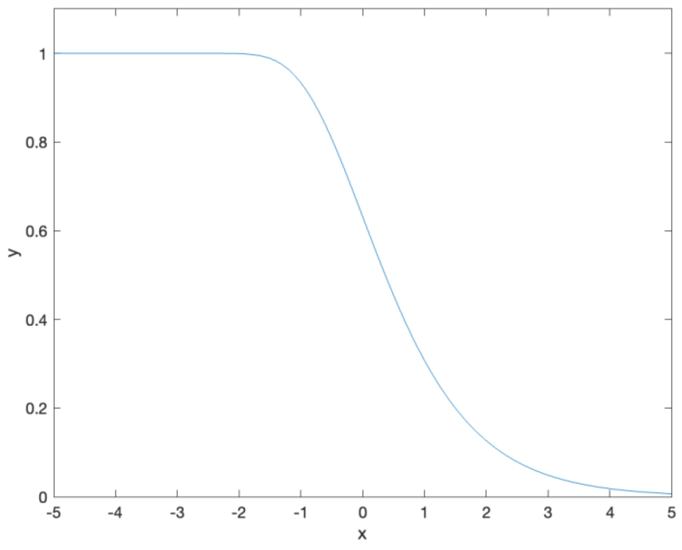


Fig. 1. Gompertz function.

The graph of selecting the right side of the y-axis as the inertia weight ω of the GWO algorithm is shown in Fig. 2:

As shown in Fig. 2, the Gompertz inertia weight remains large in the early stages of iteration, which is conducive to maintaining a large search range of the algorithm, making the algorithm less likely to fall into the local optimal solution; as the number of iterations of the algorithm increases, the curve in the middle section will decrease rapidly and eventually stabilize at a smaller value, which will help the algorithm have a smaller inertia weight in the later stages of the iteration, which will help find the optimal solution more thoroughly in local area and during long periods.

B. *Gompertz Adaptive Position Update Strategy*

Gompertz function is used to construct the adaptive weight Φ_i strategy as shown in Eq. (10):

$$\Phi_i = -e^{\frac{f_i}{f_{avg}}+1-e^{\frac{f_i}{f_{avg}}}} + 1, (i = 1, 2, 3) \tag{10}$$

Among them, f_1, f_2, and f_3 represent the fitness of the α, β and δ layer wolves respectively. f_{avg} is as shown in Eq. (11):

$$f_{avg} = \frac{f_1 + f_2 + f_3}{3} \tag{11}$$

Gompertz adaptive weight Φ_i is shown in Fig. 3:

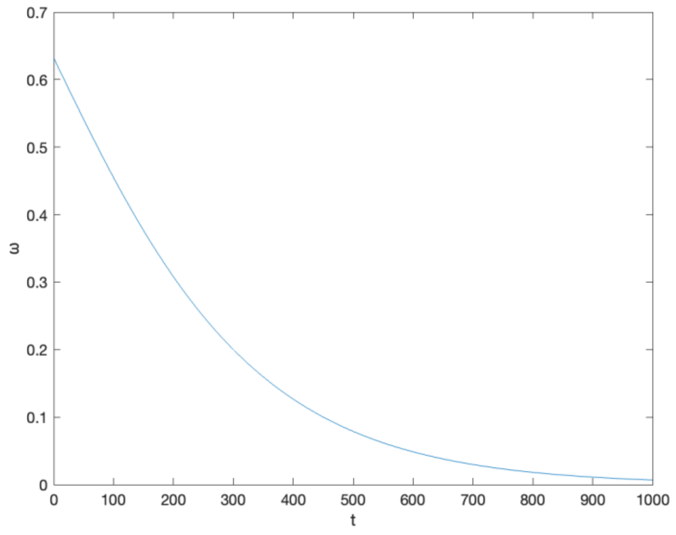


Fig. 2. Gompertz inertia weight.

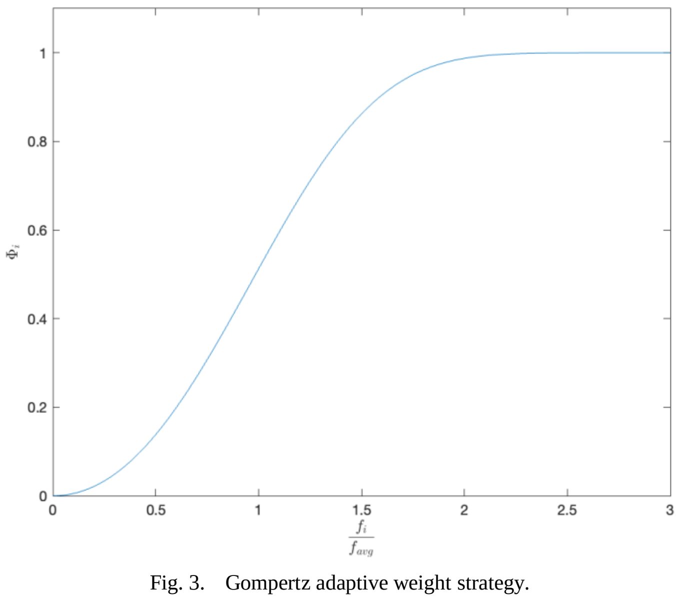


Fig. 3. Gompertz adaptive weight strategy.

As shown in Fig. 3, the Gompertz adaptive weight is close to 0 when the corresponding wolf fitness value is relatively small, indicating that the wolf is close to the prey. This step control is extremely small, which is conducive to more thoroughly finding the optimal value locally; As the corresponding wolf fitness value ratio increases, it indicates that the wolf is far away from the prey, so the Gompertz adaptive weight increases rapidly to prevent falling into the local optimum, and the step size increases, which is conducive to searching for the optimum in the global scope.

C. Adaptive Gray Wolf Optimization Algorithm based on Gompertz Inertia Weight Strategy

Based on the Gompertz inertia weight strategy, the position update formula is modified as Eq. (12):

$$\overrightarrow{X_1} = w\cdot\overrightarrow{X_\alpha} - \overrightarrow{A_1}\cdot\overrightarrow{D_\alpha} \quad \overrightarrow{X_2} = w\cdot\overrightarrow{X_\beta} - \overrightarrow{A_2}\cdot\overrightarrow{D_\beta}$$
$$\overrightarrow{X_3} = w\cdot\overrightarrow{X_\delta} - \overrightarrow{A_3}\cdot\overrightarrow{D_\delta} \qquad (12)$$

Based on the Gompertz adaptive weight strategy, the new gray wolf position $\vec{X}$ is obtained as shown in Eq. (7), (13):

$$\overrightarrow{X_1} = \Phi_1\cdot\overrightarrow{X_1} \quad \overrightarrow{X_2} = \Phi_2\cdot\overrightarrow{X_2} \quad \overrightarrow{X_3} = \Phi_3\cdot\overrightarrow{X_3} \qquad (13)$$

The steps of the GGWO algorithm are as follows:

1) *Set* the relevant parameters $\vec{a}$, $\vec{r_1}$, $\vec{r_2}$, $\vec{A}$, $\vec{C}$ according to Eq. (2), (3) and (4);

2) *Define* α, β and δ wolves;

3) *Initialize* the position of the population;

4) *Calculate* the fitness value according to Eq. (5) sort the fitness value from large to small, and filter out the top three D_α, D_β and D_δ corresponding to α, β and δ wolves respectively;

5) *Update* the positions of α, β and δ wolves according to Eq. (12) by adding inertia weight ω;

6) *Then* update the positions according to Eq. (7), (13) by adding an adaptive weight Φ_i and performing boundary check;

7) *If* the maximum number of iterations is reached, the algorithm stops and the optimal value is output; otherwise, return to step 4.

The flow chart of GGWO is shown in Fig. 4:

As shown in Fig. 4, the basic parameters of the GGWO algorithm are set to initialize the wolves, and then start iteration. In the process of continuous iteration, the position of the wolves is updated through various strategies like inertial weight, adaptive weight, and bounds check. Finally, after reaching the maximum number of iterations, the optimal value is output. The algorithm in this article uses the Gompertz inertia weight strategy and adaptive position update strategy to balance the global search and local search of the gray wolf population, which can effectively consider all information during the iteration process. Gompertz inertia weight strategy calculates the iteration of the algorithm, giving the same inertia weight value to all gray wolf individuals. This is to balance the global and local search performance of all gray wolf individuals from a global perspective. The adaptive position update strategy allows all gray wolf individuals to give different position update schemes to different gray wolf individuals when the number of iterations is fixed. This reflects the different fitness values of different particles and should be given different inertia weights, which is very scientific and necessary. The unity of the Gompertz inertia weight strategy and adaptive position update strategy can use all particles in the population to have targeted inertia weights and position update adjustment strategies, which is a non-multiplicative and scientifically reasonable solution.

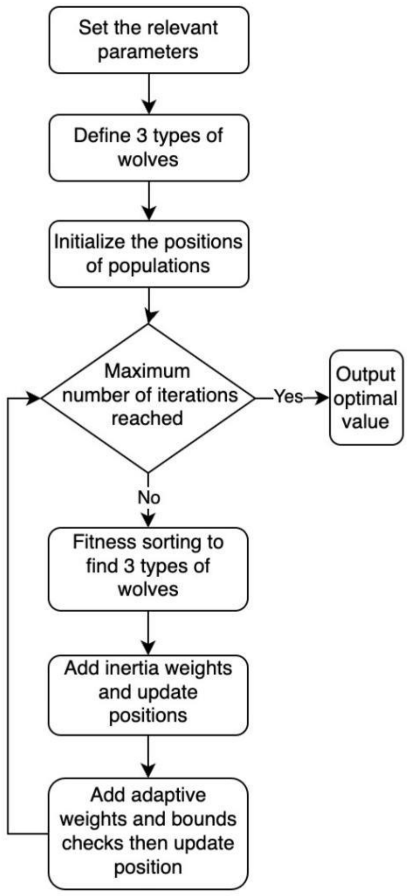

Set the relevant parameters
Define 3 types of wolves
Initialize the positions of populations
Maximum number of iterations reached
Yes
Output optimal value
No
Fitness sorting to find 3 types of wolves
Add inertia weights and update positions
Add adaptive weights and bounds checks then update position

Fig. 4. The flow chart of GGWO.

V. Simulation Experiment and Result Analysis

Simulation environment: MacOS, memory: 256GB, machine frequency 3.49GHz, MATLAB R2022a.

A. Test Function and Parameter Settings

The six test functions used in the simulation experiments of this article are shown in Table I:

Table I introduces the expressions, upper and lower limits, and optimal values of the six test functions used in this simulation.

As shown in Table I, $f_1(x)$ is a simple sum of squares function, which is smooth and convex. It is typically used to assess the basic performance of an optimization algorithm. $f_2(x)$ combines a linear sum component and a multiplicative component, introducing both global structure and local minima. It tests an algorithm's ability to handle non-separable and multimodal functions. $f_3(x)$ is a nested sum of squares, adding complexity by testing the algorithm's performance on hierarchical problems where optimization at one level depends on the optimization at another. $f_4(x)$ is a maximization function that tests the algorithm's ability to find the largest element in a vector, which can be useful for problems that require selection from a set of alternatives. $f_5(x)$ is reminiscent of the Rastrigin function, which introduces a large number of local minima, making it a challenge for algorithms to find the global minimum. $f_6(x)$ resembles a modified Schwefel function with a sinusoidal component, which is very challenging due to its complex landscape with many local optima.

TABLE I. TABLE OF TEST FUNCTIONS

Test Functions	Expressions of Functions	Domian	Optimal
F1	$f_1(x)=\sum_{i=1}^{n} x_i^2$	[-100,100]	0
F2	$f_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	[-10 ,10]	0
F3	$f_3(x)=\sum_{i=1}^{n}(\sum_{j=1}^{i} x_j)^2$	[-100,100]	0
F4	$f_4(x)=\max_i\{\lvert x_i\rvert, 1\le i\le n\}$	[-10,10]	0
F5	$f_5(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	[-5.12,5.12]	0
F6	$f_6(x)=\frac{1}{4000}\sum_{i=1}^{n} x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1$	[-600,600]	0

B. Experimental Results Analysis

1) Comparison of average convergence curves of 11 algorithms

Select 8 classic swarm intelligence optimization algorithms [33-40], and then add the three algorithms improved in this article, which are Gray Wolf Optimization Algorithm Adding Inertia Weight (GIGWO), Gray Wolf Optimization Algorithm Adding Self-adaptive Weights (GSGWO) and Adaptive gray wolf optimization algorithm based on Gompertz inertia weight strategy (GGWO) for a total of 11 optimization methods. The average convergence curves of these 11 algorithms in three dimensions on 6 test functions are shown in Fig. 5 to 10:

In Fig. 5 to10, the abscissa reflects the number of current iterations, and the ordinate represents the logarithm of the fitness value. In low dimension (D=30) and high dimension (D=200, 300), As the iteration proceeds, the other eight algorithms except GIGWO, GSGWO and GGWO converge slowly and easily fall into local optimal, the evolution curve of the GGWO algorithm is the unique algorithm with the most obvious decline, the highest solution accuracy, and the fastest convergence speed, and will not fall into the local optimal.

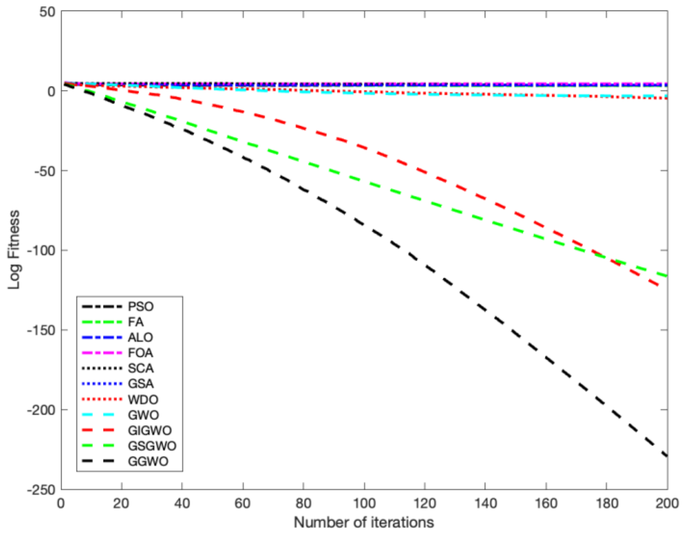
Except for the initial convergence speed of the F2 test function in high dimensions, the other eight swarm optimization algorithms are close to the optimized algorithm GIGWO, GSGWO, and GGWO. However, they will easily fall into local optimality when the number of iterations grows, and the convergence speed and accuracy are also far inferior to those of the optimized ones. In addition, the standard GWO algorithm based on GIGWO, GSGWO, and GGWO, such as the high-dimensional F6 test function, has slightly lower accuracy and convergence speed than WDO. However, after the optimization, the convergence speed of the three algorithms GIGWO, GSGWO, and GGWO are all much higher than that of WDO, which shows that the optimization strategy in this article is quite effective.

And the convergence speed and solution accuracy of GIGWO and GSGWO are far better than GWO. The convergence speed of the GGWO algorithm on F1-F4 test functions is much higher than that of the GIGWO and GSGWO algorithms, indicating that both Gompertz inertia weight and adaptive optimization strategies are effective. On F4 and F5, the convergence speed of GGWO is far better than that of GIGWO and slightly better than GSGWO. The solution accuracy of the three algorithms is close and far better than the other eight algorithms, reflecting that the superposition of the two optimization strategies is still effective because the convergence results of GGWO are better in more cases.

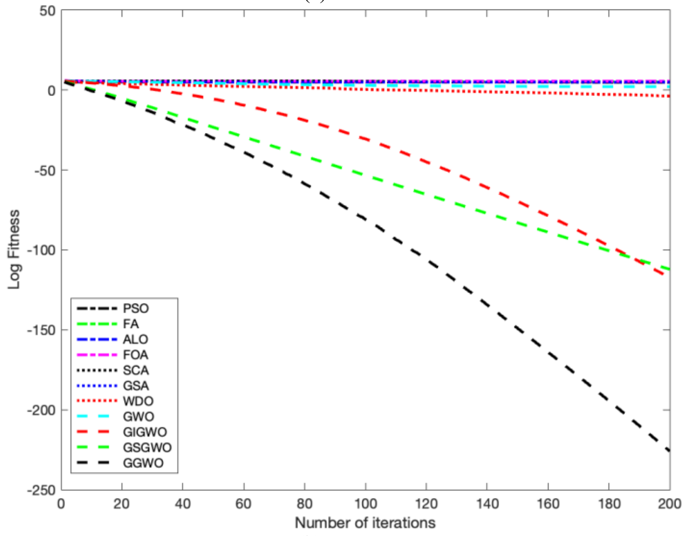
As can be seen from Fig. 5 to 10, the GGWO has the fastest decline rate and the smallest final fitness value. In the high-dimensional case of F4, although GIGWO is not completely stuck in the local optimum, the curve is stable at first, indicating that it is still stuck in the local optimum at the beginning of the iteration, which makes it impossible to search for the global optimal solution as quickly as possible. Similarly, although GSGWO has not completely stuck into the local optimal solution in the F2 high dimension, the curve gradually stabilizes as the iterations proceed, indicating that it has fallen into the local optimal, which is also not conducive to the search for the global optimal.

The optimization performance and stability of 11 algorithms in low dimension (D=30) and high dimension (D=200, 300) are shown in Tables II to IV:

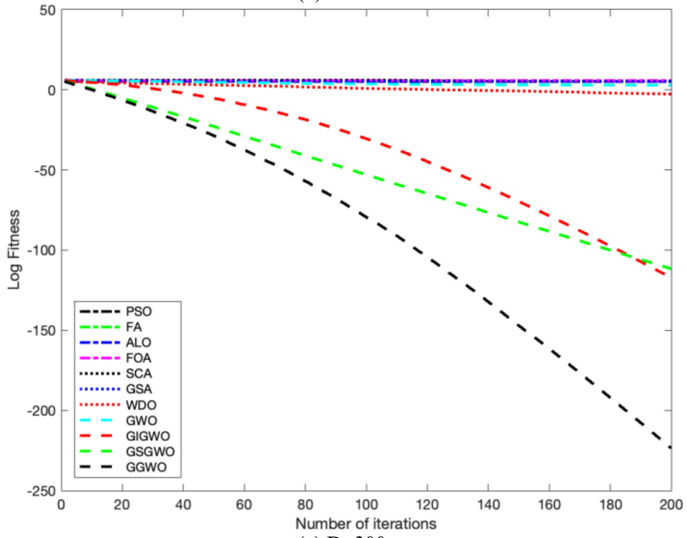
In Tables II to IV, the optimization performance and stability of 11 algorithms in low dimension (D=30) and high dimension (D=200, 300) are reflected by calculating the mean and variance. Among them, the bold data is the minimum value of the mean or standard deviation among the 11 algorithms.



(a) D=30

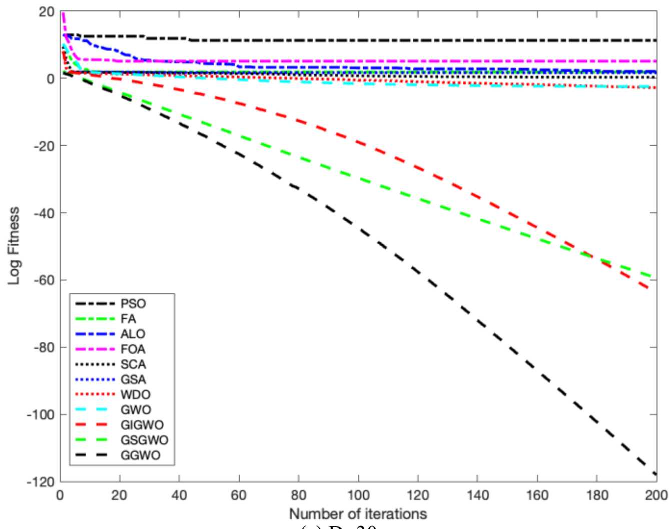


(b) D=200

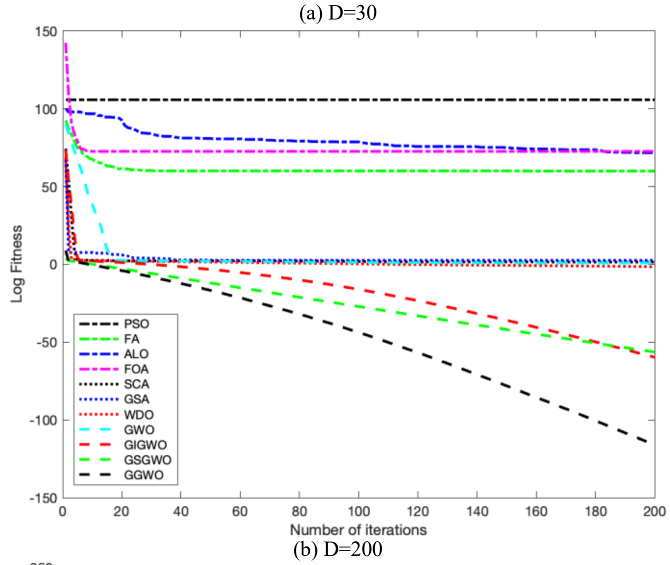


(c) D=300

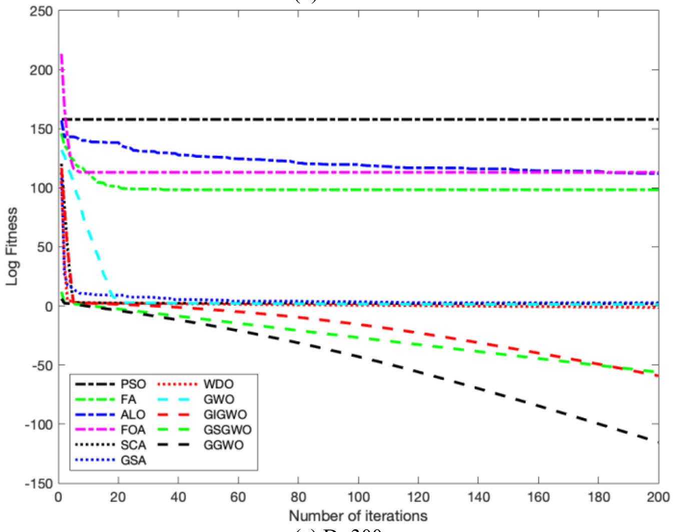
Fig. 5. Comparison chart of average convergence curve of F1.



(a) D=30

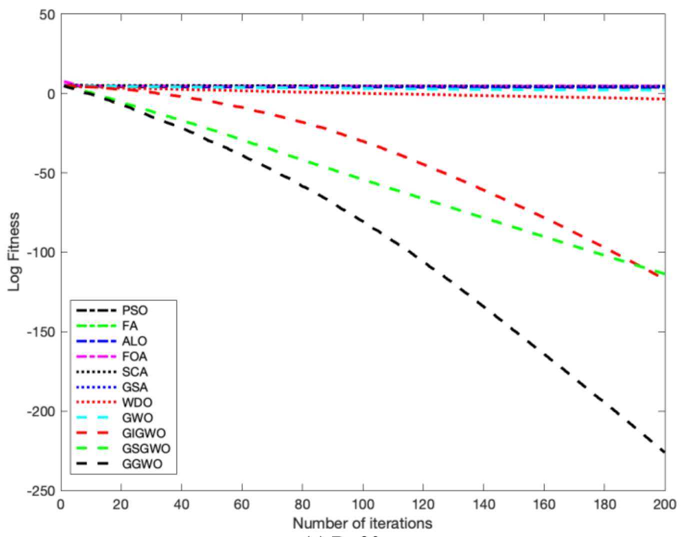


(b) D=200

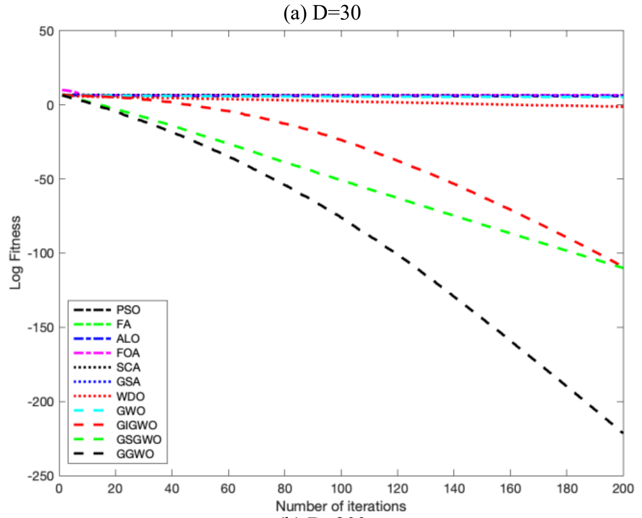


(c) D=300

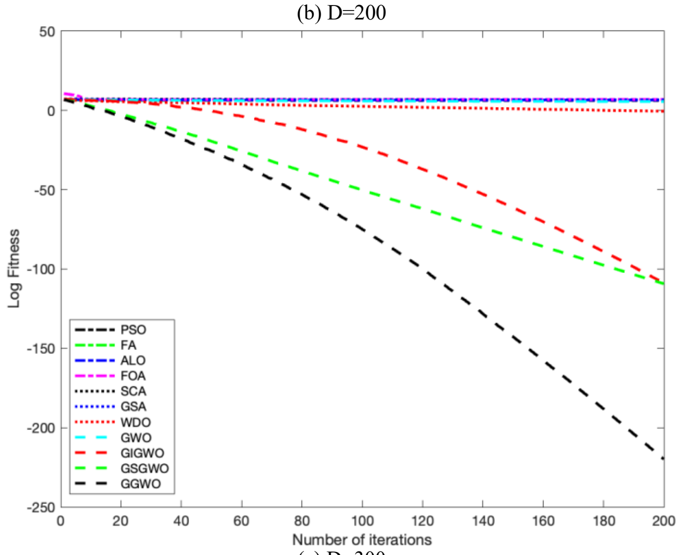
Fig. 6. Comparison chart of average convergence curve of F2.



(a) D=30

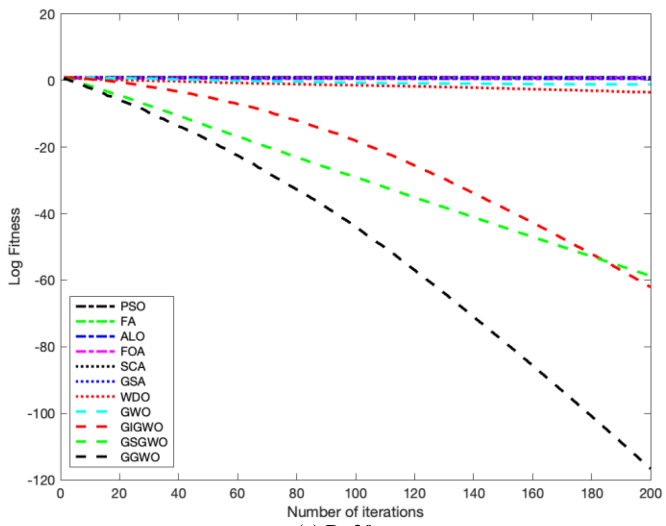


(b) D=200

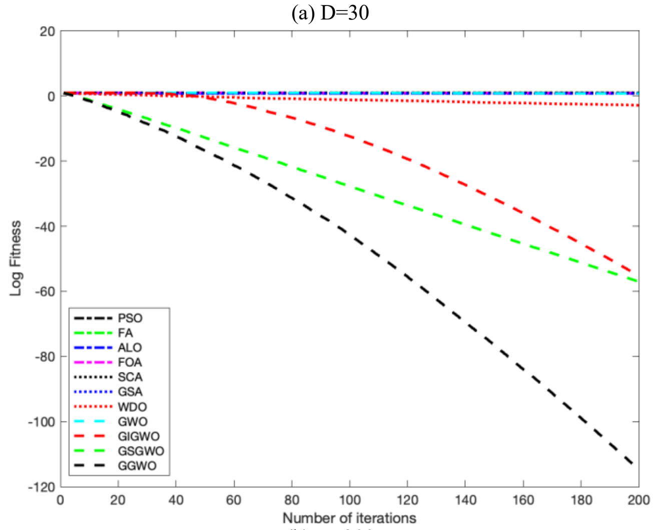


(c) D=300

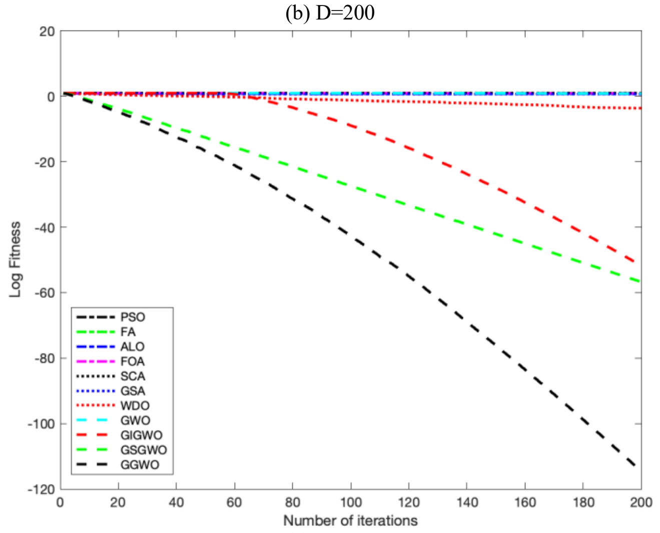
Fig. 7. Comparison chart of average convergence curve of F3.



(a) D=30

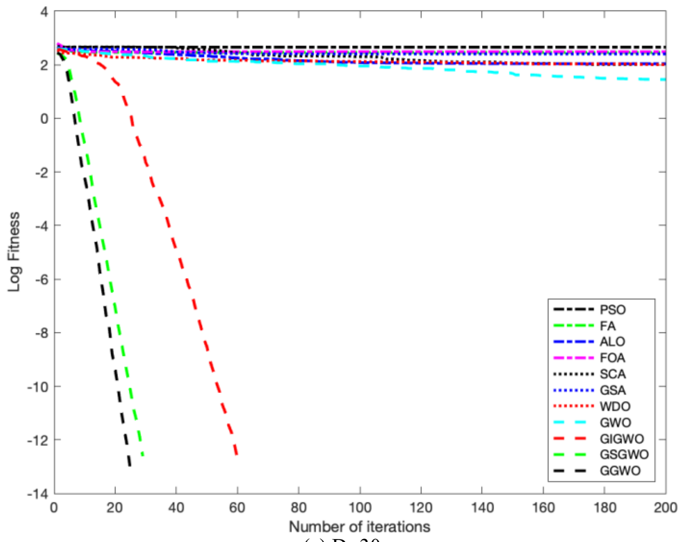


(b) D=200

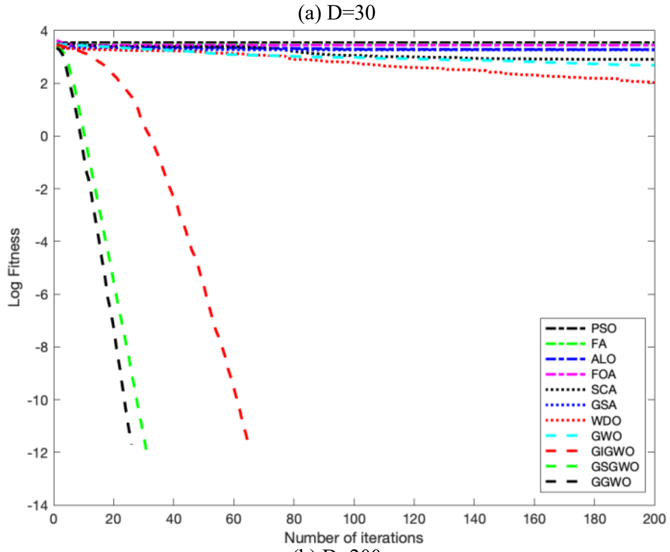


(c) D=300

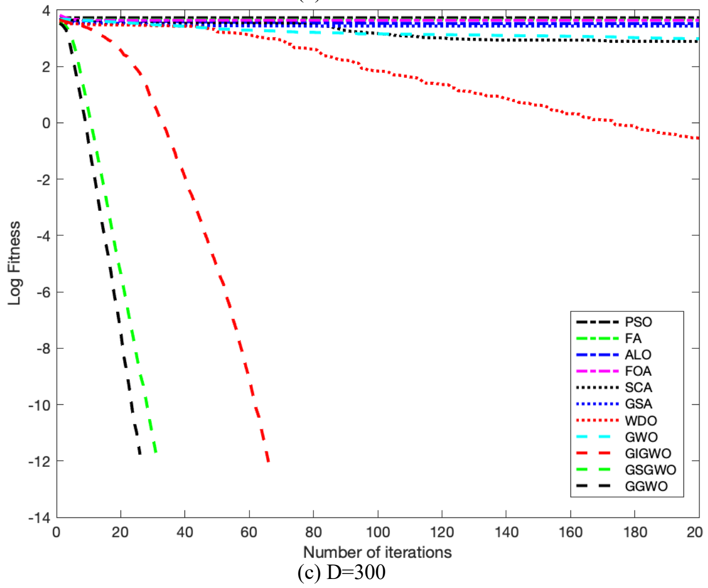
Fig. 8. Comparison chart of average convergence curve of F4.



(a) D=30

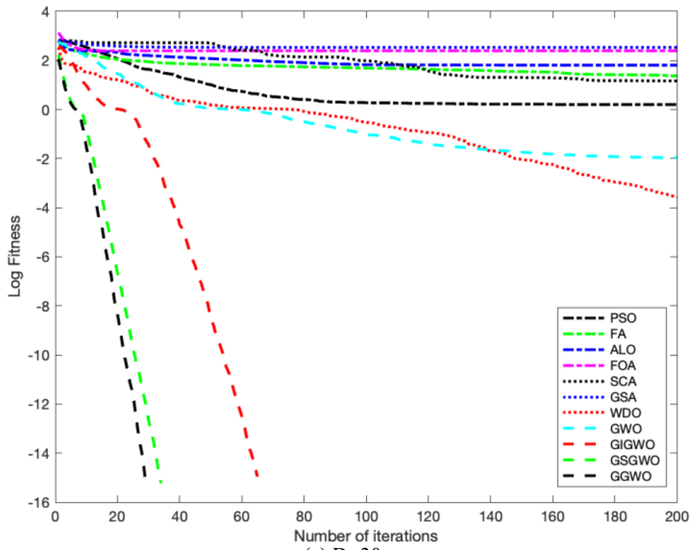


(b) D=200

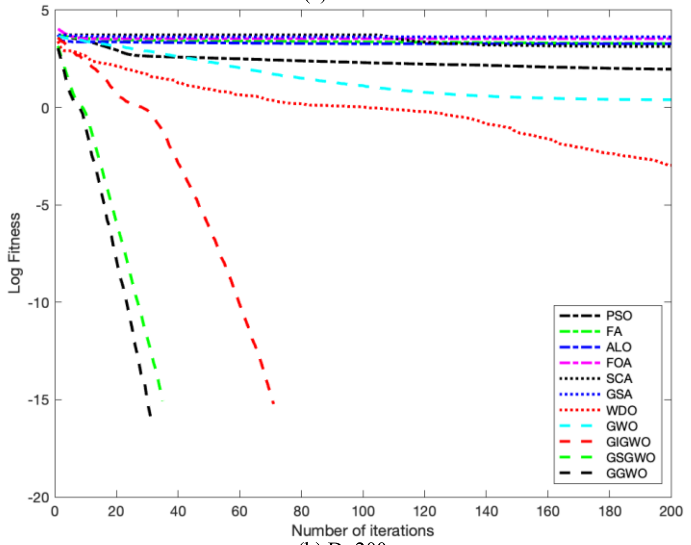


(c) D=300

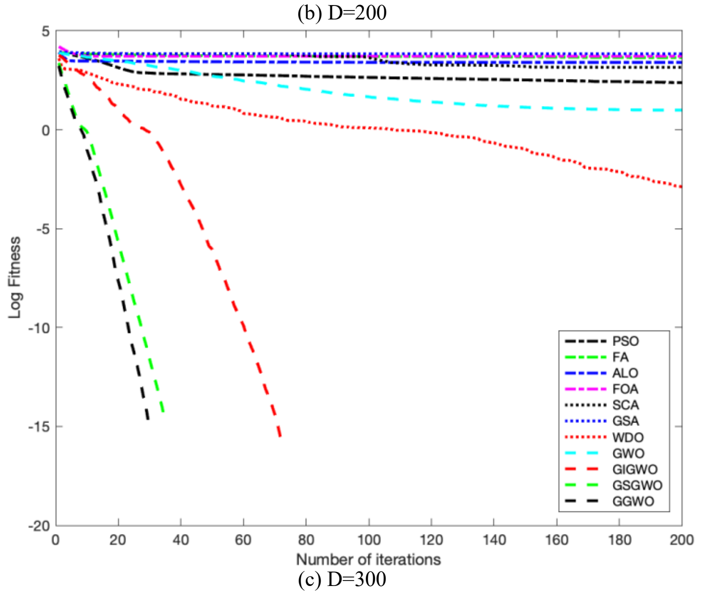
Fig. 9. Comparison chart of average convergence curve of F5.



(a) D=30



(b) D=200



(c) D=300

Fig. 10. Comparison chart of average convergence curve of F6.

2) Comparison of global optimal values of 11 algorithms

Whether it is high-dimensional or low-dimensional, GGWO has the smallest mean or standard deviation, which shows that GGWO has extremely strong optimization performance and stability. On the two test functions F5 and F6, GIGWO, GSGWO, and GGWO found the global optimal solution 0 in every experiment. The performance of GWO is lower than that of WDO, but after adding Gompertz inertia weight or adaptive weight, the performance and stability are much higher than that of WDO, which shows that the optimization strategy for GWO is quite effective.

TABLE II. TEST RESULTS OF 11 ALGORITHMS ON 6 TEST FUNCTIONS (D=30)

Functions	F1		F2		F3		F4		F5		F6	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
PSO	2.29E+03	2.79E+02	1.47E+12	2.62E+12	8.42E+03	3.20E+03	9.38E+00	2.09E-01	4.41E+02	3.31E+01	1.62E+00	1.98E-02
FA	1.30E+03	5.56E+02	5.52E+01	2.35E+01	2.96E+04	5.64E+03	7.73E+00	8.52E-01	3.11E+02	1.69E+01	1.08E+01	1.03E+01
ALO	1.03E+04	5.63E+03	1.49E+04	3.20E+04	4.49E+04	3.28E+04	4.09E+00	5.69E-01	1.49E+02	2.93E+01	7.88E+01	2.64E+01
FOA	2.87E+04	4.69E+03	1.43E+06	2.55E+06	6.11E+04	1.29E+04	6.43E+00	5.36E-01	3.39E+02	1.20E+01	2.85E+02	3.95E+01
SCA	1.35E+03	8.05E+02	3.85E+00	2.59E+00	3.63E+04	1.41E+04	6.65E+00	5.74E-01	8.93E+01	8.48E+01	6.48E+00	5.65E+00
GSA	5.14E+03	1.69E+03	5.05E+01	1.74E+01	1.19E+04	6.54E+03	5.61E+00	1.16E+00	2.32E+02	3.19E+01	3.19E+02	2.70E+01
WDO	3.90E-05	4.32E-05	4.19E-03	3.63E-03	6.23E-04	5.15E-04	4.14E-04	2.95E-04	1.46E+02	2.17E+01	5.54E-02	1.24E-01
GWO	1.49E-03	2.16E-03	2.05E-03	5.31E-04	9.82E+01	1.76E+01	9.74E-02	4.75E-02	2.54E+01	1.00E+01	5.02E-02	4.58E-02
GIGWO	1.19E-124	1.59E-124	1.40E-64	8.69E-65	3.36E-118	2.36E-118	1.35E-62	1.05E-62	0.00E+00	0.00E+00	0.00E+00	0.00E+00
GSGWO	5.73E-117	6.47E-117	7.42E-60	3.01E-60	2.80E-114	4.14E-114	3.70E-59	2.96E-59	0.00E+00	0.00E+00	0.00E+00	0.00E+00
GGWO	**8.33E-227**	**0.00E+00**	**1.20E-118**	**2.35E-118**	**3.45E-225**	**0.00E+00**	**6.19E-117**	**1.12E-116**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**

TABLE III. TEST RESULTS OF 11 ALGORITHMS ON 6 TEST FUNCTIONS (D=200)

Functions	F1		F2		F3		F4		F5		F6	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
PSO	1.03E+05	4.46E+03	2.10E+103	2.46E+103	6.95E+05	1.73E+05	9.87E+00	6.04E-02	3.46E+03	1.11E+02	7.72E+01	1.11E+01
FA	1.97E+05	2.18E+04	5.74E+72	1.28E+73	1.57E+06	2.18E+05	9.75E+00	8.21E-02	2.97E+03	1.08E+02	2.30E+03	2.60E+02
ALO	2.06E+05	5.68E+04	4.64E+86	1.04E+87	1.46E+06	7.02E+05	6.40E+00	5.04E-01	1.82E+03	1.19E+02	1.90E+03	2.36E+02
FOA	3.73E+05	7.12E+03	7.58E+77	1.69E+78	3.53E+06	1.59E+06	8.75E+00	4.70E-02	2.86E+03	5.08E+01	3.32E+03	1.24E+02
SCA	1.14E+05	1.95E+04	5.22E+01	3.03E+01	1.30E+06	4.11E+05	9.79E+00	4.73E-02	5.53E+02	2.85E+02	9.72E+02	3.95E+02
GSA	9.04E+04	6.60E+03	3.56E+02	2.20E+01	1.77E+06	8.79E+05	5.49E+00	3.72E-01	1.86E+03	7.78E+01	4.28E+03	1.13E+02
WDO	1.30E-02	2.34E-02	7.84E-02	7.50E-02	1.67E-01	2.18E-01	4.24E-04	3.49E-04	9.07E+02	8.30E+02	2.95E-03	3.05E-03
GWO	2.10E+02	2.49E+01	6.50E+00	1.03E+00	1.98E+05	4.72E+04	5.79E+00	4.61E-01	5.25E+02	7.61E+01	2.75E+00	6.56E-01
GIGWO	5.69E-118	9.94E-118	2.08E-60	1.43E-60	2.17E-110	1.26E-110	3.19E-54	3.89E-54	0.00E+00	0.00E+00	0.00E+00	0.00E+00
GSGWO	1.23E-112	1.11E-112	4.16E-57	1.44E-57	3.15E-110	3.71E-110	8.39E-58	2.33E-58	0.00E+00	0.00E+00	0.00E+00	0.00E+00
GGWO	**1.13E-225**	**0.00E+00**	**1.50E-116**	**1.04E-116**	**1.58E-221**	**0.00E+00**	**2.75E-115**	**1.56E-115**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**

TABLE V. TEST RESULTS OF 11 ALGORITHMS ON 6 TEST FUNCTIONS (D=300)

Functions	F1		F2		F3		F4		F5		F6	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
PSO	1.93E+05	2.47E+04	4.66E+159	Inf	1.44E+06	3.70E+05	9.86E+00	6.02E-02	5.33E+03	1.14E+02	2.55E+02	1.59E+01
FA	4.60E+05	5.14E+04	1.50E+118	3.15E+118	3.15E+06	8.46E+05	9.84E+00	2.44E-02	4.66E+03	1.17E+02	4.18E+03	4.56E+02
ALO	3.26E+05	4.78E+04	2.46E+147	5.51E+147	2.74E+06	1.14E+06	6.57E+00	6.14E-01	3.15E+03	1.69E+02	2.55E+03	3.25E+02
FOA	6.14E+05	2.51E+04	1.32E+120	2.95E+120	2.41E+07	1.02E+07	8.92E+00	9.37E-02	4.38E+03	8.33E+01	5.39E+03	3.12E+02
SCA	1.84E+05	2.49E+04	1.23E+02	4.22E+01	3.85E+06	5.83E+05	9.90E+00	4.23E-02	1.33E+03	6.70E+02	1.70E+03	3.63E+02
GSA	2.10E+05	1.08E+04	5.23E+02	3.14E+01	4.49E+06	1.88E+06	5.29E+00	3.65E-01	2.74E+03	6.60E+01	6.95E+03	2.51E+02
WDO	1.07E-03	1.02E-03	9.01E-02	5.76E-02	9.67E-02	5.88E-02	9.56E-04	4.91E-04	9.90E+02	1.36E+03	4.88E-04	4.13E-04
GWO	9.48E+02	2.23E+02	2.17E+01	3.17E+00	4.72E+05	8.27E+04	7.04E+00	5.97E-01	8.33E+02	2.47E+01	9.61E+00	7.45E-01
GIGWO	9.35E-118	5.23E-118	1.05E-59	6.71E-60	6.09E-109	7.32E-109	1.81E-52	2.53E-52	0.00E+00	0.00E+00	0.00E+00	0.00E+00
GSGWO	3.53E-112	3.51E-112	1.05E-56	3.26E-57	4.20E-109	9.09E-109	2.62E-57	1.30E-57	0.00E+00	0.00E+00	0.00E+00	0.00E+00
GGWO	**3.53E-224**	**0.00E+00**	**7.52E-115**	**1.02E-114**	**1.01E-219**	**0.00E+00**	**4.41E-115**	**4.04E-115**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**

In conclusion, the stability and convergence performance of GGWO is also the best among the 11 algorithms.

C. GGWO Time Complexity Analysis

The time complexity of GWO is $O(nmD)$, where n is the gray wolves' total number of in populations, m is the maximum number of iterations, and D is the dimension of the corresponding optimization problem. Moreover, GWO has one of the smallest time complexity among the eight algorithms in this article because the time complexity of other algorithms such as FA and GSA is as high as $O(n^2mD)$. GGWO uses Gompertz inertia weights and adaptive weights to update the position in the algorithm, which is essentially equivalent to linearly multiplying a constant in the formula during each iteration of the standard gray wolf optimization algorithm. Therefore, GGWO does not increase the time complexity of the original algorithm GWO, which means the time complexity of the improved algorithm GGWO in this article is also the smallest, $O(nmD)$.

GGWO greatly improves the algorithm's convergence speed, ability to jump out of local optima, and stability without additional increase in time complexity. In comparison with the other 10 population intelligent optimization algorithms, it clearly shows that the optimization performance far exceeds that of other algorithms.

VI. Conclusion

An adaptive gray wolf optimization algorithm based on the Gompertz inertia weight strategy is proposed, which uses the Gompertz function to improve the inertia weight and position update formulas. By comparing the simulation experiments, 11 different swarm intelligence algorithms were used on 6 test functions to draw the average convergence curve, and the average value of 10 runs was taken as the final display result. Experimental results show that GGWO has the smallest variance in the test functions, proving that it has the best stability. In addition, the average convergence curve of GGWO decreases the fastest, indicating that it has the fastest convergence speed. Moreover, the time complexity of GGWO is $O(nmD)$, which has a certain application potential. From the comparative analysis of standard deviation, convergence curve, time complexity, and other angles, the experimental results show that the improved algorithm GGWO has the characteristics of good stability, fast convergence speed, and high solution accuracy.

Although GGWO has made certain improvements in solution accuracy, speed, and stability, there are still some areas for future improvements. GGWO's work mainly focuses on adjusting inertial weights; we can consider other position update formulas in the future. In addition, the population initialization of GGWO is too random. Due to this, Latin hypercube sampling will be considered to initialize the population operation. Besides, while GGWO improved sharply on the simple or unimodal test functions, for processing some complex test functions or data, the effect might not greatly improved. Based on various industrial applications, GGWO can be an efficient optimization tool that can be used to deal with many practical optimization problems. In the future, GGWO will be used in some practical problems, such as medical image recognition, fault detection, UAV path planning, quantum neural network optimization, and other issues.

Data Availability

The data supporting the findings of this study are available from the GGWO repository at this link: GGWO

Conflicts of Interest

The author declares that there is no conflict of interest regarding the publication of this paper.

References

[1] Seyedali Mirjalili. How effective is the grey wolf optimizer in training multi-layer perceptrons. Applied Intelligence, 43:150–161, 2015.

[2] Satyajit Mohanty, Bidyadhar Subudhi, and Pravat Kumar Ray. A new mppt design using grey wolf optimization technique for photovoltaic system under partial shading conditions. IEEE Transactions on Sustainable Energy, 7(1):181–188, 2015.

[3] Seyedali Mirjalili, Shahrzad Saremi, Seyed Mohammad Mirjalili, and Leandro dos S Coelho. Multi-objective grey wolf optimizer: a novel

algorithm for multi-criterion optimization. Expert systems with applications, 47:106–119, 2016.

[4] Eid Emary, Hossam M Zawbaa, and Aboul Ella Hassanien. Binary grey wolf optimization approaches for feature selection. Neurocomputing, 172:371–381, 2016.

[5] Ali Asghar Heidari and Parham Pahlavani. An efficient modified grey wolf optimizer with levy flight for optimization tasks. Applied Soft Computing, 60:115–134, 2017.

[6] Hossam Faris, Ibrahim Aljarah, Mohammed Azmi Al-Betar, and Seyedali Mirjalili. Grey wolf optimizer: a review of recent variants and applications. Neural computing and applications, 30:413–435, 2018.

[7] Mehak Kohli and Sankalap Arora. Chaotic grey wolf optimization algorithm for constrained optimization problems. Journal of computational design and engineering, 5(4):458–472, 2018.

[8] Shubham Gupta and Kusum Deep. A novel random walk grey wolf optimizer. Swarm and evolutionary computation, 44:101–112, 2019.

[9] Mohammad H Nadimi-Shahraki, Shokooh Taghian, and Seyedali Mirjalili. An improved grey wolf optimizer for solving engineering problems. Expert Systems with Applications, 166:113917, 2021.

[10] Qasem Al-Tashi, Said Jadid Abdul Kadir, Helmi Md Rais, Seyedali Mirjalili, and Hitham Alhussian. Binary optimization using hybrid grey wolf optimization for feature selection. Ieee Access, 7:39496–39508, 2019.

[11] Ayta¸c Altan, Se¸ckin Karasu, and Enrico Zio. A new hybrid model for wind speed forecasting combining long short-term memory neural network, decomposition methods, and grey wolf optimizer. Applied Soft Computing, 100:106996, 2021.

[12] Xuehua Zhao, Xiang Zhang, Zhennao Cai, Xin Tian, Xianqin Wang, Ying Huang, Huiling Chen, and Lufeng Hu. Chaos enhanced grey wolf optimization wrapped elm for diagnosis of paraquat-poisoned patients. Computational biology and chemistry, 78:481–490, 2019.

[13] T Jayabarathi, T Raghunathan, BR Adarsh, and Ponnuthurai Nagaratnam Suganthan. Economic dispatch using hybrid grey wolf optimizer. Energy, 111:630–641, 2016.

[14] Mohd Herwan Sulaiman, Zuriani Mustaffa, Mohd Rusllim Mohamed, and Omar Aliman. Using the gray wolf optimizer for solving optimal reactive power dispatch problem. Applied Soft Computing, 32:286–292, 2015.

[15] Mahdi Shariati, Mohammad Saeed Mafipour, Behzad Ghahremani, Fazel Azarhomayun, Masoud Ahmadi, Nguyen Thoi Trung, and Ali Shariati. A novel hybrid extreme learning machine– grey wolf optimizer (elm-gwo) model to predict compressive strength of concrete with partial replacements for cement. Engineering with Computers, pages 1–23, 2022.

[16] SR Jino Ramson, K Lova Raju, S Vishnu, and Theodoros Anagnostopoulos. Nature inspired optimization techniques for image processing—a short review. Nature inspired optimization techniques for image processing applications, pages 113–145, 2019.

[17] T Ramakrishnan and B Sankaragomathi. A professional estimate on the computed tomography brain tumor images using svm-smo for classification and mrg-gwo for segmentation. Pattern Recognition Letters, 94:163–171, 2017.

[18] Tianhua Jiang and Chao Zhang. Application of grey wolf optimization for solving combinatorial problems: Job shop and flexible job shop scheduling cases. Ieee Access, 6:26231–26240, 2018.

[19] Yan Wei, Ni Ni, Dayou Liu, Huiling Chen, Mingjing Wang, Qiang Li, Xiaojun Cui, and Haipeng Ye. An improved grey wolf optimization strategy enhanced svm and its application in predicting the second major. Mathematical Problems in Engineering, 2017:1–12, 2017.

[20] Bo Yang, Xiaoshun Zhang, Tao Yu, Hongchun Shu, and Zihao Fang. Grouped grey wolf optimizer for maximum power point tracking of doubly-fed induction generator based wind turbine. Energy conversion and management, 133:427–443, 2017.

[21] Jian Zhou, Shuai Huang, Mingzheng Wang, and Yingui Qiu. Performance evaluation of hybrid ga–svm and gwo–svm models to predict earthquake-induced liquefaction potential of soil: a multi-dataset investigation. Engineering with Computers, pages 1–19, 2021.

[22] Abdul Kayom Md Khairuzzaman and Saurabh Chaudhury. Multilevel thresholding using grey wolf optimizer for image segmentation. Expert Systems with Applications, 86:64–76, 2017.

[23] Karuna Panwar and Kusum Deep. Discrete grey wolf optimizer for symmetric travelling salesman problem. Applied Soft Computing, 105:107298, 2021.

[24] Mostefa Kermadi, Zainal Salam, Jubaer Ahmed, and El Madjid Berkouk. An effective hybrid maximum power point tracker of photovoltaic arrays for complex partial shading conditions. IEEE Transactions on Industrial Electronics, 66(9):6990–7000, 2018.

[25] Chunying Wu, Jianzhou Wang, Xuejun Chen, Pei Du, and Wendong Yang. A novel hybrid system based on multi-objective optimization for wind speed forecasting. Renewable energy, 146:149–165, 2020.

[26] Ashokkumar Lakum and Vasundhara Mahajan. Optimal placement and sizing of multiple active power filters in radial distribution system using grey wolf optimizer in presence of nonlinear distributed generation. Electric Power Systems Research, 173:281–290, 2019.

[27] Sankalap Arora, Harpreet Singh, Manik Sharma, Sanjeev Sharma, and Priyanka Anand. A new hybrid algorithm based on grey wolf optimization and crow search algorithm for unconstrained function optimization and feature selection. Ieee Access, 7:26343–26361, 2019.

[28] Xiaodong Sun, Changchang Hu, Gang Lei, Youguang Guo, and Jianguo Zhu. State feedback control for a pm hub motor based on gray wolf optimization algorithm. IEEE Transactions on Power Electronics, 35(1):1136–1146, 2019.

[29] Ali M Eltamaly and Hassan MH Farh. Dynamic global maximum power point tracking of the pv systems under variant partial shading using hybrid gwo-flc. Solar Energy, 177:306–316, 2019.

[30] Noor Zaihah Jamal, Mohd Herwan Sulaiman, Omar Aliman and Zuriani Mustaffa, “Optimal Overcurrent Relays Coordination using an Improved Grey Wolf Optimizer” International Journal of Advanced Computer Science and Applications(ijacsa), 9(11), 2018.

[31] Telugu Maddileti, Govindarajulu Salendra and Chandra Mohan Reddy Sivappagari, “Design Optimization of Power and Area of Two-Stage CMOS Operational Amplifier Utilizing Chaos Grey Wolf Technique” International Journal of Advanced Computer Science and Applications(IJACSA), 11(7), 2020.

[32] Kathleen MC Tjørve and Even Tjørve. The use of gompertz models in growth analyses, and new gompertz-model approach: An addition to the unified-richards family. PloS one, 12(6):e0178691, 2017.

[33] James Kennedy and Russell Eberhart. Particle swarm optimization. In Proceedings of ICNN’95-international conference on neural networks, volume 4, pages 1942–1948. IEEE, 1995.

[34] Xin-She Yang. Firefly algorithms for multimodal optimization. In International symposium on stochastic algorithms, pages 169–178. Springer, 2009.

[35] Seyedali Mirjalili. The ant lion optimizer. Advances in engineering software, 83:80–98, 2015.

[36] Wen-Tsao Pan. A new fruit fly optimization algorithm: taking the financial distress model as an example. Knowledge-Based Systems, 26:69–74, 2012.

[37] Seyedali Mirjalili. Sca: a sine cosine algorithm for solving optimization problems. Knowledgebased systems, 96:120–133, 2016.

[38] Esmat Rashedi, Hossein Nezamabadi-Pour, and Saeid Saryazdi. Gsa: a gravitational search algorithm. Information sciences, 179(13):2232–2248, 2009.

[39] Zikri Bayraktar, Muge Komurcu, Jeremy A Bossard, and Douglas H Werner. The wind driven optimization technique and its application in electromagnetics. IEEE transactions on antennas and propagation, 61(5):2745–2757, 2013.

[40] Seyedali Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis. Grey wolf optimizer. Advances in engineering software, 69:46–61, 2014.