IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 30, 2019, accepted July 31, 2019, date of publication August 13, 2019, date of current version August 28, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2934994

A Random Opposition-Based Learning Grey Wolf Optimizer

WEN LONG[1,2], JIANJUN JIAO[2], XIMING LIANG[3], SHAOHONG CAI[1], AND MING XU[2]

[1]Key Laboratory of Economics System Simulation, Guizhou University of Finance and Economics, Guiyang 550025, China
[2]School of Mathematics and Statistics, Guizhou University of Finance and Economics, Guiyang 550025, China
[3]School of Science, Beijing University of Civil Engineering and Architecture, Beijing 100044, China

Corresponding author: Shaohong Cai (gzcd_csh58@126.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61463009, in part by the Program for the Science and Technology Top Talents of Higher Learning Institutions of Guizhou under Grant KY[2017]070, in part by the Science and Technology Foundation of Guizhou Province under Grant [2016]1022, and in part by the Fundamental Research Funds for the Beijing University of Civil Engineering and Architecture under Grant X18193.

ABSTRACT Grey wolf optimizer (GWO) algorithm is a swarm intelligence optimization technique that is recently developed to mimic the hunting behavior and leadership hierarchy of grey wolves in nature. It has been successfully applied to many real world applications. In the GWO algorithm, "**C**" is an important parameter which favoring exploration. At present, the researchers are few study the parameter "**C**" in GWO algorithm. In addition, during the evolution process, the other individuals in the population move towards to the α, β, and δ wolves which are to accelerate convergence. However, GWO is easy to trap in the local optima. This paper presents a modified parameter "**C**" strategy to balance between exploration and exploitation of GWO. Simultaneously, a new random opposition-based learning strategy is proposed to help the population jump out of the local optima. The experiments on 23 widely used benchmark test functions with various features, 30 benchmark problems from IEEE CEC 2014 Special Session, and three engineering design optimization problems. The results reveal that the proposed algorithm shows better or at least competitive performance against other compared algorithms on not only global optimization but also engineering design optimization problems.

INDEX TERMS Grey wolf optimizer, random opposition learning, global optimization, engineering design optimization, exploration, exploitation.

I. INTRODUCTION

A wide variety of nature-inspired optimization algorithms are there in swarm intelligence and evolutionary computation literatures. Such as genetic algorithm (GA) [1], particle swarm optimization (PSO) [2], differential evolution (DE) [3], artificial bee colony (ABC) [4], crow search algorithm (CSA) [5], cuckoo search (CS) [6], bat algorithm (BA) [7], ant colony optimization (ACO) [8], firefly algorithm (FA) [9], harmony search (HS) [10], teaching-learning-based optimization (TLBO) [11], grey wolf optimizer (GWO) [12], ant lion optimizer (ALO) [13], whale optimization algorithm (WOA) [14], moth-flame optimization (MFO) [15], etc. are a few of them. The main advantage of these algorithms is searching for solutions using the principle of "trial-and-error". Thus, these algorithms have been successfully applied to solve global optimization and real-world applications.

Concretely speaking, this paper focuses on the grey wolf optimizer (GWO), which was developed by Mirjalili et al. [12] and mimicked the hunting behavior and leadership hierarchy of grey wolves in nature. It possesses better performance than other prevailing population-based techniques, such as GA, PSO, GSA, and DE. The last four years have witnessed a rapid growth in the use of GWO for different applications [16]. In 2015, Sulaiman *et al.* [17] used the conventional GWO algorithm to find the best combination of control variables in optimal reactive power dispatch (ORPD) problem, such as generator voltages, tap changing transformers ratios, and the amount of reactive compensation devices. In [18], the basic GWO algorithm has been applied to gain the solution of non-convex and dynamic

The associate editor coordinating the review of this article and approving it for publication was Tao Zhang.

economic load dispatch problem (ELDP) of electric power system. Qais *et al.* [19] utilized the classical GWO algorithm to optimize the parameters of proportional integral controller. The controller was applied to control the permanent-magnet synchronous generator. In 2017, Khairuzzaman and Chaudhury [20] applied the canonical GWO algorithm to multilevel thresholding for image segmentation using Kapur's entropy and Otsu's between class variance functions. In [21], Emary et al. used a novel binary version of the GWO algorithm to find feature subset maximizing the classification accuracy while minimizing the number of selected features. Medjahed *et al.* [22] applied the conventional GWO algorithm to reduce the dimensionality of hyperspectral images in band selection problem. In [23], a maximum power point tracking (MPPT) design is proposed for a photo- voltaic (PV) system under partial shading conditions using the basic GWO algorithm. In 2018, Panwar *et al.* [24] utilized the binary GWO algorithm for determining the commitment schedule of large scale unit commitment (UC) problem. In [25], an improved version of GWO, named Intelligent GWO (IGWO), was applied to frame bidding strategy for a generating company in uniform price spot market. In [26], Zhang et al. used the conventional GWO algorithm for solving unmanned combat aerial vehicle (UCAV) path planning problem. In [27], the canonical GWO algorithm has been used to optimize the parameters of kernel extreme learning machine for bankruptcy prediction. In 2017, Sanjay *et al.* [28] proposed a novel hybrid GWO algorithm based on crossover and mutation operators for optimizing the configuration of distributed generator units. In [29], the basic GWO algorithm has been used to solve load frequency control (LFC) problem in an interconnected power system network equipped with classical PI/PID controller. In 2016, Precup *et al.* [30] used the standard GWO algorithm to tune the fuzzy control systems with reduced parametric sensitivity. These applications have shown the ability of GWO in terms of exploration strength compared to other population-based algorithms.

Although GWO has shown the better performance on real-world applications compared to other population- based algorithm. However, GWO also confronts some challenging problems. For example, when solving multi-modal functions, the conventional GWO algorithm can be easily trapped in the local optima, and the convergence rate will decrease considerably in the later period evolution [31]. Therefore, a number of variants of GWO have been developed to overcome the above two aspects. In [32], the evolutionary population dynamics (EPD) operator based on the theory of self-organizing criticality (SOC) was introduced into the basic GWO algorithm to enhance exploitation and promote exploration. In 2017, Heidari and Pahlavani [33] proposed an efficient modified version of GWO based on the Lévy flight operator to improve the exploration ability of the standard GWO algorithm. In 2018, inspired by PSO, Long *et al.* [31] presented a novel variant of GWO based on the nonlinear control parameter and the modified position-updating equation to balance between exploration and exploitation of the conventional GWO algorithm. In [34], a new fuzzy hierarchical operator was introduced to the simulation of the hunting process in the GWO algorithm and constructed a novel variant of GWO to improve the performance of the basic GWO algorithm. In 2019, Long *et al.* [35] proposed a novel GWO variant based on refraction learning for solving global optimization problem. In [36], the astrophysics concepts were merged into GWO to guide the grey wolves toward more promising areas of the search space and presented an improved version of GWO to solve numerical and engineering optimization problems. In 2017, Ibrahim *et al.* [37] developed a novel variant of GWO by using the chaotic logistic map, the opposition-based learning (OBL), the differential evolution, and the disruption operator to solve global optimization problems. In 2018, Long *et al.* [38] designed an improved version of GWO (EEGWO) for solving function and engineering optimization problems. The proposed EEGWO introduced the modified position-updating equation to enhance the exploration ability of the standard GWO algorithm. In 2018, Gupta and Deep [39] proposed a modified version of GWO based on random walk strategy to improve the global search ability of the basic GWO algorithm. In [40], Alomoush et al. presented a hybrid harmony search and GWO (CGWO) with opposition learning strategy for solving global optimization and feature selection problems. In 2016, Mittal *et al.* [41] developed a modified version of GWO (mGWO) for global engineering optimization. The proposed mGWO employs exponential function for the decay of parameter **a** over the course of iterations to balance between exploration and exploitation.

The No-free-Lunch (NFL) theorem [42] has logically proven that there is no population-based optimization technique appropriately suited for solving all optimization problems. For example, a particular population-based optimization algorithm can produce very promising results for a set of problems, but the same algorithm can show poor performance in a set of different problems. Therefore, a novel variant of GWO is proposed to improve the search performance of the conventional GWO algorithm. This paper presents a deep analysis of the improved GWO algorithm (called ROL-GWO), and it has the following main contributions:

- A new framework of GWO is proposed and does not affect the configuration of conventional GWO algorithm.
- A modified control parameter "**C**" strategy is prese-nted to balance between global exploration and local exploitation of the standard GWO algorithm.
- A random opposition-based learning strategy is proposed to help the population jump out of local optima.
- We investigate the performance of ROL-GWO using 23 widely used benchmark test functions and 30 benchmark test functions taken from IEEE CEC2014.
- The ROL-GWO is compared with several well-known population-based algorithms. The experimental results show that the ROL-GWO performs more effectively and accurately than other algorithms in most cases.

• We evaluate the proposed algorithm using three well-known engineering design optimization problems, i.e., spring design, welded beam design, and pressure vessel design problems.

The rest of this paper is organized as follows. Section 2 introduces the basic GWO algorithm briefly. Section 3 analyzes the standard GWO algorithm and presents an improved version of the GWO algorithm. The benchmark problems utilized to verify the performance of the proposed algorithm are presented and analyzed in Section 4. Finally, Section 5 concludes this investigation.

II. GREY WOLF OPTIMIZER

GWO is a newly developed population-based optimization algorithm inspired by Canis-lupus and proposed by Mirjalili *et al.* [12]. It mimics the leadership hierarchy and hunting mechanism of grey wolves in nature. In the GWO algorithm, the best solution in the population is called alpha (α). The second- and third- best solutions are named beta (β) and delta (δ), respectively. The other solutions in the population are assumed as omega (ω).

To mathematically model encircling behavior, the following equations are used [12]:

$$\mathbf{X}(t+1) = \mathbf{X}_p(t) - \mathbf{A} \cdot |\mathbf{C} \cdot \mathbf{X}_p(t) - \mathbf{X}(t)| \tag{1}$$

where $\mathbf{X}$ is the position vector of a wolf, t is the current iteration, $\mathbf{X}_p$ is the position vector of the prey, $\mathbf{A}$ and $\mathbf{C}$ are the coefficient matrix, respectively and are calculated as follows:

$$\mathbf{A} = 2\mathbf{a} \cdot \mathbf{r}_1 - \mathbf{a} \tag{2}$$

$$\mathbf{C} = 2 \cdot \mathbf{r}_2 \tag{3}$$

where $\mathbf{r}_1$ and $\mathbf{r}_2$ are the randomly generated vector from [0,1], respectively. $\mathbf{a}$ is linearly decreasing vector from 2 to 0 over iterations:

$$\mathbf{a}(t) = 2 - 2 \cdot \frac{t}{MaxIter} \tag{4}$$

where *MaxIter* is the maximum number of iterations.

The other individuals update their positions according to the positions of α, β, and δ wolves as follows [12]:

$$\mathbf{X}_1(t) = \mathbf{X}_\alpha(t) - \mathbf{A}_1 \cdot |\mathbf{C}_1 \cdot \mathbf{X}_\alpha(t) - \mathbf{X}(t)| \tag{5}$$

$$\mathbf{X}_2(t) = \mathbf{X}_\beta(t) - \mathbf{A}_2 \cdot |\mathbf{C}_2 \cdot \mathbf{X}_\beta(t) - \mathbf{X}(t)| \tag{6}$$

$$\mathbf{X}_3(t) = \mathbf{X}_\delta(t) - \mathbf{A}_3 \cdot |\mathbf{C}_3 \cdot \mathbf{X}_\delta(t) - \mathbf{X}(t)| \tag{7}$$

$$\mathbf{X}(t+1) = \frac{\mathbf{X}_1(t) + \mathbf{X}_2(t) + \mathbf{X}_3(t)}{3} \tag{8}$$

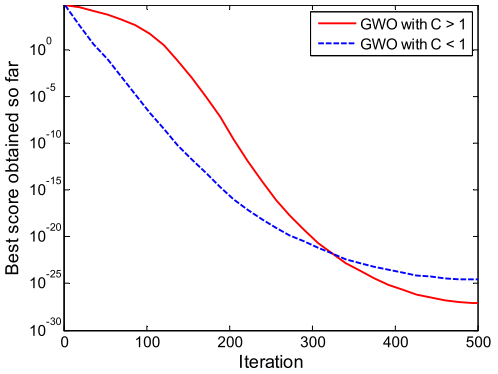
where $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}_3$ are similar to $\mathbf{A}$, $\mathbf{C}_1$, $\mathbf{C}_2$, and $\mathbf{C}_3$ are similar to $\mathbf{C}$.

The pseudo code of the standard GWO algorithm is described in [12].

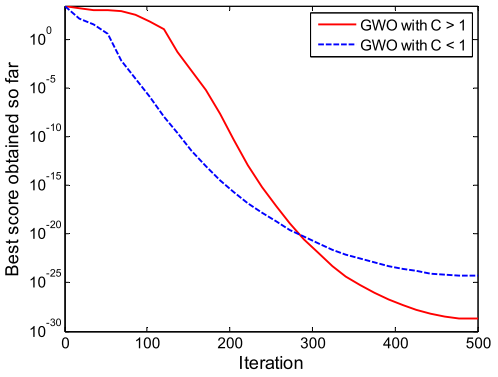
III. PROPOSED ROL-GWO ALGORITHM

A. MODIFIED PARAMETER "C" STRATEGY

All population-based optimization techniques aim to achieve a balance in both the exploration and exploitation to obtain the promising regions of the search space and eventually converge to the global optimum. After the detailed study and investigation of GWO, it has been observed that it has an obliging procedure to balance the exploration and exploitation. According to the Eq. (1), the control parameter $\mathbf{C}$ is favored to the exploration capability of GWO [12], [36]. However, the $\mathbf{C}$ vector contains random values in [0, 2]. The values of $\mathbf{C}$ are to increase ($\mathbf{C} > 1$) or reduce ($\mathbf{C} < 1$) the difficulty of the population close to the prey. This assists GWO to show a more random behavior throughout optimization, favoring exploration and local optima avoidance. Fig. 1(a) and 1(b) show the evaluation of fitness function over iterations on two benchmark test functions (i.e., Sphere and Levy), where Sphere is a unimodal function and Levy is a multimodal function, the population size is set to 30, and the dimensions of the two functions are set to 30.



(a) Sphere function



(b) Levy function

FIGURE 1. Convergence graphs of GWO on two functions.

As can be seen from Fig. 1, GWO with $\mathbf{C} > 1$ provides better exploitation that GWO with $\mathbf{C} < 1$ for Sphere and Levy functions. In addition, the graphs also show that GWO with $\mathbf{C} > 1$ provides better exploration of search space during earlier iterations as depicted.

To balance between exploration and exploitation of GWO, the novel modification has been proposed in the control parameter $\mathbf{C}$. The value of $\mathbf{C}$ changes dynamically with the number of iterations as follows:

$$\mathbf{C} = 2 \times r_2 - \frac{2}{3} \times \mathbf{a} \tag{9}$$

where $\mathbf{r}_2$ is random vector in [0, 1] and $\mathbf{a}$ is calculated by Eq. (4).

Compared with the Eq. (2), the Eq. (9) has the following characteristics. Since the parameter $\mathbf{a}$ is linearly decreased from 2 to zero over the course of iterations, the probability of $\mathbf{C} < 1$ is increased in the earlier stage of search. In this stage, the GWO algorithm has a stronger ability of exploitation. Generally speaking, the population has a good diversity in the earlier stage. The purpose of this stage is to accelerate convergence. The Eq. (9) can better meet this objective. The value of $\mathbf{a}$ decreases gradually as the number of iterations increases. According to the Eq. (9), the probability of $\mathbf{C} > 1$ is increased in the latter stage of search. Accelerating local search is particularly important in this stage. However, too much emphasis on local exploit tends to make the algorithm fall into local optimum. Based on the above facts, the Eq. (9) can effectively balance between global exploration and local exploitation of the GWO algorithm.

B. RANDOM OPPOSITON LEARNING STRATEGY

As shown in the Eqs. (5)-(8), the swarm of grey successfully complete their process of hunting by the guidance of the α, β, and δ wolves. Mathematically speaking, in GWO, each wolf updates its position with the help of these leading wolves. Thus, α, β, and δ wolves are the leading responsible search agents in updating the position of each wolf and provides an optimum direction towards the prey. It is very important that in each iteration these leading wolves should be the best (in term of fitness), so that each wolf will get an optimum guidance to approach a prey [39]. However, this search scheme promotes exploitation since all candidate wolves (candidate solutions) are attracted toward the α, β, and δ wolves, thereby converging faster toward these grey wolves. As a result of such a strong exploitation effect, the search diversity would be hampered in a sense. Finally, the GWO is prone to stagnation in local optima. Moreover, it has also been noticed that there is a scope to further increase the exploration ability of GWO. In this paper, the random opposition-based learning strategy is proposed to enhance the diversity and help the population jump out from the local optima.

Opposition-based learning (OL) is one of the powerful optimization tools developed by Tizhoosh [43]. The OL concept has successfully been used in various meta-heuristics [44]–[46] used to enhance the convergence speed.

Definition 1: Opposite number. The opposite of real number $x \in [l, u]$ is given by $\hat{x}$ [43]:

$$\hat{x} = l + u - x \tag{10}$$

where l and u are the lowest and upper bound of search space, respectively.

Definition 2: Opposite point. Suppose $\mathbf{X} = [x_1, x_2, \ldots, x_n]$, where $x_1, x_2, \ldots, x_n \in R$ and $x_j \in [l_j, u_j]$. The opposite point $\hat{\mathbf{X}} = [\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n]$ is defined by:

$$\hat{x}_j = l_j + u_j - x_j, \quad j = 1, 2, \ldots, n \tag{11}$$

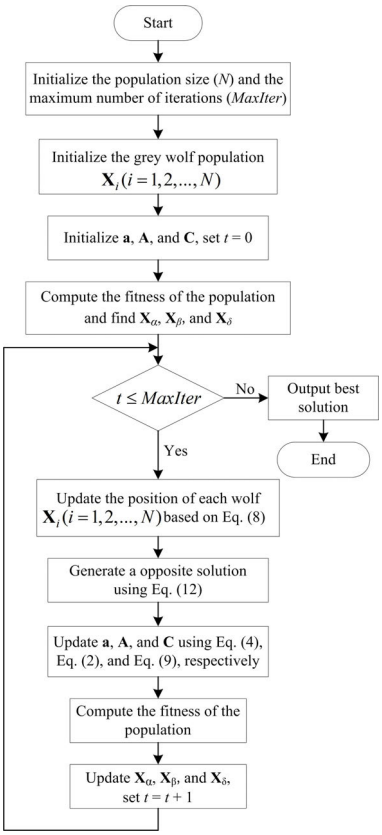


FIGURE 2. The flow chart of ROL-GWO algorithm.

In opposition-based optimization strategy, the opposite point $\hat{\mathbf{X}}$ is replaced with it is corresponding solution $\mathbf{X}$ based on the fitness function. If $f(\mathbf{X})$ is better than $f(\hat{\mathbf{X}})$ then $\mathbf{X}$ not changed, otherwise, $\mathbf{X} = \hat{\mathbf{X}}$, therefore, the solutions population are updated based on the better value of $\mathbf{X}$ and $\hat{\mathbf{X}}$.

Different from Eq. (11), this paper presents a new OL strategy, called random OL, and is defined by:

$$\hat{x}_j = l_j + u_j - r_3 \times x_j, \quad j = 1, 2, \ldots, n \tag{12}$$

where r_3 is a random number in [0,1]. Compared with the Eq. (11), the opposite solution $\hat{x}_j$ described by Eq. (12) is random enough for exploration. Therefore, the Eq. (12) can effectively enhance the diversity of the population and help the population jump out to the local optima.

The framework of the proposed ROL-GWO algorithm is shown in Fig. 2.

IV. EXPERIMENTS AND DISCUSSION

In this section, to investigate the performance of the proposed ROL-GWO algorithm, a set of experiments is performed using two sets of global benchmark test functions. One is the 23 widely used benchmark functions from literatures; the other is the 30 benchmark test functions from the IEEE CEC2014 special session [47]. In addition, to test the proposed ROL-GWO algorithm as a practical method, three engineering design optimization problems are used.

TABLE 1. 23 widely used benchmark test functions.

Function category	Function name	Function equation	Search range
Unimodal function	Sphere	$f_1(\mathbf{X}) = \sum_{i=1}^{D} x_i^2$	[−100, 100]
	Schwefel's 2.22	$f_2(\mathbf{X}) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	[−10, 10]
	Schwefel's 1.2	$f_3(\mathbf{X}) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	[−100, 100]
	Schwefel's 2.21	$f_4(\mathbf{X}) = \max_i \{ \lvert x_i \rvert, 1 \le x_i \le D \}$	[−100, 100]
	Rosenbrock	$f_5(\mathbf{X}) = \sum_{i=1}^{D} [100 (x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	[−30, 30]
	Step	$f_6(\mathbf{X}) = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	[−100, 100]
	Noise	$f_7(\mathbf{X}) = \sum_{i=1}^{D} i x_i^4 + random[0,1)$	[−1.28, 1.28]
	Quartic	$f_8(\mathbf{X}) = \sum_{i=1}^{D} i x_i^4$	[−1.28, 1.28]
	Sum-Square	$f_9(\mathbf{X}) = \sum_{i=1}^{D} i x_i^2$	[−10, 10]
	Sum-Power	$f_{10}(\mathbf{X}) = \sum_{i=1}^{D} \lvert x_i \rvert^{(i+1)}$	[−1, 1]
	Elliptic	$f_{11}(\mathbf{X}) = \sum_{i=1}^{D} (10^6)^{(i-1)/(n-1)} x_i^2$	[−100, 100]
Multimodal function	Rastrigin	$f_{12}(\mathbf{X}) = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$	[−5.12, 5.12]
	Ackley	$f_{13}(\mathbf{X}) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2} \right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D} \cos(2\pi x_i) \right) + 20 + e$	[−32, 32]
	Griewank	$f_{14}(\mathbf{X}) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	[−600, 600]
	Alpine	$f_{15}(\mathbf{X}) = \sum_{i=1}^{D} \lvert x_i \sin(x_i) + 0.1 x_i \rvert$	[−10, 10]
	Levy	$f_{16}(\mathbf{X}) = \sum_{i=1}^{D} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] + \sin^2(3\pi x_1) + \lvert x_D - 1 \rvert [1 + \sin^2(3\pi x_D)]$	[−10, 10]
	Salomon	$f_{17}(\mathbf{X}) = 1 - \cos\left(2\pi\sqrt{\sum_{i=1}^{D} x_i^2} \right) + 0.1\sqrt{\sum_{i=1}^{D} x_i^2}$	[−100, 100]
	Inverted Cosine Mixture	$f_{18}(\mathbf{X}) = 0.1D - \left(0.1\sum_{i=1}^{D} \cos(5\pi x_i) - \sum_{i=1}^{D} x_i^2 \right)$	[−1, 1]
	Pathological	$f_{19}(\mathbf{X}) = \sum_{i=2}^{D} 0.5 + \frac{\sin^2\left(\sqrt{100 x_{i-1}^2 + x_i^2} \right) - 0.5}{1 + 0.001 (x_{i-1}^2 - 2x_{i-1}x_i + x_i^2)^2}$	[−100, 100]
	Levy and Montalo	$f_{20}(\mathbf{X}) = 0.1(\sin^2(3\pi x_1) + \sum_{i=1}^{D-1} (x_i - 1)^2 (1 + \sin^2(3\pi x_{i+1})) + (x_D - 1)^2 (1 + \sin^2(2\pi x_D)))$	[−5, 5]
	Schaffer	$f_{21}(\mathbf{X}) = 0.5 + \frac{\sin^2\left(\sqrt{\sum_{i=1}^{D} x_i^2} \right) - 0.5}{\left(1 + 0.001 \cdot \sum_{i=1}^{D} x_i^2 \right)^2}$	[−100, 100]
	Easom	$f_{22}(\mathbf{X}) = (-1)^{n+1} \prod_{i=1}^{D} \cos(x_i) \cdot \exp[-\sum_{i=1}^{D} (x_i - \pi)^2]$	[−100, 100]
	Stretched V-sine	$f_{23}(\mathbf{X}) = \sum_{i=1}^{D-1} (x_i^2 + 2x_{i+1}^2)^{0.25} \cdot ((\sin 50 (x_i^2 + x_{i+1}^2)^{0.1})^2 + 1)$	[−10, 10]

A. BENCHMARK TEST FUNCTIONS

The definition of the 23 widely used benchmark test functions is given in Table 1. In Table 1, the test functions are divided into two groups: unimodal and multimodal functions. The unimodal functions ($f_1 - f_{11}$) are suitable for benchmarking the exploitation of algorithms since they have one global optima and no local optimum. On the contrary, multimodal functions ($f_{12} - f_{23}$) have a large number of local optimum avoidance of algorithm [38], [41].

B. COMPARED WITH GWO AND ITS VARIANTS

In this subsection, to study the performance of the proposed algorithm for the 23 benchmark test functions provided in Table 1, we compared the ROL-GWO with the basic GWO algorithm, the modified GWO (mGWO) [41] algorithm, the modified GWO-I (MGWO-I) algorithm [36], and the exploration-enhanced GWO (EEGWO) [38] algorithm using several independent experiments. We set the same common control parameters for all of the algorithms for a fair comparison. For each algorithm, the population size (N) is set to 30 and the maximum number of iterations ($MaxIter$) is set to 500 (i.e., the maximum number of fitness function evaluations (FFEs) is 15,000) on all of the simulations. The dimension of each function is set to 30. We tested each function 30 times. Table 2 provides the average (mean) and standard deviation (St.dev) of function results found by ROL-GWO and other four algorithms on 23 test functions from Table 1,

TABLE 2. Comparisons of ROL-GWO and other four selected algorithms on 23 benchmark test functions with 30D in Table 1.

Func	GWO		mGWO		MGWO-I		EEGWO		ROL-GWO	
	Mean	St.dev	Mean	St.dev	Mean	St.dev	Mean	St.dev	Mean	St.dev
f_1	1.59E-29	1.09E-29	1.11E-36	1.14E-36	1.13E-42	1.25E-42	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_2	6.80E-18	4.36E-18	4.32E-22	5.51E-22	1.37E-25	1.55E-25	3.42E-238	0.00E+00	0.00E+00	0.00E+00
f_3	1.81E-05	2.08E-05	1.91E-08	1.72E-08	1.28E-12	2.50E-12	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_4	5.62E-07	2.95E-07	9.96E-10	4.81E-10	2.89E-13	4.74E-13	2.31E-214	0.00E+00	0.00E+00	0.00E+00
f_5	2.75E+01	7.34E-01	2.71E+01	6.12E-01	2.80E+01	5.90E-01	2.90E+01	1.27E-02	2.90E+01	1.41E-02
f_6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_7	1.39E-03	6.15E-04	1.44E-03	2.97E-04	5.90E-04	2.23E-04	1.41E-04	1.20E-04	4.51E-05	2.72E-05
f_8	2.80E-52	3.83E-52	2.84E-64	2.35E-64	1.48E-78	1.87E-78	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_9	3.25E-30	3.41E-30	6.77E-37	5.72E-37	1.11E-42	1.61E-42	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{10}	8.12E-126	1.20E-125	1.85E-126	3.40E-126	2.84E-150	2.40E-150	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{11}	2.01E-24	2.46E-24	2.69E-40	4.32E-40	5.68E-39	5.77E-39	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{12}	6.31E-01	9.52E-01	2.27E-14	3.11E-14	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{13}	6.84E-14	6.47E-15	2.93E-14	4.78E-15	1.51E-14	0.00E+00	4.44E-15	0.00E+00	8.88E-16	0.00E+00
f_{14}	2.46E-03	5.26E-03	1.53E-03	4.83E-03	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{15}	6.29E-04	3.85E-04	2.05E-20	1.78E-20	5.97E-05	2.65E-05	2.80E-239	0.00E+00	0.00E+00	0.00E+00
f_{16}	1.85E-31	1.07E-31	2.50E-39	1.23E-39	2.51E-43	3.52E-43	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{17}	2.00E-01	0.00E+00	1.40E-01	5.48E-02	9.99E-02	0.00E+00	2.00E-02	4.47E-02	0.00E+00	0.00E+00
f_{18}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{19}	6.63E+00	2.60E+00	1.07E+01	9.92E-01	8.45E+00	1.56E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{20}	1.13E-32	2.32E-32	2.11E-39	3.55E-39	2.18E-46	2.87E-46	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{21}	3.72E-02	0.00E+00	2.62E-02	1.51E-02	9.72E-03	0.00E+00	1.94E-03	4.35E-03	0.00E+00	0.00E+00
f_{22}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{23}	5.65E-07	3.85E-07	1.59E-09	8.16E-10	8.48E-12	2.66E-12	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Average ranking	4.2174		3.4783		2.6957		1.4348		1.1304	
Overall ranking	5		4		3		2		1	

and the results of the Friedman's ranking test. All algorithms are coded in Matlab R2014a, and all of the experiments are performed on a computer with Intel(R), Core(TM)2, Quad CPU Q8300 @ 2.50 GHz and 4.00 GB RAM in the Windows 10 environment.

From Table 2, the ROL-GWO algorithm could obtain theoretical optima (0) for all of the test functions except for f_5, f_7, and f_{13}. Compared with the basic GWO and mGWO algorithms, ROL-GWO found better and similar results on 19 and three test functions, respectively. However, the better result was obtained by GWO and mGWO for f_5. With respect to MGWO-I algorithm, ROL-GWO provided better and similar results on 17 and 5 test functions, respectively. For f_5, the better result was obtained by MGWO-I. In addition, the ROL-GWO significantly outperformed the EEGWO algorithm for seven test functions (i.e., f_2, f_4, f_7, f_{13}, f_{15}, f_{17}, and f_{21}). For the rest functions, ROL-GWO and EEGWO algorithms found similar results. From Table 2, ROL-GWO ranked the first in the Friedman's test.

For further illustration, the convergence graphs of the GWO, mGWO, MGWO-I, EEGWO and ROL-GWO algorithms on 12 representative test functions are shown in Fig. 3. As can be seen from Fig. 3, ROL-GWO converges faster than the other four algorithms on all of these test functions.

To further investigate the scalability of the proposed method, ROL-GWO was tested with higher problems. In terms of the 23 benchmark test functions with $D = 500$ from Table 1. The average (mean) and standard deviation (st.dev) of objective functions values found by ROL-GWO and other four algorithms, and the results of Friedman's test are shown in Table 3. The results of all the algorithms were averaged over 30 independent runs. It is noted that we used the same parameter settings as in the above experiments, and no increase in population size or number of function evaluations was required.

From Table 3, ROL-GWO has shown very good scalability to the search dimension, i.e., the performance of ROL-GWO did not deteriorate seriously as the dimension increased. It must be emphasized that problem optimization for 500 dimensions was very challenging for GWO because it does not use any particular operators tailored to solve high dimensional optimization problems. Compared with the GWO and MGWO-I algorithms, ROL-GWO found better and similar results on 20 and two test functions (f_6 and f_{22}), respectively. However, the better result was obtained by GWO and MGWO-I for function f_5. With respect to the mGWO algorithm, ROL-GWO get better and similar results on 20 and three test functions

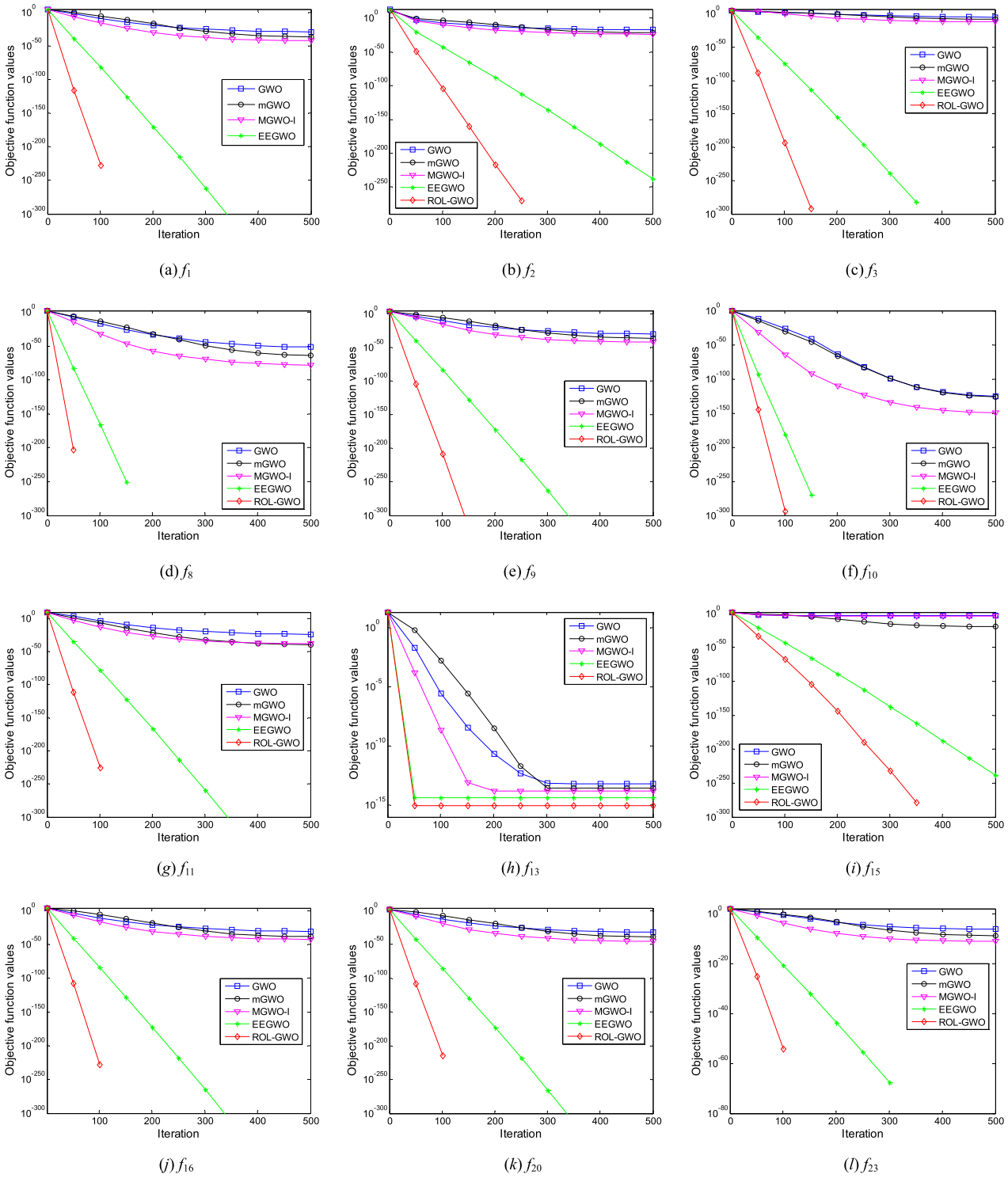


FIGURE 3. Convergence graphs of ROL-GWO and other four selected algorithms on 12 representative test functions with 30*D* in Table 1.

(f_5, f_6, and f_{22}), respectively. Compared to the EEGWO algorithm, ROL-GWO obtained better and similar results on 7 and 15 test functions, respectively. For f_5, the better result obtained by EEGWO. From Table 3, ROL-GWO ranked the first in the Friedman's test.

The convergence curves of the average function values derived from ROL-GWO and the other four algorithms are plotted in Fig. 4 for 12 representative test functions with 500 dimensions. From Fig. 4, ROL-GWO converges faster than other four algorithms on all these 12 test functions with $D = 500$.

Furthermore, we applied statistical significance testing, which is a meaningful method to investigate the difference between any two stochastic algorithms, to make

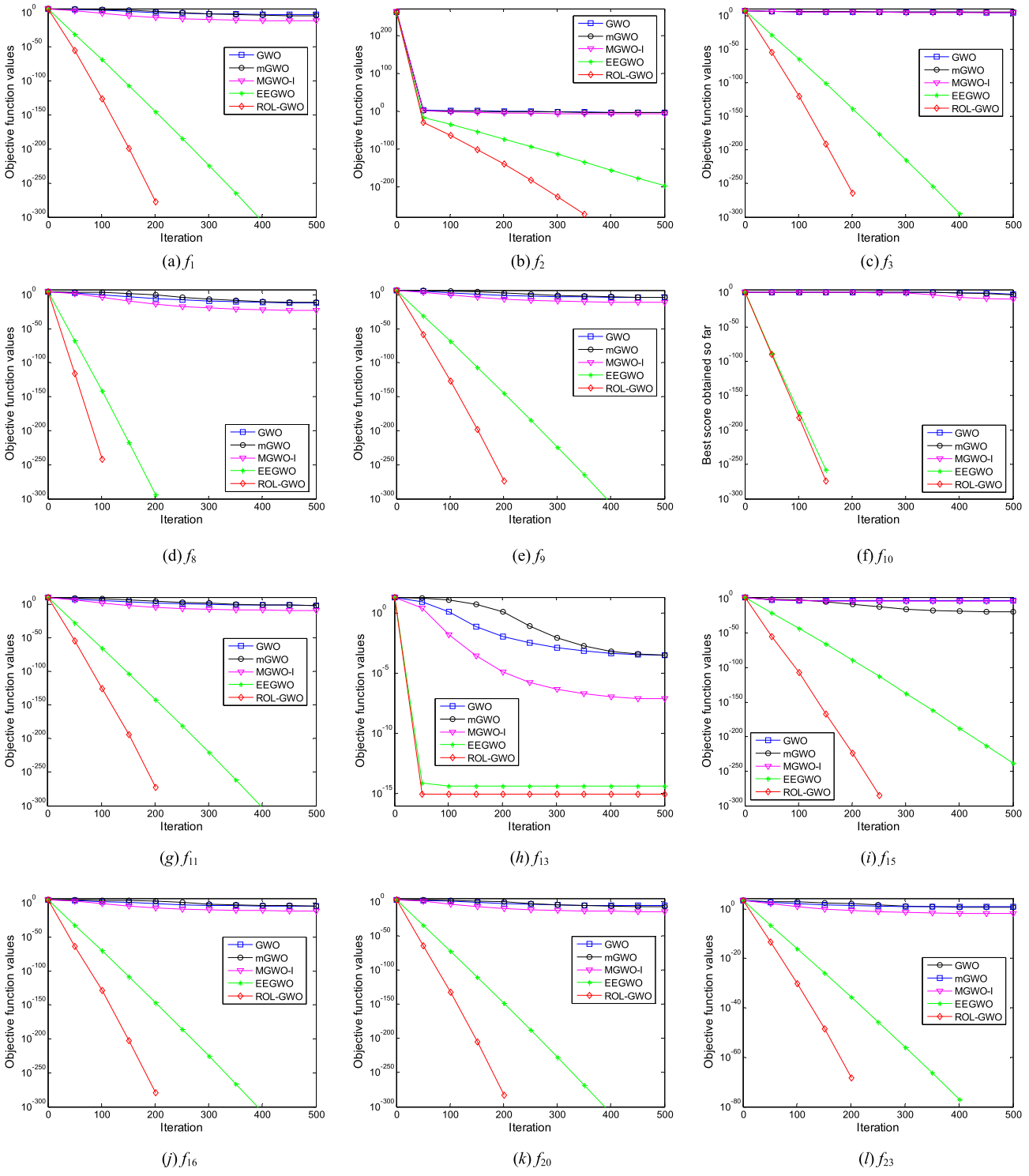


FIGURE 4. **Convergence graphs of ROL-GWO and other four selected algorithms on 12 representative test functions with 500*D* in Table 1.**

comparisons between the GWO and its four variants. The multiple-problem Wilcoxon's test and Wilcoxon's rank sum test were used to check the behaviors of the five algorithms. The statistical results based on the average function values are listed in Table 4, where $R = R^+$ or R^- is the sum of ranks based on the absolute value of the difference between two tested algorithms.

As can be seen from Table 4 shows that ROL-GWO provided higher R^+ values than R^- values in all cases. Based on the Wilcoxon's test, when $\alpha = 0.05$, a significant difference

TABLE 3. Comparisons of ROL-GWO and other four selected algorithms on 23 benchmark test functions with 500D in Table 1.

Func	GWO		mGWO		MGWO-I		EEGWO		ROL-GWO	
	Mean	St.dev	Mean	St.dev	Mean	St.dev	Mean	St.dev	Mean	St.dev
f_1	2.10E-03	7.79E-04	3.63E-05	7.21E-06	1.85E-12	5.04E-13	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_2	1.63E-03	2.44E-04	1.30E-03	1.41E-04	1.26E-07	5.27E-08	7.48E-198	0.00E+00	0.00E+00	0.00E+00
f_3	2.10E+05	9.61E+04	3.49E+05	4.43E+04	4.87E+04	6.21E+04	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_4	6.43E+01	4.62E+00	6.96E+01	3.74E+00	6.10E+01	5.87E+00	7.63E-191	0.00E+00	0.00E+00	0.00E+00
f_5	4.98E+02	1.12E-01	4.99E+02	1.57E-02	4.98E+02	8.86E-02	4.98E+02	2.29E-01	4.99E+02	1.73E-02
f_6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_7	2.75E-02	5.59E-03	3.57E-02	8.66E-03	6.34E-03	3.68E-03	4.39E-04	2.46E-04	8.51E-05	1.34E-04
f_8	1.87E-12	1.41E-12	4.36E-12	2.64E-12	8.38E-24	6.82E-24	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_9	9.45E-05	6.89E-05	1.12E-04	6.03E-05	5.01E-12	1.51E-12	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{10}	8.40E-03	9.70E-03	6.19E-04	3.45E-04	1.63E-10	3.86E-10	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{11}	8.82E-02	2.85E-02	7.88E-02	3.96E-02	2.56E-09	1.48E-09	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{12}	2.34E+01	1.08E+01	1.29E+01	9.11E+00	2.18E-11	1.13E-11	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{13}	3.10E-04	4.03E-05	3.22E-04	1.03E-04	7.43E-08	2.08E-08	4.44E-15	0.00E+00	8.88E-16	0.00E+00
f_{14}	1.40E-02	3.12E-02	6.77E-06	3.65E-06	4.54E-13	2.71E-13	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{15}	2.31E-02	3.50E-03	2.11E-02	8.05E-03	5.18E-04	2.94E-04	8.06E-199	0.00E+00	0.00E+00	0.00E+00
f_{16}	3.26E-05	2.87E-05	8.22E-05	1.20E-04	3.23E-12	2.74E-12	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{17}	6.80E-01	8.37E-02	1.14E+00	1.52E-01	2.80E-01	4.47E-02	9.99E-02	2.21E-05	0.00E+00	0.00E+00
f_{18}	1.25E-06	6.66E-07	4.43E-08	1.71E-08	4.26E-14	1.12E-14	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{19}	2.39E+02	2.26E+00	2.40E+02	1.36E+00	2.17E+02	2.39E+01	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{20}	1.19E-06	5.98E-07	4.33E-08	2.08E-08	2.79E-15	1.85E-15	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{21}	2.64E-01	2.01E-02	2.08E-01	2.71E-02	7.00E-02	1.83E-02	9.78E-03	9.27E-05	0.00E+00	0.00E+00
f_{22}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{23}	7.62E+00	5.79E-01	4.20E+00	8.23E-01	1.30E-02	1.57E-02	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Average ranking	4.0870		4.1304		2.7391		1.3043		1.1304	
Overall ranking	4		5		3		2		1	

TABLE 4. Results of the statistical test for ROL-GWO, GWO, mGWO, MGWO-I, and EEGWO at a 0.05 significance level.

Algorithm	Dimension	R^+	R^-	p-value	$\alpha = 0.05$
ROL-GWO vs. GWO	$D = 30$	224.0	52.0	1.3961E-05	Yes
ROL-GWO vs. mGWO		224.0	52.0	1.9197E-05	Yes
ROL-GWO vs. MGWO-I		206.5	69.5	1.1338E-04	Yes
ROL-GWO vs. EEGWO		152.0	124.0	1.0950E-01	No
ROL-GWO vs. GWO	$D = 500$	237.5	38.5	6.3160E-07	Yes
ROL-GWO vs. mGWO		243.0	33.0	7.5540E-07	Yes
ROL-GWO vs. MGWO-I		235.5	40.5	1.8258E-06	Yes
ROL-GWO vs. EEGWO		148.0	128.0	1.1280E-01	No

can be observed in three cases (i.e., ROL-GWO vs. GWO, ROL-GWO vs. mGWO, and ROL-GWO vs. MGWO-I), meaning ROL-GWO is significantly better than GWO, mGWO, and MGWO-I on 23 test functions with $\alpha = 0.05$.

C. COMPARED WITH OTHER ALGORITHMS

In this subsection, ROL-GWO was also compared to four other meta-heuristic algorithms, such as DIW-PSO [48], which was developed by Jiao et al., is a new particle swarm optimization (PSO) with dynamic inertia weight; GABC [49], which was proposed by Zhu and Kwong, is a novel artificial bee colony (ABC) algorithm with gbest inform-ation; ODE [44], which was presented by Rahnamayan et al., a modified differential evolution with opposition-based learning strategy; and OTLBO [50], which was presented by Roy et al., is an improved teaching learning based optimization (TLBO) with opposition learning strategy. The reasons for selecting these four algorithms for comparison are: (1) DIW-PSO, GABC, ODE, and OTLBO represent the best-performing

TABLE 5. Comparisons of ROL-GWO and DIW-PSO, GABC, ODE, and OTLBO algorithms on 23 test functions with 30D in Table 1.

Func	DIW-PSO		GABC		ODE		OTLBO		ROL-GWO	
	Mean	St.dev	Mean	St.dev	Mean	St.dev	Mean	St.dev	Mean	St.dev
f_1	1.20E-15	1.41E-15	2.25E-28	3.62E-28	5.57E-30	8.04E-30	7.15E-96	8.39E-96	0.00E+00	0.00E+00
f_2	2.22E-09	2.91E-09	2.68E-15	2.19E-15	5.29E-17	2.02E-17	2.46E-48	1.81E-48	0.00E+00	0.00E+00
f_3	2.19E-14	2.23E-14	5.33E-28	6.63E-28	2.27E-29	4.01E-29	2.95E-95	3.88E-95	0.00E+00	0.00E+00
f_4	2.49E+00	7.58E-01	3.55E-01	3.61E-01	2.96E-10	3.05E-10	8.39E-40	6.53E-40	0.00E+00	0.00E+00
f_5	3.88E+01	4.27E+01	2.31E+01	3.68E+00	2.81E+01	4.14E-01	2.69E+01	4.59E-01	2.90E+01	1.41E-02
f_6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_7	1.97E-02	8.40E-03	8.80E-03	1.72E-03	3.96E-03	1.30E-03	8.49E-04	3.26E-04	4.51E-05	2.72E-05
f_8	4.20E-20	5.69E-20	2.47E-47	4.70E-47	5.63E-53	5.04E-53	9.74E-186	0.00E+00	0.00E+00	0.00E+00
f_9	2.14E-16	2.08E-16	4.07E-29	5.05E-29	2.97E-31	3.03E-31	1.45E-95	2.68E-95	0.00E+00	0.00E+00
f_{10}	4.56E-34	8.80E-34	4.93E-64	8.38E-64	1.84E-105	4.10E-105	3.91E-197	0.00E+00	0.00E+00	0.00E+00
f_{11}	2.39E-10	3.72E-10	2.68E-24	9.95E-25	2.07E-26	2.20E-26	3.42E-91	3.60E-91	0.00E+00	0.00E+00
f_{12}	3.60E+01	1.53E+01	3.60E+01	1.09E+01	5.10E+01	1.97E+01	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{13}	1.40E-07	2.42E-07	1.20E-14	6.94E-15	2.22E-15	0.00E+00	2.22E-15	0.00E+00	8.88E-16	0.00E+00
f_{14}	7.79E-15	9.71E-15	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{15}	1.87E-08	3.49E-08	1.53E-13	2.31E-13	3.84E-17	2.49E-17	4.28E-49	2.70E-49	0.00E+00	0.00E+00
f_{16}	1.58E+00	1.33E+00	1.58E+00	1.13E+00	2.95E-31	2.00E-31	1.36E-93	1.97E-93	0.00E+00	0.00E+00
f_{17}	5.60E-01	5.40E-02	3.00E-01	7.07E-02	2.20E-01	4.47E-02	9.99E-02	0.00E+00	0.00E+00	0.00E+00
f_{18}	2.07E-01	1.68E-01	2.36E-01	1.69E-01	3.22E-33	5.37E-33	9.50E-99	1.03E-98	0.00E+00	0.00E+00
f_{19}	9.64E+00	9.22E-01	1.26E+01	1.66E-01	3.58E+00	4.27E-01	6.27E+00	2.74E-01	0.00E+00	0.00E+00
f_{20}	2.06E-18	1.65E-18	5.51E-31	3.46E-31	2.42E-33	2.64E-33	6.04E-98	9.92E-98	0.00E+00	0.00E+00
f_{21}	1.37E-01	2.29E-02	7.16E-02	3.71E-02	3.72E-02	0.00E+00	9.72E-03	1.41E-07	0.00E+00	0.00E+00
f_{22}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{23}	4.12E+00	1.40E+00	2.47E-01	1.37E-01	1.75E-07	3.59E-08	6.40E-23	3.13E-23	0.00E+00	0.00E+00
Average ranking	4.4348		3.5217		2.7826		1.8261		1.1304	
Overall ranking	5		4		3		2		1	

variants of PSO, ABC, DE, and TLBO techniques, respectively. (2) Their numerical performances are very competitive. The population sizes of DIW-PSO, GABC, ODE, and OTLBO were set to 60, 60, 60, and 30, respectively. For four algorithms, the maximum number of iterations was set to 500. The other parameters of four algorithms are the same as their origin papers. The dimension of each function is set to 30. Tables 5-6 provide the average (mean) and standard deviation (st.dev) of function values results found by ROL-GWO and other four algorithms on 23 test functions from Table 1 with $D = 30$, the results of the Friedman's test, the results of the multi-problem Wilcoxon's test, the results of the Wilcoxon's rank sum test, respectively. The results of all the algorithms were averaged over 30 independent runs.

From Table 5, the total performance of ROL-GWO is best. Compared with DIW-PSO algorithm, ROL-GWO found better and similar results on 21 and two test functions (f_6 and f_{22}), respectively. With respect to the GABC and ODE algorithms, ROL-GWO provided better and similar results on 19 and three test functions (f_6, f_{14}, and f_{22}), respectively. Compared to the OTLBO algorithm, ROL-GWO could get better and similar results on 18 and 4 functions (f_6, f_{12}, f_{14}, and f_{22}), respectively. However, for f_5, the better results were obtained by GABC, ODE, and OTLBO algorithms. In addition, ROL-GWO ranked the first in the Friedman's test.

TABLE 6. Results of the multiple-problem Wilcoxon's test for ROL-GWO and other four selected algorithms on 23 functions with 30*D*.

ROL-GWO vs	R^+	R^-	*p*-value	$\alpha = 0.05$
DIW-PSO	253.5	22.5	2.0491E-06	Yes
GABC	225.0	51.0	9.0625E-06	Yes
ODE	225.0	51.0	2.1325E-05	Yes
OTLBO	215.0	61.0	8.2590E-05	Yes

From Table 6, ROL-GWO provided higher R^+ values than R^- values in all cases. Based on the Wilcoxon's test, when $\alpha = 0.05$, a significant difference can be observed in all of the cases, meaning ROL-GWO is significantly better than DIW-PSO, GABC, ODE, and OTLBO on 23 test functions with $\alpha = 0.05$.

Furthermore, the evolution curves of the mean objective function values of the five algorithms in six representative test functions are shown in Fig. 5. As seen in Fig. 5, ROL-GWO was faster than other four algorithms for six repre-sentative test functions.

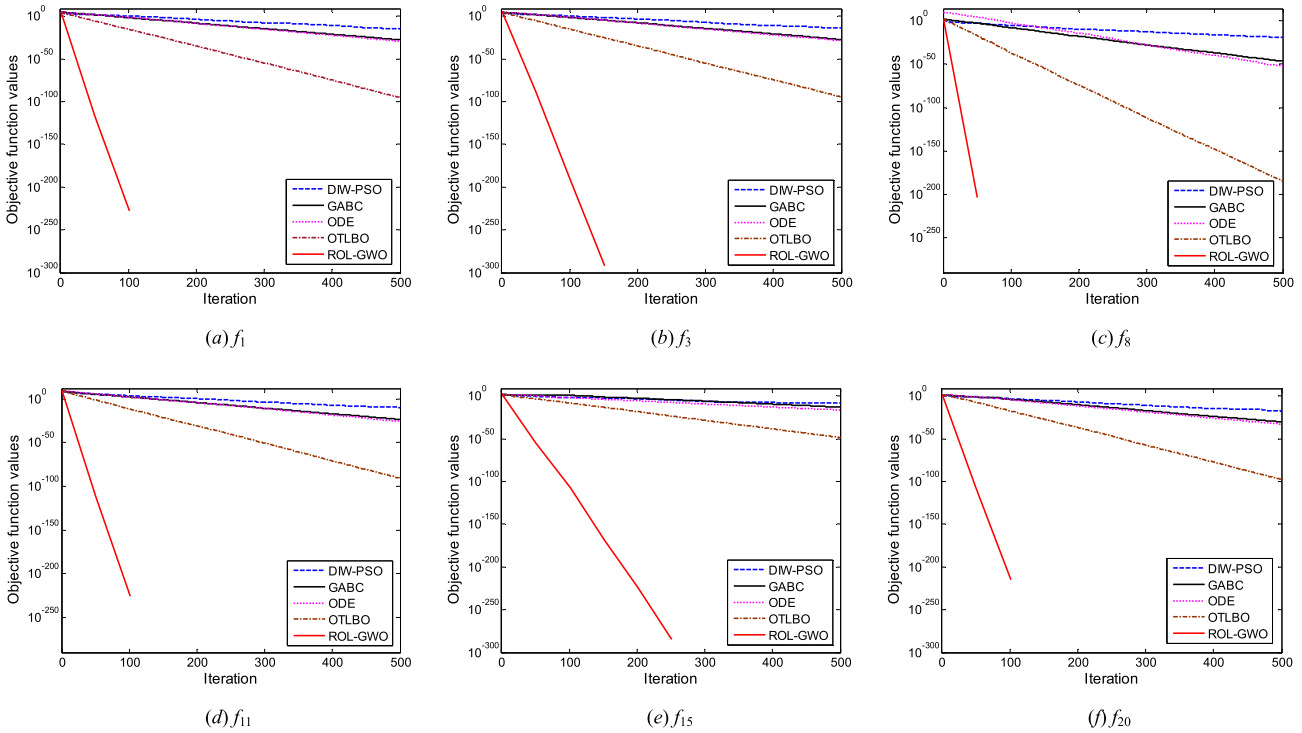


FIGURE 5. Convergence graphs of ROL-GWO and other four selected algorithms on six representative test functions with 30*D* in Table 1.

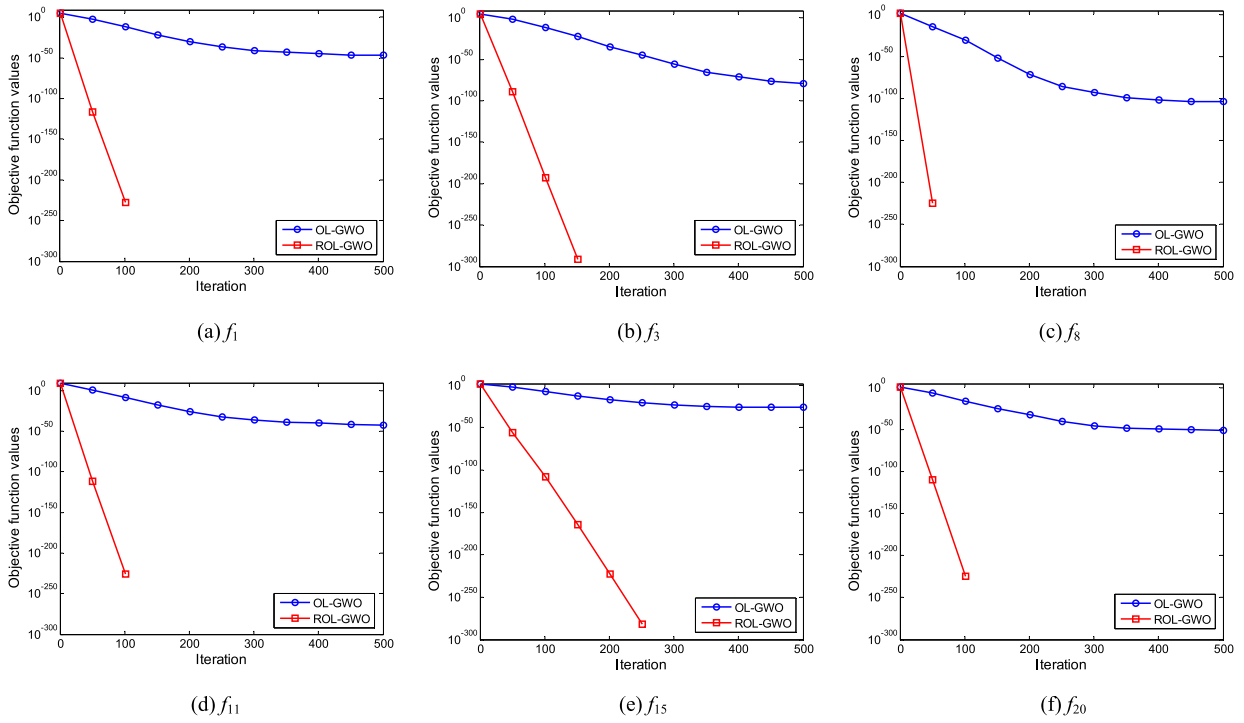


FIGURE 6. Convergence graphs of RL-GWO and OL-GWO on six representative test functions with 30*D* in Table 1.

D. COMPARED WITH OPPOSITION-BASED LEARNING

In this subsection, to further verify the performance, ROL-GWO is compared to GWO with opposition-based learning (denoted as OL-GWO) strategy. The 23 benchmark test functions with 30 dimensions from Table 1 were used to conduct experiment. In this experiment, ROL-GWO adopts the random opposition-based learning strategy (i.e., Eq. (12)), while OL-GWO utilizes the opposition-based

TABLE 7. Comparisons of OL-GWO and ROL-GWO algorithms on 23 test functions with 30*D* in Table 1.

Function	OL-GWO				ROL-GWO			
	Best	Mean	Worst	St.dev	Best	Mean	Worst	St.dev
f_1	2.06E-54	1.99E-47	7.79E-47	3.63E-47	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_2	1.97E-28	1.12E-26	2.81E-26	1.38E-26	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_3	1.26E-83	1.90E-79	6.29E-79	2.66E-79	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_4	1.15E-45	1.82E-42	6.59E-42	2.72E-42	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_5	2.88E+01	2.89E+01	2.90E+01	1.17E-01	2.90E+01	2.90E+01	2.90E+01	1.41E-02
f_6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_7	1.57E-06	8.47E-05	2.27E-04	8.77E-05	1.18E-06	4.51E-05	9.81E-05	2.72E-05
f_8	2.03E-113	3.06E-104	2.64E-103	1.20E-103	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_9	1.05E-51	6.19E-48	3.04E-47	1.35E-47	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{10}	1.04E-210	1.37E-195	6.86E-195	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{11}	9.05E-48	5.78E-43	1.92E-42	8.26E-43	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{12}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{13}	4.44E-15	5.86E-15	7.99E-15	1.94E-15	8.88E-16	8.88E-16	8.88E-16	0.00E+00
f_{14}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{15}	6.13E-32	6.03E-27	1.23E-26	5.41E-27	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{16}	3.43E-57	3.71E-47	1.85E-46	8.27E-47	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{17}	8.16E-23	4.00E-02	9.99E-02	5.47E-02	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{18}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{19}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{20}	8.65E-54	3.87E-51	2.04E-50	9.06E-51	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{21}	9.70E-03	9.72E-03	9.80E-03	4.47E-05	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{22}	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f_{23}	1.93E-13	6.69E-13	1.59E-12	5.39E-13	0.00E+00	0.00E+00	0.00E+00	0.00E+00

learn-ing strategy (i.e., Eq. (11)). We implemented 30 runs for each test function. It is noted that we use the same parameter settings as in the above experiments. Table 7 provides the best, mean, worst, and standard deviation values obtained by ROL-GWO and OL-GWO algorithms.

From Table 7, compared with OL-GWO, ROL-GWO provided better and similar results on 16 and 6 test func-tions (f_6, f_{12}, f_{14}, f_{18}, f_{19}, and f_{22}), respectively. For f_5, the better results of the ''best'' and ''mean'' were obtained by OL-GWO. In addition, ROL-GWO found similar ''worst'' and better ''st.dev'' values. Based on the above mentioned comparisons, a significant difference can be observed in most test functions, meaning ROL-GWO is significantly better than OL-GWO in most cases. In other words, GWO with random opposition-based learning (ROL) strategy can obtain better performance than GWO with opposition-based learning (OL) strategy in most cases.

Fig. 6 provided the convergence curves of six representative test functions. As can be seen from Fig. 6, the ROL-GWO has better convergence accuracy and faster convergence speed than OL-GWO.

E. ROL-GWO APPLIED TO THE CEC 2014 PROBLEMS

In this subsection, we further investigated the performance of ROL-GWO by using the other 30 benchmark test func-tions from IEEE CEC2014, which are more complicated than the 23 benchmark test functions in Table 1. These 30 test functions can be divided into four classes: 1) unimodal functions (Fc01-Fc03); 2) multi-modal functions (Fc04-Fc16); 3) hybrid functions (Fc17-Fc22); and composition functions (Fc23-Fc30). A detailed description of these 30 test functions can be shown in [47]. In these 30 test functions, the search range is $[-100,100]$, and the dimen-sions are set to 30.

To verify the efficiency and effectiveness of ROL-GWO, the results are compared with four state-of-the-art optimization techniques such as CLPSO [51] is an improved version of PSO based on comprehensive learning concept; CoDE [52] is a modified DE with composite trial vector generation strategies and control parameters; MoABC [53] is a modified version of ABC and uses a grid-based approach; HSCA [6] is a novel hybrid version of CSA based on self-adaptive control parameters and a linear population reduction. The above mentioned four optimization techniques represent the state-of-the-art in PSO, DE, ABC, and SCA algorithms respectively and their performances are very competitive. The parameters of these four algorithms are set as the same of their original papers. To ensure a fair comparison, the same maximum number of function evaluations is set to $3.00\text{E}+05$, which is the stopping criterion for five algorithms. For each function, the error values $(f(x)-f(x_0))$ are compared during 30 independent trials. Note that x is the best result when a method ends and x_0 is the global optimum. Table 8 shows the average (mean) and standard deviation (st.dev) of function values, and the results of the Friedman's ranking test, respectively.

TABLE 8. Comparisons of ROL-GWO and other four selected algorithms on 30 test functions from IEEE CEC 2014.

Function	CLPSO		CoDE		MoABC		HSCA		ROL-GWO	
	Mean	St.dev	Mean	St.dev	Mean	St.dev	Mean	St.dev	Mean	St.dev
Fc01	1.48E+08	2.57E+07	1.21E+07	4.48E+06	2.81E+07	1.01E+07	3.50E+07	2.49E+07	2.17E+07	1.24E+06
Fc02	6.81E+09	1.12E+09	1.89E+07	9.45E+06	2.88E+04	4.11E+04	1.95E+07	5.49E+07	8.66E+06	3.59E+05
Fc03	9.86E+04	1.68E+04	4.16E+03	1.89E+03	1.06E+04	3.66E+03	3.10E+04	1.36E+04	1.26E+03	5.72E+02
Fc04	9.77E+02	1.39E+02	1.44E+02	1.55E+01	1.59E+02	2.76E+01	2.03E+02	6.69E+01	1.71E+02	2.51E+01
Fc05	2.11E+01	4.87E-02	2.10E+01	6.56E-02	2.04E+01	3.53E-02	2.00E+01	2.28E-03	2.01E+01	9.02E-03
Fc06	5.08E+01	2.46E+00	5.57E+01	2.67E+00	3.78E+01	2.65E+00	3.23E+01	3.27E+00	1.15E+01	8.39E-01
Fc07	6.32E+01	8.63E+00	1.20E+00	7.20E-02	5.72E-01	1.36E-01	1.79E+00	2.19E+00	3.66E-01	1.83E-01
Fc08	2.92E+02	1.87E+01	2.30E+02	1.45E+01	1.26E+01	1.74E+01	1.71E+02	3.46E+01	5.24E+01	2.22E+01
Fc09	4.73E+02	2.13E+01	3.80E+02	1.89E+01	2.58E+02	2.83E+01	2.80E+02	5.16E+01	8.02E+01	1.16E+01
Fc10	7.62E+03	5.19E+02	7.26E+03	3.84E+02	2.29E+02	1.07E+02	2.66E+03	5.34E+02	8.98E+02	5.43E+01
Fc11	1.14E+04	5.09E+02	1.21E+04	4.27E+02	5.74E+03	3.27E+02	4.13E+03	5.35E+02	2.62E+03	3.43E+02
Fc12	2.67E+00	3.28E-01	2.47E+00	2.74E-01	4.71E-01	5.73E-02	5.11E-01	2.56E-01	3.20E-01	3.19E-01
Fc13	7.61E-01	8.74E-02	6.53E-01	6.56E-02	4.51E-01	4.11E-02	4.81E-01	1.17E-01	2.60E-01	2.30E-01
Fc14	1.60E+01	3.71E+00	4.31E-01	8.50E-02	2.98E-01	2.50E-02	3.08E-01	5.64E-02	4.20E-01	2.88E-01
Fc15	3.31E+03	1.93E+03	3.78E+01	2.26E+00	3.14E+01	6.02E+00	9.80E+01	3.02E+01	1.12E+01	4.34E+00
Fc16	2.24E+01	2.33E-01	2.28E+01	3.26E-01	1.97E+01	4.02E-02	1.27E+01	5.01E-01	1.01E+01	5.55E-01
Fc17	1.77E+07	4.74E+06	1.81E+05	1.24E+05	1.01E+07	4.96E+06	1.48E+06	1.21E+06	7.65E+05	8.67E+05
Fc18	2.51E+07	8.22E+06	3.62E+03	2.31E+03	9.92E+03	9.94E+03	7.67E+03	6.70E+03	3.88E+03	2.24E+03
Fc19	9.23E+01	1.19E+01	3.62E+01	1.08E+01	3.33E+01	1.06E+01	5.33E+01	3.63E+01	1.88E+01	3.18E+00
Fc20	5.17E+04	1.06E+04	5.04E+02	3.17E+02	3.96E+04	1.29E+04	3.93E+04	2.20E+04	1.76E+04	9.12E+03
Fc21	5.71E+06	2.15E+06	2.12E+04	1.61E+04	7.30E+06	4.36E+06	3.54E+05	3.48E+05	5.85E+05	2.09E+05
Fc22	1.36E+03	1.71E+02	1.44E+03	1.59E+02	1.14E+03	1.89E+02	9.47E+02	3.31E+02	4.27E+02	1.60E+02
Fc23	3.90E+02	8.19E+00	3.55E+02	1.77E-01	3.57E+02	7.30E+00	3.29E+02	7.51E+00	2.00E+02	0.00E+00
Fc24	3.39E+02	6.72E+00	2.83E+02	1.80E+00	2.71E+02	1.78E+00	2.78E+02	3.11E+01	2.00E+02	0.00E+00
Fc25	2.46E+02	5.30E+00	2.18E+02	1.94E+00	2.22E+02	2.80E+00	2.23E+02	9.39E+00	2.00E+02	0.00E+00
Fc26	1.10E+02	2.83E+01	1.04E+02	1.82E+01	1.01E+02	6.75E-02	1.00E+02	1.63E-01	1.01E+02	1.41E-01
Fc27	1.33E+03	3.66E+02	1.28E+03	1.47E+02	1.08E+03	3.78E+02	4.27E+02	1.96E+01	2.00E+02	0.00E+00
Fc28	3.02E+03	4.20E+02	1.92E+03	1.26E+02	2.15E+03	3.42E+02	3.49E+03	5.48E+02	2.00E+02	0.00E+00
Fc29	4.10E+05	1.64E+05	2.00E+04	7.15E+03	3.32E+03	1.46E+03	5.44E+05	2.61E+06	2.00E+02	0.00E+00
Fc30	6.00E+04	1.52E+04	1.97E+04	2.00E+03	1.61E+04	4.10E+03	2.49E+04	2.26E+04	2.00E+02	0.00E+00
Average ranking	4.7667		3.1000		2.5667		3.0333		1.5000	
Total ranking	5		4		2		3		1	

TABLE 9. Comparisons results of ROL-GWO and other algorithms for pressure vessel design problem.

Algorithm	x_1 (best)	x_2 (best)	x_3 (best)	x_4 (best)	f (best)	f (mean)	f (worst)	St.dev	Max_NFEs
GA	0.8125	0.4375	40.3239	200.0000	6288.7445	6293.8432	6308.4970	7.4133	900,000
SPGA	0.8125	0.4375	42.0974	176.6540	6059.9463	6177.2533	6469.3220	130.9297	80,000
SMES	0.8125	0.4375	42.0981	176.6405	6059.7500	6850.0000	7332.8800	426.00	/
CPSO	0.8125	0.4375	42.0913	176.7465	6061.0777	6147.1332	6363.8041	86.45	240,000
G-QPSO	0.8125	0.4375	42.0984	176.6372	6059.7208	6440.3786	7544.4925	448.4711	8,000
CDE	0.8125	0.4375	42.0984	176.6377	6059.7340	6085.2303	6371.0455	43.013	204,800
GDA	0.8125	0.4375	42.0975	176.6484	6059.8391	6149.7276	6823.6024	210.77	20,000
CSA	0.8125	0.4375	42.0984	176.6366	6059.7144	6342.4991	7332.8416	384.9454	250,000
SCA	0.817577	0.417932	41.74939	183.5727	6137.3724	6326.7606	6512.3541	126.609	30,000
EEGWO	13.09291	6.792196	42.09758	176.6495	6059.8704	6066.7220	6091.0922	10.64121	50,000
ROL-GWO	12.73387	6.781898	42.09825	176.6397	6059.7528	6066.0068	6090.6405	13.77132	50,000

When a method achieved the best performance on the corresponding test function, the average value was highlighted with a gray background.

From Table 8, the total performance of ROL-GWO is best. Compared with CLPSO, ROL-GWO found better results on all of the test functions. With respect to CoDE, ROL-GWO obtained better and worse results on 24 and six test functions ($f_1, f_4, f_{17}, f_{18}, f_{20}$, and f_{21}), respectively. Compared to the MoABC algorithm, ROL-GWO found better and worse results on 24 and five test functions (f_2, f_4, f_8, f_{10}, and f_{14}),

TABLE 10. Comparisons results of ROL-GWO and other algorithms for welded beam design problem.

Algorithm	x_1 (best)	x_2 (best)	x_3 (best)	x_4 (best)	f (best)	f (mean)	f (worst)	St.dev	Max_NFEs
SC	0.2444	6.2380	8.2886	0.2446	2.3854	3.2551	6.3997	9.60E-01	33,095
FSA	0.2443	6.2158	8.2939	0.2443	2.3811	2.4042	2.4890	/	56,243
AATM	0.2441	6.2209	8.2982	0.2444	2.3823	2.3870	2.3916	2.20E-03	30,000
HEAA	0.2444	6.2175	8.2915	0.2444	2.3810	2.3810	2.3810	1.30E-05	30,000
EEGWO	0.2444	6.2170	8.2928	0.2444	2.3813	2.3817	2.3824	4.18E-04	50,000
ROL-GWO	0.24434	6.2188	8.2916	0.24437	2.3811	2.3812	2.3813	8.37E-05	50,000

TABLE 11. Comparisons results of ROL-GWO and other algorithms for tension/compression spring design problem.

Algorithm	x_1 (best)	x_2 (best)	x_3 (best)	f (best)	f (mean)	f (worst)	St.dev	Max_NFEs
GA	10.890522	0.363965	0.051989	0.012681	0.012742	0.012973	5.90E-05	80,000
SC	10.6484423	0.3681587	0.0521602	0.012669	0.012923	0.016717	5.92E-05	30,000
CPSO	11.244543	0.357644	0.051728	0.0126747	0.012730	0.012924	5.20E-05	200,000
AATM	11.119253	0.051813	0.359690	0.0126683	0.0127081	0.0128614	4.50E-05	25,000
GSA	14.22867	0.05000	0.317312	0.0128739	0.0134389	0.0142117	1.34E-02	30,000
GWO	12.04249	0.344541	0.051178	0.0126723	0.0126971	0.0127208	2.10E-05	30,000
MVO	14.22623	0.315956	0.05000	0.0128169	0.0144644	0.0178397	1.62E-03	30,000
SCA	12.72269	0.334779	0.050780	0.0127097	0.0128396	0.0129984	7.80E-05	30,000
MGWO	11.80809	0.348197	0.051334	0.0126696	0.0126799	0.0127057	1.10E-05	30,000
EEGWO	11.3113	0.35634	0.051673	0.012665	0.012685	0.012720	2.22E-05	50,000
ROL-GWO	11.2416	0.357538	0.0517234	0.012666	0.0126796	0.012704	1.44E-05	50,000

respectively. For f_{26}, MoABC and ROL-GWO obtained similar results. ROL-GWO surpassed HSCA on 26 test functions. However, the better results obtained by HSCA on four test functions (f_5, f_{14}, f_{21}, and f_{26}). In addition, ROL-GWO ranked the first in the Friedman's test, followed by MoABC, HSCA, CoDE, and CLPSO.

F. ROL-GWO APPLIED TO ENGINEERING DESIGN PROBLEMS

To investigate the performance of ROL-GWO on real-world applications, we implemented it on three engineering design problems, namely, pressure vessel design, tension/compression spring design, and welded beam design problems. The detailed descriptions of three design problems can be shown in [38]. These three real-world application problems are constrained optimization problems. Deb's feasibility-based rule [54] is one of the most popular constraint-handling techniques. Therefore, the Deb's feasibility-based rule is introduced to deal with constraints. Tables 9-11 show the results obtained by ROL-GWO and other optimization techniques reported in the literature on pressure vessel design, tension/compression spring design, and welded beam design problems, respectively. Please note that the results of other algorithms are taken from [38].

As can be seen from Table 9, for the pressure vessel design problem, the best result was obtained by the CSA. In terms of the mean and worst indexes, the results found by ROL-GWO were better than those obtained by the other algorithms. Moreover, for the number of FEs, G-QPSO had the minimum number of FEs (8,000), while GA had a considerable number of FEs (900,000).

From Table 10, for the welded beam design problem, the better results of the "best", "mean", "worst", "st.dev" were obtained by the HEAA. Compared with SC, AATM, and EEGWO, ROL-GWO found better results. With respect to FSA, ROL-GWO provided similar "best" value. However, the better "mean" and "worst" values obtained by ROL-GWO. In addition, AATM and HEAA had the minimum number of FEs (30,000).

As seen in Table 11, for the tension/compression spring design problem, the best result was obtained by the EEGWO. Compared with other algorithms, the better "mean" and "worst" results were obtained by ROL-GWO. Moreover, the number of FEs by ROL-GWO was moderate among the compared algorithms.

V. CONCLUSION

In this paper, a modified version of GWO, called ROL-GWO, was developed to solve global optimization pro-blems. A new control parameter "**C**" strategy was proposed to balance between global exploration and local explo-itation. In addition, a novel random opposition-based learn-ing strategy was designed to enhance the ability of global search. 23 widely used benchmark test functions and 30 benchmark problems from IEEE CEC2014 have selected to verify the effectiveness of ROL-GWO. The performance of ROL-GWO was compared with basic GWO, GWO variants, and other meta-heuristic algorithms. The experimental results show that

ROL-GWO is very competitive with the compared algorithms. Moreover, ROL-GWO provided better results compared to other state of art algorithms for real world application problems. In the future, it is interesting to applied ROL-GWO to solve constrained single- and multi-objective optimization problems, and other different types of optimization problems.

REFERENCES

[1] J.-T. Tsai, T.-K. Liu, and J.-H. Chou, "Hybrid Taguchi-genetic algorithm for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 4, pp. 365–377, Aug. 2004.

[2] A. A. Nagar, F. Han, Q.-H. Ling, and S. Mehta, "An improved hybrid method combining gravitational search algorithm with dynamic multi swarm particle swarm optimization," *IEEE Access*, vol. 7, pp. 50388–50399, 2019.

[3] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution—An updated survey," *Swarm Evol. Comput.*, vol. 27, pp. 1–30, Apr. 2016.

[4] W.-F. Gao, S.-Y. Liu, and L.-L. Huang, "A novel artificial bee colony algorithm based on modified search equation and orthogonal learning," *IEEE Trans. Cybern.*, vol. 43, no. 3, pp. 1011–1024, Jun. 2013.

[5] S. Arora, H. Singh, M. Sharma, S. Sharma, and P. Anand, "A new hybrid algorithm based on grey wolf optimization and crow search algorithm for unconstrained function optimization and feature selection," *IEEE Access*, vol. 7, pp. 26343–26361, 2019.

[6] U. Mlakar, I. Fister, Jr., and I. Fister, "Hybrid self-adaptive cuckoo search for global optimization," *Swarm Evol. Comput.*, vol. 29, pp. 47–72, Aug. 2016.

[7] S. Mirjalili, S. M. Mirjalili, and X.-S. Yang, "Binary bat algorithm," *Neural Comput. Appl.*, vol. 25, nos. 3–4, pp. 663–681, Sep. 2014.

[8] V. Viswanathan and I. Krishnamurthi, "Finding relevant semantic association paths using semantic ant colony optimization algorithm," *Soft Comput.*, vol. 19, no. 1, pp. 251–260, Jan. 2015.

[9] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Mixed variable structural optimization using firefly algorithm," *Comput. Struct.*, vol. 89, nos. 23–24. pp. 2325–2336, 2011.

[10] S. M. Ashrafi and A. B. Dariane, "Performance evaluation of an improved harmony search algorithm for numerical optimization: Melody search (MS)," *Eng. Appl. Artif. Intell.*, vol. 26, no. 4, pp. 1301–1321, Apr. 2013.

[11] S. C. Satapathy and A. Naik, "Modified teaching–learning-based optimization algorithm for global numerical optimization—A comparative study," *Swarm Evol. Comput.*, vol. 16, pp. 28–37, Jun. 2014.

[12] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.

[13] S. Mirjalili, "The ant lion optimizer," *Adv. Eng. Softw.*, vol. 83, pp. 80–98, May 2015.

[14] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016.

[15] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowl.-Based Syst.*, vol. 89, pp. 228–249, Nov. 2015.

[16] H. Faris, I. Aljarah, M. A. Al-Betar, and S. Mirjalili, "Grey wolf optimizer: A review of recent variants and applications," *Neural Comput. Appl.*, vol. 30, no. 2, pp. 413–435, 2018.

[17] M. H. Sulaiman, Z. Mustaffa, M. R. Mohamed, and O. Aliman, "Using the gray wolf optimizer for solving optimal reactive power dispatch problem," *Appl. Soft Comput.*, vol. 32, pp. 286–292, Jul. 2015.

[18] V. K. Kamboj, S. K. Bath, and J. S. Dhillon, "Solution of non-convex economic load dispatch problem using grey wolf optimizer," *Neural Comput. Appl.*, vol. 27, no. 5, pp. 1301–1316, Jul. 2015.

[19] M. H. Qais, H. M. Hasanien, and S. Alghuwainem, "A grey wolf optimizer for optimum parameters of multiple PI controllers of a grid-connected PMSG driven by variable speed wind turbine," *IEEE Access*, vol. 6, pp. 44120–44128, 2018.

[20] A. K. M. Khairuzzaman and S. Chaudhury, "Multilevel thresholding using grey wolf optimizer for image segmentation," *Expert Syst. Appl.*, vol. 86, pp. 64–76, Nov. 2017.

[21] E. Emary, H. M. Zawbaa, and A. E. Hassanien, "Binary grey wolf optimization approaches for feature selection," *Neurocomputing*, vol. 172, pp. 371–381, Jan. 2016.

[22] S. A. Medjahed, T. A. Saadi, A. Benyettou, and M. Ouali, "Gray wolf optimizer for hyperspectral band selection," *Appl. Soft Comput.*, vol. 40, pp. 178–186, Mar. 2016.

[23] S. Mohanty, B. Subudhi, and P. K. Ray, "A new MPPT design using grey wolf optimization technique for photovoltaic system under partial shading conditions," *IEEE Trans. Sustain. Energy*, vol. 7, no. 1, pp. 181–188, Jan. 2016.

[24] L. K. Panwar, S. Reddy, A. Verma, B. K. Panigrahi, and R. Kumar, "Binary Grey wolf optimizer for large scale unit commitment problem," *Swarm Evol. Comput.*, vol. 38, pp. 251–266, Feb. 2018.

[25] A. Saxena, B. P. Soni, R. Kumar, and V. Gupta, "Intelligent grey wolf optimizer—Development and application for strategic bidding in uniform price spot energy market," *Appl. Soft Comput.*, vol. 69, pp. 1–13, Aug. 2018.

[26] S. Zhang, Y. Zhou, Z. Li, and W. Pan, "Grey wolf optimizer for unmanned combat aerial vehicle path planning," *Adv. Eng. Softw.*, vol. 99, pp. 121–136, Sep. 2016.

[27] M. Wang, H. Chen, H. Li, Z. Cai, X. Zhao, C. Tong, J. Li, and X. Xu, "Grey wolf optimization evolving kernel extreme learning machine: Application to bankruptcy prediction," *Eng. Appl. Artif. Intell.*, vol. 63, pp. 54–68, Aug. 2017.

[28] R. Sanjay, T. Jayabarathi, T. Raghunathan, V. Ramesh, and N. Mithulananthan, "Optimal allocation of distributed generation using hybrid grey wolf optimizer," *IEEE Access*, vol. 5, pp. 14807–14818, 2017.

[29] D. Guha, P. K. Roy, and S. Banerjee, "Load frequency control of interconnected power system using grey wolf optimization," *Swarm Evol. Comput.*, vol. 27, pp. 97–115, Apr. 2016.

[30] R. E. Precup, R.-C. David, and E. M. Petriu, "Grey wolf optimizer algorithm-based tuning of fuzzy control systems with reduced parametric sensitivity," *IEEE Trans. Ind. Electron.*, vol. 64, no. 1, pp. 527–534, Jan. 2017.

[31] W. Long, J. Jiao, X. Liang, and M. Tang, "Inspired grey wolf optimizer for solving large-scale function optimization problems," *Appl. Math. Mod.*, vol. 60, pp. 112–126, Aug. 2018.

[32] S. Saremi, S. Z. Mirjalili, and S. M. Mirjalili, "Evolutionary population dynamics and grey wolf optimizer," *Neural Comput. Appl.*, vol. 26, no. 5, pp. 1257–1263, Jul. 2015.

[33] A. A. Heidari and P. A. Pahlavani, "An efficient modified grey wolf optimizer with Lévy flight for optimization tasks," *Appl. Soft Comput.*, vol. 60, pp. 115–134, Nov. 2017.

[34] L. Rodríguez, O. Castillo, P. Melin, F. Valdez, C. I. Gonzalez, G. E. Martinez, and J. Soto, "A fuzzy hierarchical operator in the grey wolf optimizer algorithm," *Appl. Soft Comput.*, vol. 57, pp. 315–328, Aug. 2017.

[35] W. Long, T. Wu, S. Cai, X. Liang, J. Jiao, and M. Xu, "A novel grey wolf optimizer algorithm with refraction learning," *IEEE Access*, vol. 7, pp. 57805–57819, 2019.

[36] V. Kumar and D. Kumar, "An astrophysics-inspired Grey wolf algorithm for numerical optimization and its application to engineering design problems," *Adv. Eng. Softw.*, vol. 112, pp. 231–254, Oct. 2017.

[37] R. A. Ibrahim, M. A. Elaziz, and S. Lu, "Chaotic opposition-based grey-wolf optimization algorithm based on differential evolution and disruption operator for global optimization," *Expert Syst. Appl.*, vol. 108, pp. 1–27, Oct. 2018.

[38] W. Long, J. Jiao, X. Liang, and M. Tang, "An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization," *Eng. Appl. Artif. Intell.*, vol. 68, pp. 63–80, Feb. 2018.

[39] S. Gupta and K. Deep, "A novel random walk grey wolf optimizer," *Swarm Evol. Comput.*, vol. 44, pp. 101–112, Feb. 2019.

[40] A. A. Alomoush, A. A. Alsewari, H. S. Alamri, K. Aloufi, and K. Z. Zamli, "Hybrid harmony search algorithm with grey wolf optimizer and modified opposition-based learning," *IEEE Access*, vol. 7, pp. 68764–68785, 2019.

[41] N. Mittal, U. Singh, and B. S. Sohi, "Modified grey wolf optimizer for global engineering optimization," *Appl. Comput. Intell. Soft Comput.*, vol. 2016, pp. 1–16, Mar. 2016, Art. no. 7950348.

[42] D. H. Wolper and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[43] H. Tizhoosh, "Opposition-based learning: A new scheme for machine intelligence," in *Proc. Int. Conf. Comput. Intell. Modeling, Control Automat.*, Nov. 2005, pp. 695–701.

[44] S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 64–79, Feb. 2008.

[45] H. Wang, Z. Wu, S. Rahnamayan, Y. Liu, and M. Ventresca, "Enhancing particle swarm optimization using generalized opposition-based learning," *Inf. Sci.*, vol. 181, no. 20, pp. 4699–4714, Oct. 2011.
[46] M. A. Elaziz, D. Oliva, and S. Xiong, "An improved opposition-based sine cosine algorithm for global optimization," *Expert Sys. Appl.*, vol. 90, pp. 484–500, Dec. 2017.
[47] J. J. Liang, B. Qu, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC2014 special session and competition on single objective real-parameter numerical optimization," Comput. Intell. Lab., Zhengzhou Univ., Zhengzhou, China, Tech. Rep. 201311, 2014.
[48] B. Jiao, Z. Lian, and X. Gu, "A dynamic inertia weight particle swarm optimization algorithm," *Chaos Solitons Fractals*, vol. 37, no. 3, pp. 698–705, Aug. 2008.
[49] G. Zhu and S. Kwong, "Gbest-guided artificial bee colony algorithm for numerical function optimization," *Appl. Math. Comput.*, vol. 217, pp. 3166–3173, Dec. 2010.
[50] P. K. Roy, C. Paul, and S. Sultana, "Oppositional teaching learning based optimization approach for combined heat and power dispatch," *Int. J. Elect. Power Energy Syst.*, vol. 57, pp. 392–403, May 2014.
[51] J. J. Liang, A. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 281–295, Jul. 2006.
[52] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 55–66, Feb. 2011.
[53] R. Akbari, R. Hedayatzadeh, K. Ziarati, and B. Hassanizadeh, "A multi-objective artificial bee colony algorithm," *Swarm Evol. Comput.*, vol. 2, pp. 39–52, Feb. 2012.
[54] K. Deb, "An efficient constraint handling method for genetic algorithms," *Comput. Methods Appl. Mech. Eng.*, vol. 186, nos. 2–4, pp. 311–338, Jun. 2000.

WEN LONG received the M.S. degree in system science from Guangxi Normal University, Guilin, China, in 2008, and the Ph.D. degree in control science and engineering from Central South University, Changsha, China, in 2011.

He is currently a Professor with the Key Laboratory of Economics System Simulation of Guizhou Province, Guizhou University of Finance and Economics. His current research interests include evolutionary computation, nonlinear optimization, constrained single- and multi-optimizations, machine learning, data mining, and their practical applications.

JIANJUN JIAO received the M.S. degree in fundamental mathematics from Guangxi Normal University, Guilin, China, in 2005, and the Ph.D. degree in applied mathematics from the Dalian University of Technology, Dalian, China, in 2008.

He is currently a Professor with the School of Mathematics and Statistics, Guizhou University of Finance and Economics. His current research interests include computation intelligence, biomathematics, data mining, nonlinear modeling, and their practical applications.

XIMING LIANG received the M.S. degree in applied mathematics from the Yunnan University, Kunming, China, in 1992, and the Ph.D. degree in computational mathematics from Xi'an Jiaotong University, Xi'an, China, in 1998.

He is currently a Professor with the School of Science, Beijing University of Civil Engineering and Architecture. His current research interests include constrained optimization, evolutionary computation, large-scale numerical optimization, and their practical applications.

SHAOHONG CAI received the M.S. degree in theoretical physics from Guizhou University, Guiyang, China, in 1988, and the Ph.D. degree in industry economics from the Wuhan University of Technology, Wuhan, China, in 2009.

He is currently a Professor with the Key Laboratory of Economics System Simulation of Guizhou Province, Guizhou University of Finance and Economics. His current research interests include computation intelligence, machine learning, big-data mining, and their practical applications.

MING XU received the M.S. degree in applied mathematics from Guangxi Normal University, Guilin, China, in 2009, and the Ph.D. degree in system analysis and integration from Yunnan University, Kunming, China, in 2016.

He is currently an Associate Professor with the School of Mathematics and Statistics, Guizhou University of Finance and Economics. His current research interests include intelligence optimization, nonlinear science, machine learning, data mining, neural network, and their practical applications.