Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 389630, 20 pages
http://dx.doi.org/10.1155/2015/389630

Research Article

A Hybrid Global Optimization Algorithm Based on Wind Driven Optimization and Differential Evolution

Zongfan Bao,[1] Yongquan Zhou,[1,2] Liangliang Li,[1] and Mingzhi Ma[1]

[1] *College of Information Science and Engineering, Guangxi University for Nationalities, Nanning, Guangxi 530006, China*
[2] *Key Laboratory of Guangxi High School Complex System and Computational Intelligence, Nanning 530006, China*

Correspondence should be addressed to Yongquan Zhou; yongquanzhou@126.com

Received 7 June 2015; Accepted 22 July 2015

Academic Editor: Alessandro Salvini

This paper presents a new hybrid global optimization algorithm, which is based on the wind driven optimization (WDO) and differential evolution (DE), named WDO-DE algorithm. The WDO-DE algorithm is based on a double population evolution strategy, the individuals in a population evolved by wind driven optimization algorithm, and a population of individuals evolved from difference operation. The populations of individuals both in WDO and DE employ an information sharing mechanism to implement coevolution. This paper chose fifteen benchmark functions to have a test. The experimental results show that the proposed algorithm can be feasible in both low-dimensional and high-dimensional cases. Compared to GA-PSO, WDO, DE, PSO, and BA algorithm, the convergence speed and precision of WDO-DE are higher. This hybridization showed a better optimization performance and robustness and significantly improves the original WDO algorithm.

1. Introduction

Nature always makes people produce a lot of inspiration. In the past, many natural heuristic algorithms have been proposed for solving real-world and large scale problems which are very difficult to solve by traditional methods, for example, genetic algorithm (GA) [1], particle swarm optimization (PSO) [2, 3], ant colony optimization (ACO) [4], differential evolution (DE) [5], firefly algorithm (FA) [6], bat algorithm (BA) [7], cuckoo search (CS) [8], flower pollination algorithm (FPA) [9], wind driven optimization (WDO) [10].

Because some heuristic algorithms are not very satisfactory in many respects, in the recent years, according to the characteristics of some algorithms, the heuristic algorithm combined with each other is an increasingly popular strategy to improve algorithms. These hybridizations have been shown to be effective global optimization algorithms and have been applied to solve application problems. For example, a hybrid of genetic algorithm (GA) and particle swarm optimization (PSO) is applied to recurrent neural/fuzzy network design [11]. A hybrid metaheuristic differential evolution (DE) and cuckoo search (CS) algorithm is implemented to solve the UCAV path planning problem [12]. A hybrid particle swarm optimization (PSO) and ant colony optimization (ACO) algorithm is implemented to solve hierarchical classification [13].

The wind driven optimization (WDO) is a novel nature-inspired technique. It is a population based iterative heuristic process. The WDO was proposed by Bayraktar et al. in 2010 [10]. The inspiration of the WDO derives from the atmosphere, where wind blows in an attempt to balance the atmospheric pressure. Due to the fact that the WDO has few parameters in need of control and it is very easy to be carried out, it has been paid more attention by the academic community. But, in the early stage of solving optimization problems, the convergence speed of the WDO is quicker than others; when all the individuals are close to the optimal one in the late stage of solving optimization problems, it will lead to the loss of population diversity, and it is easy to fall into local optimum in finding better solutions.

In 1997, Storn and Price have introduced the DE algorithm [5]. In the next time, the DE algorithm has been applied to solve optimization problems in diverse fields. After the population initialization, the population through mutation, crossover, and selection operators generates new population,

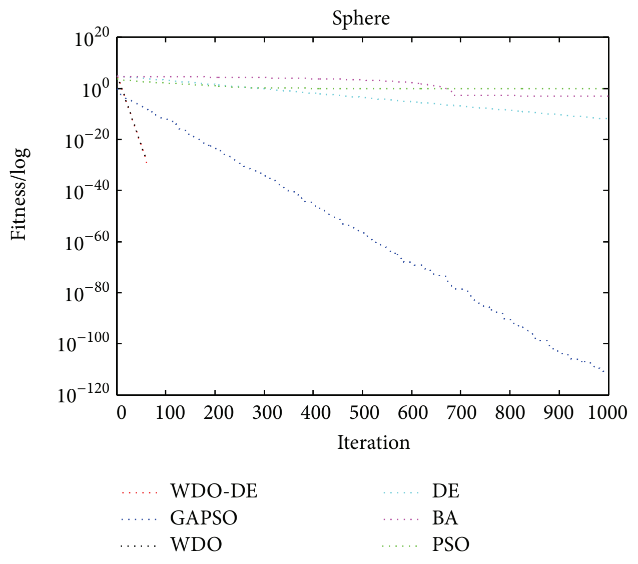

Figure 1: Comparison of performance of algorithms for minimization of f_1 ($D = 30$).

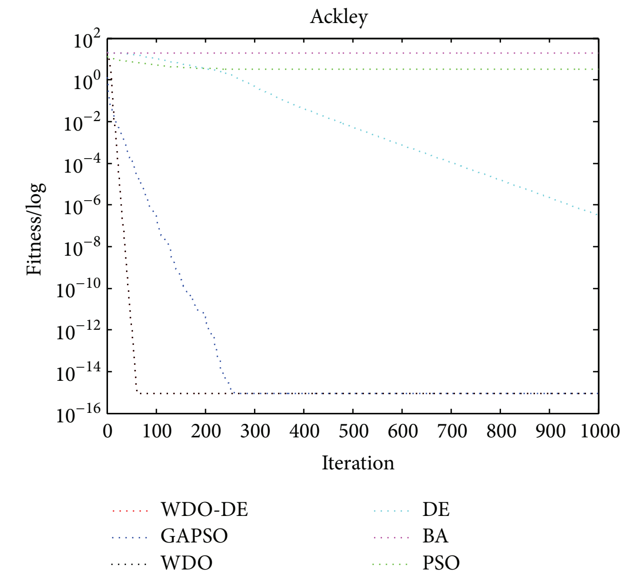

Figure 2: Comparison of performance of algorithms for minimization of f_2 ($D = 30$).

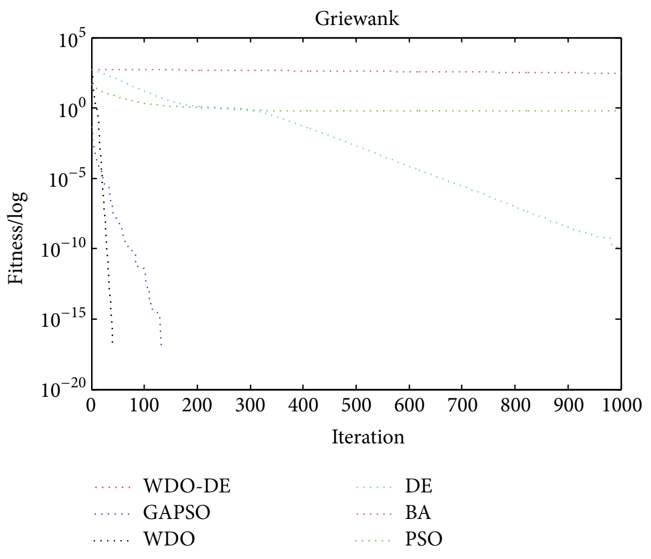

Figure 3: Comparison of performance of algorithms for minimization of f_3 ($D = 30$).

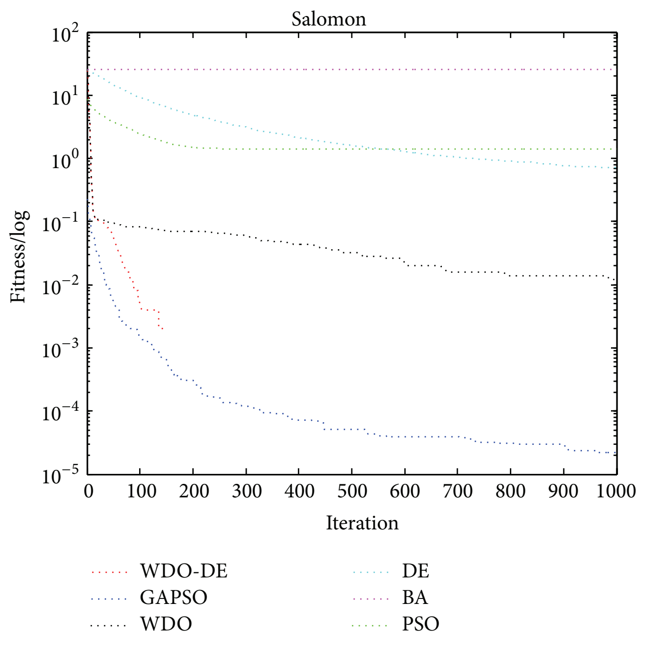

Figure 4: Comparison of performance of algorithms for minimization of f_4 ($D = 30$).

it is able to maintain diversity of population, which can achieve the search of global optimal solution after several iterations. But it also has flaws. Its disadvantages contain the slow pace of convergence, and it is easy to fall into local optimum.

WDO algorithm can easily suffer from the premature convergence when solving global optimization problems. It is an important method for relieving the premature convergence to control the population diversity. In order to overcome the deficiencies of a single algorithm in solving a global optimization problem, in this paper, we propose a new hybrid global optimization algorithm based on the wind driven optimization and differential evolution. This evolution strategy allows WDO and DE algorithms to give full play to their respective advantages. DE algorithm enables keeping the diversity of the population. This can be a good remedy defect of WDO algorithm, so it can avoid falling into a local optimum due to the loss of population diversity. And WDO algorithm converges faster; it can be good to make up for the shortcomings of DE algorithm in convergence speed, utilizing the individual of DE algorithm to guide the evolution of the individual of WDO algorithm, which reduces the risk of falling into a local optimal solution. It is not only to ensure

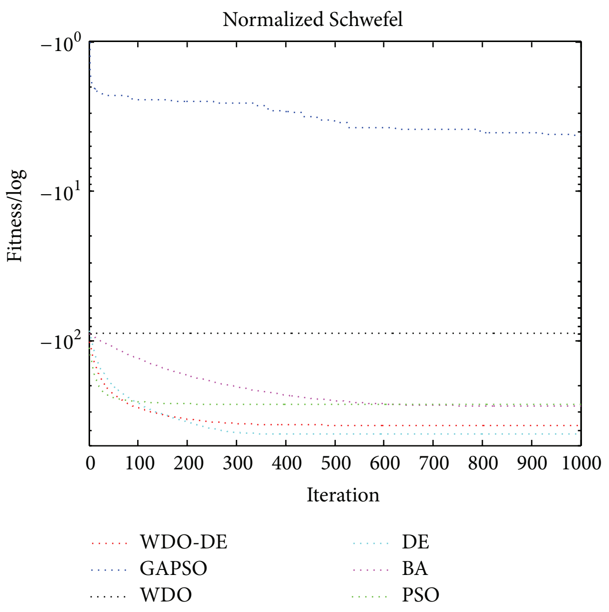

Figure 5: Comparison of performance of algorithms for minimization of f_5 ($D = 30$).

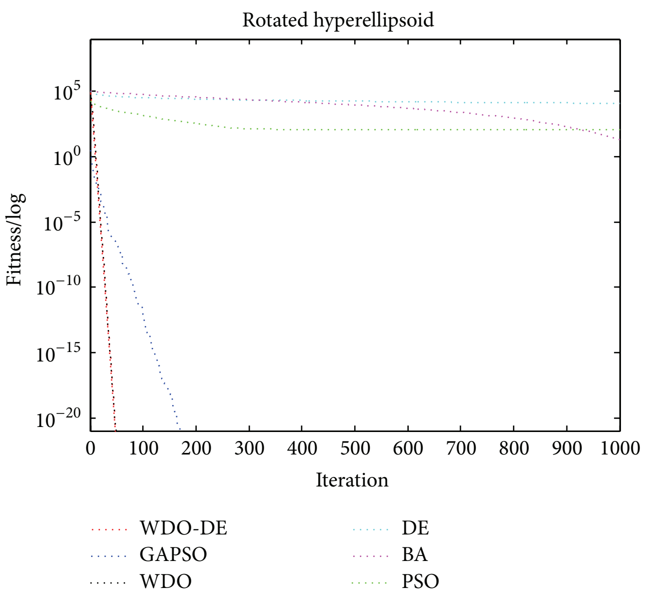

Figure 7: Comparison of performance of algorithms for minimization of f_7 ($D = 30$).

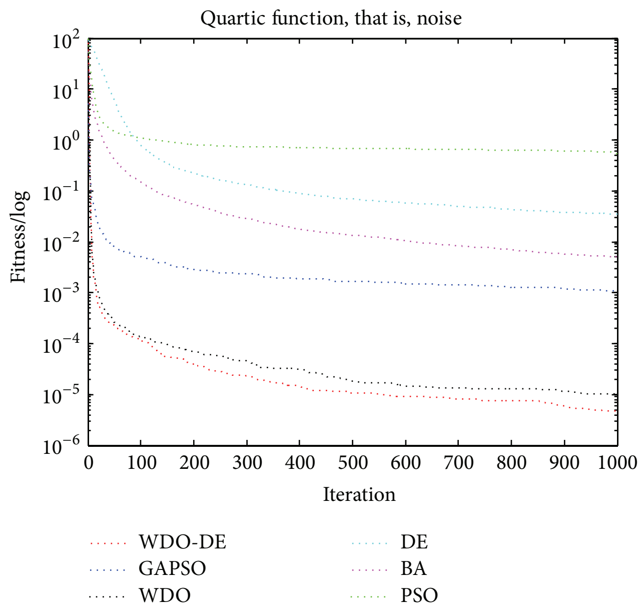

Figure 6: Comparison of performance of algorithms for minimization of f_6 ($D = 30$).

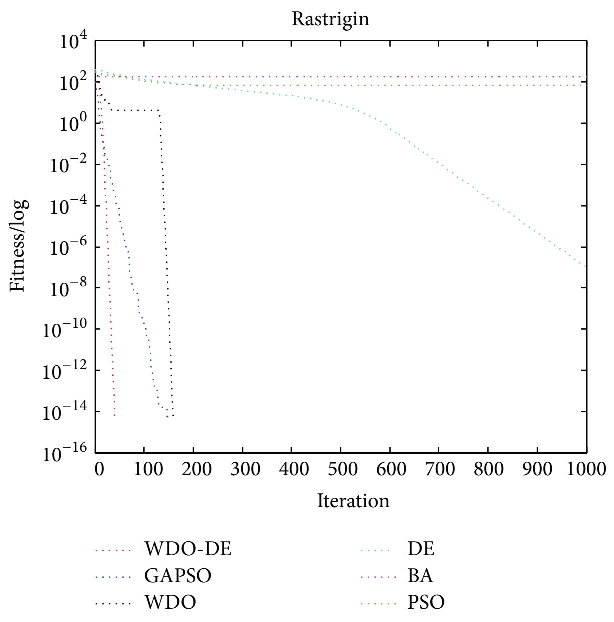

Figure 8: Comparison of performance of algorithms for minimization of f_8 ($D = 30$).

the accuracy of the algorithm, but also to guarantee the speed of solving problems. Finally, the fifteen benchmark functions are tested; the experimental results show that the proposed algorithm can be feasible in both low-dimensional and high-dimensional cases. Compared to GA-PSO, WDO, DE, PSO, and BA algorithm, the convergence speed and precision of WDO-DE are higher. This hybridization showed a better optimization performance and robustness and significantly improves the original WDO algorithm.

2. A Brief Introduction on WDO and DE Algorithm

2.1. Wind Driven Optimization. The inspiration of the proposed WDO derives from the atmosphere. In the atmosphere, wind blows in an attempt to balance the imbalance of pressure. It flows from high pressure areas to low pressure areas at a velocity. Depending on the above analysis, some theoretical assumptions are formulated in derivation of the WDO algorithm. The starting point of WDO algorithm is

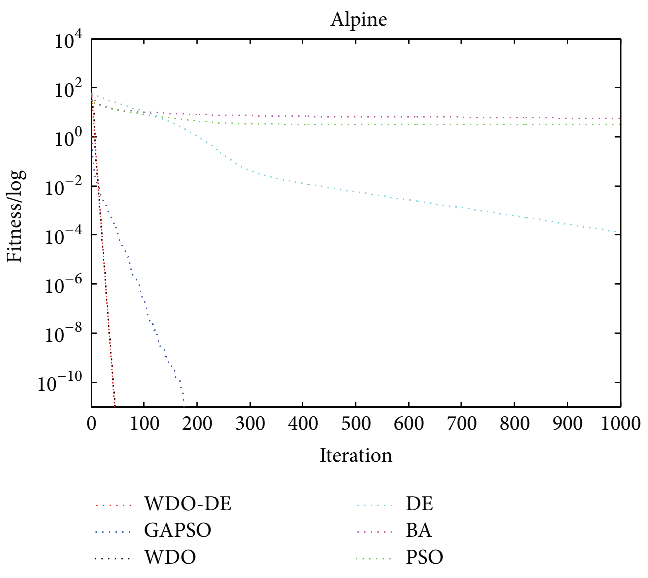

FIGURE 9: Comparison of performance of algorithms for minimization of f_9 ($D = 30$).

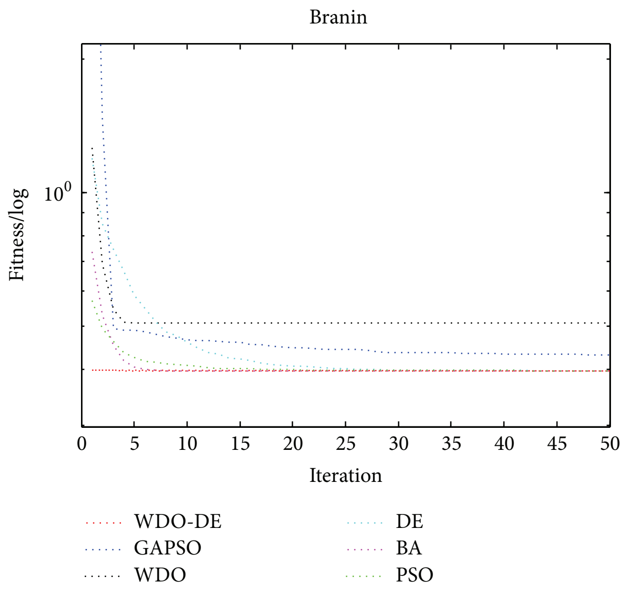

FIGURE 10: Comparison of performance of algorithms for minimization of f_{10} ($D = 2$).

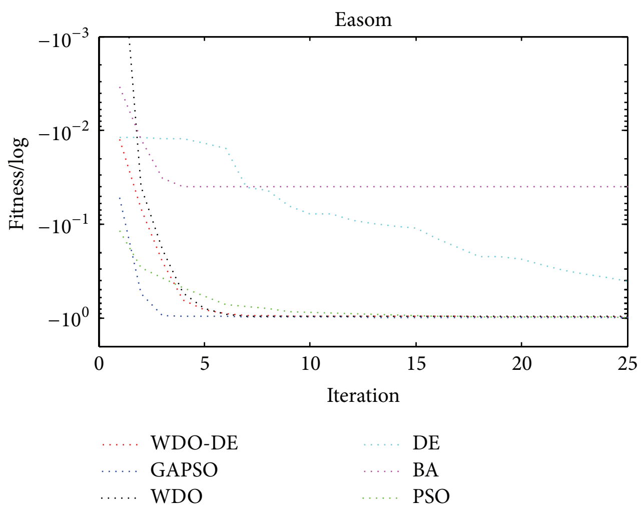

FIGURE 11: Comparison of performance of algorithms for minimization of f_{11} ($D = 2$).

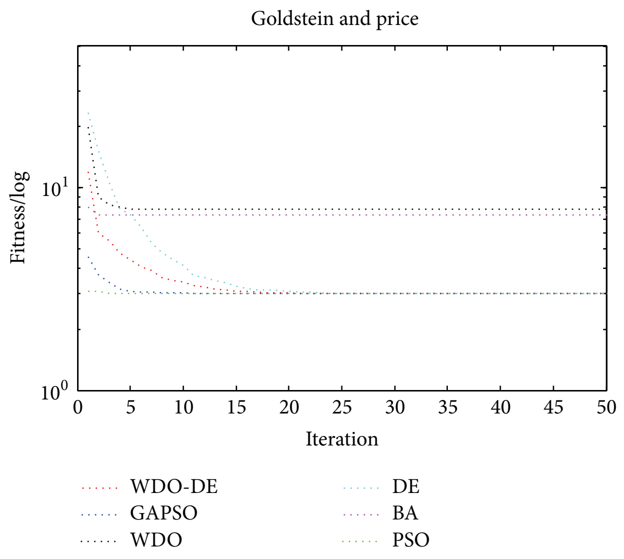

FIGURE 12: Comparison of performance of algorithms for minimization of f_{12} ($D = 2$).

Newton's second law of motion, which is used to provide accurate results for the analysis of atmospheric motion in the Lagrangian description [14, 15]

$$\rho\vec{\alpha} = \sum \vec{F}_i, \tag{1}$$

where $\vec{\alpha}$ is the acceleration, ρ is the air density for an infinitesimal air parcel, and $\vec{F}_i$ are all the forces acting on the air parcel. In order to let air pressure establish the equation relationship with the air parcel's density and temperature, the ideal gas law is given by

$$P = \rho RT, \tag{2}$$

where P is the pressure, R is the universal gas constant, and T is the temperature.

The cause of the air movement is due to the combination of many forces, mainly including gravitational force ($\vec{F}_G$), pressure gradient force ($\vec{F}_{\mathrm{PG}}$), Coriolis force ($\vec{F}_C$), and friction force ($\vec{F}_F$). The physical equations of the above mentioned forces are as follows:

$$\begin{aligned} \vec{F}_G &= \rho\delta V\vec{g}, \\ \vec{F}_{\mathrm{PG}} &= -\nabla P\delta V, \\ \vec{F}_C &= -2\Omega\times\vec{u}, \\ \vec{F}_F &= -\rho\alpha\vec{u}, \end{aligned} \tag{3}$$

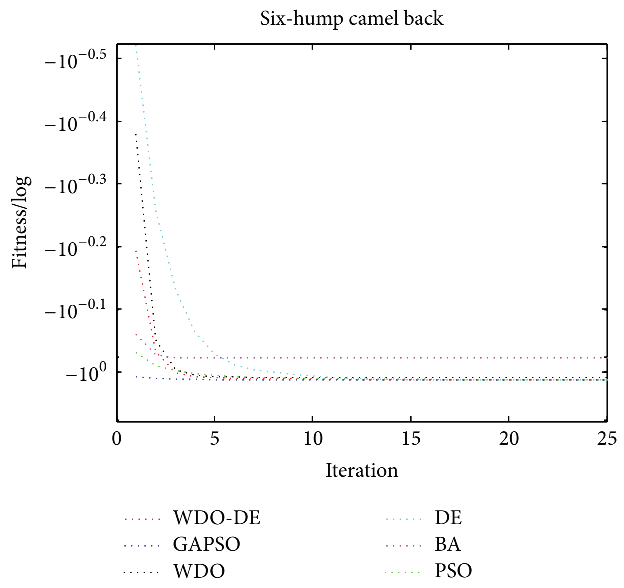

FIGURE 13: Comparison of performance of algorithms for minimization of f_{13} ($D = 2$).

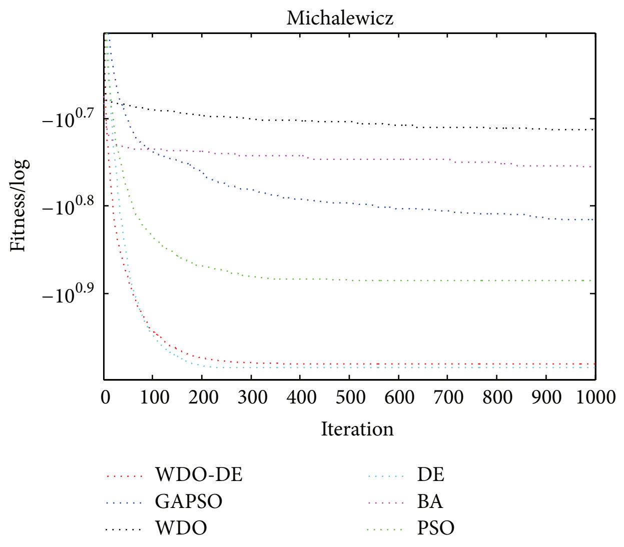

FIGURE 15: Comparison of performance of algorithms for minimization of f_{15} ($D = 10$).

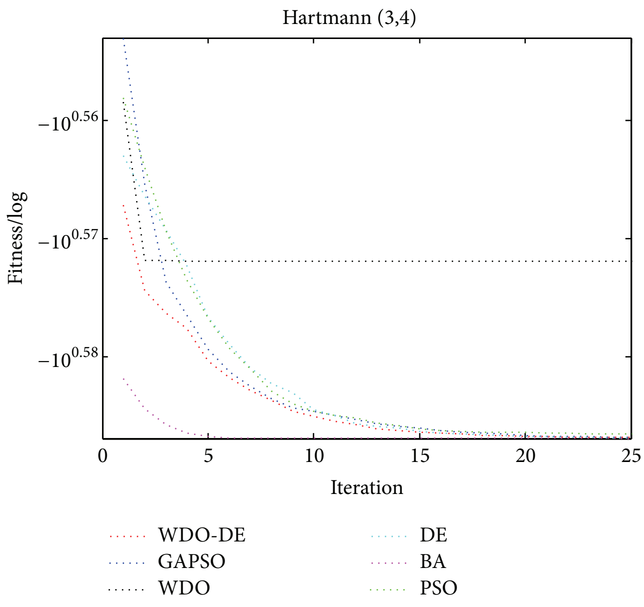

FIGURE 14: Comparison of performance of algorithms for minimization of f_{14} ($D = 3$).

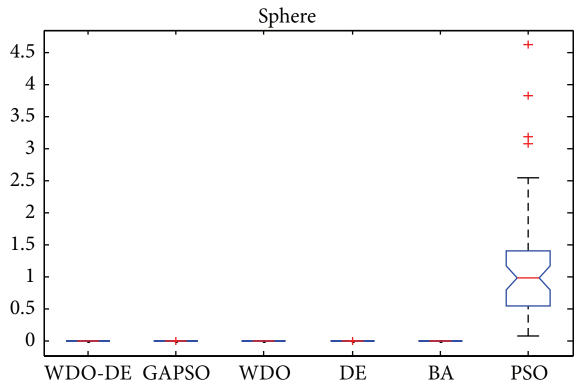

FIGURE 16: ANOVA tests of the global minimum values for f_1 ($D = 30$).

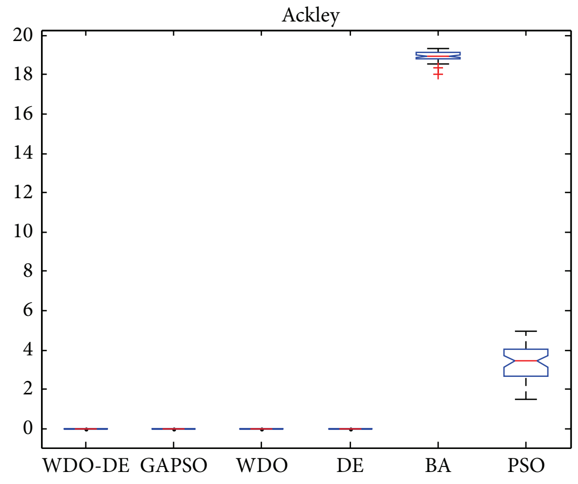

FIGURE 17: ANOVA tests of the global minimum values for f_2 ($D = 30$).

where δV is finite volume of the air, $\vec{g}$ represents the gravitational acceleration, ∇P represents the pressure gradient, Ω is rotation of the Earth, $\vec{u}$ represents the velocity vector of the wind, and α is the friction coefficient.

The forces mentioned above can be added to (1). The equation can be described as in

$$\rho \frac{\Delta \vec{u}}{\Delta t} = (\rho \delta V \vec{g}) + (-\nabla P \delta V) + (-2\Omega \times \vec{u}) + (-\rho \alpha \vec{u}), \qquad (4)$$

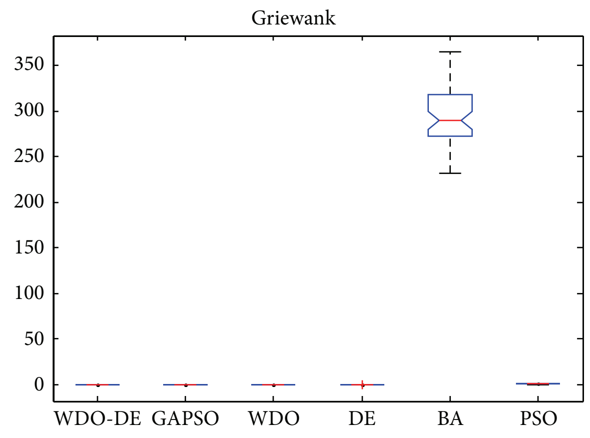

Figure 18: ANOVA tests of the global minimum values for f_3 ($D = 30$).

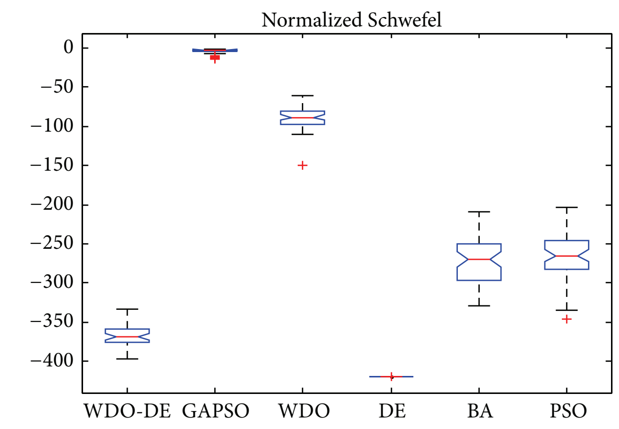

Figure 20: ANOVA tests of the global minimum values for f_5 ($D = 30$).

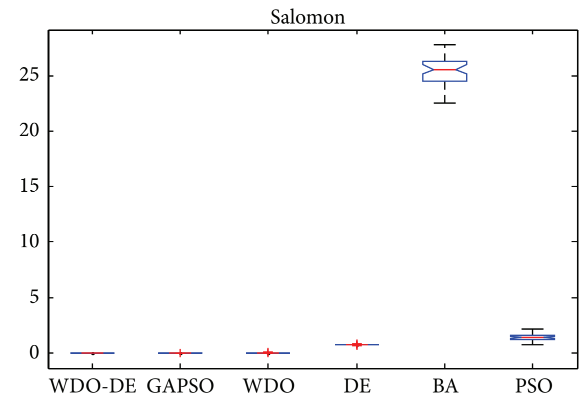

Figure 19: ANOVA tests of the global minimum values for f_4 ($D = 30$).

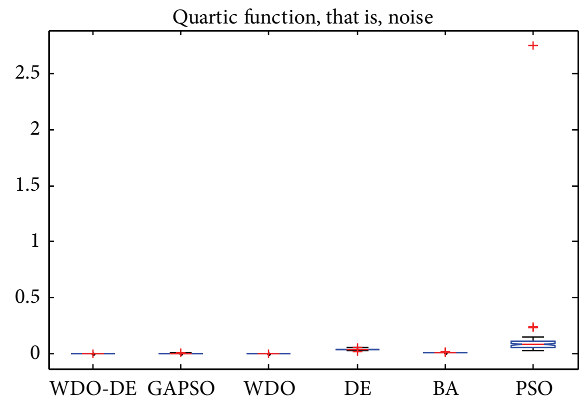

Figure 21: ANOVA tests of the global minimum values for f_6 ($D = 30$).

where the acceleration $\vec{\alpha}$ in (1) is rewritten as $\vec{\alpha} = \Delta\vec{u}/\Delta t$; for simplicity, set $\Delta t = 1$; for an infinitesimal air parcel, set $\delta V = 1$, which simplifies (4) to

$$\rho\Delta\vec{u} = (\rho\vec{g}) + (-\nabla P) + (-2\Omega\times\vec{u}) + (-\rho\alpha\vec{u}). \qquad (5)$$

On the basis of (2), the density ρ can be written in terms of the pressure; thus (5) can be rewritten as

$$\Delta\vec{u} = \vec{g} + \left(-\nabla P\frac{RT}{P_{\text{cur}}}\right) + \left(\frac{-2\Omega\times\vec{u}RT}{P_{\text{cur}}}\right) + (-\alpha\vec{u}), \qquad (6)$$

where P_{cur} is the pressure of current location. It is assumed in the WDO algorithm that velocity and position of the air parcel are changing at each iteration. Thus, $\Delta\vec{u}$ can be written as $\Delta\vec{u} = \vec{u}_{\text{new}} - \vec{u}_{\text{cur}}$, where $\vec{u}_{\text{new}}$ represents the velocity in next iteration and $\vec{u}_{\text{cur}}$ is the velocity at the current iteration. $\vec{g}$ and ∇P are vectors, they can be broken down in direction and magnitude as $\vec{g} = |g|(0 - x_{\text{cur}})$, $-\nabla P = |P_{\text{opt}} - P_{\text{cur}}|(x_{\text{opt}} - x_{\text{cur}})$, P_{opt} is the optimum pressure point that has been found so far, x_{opt} is the optimum location that has been found so far, and x_{cur} is the current location; updating (6) with the new equations, (6) can be rewritten as

$$\vec{u}_{\text{new}} = (1-\alpha)\vec{u}_{\text{cur}} - gx_{\text{cur}} + \left(\frac{RT}{P_{\text{cur}}}\left|P_{\text{opt}} - P_{\text{cur}}\right|\left(x_{\text{opt}} - x_{\text{cur}}\right)\right) + \left(\frac{-2\Omega\times\vec{u}RT}{P_{\text{cur}}}\right). \qquad (7)$$

Finally, there are three additional substitutions needed. Firstly, the influence of the Coriolis force $(\Omega\times\vec{u})$ is replaced by the velocity influence from another dimension $\vec{u}_{\text{cur}}^{\text{other dim}}$. Secondly, all the coefficients are combined together; that is, $c = -2RT$. Thirdly, in some cases where the pressure is extremely large, the updated velocities are too large to become meaningless and the efficiency of the WDO algorithm will be reduced. So the actual pressure value is replaced by rank among all air parcels based on their pressure values, the resulting equation of updating the velocity can be described

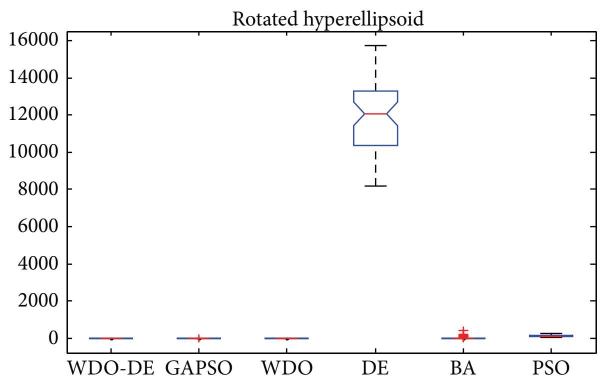

FIGURE 22: ANOVA tests of the global minimum values for f_7 ($D = 30$).

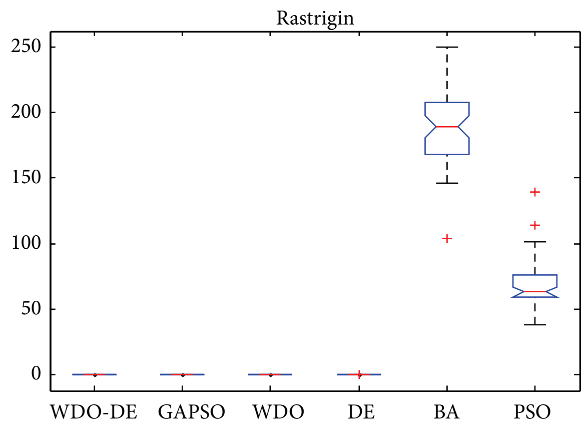

FIGURE 23: ANOVA tests of the global minimum values for f_8 ($D = 30$).

as in (8), and the equation of updating the location can be described as in (9):

$$\vec{u}_{\text{new}} = (1-\alpha)\,\vec{u}_{\text{cur}} - g x_{\text{cur}} + \left(RT\left|1-\frac{1}{i}\right|\left(x_{\text{opt}} - x_{\text{cur}}\right)\right) + \left(\frac{c\vec{u}_{\text{cur}}^{\text{other dim}}}{i}\right), \quad (8)$$

$$\vec{x}_{\text{new}} = \vec{x}_{\text{cur}} + \left(\vec{u}_{\text{new}} \times \Delta t\right), \quad (9)$$

where i is the ranking among all air parcels and $\vec{x}_{\text{new}}$ represents the new location for the next iteration.

WDO is similar to other nature-inspired optimization algorithms, but compared to other optimization algorithms, the code of WDO is more simple and easy to implement; it has less few control variables that need adjustment.

2.2. Differential Evolution. Differential evolution is introduced by Storn and Price in 1997 [5]. DE is an effective and simple global optimization algorithm.

First of all, a population is generated randomly, it may be represented as $X_{i,G}$ ($i = 1, 2, \ldots, \text{NP}$) $= \{x_{i,G}^1, x_{i,G}^2, \ldots, x_{i,G}^D\}$,

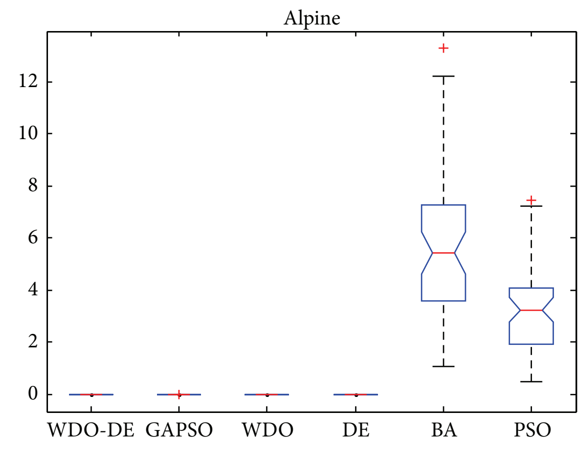

FIGURE 24: ANOVA tests of the global minimum values for f_9 ($D = 30$).

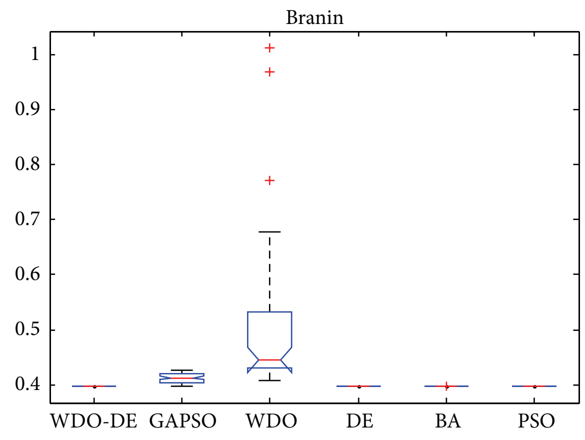

FIGURE 25: ANOVA tests of the global minimum values for f_{10} ($D = 30$).

NP is the number of population, D is the number of dimensions, and G denotes the generation of the population. There are three operators—mutation, crossover, and selection. Then the original population will through three operators generate a new population. The main progress of DE in detail is as follows.

(1) Mutation. After initialization, mutation operation is used to generate the mutant vectors $V_{i,G} = \{v_{i,G}^1, v_{i,G}^2, \ldots, v_{i,G}^D\}$. Mutation operation usually has the following five most frequently implemented strategies.

DE/rand/1:

$$V_{i,G} = X_{r_1^i,G} + F \cdot \left(X_{r_2^i,G} - X_{r_3^i,G}\right). \quad (10)$$

DE/best/1:

$$V_{i,G} = X_{\text{best},G} + F \cdot \left(X_{r_1^i,G} - X_{r_2^i,G}\right). \quad (11)$$

DE/current-to-best/1:

$$V_{i,G} = X_{i,G} + F \cdot \left(X_{\text{best},G} - X_{i,G}\right) + F \cdot \left(X_{r_1^i,G} - X_{r_2^i,G}\right). \quad (12)$$

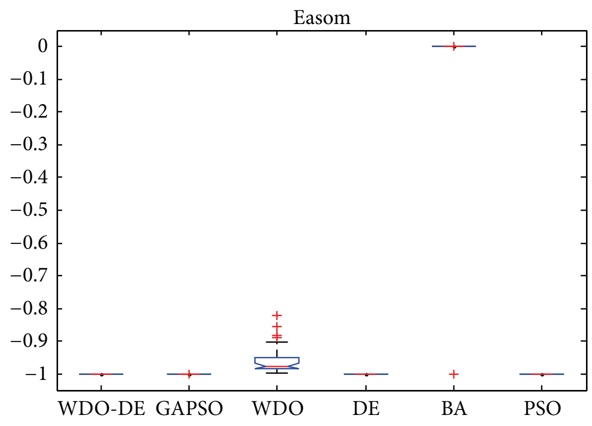

FIGURE 26: ANOVA tests of the global minimum values for f_{11} ($D = 2$).

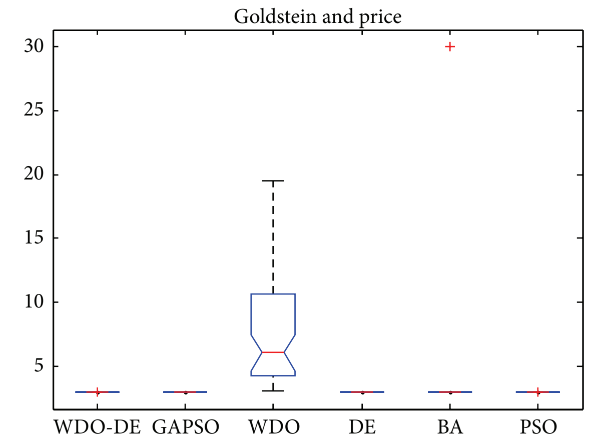

FIGURE 27: ANOVA tests of the global minimum values for f_{12} ($D = 2$).

DE/best/2:

$$V_{i,G} = X_{\text{best},G} + F \cdot \left(X_{r_1^i,G} - X_{r_2^i,G}\right) + F \cdot \left(X_{r_3^i,G} - X_{r_4^i,G}\right). \tag{13}$$

DE/rand/2:

$$V_{i,G} = X_{r_1^i,G} + F \cdot \left(X_{r_2^i,G} - X_{r_3^i,G}\right) + F \cdot \left(X_{r_4^i,G} - X_{r_5^i,G}\right). \tag{14}$$

The subscripts $r_1^i, r_2^i, r_3^i, r_4^i, r_5^i$ are mutually exclusive integers within the range $[1, \text{NP}]$ and F is the scale factor of difference vector.

(2) Crossover. After the mutation operation is completed, DE will utilize crossover operation to generate a trial vector $U_{i,G} = \{u_{i,G}^1, u_{i,G}^2, \ldots, u_{i,G}^D\}$:

$$u_{i,G}^j = \begin{cases} v_{i,G}^j & \text{if } \text{rand}_j\,[0,1) \le CR \text{ or } \left(j = j_{\text{rand}}\right), \\ x_{i,G}^j & \text{others}, \end{cases} \tag{15}$$

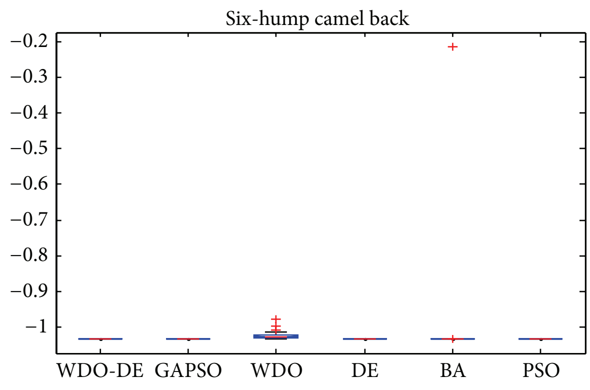

FIGURE 28: ANOVA tests of the globalminimum values for f_{13} ($D = 2$).

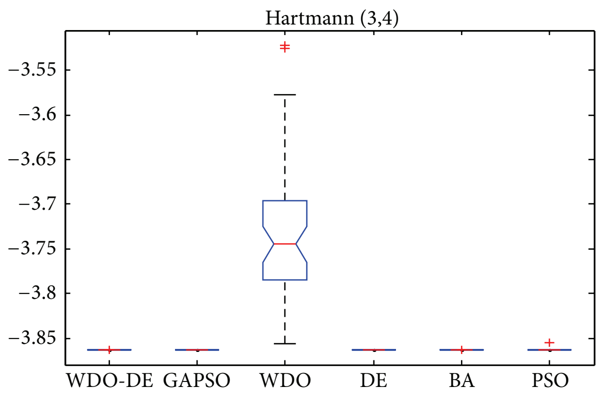

FIGURE 29: ANOVA tests of the global minimum values for f_{14} ($D = 2$).

where $CR \in [0, 1]$; it is a crossover constant. j_{rand} is a random integer within the range $[1, D]$.

(3) Selection. After completing the first two operators, according the fitness value of trial vectors, DE utilizes selection operation to select the best one for the next generation:

$$X_{i,G+1} = \begin{cases} U_{i,G} & \text{if } f\left(U_{i,G}\right) \le f\left(X_{i,G}\right), \\ X_{i,G} & \text{otherwise.} \end{cases} \tag{16}$$

After gradual iteration, DE can achieve the search of global optimal solution.

3. WDO-DE Algorithm

This section focuses on the rationale of the WDO-DE algorithm. WDO and DE are based on the population of the global search techniques. And the WDO-DE is based on a double population evolution strategy [16]. The individuals both in WDO and DE employ an information sharing mechanism to implement coevolution. The strategy makes WDO-DE enjoy the advantages of two algorithms. It can maintain diversity of the populations, and the WDO-DE algorithm has the capability to jump out of the local optimal

```
Step 1. Initialize parameters.
N (Population size); G (Max number of generations);
Parameters of WDO: RT (RT coefficient); $\alpha$ (The friction coefficient); max V (Maximum allowed speed); g (Gravitational
constant); c (Constant in the update equation).
Parameters of DE: F (Mutation scale factor); Pc (Crossover probability).
Step 2. Initialization populations.
    Step 2.1. Generate one initial population with $N/2$ air particles, each air particle assign random location $P_{i,G}^{\text{wdo}}$ and velocity
              $V_{i,G}^{\text{wdo}}$, evaluation the population and identify the best solution of WDO algorithm $P_{\text{best},G}^{\text{wdo}}$;
    Step 2.2. Generate one initial population $P_{i,G}^{\text{de}}$ with $N/2$ individuals, evaluation the population and identify the best
              solution $P_{\text{best},G}^{\text{de}}$.
Step 3. Identify the best solution $P_{\text{best}}$ among all particles in WDO and DE.
Step 4. While stopping criterion is not satisfied
    Step 4.1. Running process of the WDO algorithm
              for $i = 1$ to the $N/2$ do
                  $P_{\text{best},G}^{\text{wdo}} = P_{\text{best}}$
                  Generate the trial velocity according to (8)
                  Generate the trial location $U_{i,G}$ by (9)
                  Evaluate the trial location $U_{i,G}$
                          If $f(U_{i,G}) \le f(P_{i,G}^{\text{wdo}})$
                              $P_{i,G+1}^{\text{wdo}} = U_{i,G}$,    $f(P_{i,G+1}^{\text{wdo}}) = f(U_{i,G})$
                              If $f(U_{i,G}) \le f(P_{\text{best},G}^{\text{wdo}})$
                                  $P_{\text{best},G}^{\text{wdo}} = U_{i,G}$,    $f(P_{\text{best},G}^{\text{wdo}}) = f(U_{i,G})$
                              End if
                          End if
              End for
    Step 4.2. Running process of the DE algorithm
              for $i = 1$ to the $N/2$ do
                  $P_{\text{best},G}^{\text{de}} = P_{\text{best}}$
                  Generate $V_{i,G}$ using (11)
                  Generate the trial vector $U_{i,G}$ by (15)
                  Evaluate the trial vector $U_{i,G}$
                          If $f(U_{i,G}) \le f(P_{i,G}^{\text{de}})$
                              $P_{i,G+1}^{\text{de}} = U_{i,G}$,    $f(P_{i,G+1}^{\text{de}}) = f(U_{i,G})$
                              If $f(U_{i,G}) \le f(P_{\text{best},G}^{\text{de}})$
                                  $P_{\text{best},G}^{\text{de}} = U_{i,G}$,    $f(P_{\text{best},G}^{\text{de}}) = f(U_{i,G})$
                              End if
                          End if
              End for
    Step 4.3. Identify the best solution $P_{\text{best}}$
        If $f(P_{\text{best},G}^{\text{wdo}}) \le f(P_{\text{best},G}^{\text{de}})$
                  $P_{\text{best}} = P_{\text{best},G}^{\text{wdo}}$
        Else
                  $P_{\text{best}} = P_{\text{best},G}^{\text{de}}$
        End if
    Step 4.4. Increment the generation count $G = G + 1$
Step 5. End while
```

Algorithm 1: WDO-DE.

solution. Based on the above description, the main procedure of WDO-DE is as shown in Algorithm 1.

4. Experimental Results

4.1. Experimental Setup. All algorithms are implemented in MATLAB R2012a, and experiments are performed on a Pentium 3.00 GHz Processor with 4.0 GB of memory, Windows 7 operating system.

4.2. Benchmark Test Functions. To test the performance of WDO-DE algorithm, we use 15 benchmark functions [17, 18] which have been widely used in the test. Among these benchmarks, part I contains the nine high-dimensional functions and part II contains six low-dimensional functions. Table 1 has shown the benchmark functions.

4.3. Parameters Setting. In this section, the parameters setting are presented. Bayraktar et al. did a lot of research

TABLE 1: Description of the benchmark function used in our experiment.

Category	Test functions	Name	Range of search	Minimum value
I	$f_1(x) = \sum_{i=1}^{n} x_i^2$	Sphere	[−100, 100]	0
	$f_2(x) = -20\exp\left[-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right] - \exp\left[\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right] + 20 + e$	Ackley	[−32, 32]	0
	$f_3(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	Griewank	[−600, 600]	0
	$f_4(x) = -\cos\left(2\pi\sqrt{\sum_{i=1}^{n} x_i^2}\right) + 0.1\sqrt{\sum_{i=1}^{n} x_i^2} + 1$	Salomon	[−100, 100]	0
	$f_5(x) = -\frac{1}{n}\sum_{i=1}^{n} x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)$	Normalized Schwefel	[−512, 512]	−418.9829
	$f_6(x) = \sum_{i=1}^{n} i x_i^4 + \text{rand}[0, 1)$	Quartic function, that is, noise	[−1.28, 1.28]	0
	$f_7(x) = \sum_{i=1}^{n}\sum_{j=1}^{i} x_j^2$	Rotated hyperellipsoid	[−100, 100]	0
	$f_8(x) = 10n + \sum_{i=1}^{n}\left[x_i^2 - 10\cos(2\pi x_i)\right]$	Rastrigin	[−5.12, 5.12]	0
	$f_9(x) = \sum_{i=1}^{n}\sqrt{x_i \sin(x_i) + 0.1x_i}$	Alpine	[−10, 10]	0
II	$f_{10}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$	Branin	[−5, 15]	0.397887
	$f_{11}(x) = -\cos(x_1)\cos(x_2)\exp\left[-(x_1-\pi)^2 + (x_2-\pi)^2\right]$	Easom	[−100, 100]	−1
	$f_{12}(x) = \left[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \times \left[30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$	Goldstein and Price	[−2, 2]	3
	$f_{13}(x) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2$	Six Hump Camel back	[−5, 5]	−1.0316
	$f_{14}(x) = -\sum_{i=1}^{4}\alpha_i \exp\left(-\sum_{j=1}^{3}A_{ij}\left(x_i - P_{ij}\right)^2\right), \quad \alpha = (1.0, 1.2, 3.0, 3.2)^T,$ $A = [3.0\ 10\ 30; 0.1\ 10\ 35; 3.0\ 10\ 30; 0.1\ 10\ 35],$ $P = [0.3689\ 0.117\ 0.2673; 0.4699\ 0.4387\ 0.747; 0.1091\ 0.8732\ 0.5547; 0.0381\ 0.5743\ 0.8828]$	Hartmann (3,4)	[0, 1]	−3.86278
	$f_{15}(x) = -\sum_{i=1}^{n}\sin(x_i)\sin^{2m}\left(\frac{i x_i^2}{\pi}\right)$	Michalewicz	[0, pi]	−9.66015

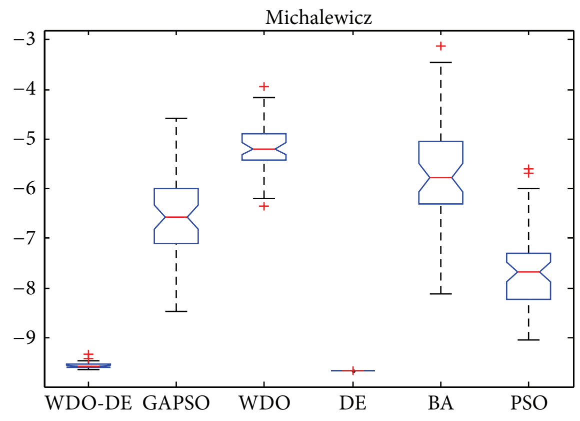

Figure 30: ANOVA tests of the global minimum values for f_{15} ($D = 3$).

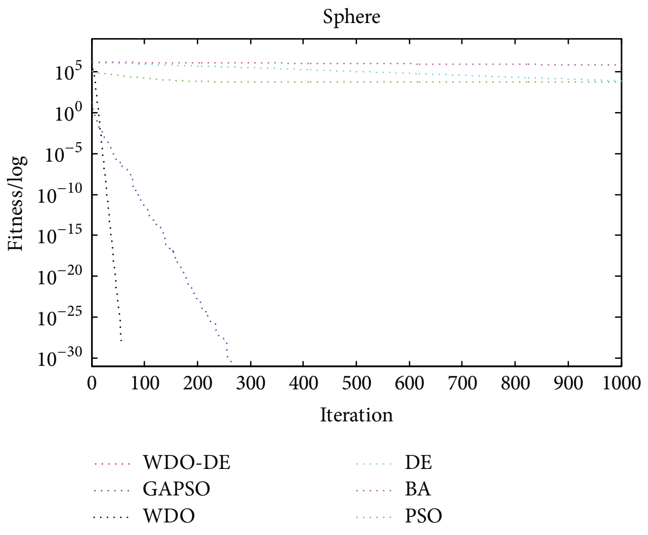

Figure 31: Comparison of performance of algorithms for minimization of f_1 ($D = 500$).

Table 2: The parameters set of WDO.

Parameters	Value
Population size	50
Max number of iterations	1000
RT coefficient	1
Constants in the update equation	0.8
Maximum allowed speed	0.3
Gravitational constant	0.6
Coriolis effect	0.7

for the parameters set of WDO [19]. The parameters set of WDO and DE is based on the practical experience to take the appropriate value. Tables 2 and 3 represent the necessary parameters in our experiment [20].

4.4. Algorithm Performance Comparison. In this section, in order to test the performance of WDO-DE algorithm,

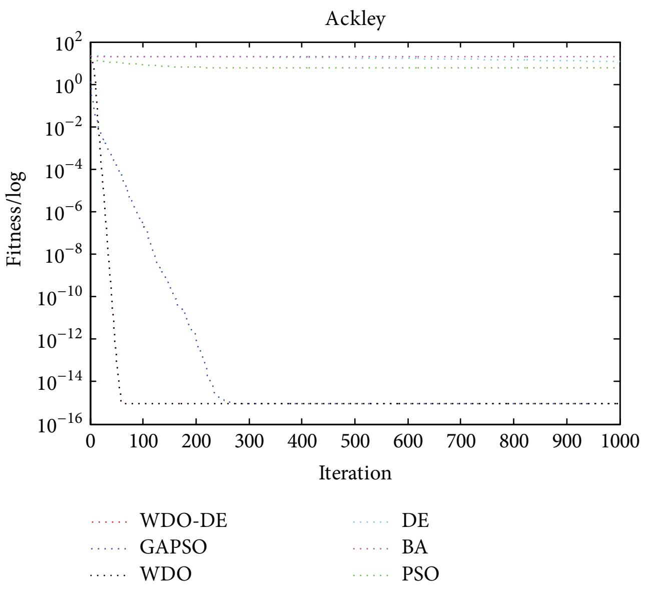

Figure 32: Comparison of performance of algorithms for minimization of f_2 ($D = 1000$).

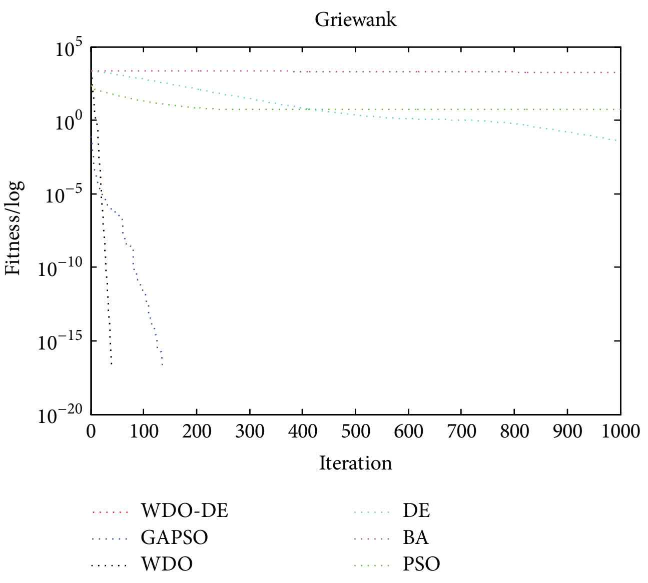

Figure 33: Comparison of performance of algorithms for minimization of f_3 ($D = 100$).

Table 3: The parameters set of DE.

Parameters	Value
Population size	50
Max number of iterations	1000
Mutation scale factor	0.2
Crossover probability	0.03

WDO-DE algorithm has been compared with the algorithms GA-PSO, WDO, DE, BA, and PSO in low dimension and high dimension. The mean results, standard deviation (Std.)

Table 4: Comparison of performance of algorithms in low dimension.

Test functions	Result	Algorithm					
		PSO	BA	DE	WDO	GAPSO	WDO-DE
f_1 $(D = 30)$	Mean	1.198565069	0.001259031	1.15546*E* − 12	0	8.938*E* − 112	**0**
	Std.	0.986982066	0.000164979	3.96904*E* − 13	0	6.2926*E* − 111	**0**
	Worst	4.623331223	0.001579358	2.31909*E* − 12	0	4.4499*E* − 110	**0**
	Best	0.077133013	0.000941037	6.51508*E* − 13	0	1.7446*E* − 133	**0**
	Rank	6	5	4	1	3	**1**
f_2 $(D = 30)$	Mean	3.23151972	18.95562594	3.29757*E* − 07	8.88178*E* − 16	8.88178*E* − 16	**8.88178E − 16**
	Std.	0.928231321	0.240934834	4.98931*E* − 08	0	0	**0**
	Worst	4.958505686	19.31856935	4.15132*E* − 07	8.88178*E* − 16	8.88178*E* − 16	**8.88178E − 16**
	Best	1.51521671	18.06057448	2.41907*E* − 07	8.88178*E* − 16	8.88178*E* − 16	**8.88178E − 16**
	Rank	5	6	5	1	1	**1**
f_3 $(D = 30)$	Mean	0.617836326	293.8529401	9.92471*E* − 11	0	0	**0**
	Std.	0.243522453	31.91402142	2.04264*E* − 10	0	0	**0**
	Worst	1.060834843	364.9175036	1.24088*E* − 09	0	0	**0**
	Best	0.118835584	231.9851842	6.60083*E* − 12	0	0	**0**
	Rank	5	6	4	1	1	**1**
f_4 $(D = 30)$	Mean	1.399873346	25.41987335	0.703241831	0.011986233	2.22495*E* − 05	**0**
	Std.	0.257935287	1.292363757	0.060062328	0.032788391	2.93452*E* − 05	**0**
	Worst	2.099873346	27.79987335	0.800431238	0.099920546	0.000122472	**0**
	Best	0.699873346	22.59987335	0.599891695	0	0	**0**
	Rank	5	6	4	3	2	**1**
f_5 $(D = 30)$	Mean	−265.951733	−272.200713	−418.982887	−89.1395986	−4.234983	−366.236378
	Std.	30.56577124	27.19605457	8.05405*E* − 12	14.48624741	3.304955999	12.75651852
	Worst	−202.664961	−208.394008	−418.982887	−60.8635682	−1.84114041	−332.945962
	Best	−346.349165	−328.828387	−418.982887	−149.39382	−14.3267589	−397.157289
	Rank	4	3	1	5	6	2
f_6 $(D = 30)$	Mean	1.39*E* − 01	0.005067	0.035133193	1.02406*E* − 05	0.001098329	**4.81416E − 06**
	Std.	3.80*E* − 01	0.001999654	0.006615916	1.03384*E* − 05	0.000919425	**5.23045E − 06**
	Worst	2.75*E* + 00	0.013185707	0.053066164	4.44353*E* − 05	0.004461194	**2.57286E − 05**
	Best	2.30*E* − 02	0.002422956	0.019201609	7.08496*E* − 08	2.49604*E* − 05	**6.73402E − 08**
	Rank	6	4	5	2	3	**1**
f_7 $(D = 30)$	Mean	120.2398214	20.32158962	11826.12725	0	1.5811*E* − 114	**0**
	Std.	71.10080897	67.06165947	1857.70219	0	1.0078*E* − 113	**0**
	Worst	273.1473491	408.7930042	15721.31651	0	7.097*E* − 113	**0**
	Best	20.21367214	0.004650193	8201.656294	0	6.0955*E* − 138	**0**
	Rank	5	4	6	1	3	**1**
f_8 $(D = 30)$	Mean	68.83078526	188.2789504	9.98004*E* − 08	0	0	**0**
	Std.	18.36191672	28.50077605	1.1162*E* − 07	0	0	**0**
	Worst	139.4625725	249.9566	6.78839*E* − 07	0	0	**0**
	Best	38.07588886	103.6807268	1.22403*E* − 08	0	0	**0**
	Rank	5	6	4	1	1	**1**
f_9 $(D = 30)$	Mean	3.257708808	5.701336524	0.000124894	0	3.10916*E* − 59	**0**
	Worst	1.752511346	2.663533038	4.35934*E* − 05	0	1.26282*E* − 58	**0**
	Best	7.423856504	13.2677015	0.000220385	0	6.7331*E* − 58	**0**
	Worst	0.503649602	1.083438608	5.01768*E* − 05	0	4.23177*E* − 67	**0**
	Rank	5	6	4	1	3	**1**

Table 4: Continued.

Test functions	Result	Algorithm					
		PSO	BA	DE	WDO	GAPSO	WDO-DE
f_{10} ($D = 2$)	Mean	0.397887358	0.397887361	0.397887358	0.507404422	0.411708093	**0.397887358**
	Std.	3.36448E − 16	4.03531E − 09	3.36448E − 16	0.134086235	0.008420321	**3.36448E − 16**
	Worst	0.397887358	0.397887373	0.397887358	1.01191027	0.427364689	**0.397887358**
	Best	0.397887358	0.397887358	0.397887358	0.407529279	0.398014086	**0.397887358**
	Rank	1	4	1	6	5	**1**
f_{11} ($D = 2$)	Mean	−1	−0.04000324	−1	−0.96070704	−0.99999566	**−1**
	Std.	0	0.197947995	0	0.039519354	7.81918E − 06	**0**
	Worst	−1	0	−1	−0.82029884	−0.99995866	**−1**
	Best	−1	−1	−1	−0.99846688	−1	**−1**
	Rank	1	6	1	5	4	**1**
f_{12} ($D = 2$)	Mean	3	7.320000769	3	7.838633652	3	**3**
	Std.	4.29203E − 15	9.998857372	3.99781E − 15	4.897011481	4.12124E − 15	**3.48119E − 15**
	Worst	3	30.00000257	3	19.47699347	3	**3**
	Best	3	3.000000005	3	3.024309606	3	**3**
	Rank	4	5	2	6	3	**1**
f_{13} ($D = 2$)	Mean	−1.03162845	−0.95001198	−1.03162845	−1.02310481	−1.03162845	−1.03162845
	Std.	2.24299E − 16	0.247335237	2.24299E − 16	0.010122307	2.24299E − 16	3.58878E − 16
	Worst	−1.03162845	−0.21546376	−1.03162845	−0.97714288	−1.03162845	−1.03162845
	Best	−1.03162845	−1.03162845	−1.03162845	−1.03147363	−1.03162845	−1.03162845
	Rank	1	6	1	5	1	4
f_{14} ($D = 3$)	Mean	−3.86199199	−3.86277843	−3.86277979	−3.73158405	−3.86277979	**−3.86277979**
	Std.	0.002387392	8.83669E − 07	4.93458E − 15	0.076497205	4.93458E − 15	**4.18232E − 15**
	Worst	−3.8549018	−3.86277528	−3.86277979	−3.52239202	−3.86277979	**−3.86277979**
	Best	−3.86277979	−3.86277961	−3.86277979	−3.85565215	−3.86277979	**−3.86277979**
	Rank	5	4	2	6	2	**1**
f_{15} ($D = 10$)	Mean	−7.67025031	−5.6907234	−9.66015172	−5.16498045	−6.53572784	−9.56777535
	Std.	0.781635349	1.010252541	4.39534E − 16	0.511107324	0.820311453	0.060299252
	Worst	−5.60672009	−3.1283389	−9.66015172	−3.93161423	−4.57905773	−9.32718849
	Best	−9.0444038	−8.11751567	−9.66015172	−6.35730534	−8.47351627	−9.64648266
	Rank	3	5	1	6	4	2
Average rank		4.06	5.06	3	3.33	2.8	**1.33**
Overall rank		5	6	3	4	2	**1**

results, the optimal fitness value, the worst fitness value, and rank results between the algorithms of 50 independent runs for $f_1 \sim f_{15}$ are shown in Table 4. Bold and italicized results mean that WDO-DE is better. Population size of other algorithms is 100. Max number of iterations of all tests is 1000.

For the low-dimensional case, according to Table 4, test results of WDO-DE are better than the other algorithms except f_5, f_{13}, and f_{15}. For f_5 and f_{15}, the DE algorithm gives the better results. Although the result of f_{13} function is worse than GA-PSO, DE, and PSO algorithm, it has already reached the theoretical optimal value. What is more, WDO-DE can find the theoretical optimum values for twelve benchmark functions ($f_1 \sim f_4$, $f_7 \sim f_{14}$) and has a very strong robustness. The novel hybrid global optimization algorithm is better than the original algorithm.

In the last, we calculated the average rank based on these fifteen functions' ranking [18]. Then, we rank the average rank and obtain the overall rank. From the average rank of each algorithm, we can learn that WDO-DE is very robust and efficient.

For the benchmark function f_{11}, the solution of WDO-DE algorithm is the closest to the theoretical optimal solution; for the benchmark function $f_{12} \sim f_{14}$, the original WDO algorithm cannot be close to the theoretical optimal solution; however the WDO-DE algorithm is close to the theoretical optimal solution (Table 5). And we can find that the convergence speed of the WDO-DE algorithm is quicker than other algorithms. In benchmark functions f_{11} and f_{13}, the WDO-DE algorithm has converged within 100 generations. And in benchmark functions f_{12} and f_{14}, the WDO-DE algorithm also has converged within 200 generations.

Meanwhile, Figures 1–15 have shown the evolutionary process of fitness value (the vertical axis is logarithmic fitness value). And Figures 16–30 are the ANOVA tests of the global

Table 5: Comparison of optimal solution of algorithms in low dimension.

Test functions	Numbers of iterations	Algorithm						Theoretical optimal solution
		PSO	BA	DE	WDO	GAPSO	WDO-DE	
f_{11} $(D = 2)$	50	(3.129031847760726, 3.134768665032161)	(3.141553103558010, −14.166037126201127)	(3.171893641024008, 3.131824693854047)	(0.0312746919874603, 0.0313915996157868)	(3.10191140474814, 3.10191140474814)	(3.14140267832352, 3.14201661395468)	(π, π)
	100	(3.140173624854166, 3.142584336878753)	(3.141553103558010, −14.166037126201127)	(3.141594887583755, 3.141597340960960)	(0.031336085130091, 0.0314633376198137)	(3.1630840677713, 3.1630840677713)	(3.14159264694396, 3.14159265459054)	
	200	(3.141597860814148, 3.141592422127699)	(3.141553103558010, −14.166037126201127)	(3.141592746546202, 3.141592336946040)	(0.031336085130091, 0.0314633376198137)	(3.14537991099944, 3.14537991099944)	(3.14159264694396, 3.14159265459054)	
	500	(3.141592649491149, 3.141592658550973)	(3.141553103558010, −14.166037126201127)	(3.141592746546202, 3.141592336946040)	(0.031336085130091, 0.0314633376198137)	(3.14380966948824, 3.14380966948824)	(3.14159264694396, 3.14159265459054)	
	1000	(3.141592649491149, 3.141592658550973)	(3.141643909560600, −14.166039618443593)	(3.141592746546202, 3.141592336946040)	(0.031336085130091, 0.0314633376198137)	(3.14241959184766, 3.14241959184766)	(3.14159264694396, 3.14159265459054)	
f_{12} $(D = 2)$	50	(−0.000274778503056, −1.000060523567713)	(−0.000239117085666, −1.000035528802250)	(−0.000019254003131, −1.000054781208720)	(0.0282917779333518, −0.448857319887312)	(−1.48794259472975e− 05, −0.999978563799561)	(3.40024382036861e − 07, −1.00000055778692)	(0, −1)
	100	(−0.000274778503056, −1.000060523567713)	(−0.000170307371325, −1.000123380488044)	(−0.000000003959147, −0.999999996915653)	(0.0282917779333518, −0.448857319887312)	(−1.93260130983304e− 09, −0.999999998297379)	(−1.81504593815985e − 09, −1.00000000299309)	
	200	(0.000000218751157, −0.999999566676768)	(−0.000109683118045, −1.000005656513546)	(0.000000000326953, −1.000000001047802)	(0.0282917779333518, −0.448857319887312)	(1.72540643351734e − 09, −1.00000000037874)	(4.59636719429074e − 10, −1.00000000221316)	
	500	(−0.000000003246821, −1.000000000641675)	(0.000010008823714, −1.000028736158479)	(−0.000000000328438, −1.000000001077549)	(0.0282917779333518, −0.448857319887312)	(1.72540643351734e − 09, −1.00000000037874)	(4.59636719429074e − 10, −1.00000000221316)	
	1000	(−0.000000001007721, −0.999999999978878)	(0.000010008823714, −1.000028736158479)	(−0.000000005887967, −1.000000003780139)	(0.0282917779333518, −0.448857319887312)	(5.07174141681265e − 09, −0.999999995696904)	(4.59636719429074e − 10, −1.00000000221316)	
f_{13} $(D = 2)$	50	(−0.090537017481104, 0.713861505393922)	(0.089934924086771, −0.712819423269000)	(−0.089842014825500, 0.712656400621485)	(0.0150932106229768, −0.139673825192312)	(−0.0898421518832255, 0.712656330384267)	(−0.0898489505500046, 0.712661110362659)	(0.0898, −0.7126) and (−0.0898, 0.7126)
	100	(−0.089579213574223, 0.712890938477391)	(0.089669828873280, −0.712754046171649)	(−0.089842014825500, 0.712656400621485)	(0.0150932106229768, −0.139673825192312)	(−0.0898420125705887, 0.712656401610307)	(−0.0898420134082198, 0.71265640418782)	
	200	(−0.089839286217455, 0.712655891026959)	(0.089744843168425, −0.712675910202981)	(−0.089842014825500, 0.712656400621485)	(0.0150932106229768, −0.139673825192312)	(−0.0898420139216395, 0.712656402819083)	(−0.0898420134082198, 0.71265640418782)	
	500	(−0.089842013891696, 0.712656404507059)	(0.089744843168425, −0.712675910202981)	(−0.089842014825500, 0.712656400621485)	(0.0150932106229768, −0.139673825192312)	(−0.0898420139216395, 0.712656402819083)	(−0.0898420134082198, 0.71265640418782)	
	1000	(−0.089842013891696, 0.712656404507059)	(0.089806395772405, −0.712611171310040)	(−0.089842014825500, 0.712656400621485)	(0.0150932106229768, −0.139673825192312)	(−0.0898420139216395, 0.712656402819083)	(−0.0898420134082198, 0.71265640418782)	

Table 5: Continued.

Test functions	Numbers of iterations	Algorithm						Theoretical optimal solution
		PSO	BA	DE	WDO	GAPSO	WDO-DE	
f_{14} $(D = 3)$	50	(0, 0.555683378893354, 0.853290531318392)	(0.113849817052731, 0.555603849176208, 0.852558071570959)	(0.111559931280739, 0.555576225589055, 0.852504656091373)	(0.344200172275741, 0.191217218742259, 0.719829205828109)	(0.114453784913746, 0.555632415452041, 0.852564399900014)	(0.114490931509466, 0.555654876198981, 0.852559993186743)	(0.114614, 0.555649, 0.852547)
	100	(0, 0.555636156053527, 0.853073594865928)	(0.113849817052731, 0.555603849176208, 0.852558071570959)	(0.114589394834657, 0.555648863896550, 0.852546999524024)	(0.344200172275741, 0.191217218742259, 0.719829205828109)	(0.114588883123683, 0.555648895909646, 0.852546984663178)	(0.114588896201262, 0.555648895689609, 0.852546983169442)	
	200	(0, 0.555678232858826, 0.853059878827053)	(0.115127379365939, 0.555614295721604, 0.852539734887561)	(0.114588865815882, 0.555648895017175, 0.852546985276585)	(0.344200172275741, 0.191217218742259, 0.719829205828109)	(0.114588890069778, 0.555648895154575, 0.852546983865954)	(0.114588880154869, 0.555648895361092, 0.852546984294326)	
	500	(0, 0.555678096423990, 0.853059974727304)	(0.115127379365939, 0.555614295721604, 0.852539734887561)	(0.114588865815882, 0.555648895017175, 0.852546985276585)	(0.344200172275741, 0.191217218742259, 0.719829205828109)	(0.114588890069778, 0.555648895154575, 0.852546983865954)	(0.114588880154869, 0.555648895361092, 0.852546984294326)	
	1000	(0, 0.555678096423990, 0.853059974727304)	(0.115127379365939, 0.555614295721604, 0.852539734887561)	(0.114588865815882, 0.555648895017175, 0.852546985276585)	(0.344200172275741, 0.191217218742259, 0.719829205828109)	(0.114588890069778, 0.555648895154575, 0.852546983865954)	(0.114588880154869, 0.555648895361092, 0.852546984294326)	

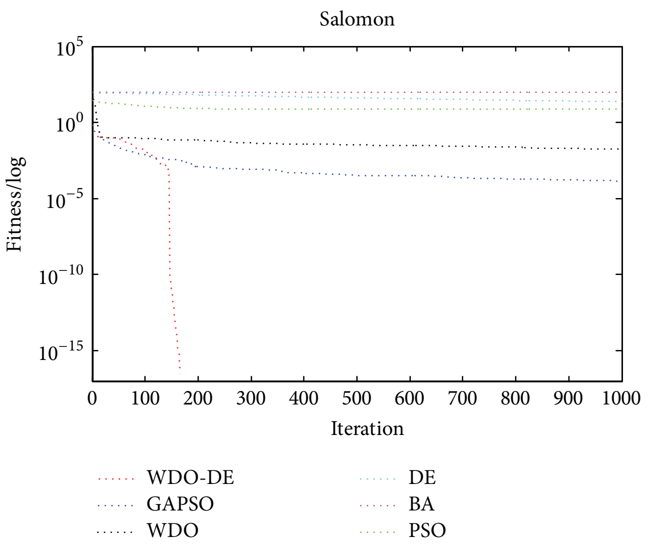

FIGURE 34: Comparison of performance of algorithms for minimization of f_4 ($D = 300$).

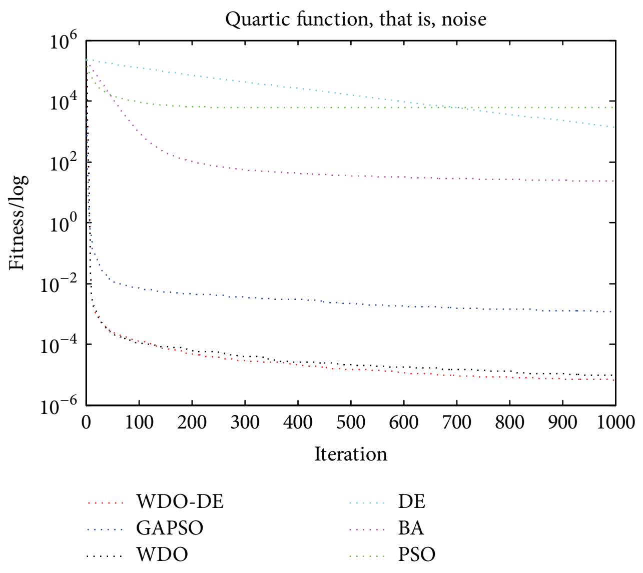

FIGURE 36: Comparison of performance of algorithms for minimization of f_6 ($D = 1000$).

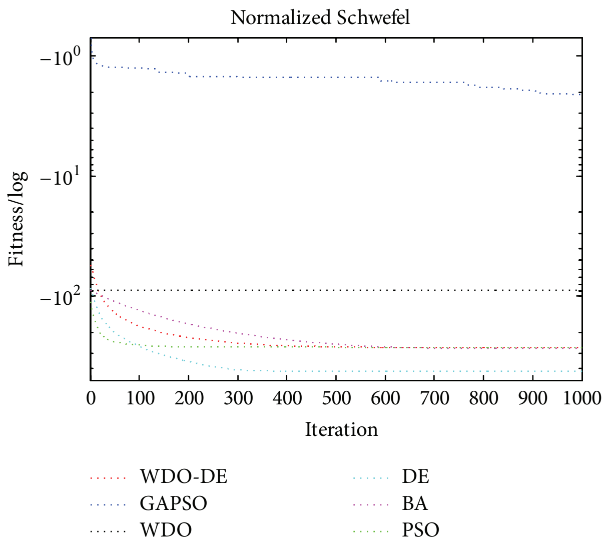

FIGURE 35: Comparison of performance of algorithms for minimization of f_5 ($D = 100$).

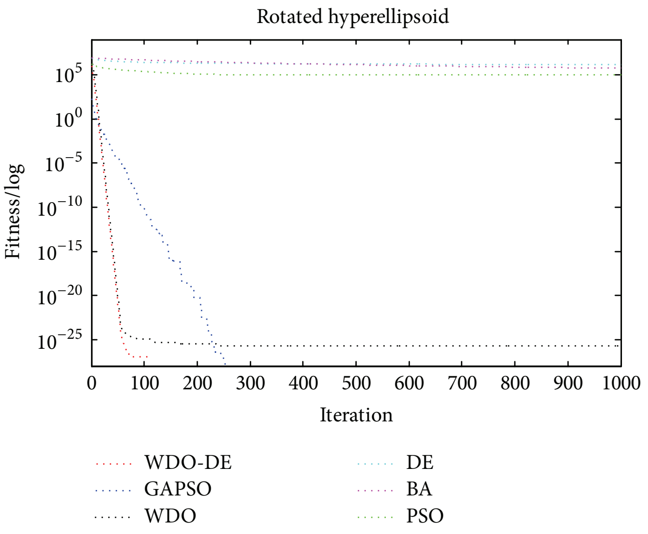

FIGURE 37: Comparison of performance of algorithms for minimization of f_7 ($D = 300$).

minimum. As can be seen from Figures 1–15, WDO-DE algorithm can converge within the maximum number of iterations except f_5 and f_{15}, and it has a faster global convergence speed in many functions and higher convergence precision. From the evolutionary process of fitness value it can be seen that the WDO-DE algorithm has a strong ability to find the optimal solutions. Moreover, as seen from Figures 16–30, we can learn that WDO-DE is the most robust in these algorithms. Therefore, WDO-DE is an effective and feasible solution for optimization problems in low-dimensional case.

In order to test the optimization ability of the algorithms in high-dimensional space, this paper selects several different dimensions for tests [15]. Among them, f_3 and f_5 were set to 100 dimensions, f_4 and f_7 set to 300 dimensions, f_1 and f_8 set to 500 dimensions, and f_2, f_6, and f_9 set to 1000 dimensions. In all the tests, the max number of iterations is 1000, and the set of other parameters is the same. The mean results, standard deviation (Std.) results, the optimal fitness value, the worst fitness value, and rank results between the algorithms of 50 independent runs for $f_1 \sim f_9$ are shown in Table 6.

For the high-dimensional case, as seen from Table 6, test results of WDO-DE are better than the other algorithms except f_5. For f_5, the results of f_5 function are secondary to DE algorithm; although the DE algorithm gives better results, it has not reached the theoretical optimal value. WDO-DE can find the theoretical optimum values for seven benchmark functions ($f_1 \sim f_4$, $f_7 \sim f_9$) and has a very strong robustness. This indicates that WDO-DE is very robust and efficient.

Table 6: Comparison of performance of algorithms in high dimension.

Test functions	Result	Algorithm					
		PSO	BA	DE	WDO	GAPSO	WDO-DE
f_1 ($D = 500$)	Mean	5942.812635	690529.75	7833.289879	0	$3.8474E-114$	**0**
	Std.	2010.942412	16105.47575	307.7206962	0	$2.6148E-113$	**0**
	Worst	12148.04486	721605.363	8350.202931	0	$1.8493E-112$	**0**
	Best	1477.889332	637825.2912	7208.53767	0	$5.6298E-132$	**0**
	Rank	4	6	5	1	3	**1**
f_2 ($D = 1000$)	Mean	6.021999854	19.58454162	6.556491165	$8.88178E-16$	$8.88178E-16$	**8.88178E − 16**
	Std.	0.963645094	0.072035802	0.086504851	0	0	**0**
	Worst	7.69115243	20.05614049	6.718429507	$8.88178E-16$	$8.88178E-16$	**8.88178E − 16**
	Best	3.224297469	19.5300771	6.318110244	$8.88178E-16$	$8.88178E-16$	**8.88178E − 16**
	Rank	4	6	5	1	1	**1**
f_3 ($D = 100$)	Mean	5.151497492	1774.455594	0.038171893	0	0	**0**
	Std.	2.161430833	103.0300286	0.006252419	0	0	**0**
	Worst	13.35711745	1927.77745	0.051473105	0	0	**0**
	Best	1.782030264	1418.804786	0.025514473	0	0	**0**
	Rank	5	6	4	1	1	**1**
f_4 ($D = 300$)	Mean	7.811873346	93.93587335	23.2983014	0.01799082	0.000134214	**0**
	Std.	1.358065852	1.176428355	0.51684138	0.038789028	0.00022591	**0**
	Worst	12.59987335	96.09987335	24.38078634	0.100231302	0.001189194	**0**
	Best	5.099873346	90.29987335	22.11997471	0	$3.67969E-07$	**0**
	Rank	4	6	5	3	2	**1**
f_5 ($D = 100$)	Mean	−188.353807	−225.492975	−415.031229	−49.9853541	−2.07455021	−266.617912
	Std.	24.35490997	13.25383666	1.616785061	9.085364316	1.865239717	16.69768835
	Worst	−110.804763	−196.772214	−409.751888	−29.977684	−0.9468723	−229.62838
	Best	−231.936138	−260.621897	−417.256124	−71.336055	−8.7435685	−294.826296
	Rank	4	3	1	5	6	2
f_6 ($D = 1000$)	Mean	6096.609068	23.0239015	1384.433152	$9.50848E-06$	0.001216642	**6.70813E − 06**
	Std.	3404.371174	1.515569145	74.20351119	$1.03931E-05$	0.000874723	**7.15954E − 06**
	Worst	14502.3679	27.87497256	1594.24402	$4.67328E-05$	0.003668518	**3.54361E − 05**
	Best	457.2497659	19.88875782	1219.60675	$1.46012E-07$	$1.20405E-05$	**3.50446E − 07**
	Rank	6	4	5	2	3	**1**
f_7 ($D = 300$)	Mean	97529.8314	570894.7164	1506962.5	$1.94072E-26$	$1.1197E-114$	**0**
	Std.	49025.64187	99081.42372	90799.2443	$1.3723E-25$	$6.0113E-114$	**0**
	Worst	254179.9951	781652.5658	1740971.299	$9.70362E-25$	$3.9713E-113$	**0**
	Best	39232.88764	383943.7522	1315295.793	0	$2.0359E-133$	**0**
	Rank	4	5	6	3	2	**1**
f_8 ($D = 500$)	Mean	3250.396382	4161.811265	3678.744509	0	0	**0**
	Std.	478.6226817	131.4154437	45.71686166	0	0	**0**
	Worst	4235.766671	4449.362401	3755.802488	0	0	**0**
	Best	2253.591512	3822.595502	3563.581287	0	0	**0**
	Rank	4	6	5	1	1	**1**
f_9 ($D = 1000$)	Mean	201.0787949	703.6914404	751.9162928	0	$2.40675E-58$	**0**
	Worst	51.1605331	35.62374915	15.21931169	0	$8.18033E-58$	**0**
	Best	385.1517001	765.0787636	779.6415024	0	$4.24827E-57$	**0**
	Worst	87.92253474	626.8641799	720.1039204	0	$1.30498E-65$	**0**
	Rank	4	5	6	1	3	**1**
Average rank		4.33	5.22	4.67	2	2.44	**1.11**
Overall rank		4	6	5	2	3	**1**

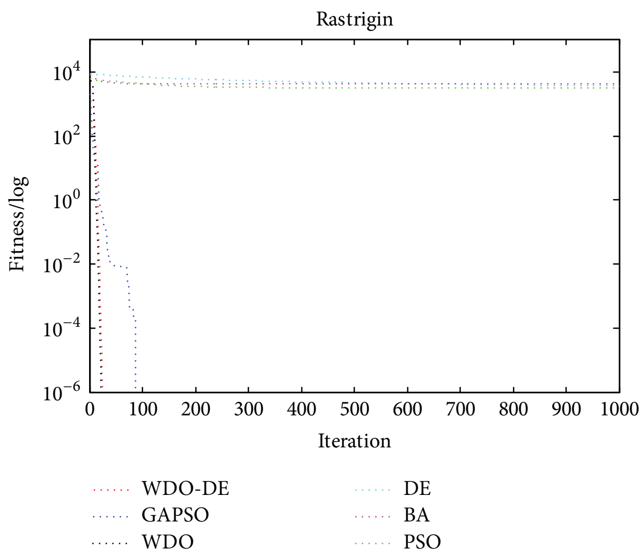

Figure 38: Comparison of performance of algorithms for minimization of f_8 ($D = 500$).

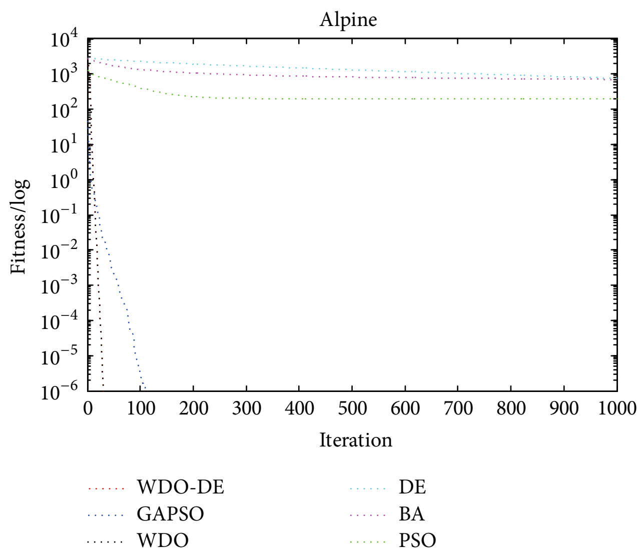

Figure 39: Comparison of performance of algorithms for minimization of f_9 ($D = 1000$).

The same as before, we calculated the average rank based on these nine functions' ranking. Then, we rank the average rank and obtain the overall rank. From the average rank of each algorithm, we can learn that WDO-DE is very robust and efficient.

The same as before, Figures 31–39 have shown the evolutionary process of fitness value (the vertical axis is logarithmic fitness value). And Figures 40–48 are the ANOVA tests of the global minimum. As can be seen from Figures 31–39, WDO-DE algorithm has the fastest global convergence speed and the highest convergence precision in all of these functions only except f_5. From the evolutionary process of fitness value it can be seen that the WDO-DE algorithm has a strong ability to find the optimal solutions. Moreover, as seen from Figures 40–48, we can learn that WDO-DE is the most robust in these algorithms. Therefore, WDO-DE is also an effective and feasible solution for optimization problems in high-dimensional case.

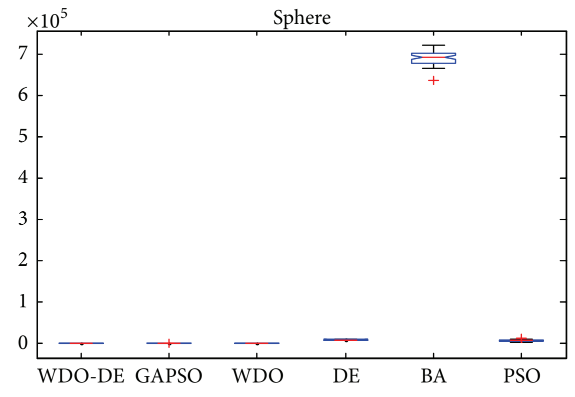

Figure 40: ANOVA tests of the global minimum values for f_1 ($D = 500$).

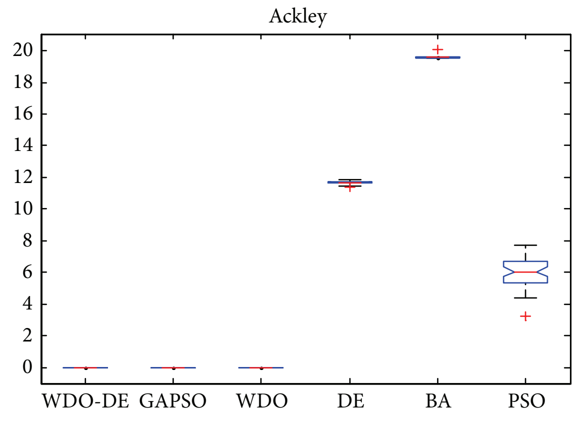

Figure 41: ANOVA tests of the global minimum values for f_2 ($D = 1000$).

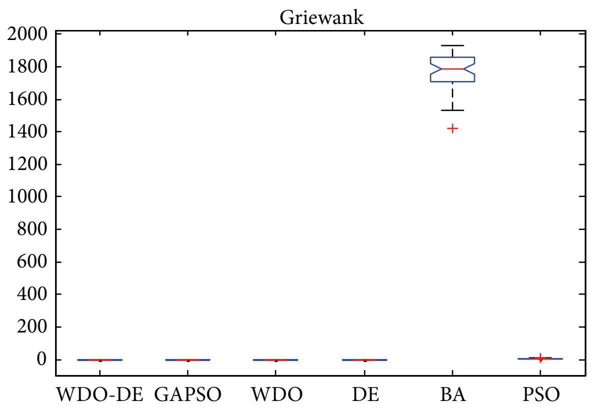

Figure 42: ANOVA tests of the global minimum values for f_3 ($D = 100$).

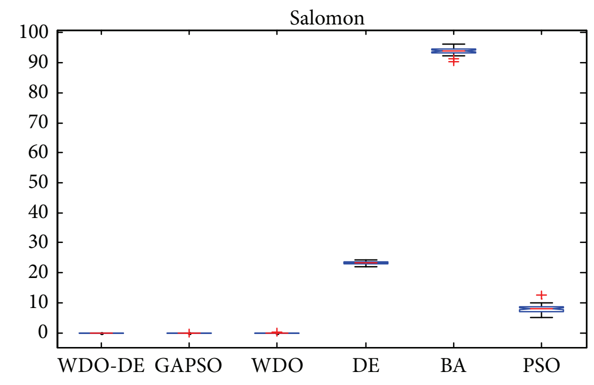

FIGURE 43: ANOVA tests of the global minimum values for f_4 (D = 300).

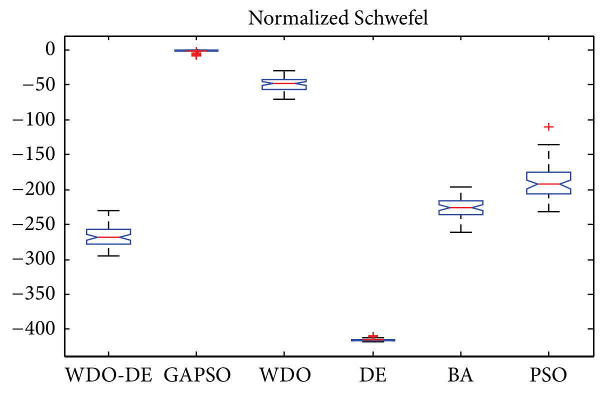

FIGURE 44: ANOVA tests of the global minimum values for f_5 (D = 100).

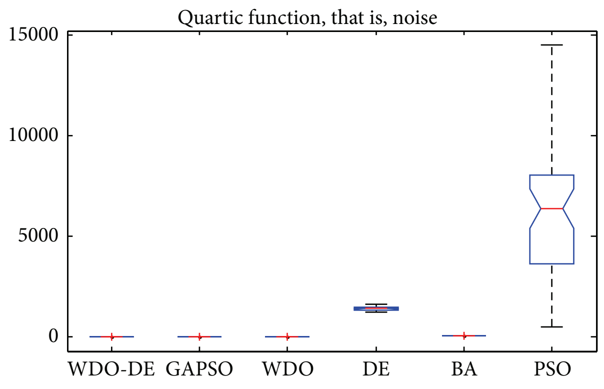

FIGURE 45: ANOVA tests of the global minimum values for f_6 (D = 1000).

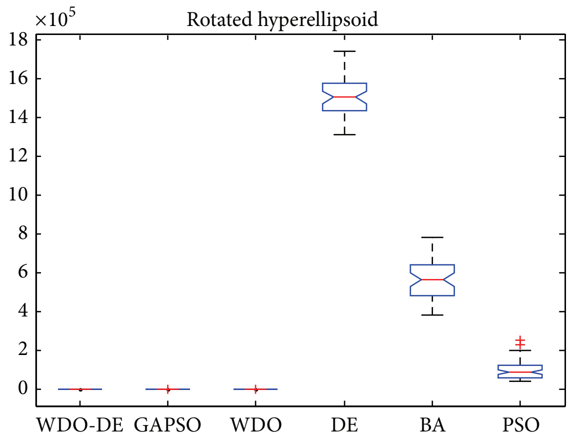

FIGURE 46: ANOVA tests of the global minimum values for f_7 (D = 300).

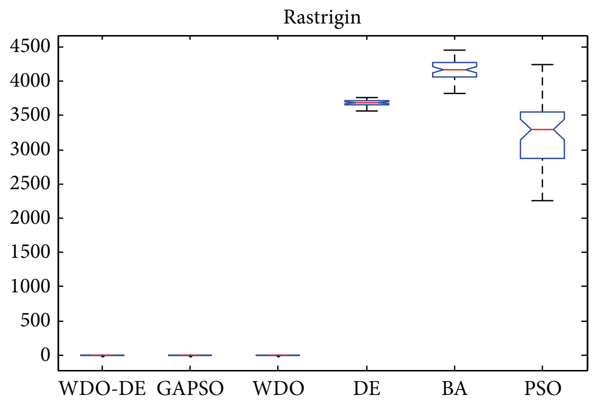

FIGURE 47: ANOVA tests of the global minimum values for f_8 (D = 500).

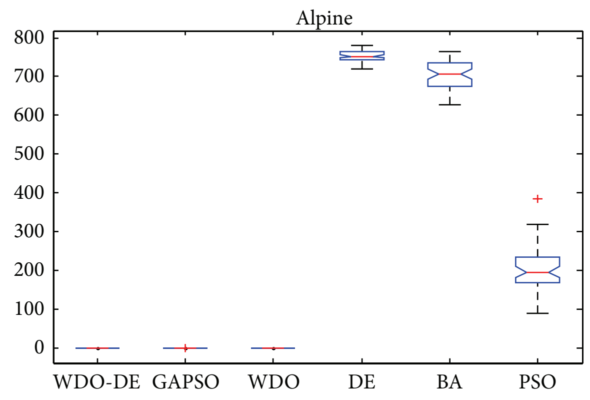

FIGURE 48: ANOVA tests of the global minimum values for f_9 (D = 1000).

5. Conclusion and Future Research

In this paper, we present a new hybrid global optimization algorithm called WDO-DE, which is based on the wind driven optimization (WDO) and differential evolution (DE) for solving optimization problems. We use 15 benchmark functions which contain unimodal, multimodal, low-dimensional, and high-dimensional unconstrained test functions to test the performance of WDO-DE algorithm. The WDO-DE algorithm can converge within the maximum number of iterations in most functions. In comparison with the GA-PSO, WDO, DE, BA, and PSO, the WDO-DE algorithm is more effective in finding better solutions and the convergence speed and precision of WDO-DE are higher. It is an effective and reliable global optimization algorithm.

Although in this paper the hybrid WDO-DE algorithm was implemented only for function optimization, in the field of optimization, there are still many aspects worthy of our study. Firstly, the hybrid algorithm proposed in this paper is based on the continuous space optimization. The future research may concentrate on discrete WDO algorithm. We can utilize many discretized strategies to discretize WDO algorithm to solve a problem characterized by discrete-valued design variables. Secondly, in engineering application, production management, and national defense construction, many optimization problems are multiobjective optimization problems, which are widely used in practical engineering. We would apply our proposed hybrid approach to solve multi-objective optimization problem. Lastly, we will learn more algorithms which have better optimization performance and analyze their characteristics. We would develop new hybrid approaches to solve the optimization problems raised above.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is supported by National Science Foundation of China under Grants nos. 61165015 and 61463007 and the Innovation Project of Guangxi Graduate Education.

References

[1] D. E. Golberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addion Wesley, 1989.

[2] R. C. Eberhart and J. A. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the 6th International Symposium on Micro Machine and Human Science (MHS '95)*, pp. 39–43, IEEE, Nagoya, Japan, October 1995.

[3] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of Machine Learning*, pp. 760–766, Springer, New York, NY, USA, 2010.

[4] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 26, no. 1, pp. 29–41, 1996.

[5] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[6] X. S. Yang, "Firefly algorithms for multimodal optimization," in *Stochastic Algorithms: Foundations and Applications*, pp. 169–178, Springer, Berlin, Germany, 2009.

[7] X. S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, pp. 65–74, Springer, Berlin, Germany, 2010.

[8] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature & Biologically Inspired Computing (NaBIC '09)*, pp. 210–214, IEEE, Coimbatore, India, December 2009.

[9] X. S. Yang, "Flower pollination algorithm for global optimization," in *Unconventional Computation and Natural Computation*, pp. 240–249, Springer, Berlin, Germany, 2012.

[10] Z. Bayraktar, M. Komurcu, and D. H. Werner, "Wind Driven Optimization (WDO): a novel nature-inspired optimization algorithm and its application to electromagnetics," in *Proceedings of the Antennas and Propagation Society International Symposium (APSURSI)*, pp. 1–4, IEEE, Toronto, Canada, July 2010.

[11] C.-F. Juang, "A hybrid of genetic algorithm and particle swarm optimization for recurrent network design," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 2, pp. 997–1006, 2004.

[12] G. Wang, L. Guo, H. Duan, H. Wang, L. Liu, and M. Shao, "A hybrid metaheuristic DE/CS algorithm for UCAV three-dimension path planning," *The Scientific World Journal*, vol. 2012, Article ID 583973, 11 pages, 2012.

[13] N. Holden and A. A. Freitas, "A hybrid particle swarm/ant colony algorithm for the classification of hierarchical biological data," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '05)*, pp. 100–107, June 2005.

[14] H. Riehl, *Introduction to the Atmosphere*, McGraw-Hill, 1978.

[15] C. D. Ahrens, *Meteorology Today: An Introduction to Weather, Climate, and the Environment*, Thomson-Brooks/Cole, Belmont, Calif, USA, 7th edition, 2003.

[16] H. Xia, Z. Wang, and Y. Zhou, "Double-population differential evolution based on logistic model," *Journal of Information and Computational Science*, vol. 11, no. 15, pp. 5549–5557, 2014.

[17] L. Li, Y. Zhou, and J. Xie, "A free search krill herd algorithm for functions optimization," *Mathematical Problems in Engineering*, vol. 2014, Article ID 936374, 21 pages, 2014.

[18] X. Li, J. Zhang, and M. Yin, "Animal migration optimization: an optimization algorithm inspired by animal migration behavior," *Neural Computing and Applications*, vol. 24, no. 7-8, pp. 1867–1877, 2014.

[19] Z. Bayraktar, M. Komurcu, J. A. Bossard, and D. H. Werner, "The wind driven optimization technique and its application in electromagnetics," *IEEE Transactions on Antennas and Propagation*, vol. 61, no. 5, pp. 2745–2757, 2013.

[20] A. K. Bhandari, V. K. Singh, A. Kumar, and G. K. Singh, "Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy," *Expert Systems with Applications*, vol. 41, no. 7, pp. 3538–3560, 2014.