

Article

An Improved Backtracking Search Algorithm for Global Optimization

Yiying Zhang 

College of Electrical Engineering and Automation, Shandong University of Science and Technology,
Qingdao 266590, China; zhangyiying@sdust.edu.cn

Abstract

Population-based optimization algorithms have been widely applied to optimization problems in many fields and have achieved encouraging computational results. To improve the performance of population-based optimization algorithms, it is necessary to adopt targeted improvement techniques to improve the algorithms, which is also a major direction for the current development of population-based optimization algorithms. To improve the performance of population-based optimization algorithms, this paper proposes a new technique for improving metaheuristic algorithms from the perspective of average operators, which defines six averaging operators called global average position, upper average position, lower average position, random global average position, random upper average position, and random lower average position. These six average operators define the average position from different perspectives. To verify the validity of the defined average operators, they are applied to the classic backtracking search algorithm (BSA), and the average operator-driven backtracking search algorithm (AOBSA) is proposed to overcome the disadvantage of BSA's single search strategy being prone to getting stuck in local optima. To investigate the performance of the AOBSA, it is used to solve 30 challenging test functions with 30 and 50 dimensions, along with two classical engineering design problems. The experimental results demonstrate the excellent performance of the AOBSA in solving numerical problems and engineering design problems. This also demonstrates the effectiveness of the defined average operators and their potential to be applied to other population-based optimization algorithms.

Keywords: average operator; backtracking search algorithm; global optimization; population-based optimization algorithm

MSC: 90C26



Academic Editor: Andrea Scozzari

Received: 11 May 2026

Revised: 26 June 2026

Accepted: 29 June 2026

Published: 2 July 2026

Copyright: © 2026 by the author.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Optimization problems exist in almost any engineering field. In recent years, using population-based optimization algorithms to solve complex engineering problems has always been a research hotspot. Liu et al. [1] designed a strength learning particle swarm optimization to solve the multi-objective multi-robot task scheduling problem. Wang et al. [2] used a reinforcement learning-based particle swarm optimization to solve the end-to-end traffic scheduling in a time-sensitive networking-5G network. Yang et al. [3] presented an enhanced gray wolf optimizer with unmanned aerial vehicle collaboration to optimize resource utilization in consumer electronics networks. Jyothi et al. [4] proposed an improved whale optimization algorithm to optimize the neural network model for

cyber-attack detection. Yu et al. [5] employed a multi-population differential evolution for feature selection with mutual information ranking. Dubey et al. [6] solved the robust and imperceptible image watermarking problem by a chaotic map-integrated quantum-inspired multi-objective cuckoo search algorithm. Liu et al. [7] handled the operation optimization problem in gas pipelines by designing a learning-based artificial bee colony algorithm. Majumdar and Mitra [8] put forward a salp swarm algorithm with lens opposition-based learning and local search to solve the feature selection problem.

The population-based optimization algorithm has a simple structure and straightforward search rules, and it does not require gradient information. This means that the population-based optimization algorithm has relatively low computational complexity. In addition, population-based optimization algorithms have three significant advantages:

- **Population:** Population-based optimization algorithms are population-based search methods. Each individual in the population represents a potential solution to the problem. All individuals in the population collaborate and share valuable information, driving the entire population to continuously find solutions that are closer to the global optimum. This collaborative and mutually supportive search approach can significantly increase the probability of population-based optimization algorithms finding the global optimum.
- **Randomness:** Population-based optimization algorithms are mostly stimulated by natural phenomena. To reflect this in the algorithm design, random numbers are usually introduced, such as those that follow the uniform distribution between 0 and 1, and those that follow the standard normal distribution. In addition, the population is also randomly selected in the feasible solution space during initialization. Randomness can weaken the correlation between the initialized population and the final solution found, thereby enhancing the stability of the algorithm. It is worth mentioning that the introduction of randomness can also enhance the algorithm's ability to escape from local optima to some extent, thereby improving its global search capability.
- **Balance Local Search and Global Search:** The search rules of population-based optimization algorithms usually take into account both local search and global search. This enables the algorithm to escape from local optima to a certain extent during the search process and find the global optimum as much as possible.

In recent years, inspired by various types of natural phenomena, a large number of population-based optimization algorithms have emerged continuously. Fu et al. [9] proposed the secretary bird optimization algorithm, inspired by the survival behavior of secretary birds in their natural environment. Pan et al. [10] designed the gannet optimization algorithm, inspired by the various unique behaviors of gannets during foraging. Dehghani et al. [11] presented the coati optimization algorithm, inspired by coatis' behavior in nature, including attacking and hunting, as well as their escape behavior when confronted by predators. Abdel-Basset et al. [12] put forward the Kepler optimization algorithm, inspired by Kepler's laws of planetary motion, to predict the position and velocity of planets at any given time. Trojovský et al. [13] presented the walrus optimization algorithm, inspired by the behaviors of walruses in nature, including the processes of feeding, migrating, escaping, and fighting predators. Zhang et al. [14] developed generalized normal distribution optimization, inspired by the generalized normal distribution model. Azizi et al. [15] contributed the energy valley optimizer, inspired by advanced physics principles regarding stability and different modes of particle decay. Deng and Liu [16] reported the snow ablation optimizer, inspired by the sublimation and melting behavior of snow in nature.

The backtracking search algorithm (BSA) is also a powerful population-based optimization algorithm, which is inspired by the return of a social group of living creatures at random intervals to hunting areas that were previously found fruitful for obtaining

nourishment [17]. The BSA has a very simple structure; it guides the search direction of the population through the vector difference between historical information and current information, and enriches the diversity of the population by introducing a binary crossover matrix. Since its proposal, the BSA has been successfully applied to solve many engineering problems. Wu et al. [18] used a two-neural-metaheuristic method based on vortex search and BSA to predict the heating load of residential buildings. Hu et al. [19] designed an adaptive BSA with a dual-learning strategy for dynamic economic dispatch with valve-point effects. Zhang et al. [20] proposed an enhanced BSA to solve the flight planning of a multi-drone-assisted commercial parcel delivery system. Dai et al. [21] reported an adaptive hybrid BSA to solve the dynamic economic dispatch with valve-point effects. Emami et al. [22] presented a backtracking search-based extreme gradient-boosting algorithm to solve soil moisture prediction. Although the application of the BSA in some engineering problems has achieved success, it is still very necessary to develop more efficient variants of BSA motivated by the following:

- The No-Free-Lunch Theorem [23]: An algorithm may demonstrate superior performance to other algorithms on certain problems, but it is impossible for it to exhibit better performance than other algorithms on all problems [23].
- The improvement techniques involved in these proposed improved versions of the BSA do not take into account the perspective of the average operator. The average operator is a comprehensive evaluation of the overall and local information of a population, which can better reflect the overall and local characteristics of the population.

Thus, this paper presents a new variant of BSA called the average operator-driven backtracking search algorithm (AOBSA). The search rule of the BSA primarily guides the search direction based on the vector difference between historical information and current information, and during the update of each individual, it relies on the vector difference between a historical individual and a current individual. Furthermore, the search rule does not utilize global information but, rather, local information. This will lead to the search mechanism being prone to getting stuck in local optimal solutions when solving complex optimization problems, due to the lack of valuable information that utilizes the entire population. To overcome the disadvantages of the BSA, the AOBSA is proposed. The main contributions of this paper can be summarized as follows:

- Six average operators are defined based on the overall information of the population.
- According to the defined average operators, a variant of BSA with six search strategies, called the AOBSA, is designed.
- The performance of the AOBSA is first tested by solving the challenging CEC 2017 test set [24], with two different dimensions, and then applied to solving complex engineering design problems.

The rest of this paper is organized as follows: Section 2 displays the basic BSA. Section 3 presents the proposed AOBSA. Section 4 discusses the experimental results of the AOBSA in solving the numerical problems. Section 5 discusses the experimental results of the AOBSA in solving the engineering problems. The conclusions are presented in Section 6.

2. BSA

The BSA is a highly convenient group-based optimization algorithm, whose framework primarily consists of four components, namely, initialization, Section-I, Section-II, and population generation [17]:

- Initialization: In the BSA, an individual represents a potential solution meeting the conditions, and all individuals constitute a population. This step aims to initialize the

population X ($X = \{x_1, x_2, \dots, x_N\}$), where N is the population size. The initialization can be expressed by

$$x_i = l + \kappa_1(u - l), i \in [1, N], \quad (1)$$

where l is the lower limit of the variables, u is the upper limit of the variables, κ is a random number following a uniform distribution between 0 and 1, and x_i is the position of individual i .

- Section-I: The historical population plays a very important role in guiding the search direction of the BSA. This step is to determine the historical population X_o during the search process. This can be written by

$$X_o = X, \text{ if } \kappa_2 \leq \kappa_3, \quad (2)$$

$$X_o = G(X_o), \text{ if } \kappa_2 > \kappa_3, \quad (3)$$

where κ_2 and κ_3 are two random numbers that follow a uniform distribution between 0 and 1, and $G(\cdot)$ is a function to randomly sort the individuals in the X_o .

- Section-II: This step is to get an initial population X_T ($X_T = \{x_{T,1}, x_{T,2}, \dots, x_{T,N}\}$) by the designed mutation and crossover operators. This step can be represented by

$$x_{T,i} = x_i + 3\kappa_4 h_i(x_{o,i} - x_i), i \in [1, N], \quad (4)$$

where κ_4 is a random number obeying the standard normal distribution and h_i is a crossover vector controlled by the mix rate. All crossover vectors can be denoted by $H = \{h_i, i = 1, 2, \dots, N\}$. The method of producing H is based on the reference [17].

- Population Generation: This step is to update the population, which can be stated as follows:

$$x_i = x_{T,i}, \text{ if } f(x_{T,i}) \leq f(x_i). \quad (5)$$

The implementation of the BSA can be referred to in Algorithm 1.

Algorithm 1 The implementation of BSA

Require: l, u, N , the number of variables D , the maximum number of iterations T_m

- 1: Initialize the population X by Equation (1);
 - 2: Initialize the historical population by Equation (1);
 - 3: Compute the fitness value of each individual in X and select the optimal solution x_{opt} ;
 - 4: **for** each $t \in [1, T_m]$ **do**
 - 5: Generate the crossover matrix H ;
 - 6: Update the historical population X_o by Equations (2) and (3);
 - 7: **for** each $i \in [1, N]$ **do**
 - 8: Get the initial individual $x_{T,i}$ by Equation (4);
 - 9: Update x_i by Equation (5);
 - 10: **end for**
 - 11: Compute the fitness value of each individual in X and update x_{opt} ;
 - 12: **end for**
 - 13: Output x_{opt} .
-

3. AOBSA

As mentioned in Equation (4), the search direction of this initial population is mainly determined by the vector difference between a vector in the historical population and a vector in the current population. In other words, Equation (4) focuses more on promoting the entire search process through local information exchange among populations, while ignoring the impact of overall population information on search quality. To overcome this drawback of the BSA, the proposed AOBSA first defines six average operators involving the overall

information of the population, and then designs six different search strategies based on these average operators.

3.1. Average Operators

The average position of the population is the most common indicator reflecting the overall characteristics of the population for the population-based optimization algorithm, which has been introduced to a lot of population-based optimization algorithms to improve their global search performance. Rao et al. [25] proposed teaching–learning-based optimization (TLBO). In TLBO, the average position of the population is the key parameter in the designed teaching phase. Zhang and Jin [26] designed the group teaching optimization algorithm (GTOA), where the average position of the population plays the important role in teaching phase I. Zhang et al. [14] presented generalized normal distribution optimization (GNDO). In GNDO, the search rule is formulated around the average position of the population. Motivated by this, the defined average operators are also based on the average position of the population, which can be expressed as follows:

- **Global Average Position:** The defined global average position is to reflect the overall characteristic of the population, which can be computed by

$$M_G = \frac{1}{N} \sum_{i=1}^N x_i, \quad (6)$$

where M_G is the global average position.

- **Upper Average Position:** The defined upper average position is to reflect the overall characteristic of the best half of the individuals in the population, which can be computed by

$$M_U = \frac{1}{0.5N} \sum_{i=1}^{0.5N} x_{a_j}, j = 1, 2, 3, \dots, 0.5N, \quad (7)$$

where M_U is the upper average position, and a is the set of the set of indices of the best half of the individuals in the population.

- **Lower Average Position:** The defined lower average position is to reflect the overall characteristic of the worst half of the individuals in the population, which can be computed by

$$M_L = \frac{1}{0.5N} \sum_{i=1}^{0.5N} x_{b_j}, j = 1, 2, 3, \dots, 0.5N, \quad (8)$$

where M_L is the lower average position, and b is the set of the set of indices of the worst half of the individuals in the population.

- **Random Global Average Position:** The defined random global average position is to reflect the overall random characteristic of the population, which can be computed by

$$M_{RG} = \lambda_1 M_G + \lambda_2 x_{\theta_1}, \quad (9)$$

where M_{RG} is the random global average position, λ_1 and λ_2 are two random numbers that follow a uniform distribution between 0 and 1, and θ_1 is a random integer selected within the range of 1 to N . In addition, λ_1 and λ_2 meet

$$\lambda_1 + \lambda_2 = 1. \quad (10)$$

- **Random Upper Average Position:** The defined random upper average position is to reflect the overall upper characteristic of the population, which can be computed by

$$M_{RU} = \lambda_3 M_U + \lambda_4 x_{\theta_2}, \quad (11)$$

where M_{RU} is the random upper average position, λ_3 and λ_4 are two random numbers that follow a uniform distribution between 0 and 1, and θ_2 is a random integer selected from a . In addition, λ_3 and λ_4 meet

$$\lambda_3 + \lambda_4 = 1. \quad (12)$$

- Random Lower Average Position: The defined random lower average position is to reflect the overall lower characteristic of the population, which can be computed by

$$M_{RL} = \lambda_5 M_L + \lambda_6 x_{\theta_3}, \quad (13)$$

where M_{RL} is the random lower average position, λ_5 and λ_6 are two random numbers that follow a uniform distribution between 0 and 1, and θ_3 is a random integer selected from b . In addition, λ_5 and λ_6 meet

$$\lambda_5 + \lambda_6 = 1. \quad (14)$$

3.2. Search Strategies

The BSA only has one search strategy, as shown in Equation (4). A single search strategy is not conducive to the BSA escaping from local optima when solving complex optimization problems, and it is prone to premature convergence. Thus, six search strategies were designed based on the defined average operators, which can be expressed as follows:

- The first search strategy is based on the defined global average position M_G , which can be written by

$$x_{T,i} = x_i + 0.5 \times 3\kappa_4 h_i(x_{o,i} - x_i + x_{o,i} - M_G), i \in [1, N]. \quad (15)$$

- The second search strategy is based on the defined upper average position M_U , which can be written by

$$x_{T,i} = x_i + 0.5 \times 3\kappa_4 h_i(x_{o,i} - x_i + x_{o,i} - M_U), i \in [1, N]. \quad (16)$$

- The third search strategy is based on the defined lower average position M_L , which can be written by

$$x_{T,i} = x_i + 0.5 \times 3\kappa_4 h_i(x_{o,i} - x_i + x_{o,i} - M_L), i \in [1, N]. \quad (17)$$

- The fourth search strategy is based on the defined random global average position M_{RG} , which can be written by

$$x_{T,i} = x_i + 0.5 \times 3\kappa_4 h_i(x_{o,i} - x_{c_1} + x_{o,i} - M_{RG}), i \in [1, N], \quad (18)$$

where c_1 is a random integer between 1 and N , and c_1 is not equal to i .

- The fifth search strategy is based on the defined random upper average position M_{RU} , which can be written by

$$x_{T,i} = x_i + 0.5 \times 3\kappa_4 h_i(x_{o,i} - x_{c_2} + x_{o,i} - M_{RU}), i \in [1, N], \quad (19)$$

where c_2 is a random integer between 1 and N , and c_2 is not equal to i .

- The sixth search strategy is based on the defined random lower average position M_{RL} , which can be written by

$$x_{T,i} = x_i + 0.5 \times 3\kappa_4 h_i(x_{o,i} - x_{c_3} + x_{o,i} - M_{RL}), i \in [1, N], \quad (20)$$

where c_3 is a random integer between 1 and N , and c_3 is not equal to i .

These six search strategies can be divided into two categories: one category includes Equations (15)–(17); the other category includes Equations (18)–(20). Compared with Equations (15)–(17), Equations (18)–(20) have stronger randomness, which is reflected in the following two aspects: Firstly, x_{c_1} , x_{c_2} , and x_{c_3} are selected randomly. Secondly, randomly selected individuals are also introduced to M_{RG} , M_{RU} , and M_{RL} , as shown in Equations (9)–(13). In population-based optimization algorithms, the randomness in the search rules facilitates the algorithm to escape from local optima and balance diversification and intensification, which is very helpful in improving the optimization performance of the algorithm, as discussed in [27].

3.3. The Implementation of the AOBSA

Algorithm 2 presents the implementation of the AOBSA. Comparing Algorithms 1 and 2, the differences between the BSA and AOBSA can be summarized as follows:

- The BSA only has one search strategy, whereas the AOBSA has six search strategies. Therefore, compared to the BSA, the AOBSA has a stronger ability to escape from local optima and maintain population diversity, which is beneficial for enhancing the global search capability of the AOBSA.
- The search strategy of the BSA only considers the local interaction between historical information and current information, while the six search strategies of the AOBSA not only consider the local interaction between historical information and current information but also consider the overall interaction between historical information and current information. Furthermore, these six search strategies are of equal significance to the search mechanism of the AOBSA. They merely update the individuals within the population from different perspectives; thus, their probabilities of being selected are also the same as shown in Algorithm 2 (see line 9 to line 12 in Algorithm 2). In addition, as mentioned above, the six search strategies belong to two categories: one category includes Equations (15)–(17); the other category includes Equations (18)–(20). Among them, Equations (15)–(17) are based on three basic operators, namely, global average position, upper average position, and lower average position. Equations (18)–(20) are based on the above three basic random operators, namely, random global average position, random upper average position, and random lower average position. These two types of search strategies have the same probability of being selected when prior knowledge is unknown, as shown in line 9 of Algorithm 2. Compared to the search strategy of the BSA, the search strategy of the AOBSA is not influenced by a single search direction but by the combined influence of two search directions, which can prevent the AOBSA falling into local optima to a certain extent.
- Compared to the search strategy of the BSA, the three search strategies (i.e., Equations (18)–(20)) of the AOBSA exhibit stronger randomness, which is primarily manifested in the following aspects: x_{c1} , x_{c2} , and x_{c3} are selected randomly, and randomly selected individuals are also introduced to M_{RG} , M_{RU} , and M_{RL} . This can also enhance the diversity of the AOBSA population from another perspective, thereby improving the global search performance of the AOBSA.
- Compared to the BSA, the computational complexity of the AOBSA does not increase significantly. As mentioned above, the increased computational complexity of the AOBSA compared to the BSA mainly lies in the computation of the six defined average operators and the formulation of the six search strategies. However, from Equations (6)–(20), this is a very simple calculation and does not incur significant computational overhead. According to Algorithm 2, the calculation process for the computational complexity of the AOBSA is as follows: line 1 is to initialize the population with N individuals, and each individual consists of D elements, whose computational complexity is $O(N \times D)$; line 2 is to generate the historical population, and its computational complexity is also $O(N \times D)$; line 3 is to evaluate the population and select the best individual, whose computational complexity is $O(N \times D)$; line 5 is to get the crossover matrix, and the worst computational complexity is $O(N \times D \times T_m)$ based on the generation method proposed by the author of the BSA [17]; line 6 is to update the historical population, whose computational complexity is $O(N \times D \times T_m)$; line 7 is to compute six average operators, whose computational complexity is $O(N \times D \times T_m)$; line 8 to line 15 are to update the individual, whose computational complexity is $O(N \times D \times T_m)$; line 16 is to select the best individual, whose computational complexity is $O(N)$. Thus, the total computational

complexity of the AOBSA is $O(N \times D \times T_m)$. Similarly, it can be concluded that the computational complexity of the BSA is $O(N \times D \times T_m)$. This method of calculating complexity ignores low-order terms and constant factors, and it focuses on examining the highest-order terms. Therefore, although the BSA and AOBSA have the same computational complexity in theoretical analysis, it should be noted that the AOBSA still has a higher computational time than the BSA due to its involvement of more operations, but it is not significantly higher than that of the BSA.

Algorithm 2 The implementation of the AOBSA

Require: l, u, N , the number of variables D , the maximum number of iterations T_m

```

1: Initialize the population  $X$  by Equation (1);
2: Initialize the historical population by Equation (1);
3: Compute the fitness value of each individual in  $X$  and select the optimal solution  $x_{opt}$ ;
4: for each  $t \in [1, T_m]$  do
5:   Generate the crossover matrix  $H$ ;
6:   Update the historical population  $X_o$  by Equations (2) and (3);
7:   Compute  $M_G, M_U, M_L, M_{RG}, M_{RU}$ , and  $M_{RL}$ ;
8:   for each  $i \in [1, N]$  do
9:     if rand < rand then
10:      Get the initial individual  $x_{T,i}$  by randomly selecting a search strategy from Equations (15)–(17);
11:     else
12:      Get the initial individual  $x_{T,i}$  by randomly selecting a search strategy from Equations (18)–(20);
13:     end if
14:     Update  $x_i$  by Equation (5);
15:   end for
16:   Update  $x_{opt}$ ;
17: end for
18: Output  $x_{opt}$ .
```

As mentioned earlier, the six search strategies designed in Algorithm 2 assume equal probability selection. When there is no prior information, equal probability is a conservative and fair choice. All conclusions in this article are based on this assumption. This article does not provide theoretical optimality proof or sensitivity analysis to different probabilities, which is a limitation of the algorithm designed in this article. Future work will delve into adaptive strategies for probabilistic selection, which will also be very helpful in promoting the development of group-based optimization algorithms.

4. Experimental Results in CEC 2017 Test Suite

This section aims to verify the performance of the AOBSA by solving the complex CEC 2017 test suite [24]. The CEC 2017 test suite consists of 30 test functions, and the vast majority of the test functions are complex multimodal functions. Compared to unimodal functions, multimodal functions have multiple local minima among their feasible solutions, posing significant challenges for algorithms to search for global optimal solutions. Therefore, the CEC 2017 test suite is highly ideal for verifying the optimization performance of the AOBSA.

4.1. Parameter Settings

To test the optimization competitiveness of the AOBSA, it was compared with the following nine algorithms: BSA [17], quadratic interpolation optimization (QIO) [28], gray wolf optimizer (GWO) [29], salp swarm algorithm (SSA) [30], Newton–Raphson-based optimizer (NRO) [31], blood-sucking leech optimizer (BSLO) [32], sine–cosine algorithm (SCA) [33], starfish optimization algorithm (SFOA) [34], and dandelion optimizer (DO) [35]. In the experiment, two dimensions of the CEC 2017 test suite were tested: 30-dimensional and the 50-dimensional. To ensure a fair comparison, the population size of all algorithms was the same (30 for solving 30-dimensional test functions and 50 for solving

50-dimensional test functions), the number of iterations for all algorithms was the same (10,000 for both 30-dimensional and 50-dimensional test functions), each test function was independently repeated 30 times for all algorithms, and other parameters of the compared algorithms were all taken from the original literature. All experiments were simulated via MATLAB R2017a installed on Windows 11. The computing environment consisted of a Core i7 system (2.3 GHz) with 16 GB of memory. The experimental result analysis tools included numerical analysis (i.e., mean and variance), the Friedman test, the Wilcoxon signed-rank test, and convergence analysis.

4.2. Discussion and Analysis of Experimental Results

Tables 1 and 2 present the experimental results produced by the AOBSA and the compared algorithms in solving the CEC 2017 test suite with 30 and 50 dimensions, respectively. In the two tables, MEAN and STD denote the mean solution and standard deviation from 30 independent repeated experiments, respectively. From Table 1, the AOBSA achieved the best MEAN in 23 test functions, including $F_1, F_3, F_4, F_5, F_6, F_7, F_9, F_{11}, F_{13}, F_{14}, F_{15}, F_{16}, F_{18}, F_{19}, F_{21}, F_{23}, F_{24}, F_{25}, F_{26}, F_{27}, F_{28}, F_{29}$, and F_{30} . The BSA obtained the best MEAN in six test functions, i.e., $F_6, F_8, F_{11}, F_{17}, F_{21}$, and F_{25} . QIO and SFOA achieved the best MEAN in four test functions (i.e., F_3, F_{11}, F_{22} , and F_{25}) and four test functions (i.e., F_3, F_{12}, F_{20} , and F_{25}), respectively. SSA, BSLO, and DO found the best MEAN in two test functions (i.e., F_3 and F_{25}), three test functions (i.e., F_6, F_{10} , and F_{25}), and three test functions (i.e., F_2, F_3 , and F_{25}), respectively. GWO, NRO, and SCA could not search for the best MEAN in any test functions. The AOBSA achieved the best STD in two-thirds of the test functions, i.e., $F_1, F_3, F_5, F_7, F_9, F_{11}, F_{13}, F_{14}, F_{15}, F_{16}, F_{17}, F_{18}, F_{19}, F_{21}, F_{23}, F_{24}, F_{25}, F_{27}, F_{29}$, and F_{30} . QIO, GWO, and BSLO obtained the best STD in two test functions (i.e., F_2 and F_{22}), three test functions (i.e., F_8, F_{22} , and F_{26}), and two test functions (i.e., F_4 and F_6), respectively. In terms of STD, BSA, SCA, SFOA, and DO only beat the other algorithms in F_{12}, F_{10}, F_3 , and F_{28} , respectively. SSA and NRO could not get the best STD in any test functions. In terms of MEAN and STD, the AOBSA achieved the best results on at least two-thirds of the test functions. This indicates that the AOBSA significantly outperforms the compared algorithms in terms of search accuracy and stability. From Table 2, the AOBSA obtained the best MEAN in 11 test functions, i.e., $F_1, F_4, F_6, F_9, F_{12}, F_{14}, F_{15}, F_{19}, F_{27}, F_{29}$, and F_{30} . The BSA also shows competitiveness, obtaining the best MEAN in nine test functions, i.e., $F_5, F_8, F_{11}, F_{13}, F_{16}, F_{17}, F_{21}, F_{23}$, and F_{24} . In terms of MEAN, QIO, GWO, NRO, and SFOA beat the other algorithms in two test functions (i.e., F_2 and F_{28}), three test functions (i.e., F_3, F_{26} , and F_{28}), three test functions (i.e., F_7, F_{10} , and F_{28}), and four test functions (i.e., F_{18}, F_{20}, F_{22} , and F_{25}), respectively. GWO, NRO, SCA, and DO did not demonstrate any superiority over other algorithms in any test functions. In terms of STD, the AOBSA was the best of all algorithms in $F_1, F_4, F_5, F_6, F_7, F_8, F_9, F_{12}, F_{14}, F_{15}, F_{16}, F_{17}, F_{18}, F_{19}, F_{23}, F_{24}, F_{26}, F_{29}$, and F_{30} . QIO, BSA, and SCA obtained the best STD in three test functions (i.e., F_2, F_{11} , and F_{28}), three test functions (i.e., F_{13}, F_{21} , and F_{25}), and three test functions (i.e., F_{10}, F_{20} , and F_{22}), respectively. In terms of STD, GWO and SFOA beat the other algorithms in F_{27} and F_3 , respectively. SSA, NRO, BSLO, and DO did not get the best STD in any test functions. That is, although the AOBSA is slightly superior to the BSA in global search capability, it is significantly better than the other compared algorithms. In terms of stability, the AOBSA is significantly superior to the comparison algorithms. Based on the analysis of the experimental results above, the AOBSA outperforms the compared algorithms in terms of stability and solution accuracy when solving the CEC 2017 test suite with 30 and 50 dimensions. This also demonstrates the excellent global search capability and robustness of the AOBSA compared to the comparative algorithms.

Table 1. The experimental results of the AOBSA and the nine compared algorithms on 30 benchmark functions from the CEC 2017 test suite with 30 dimensions.

No.	Algorithm	QIO	GWO	SSA	NRO	BSLO	BSA	SCA	SFOA	DO	AOBSA
F ₁	MEAN	3.09×10^3	2.76×10^9	5.98×10^3	1.34×10^{10}	6.58×10^3	1.53×10^2	1.27×10^{10}	5.20×10^8	5.88×10^3	1.00×10^2
	STD	3.21×10^3	2.15×10^9	4.39×10^9	4.39×10^9	6.64×10^3	2.15×10^2	1.69×10^9	2.26×10^9	6.84×10^3	3.36×10^{-8}
F ₂	MEAN	2.00×10^2	1.87×10^{31}	2.03×10^2	2.15×10^{34}	2.48×10^2	1.95×10^{43}	3.06×10^{33}	8.95×10^3	2.08×10^2	6.26×10^3
	STD	4.06×10^{-4}	1.02×10^{32}	1.11×10^{35}	1.11×10^{35}	1.41×10^2	1.07×10^{44}	1.14×10^{34}	3.87×10^4	7.83	2.64×10^4
F ₃	MEAN	3.00×10^2	3.63×10^4	3.00×10^2	3.48×10^4	3.53×10^2	3.07×10^2	3.63×10^4	3.00×10^2	3.00×10^2	3.00×10^2
	STD	2.94×10^{-3}	9.77×10^3	6.01×10^3	6.01×10^3	6.68×10^1	1.38×10^1	5.36×10^3	1.58×10^{-13}	2.11×10^{-1}	2.59×10^{-1}
F ₄	MEAN	4.59×10^2	5.65×10^2	4.84×10^2	1.56×10^3	4.86×10^2	4.75×10^2	1.44×10^3	4.44×10^2	4.85×10^2	4.22×10^2
	STD	3.70×10^1	4.40×10^1	7.93×10^2	7.93×10^2	1.80×10^1	2.36×10^1	2.72×10^2	2.72×10^1	2.53×10^1	2.80×10^1
F ₅	MEAN	6.12×10^2	5.97×10^2	6.20×10^2	8.05×10^2	6.17×10^2	5.86×10^2	7.79×10^2	6.17×10^2	6.61×10^2	5.76×10^2
	STD	2.63×10^1	3.20×10^1	3.91×10^1	3.91×10^1	2.13×10^1	1.92×10^1	1.87×10^1	3.12×10^1	3.85×10^1	1.46×10^1
F ₆	MEAN	6.08×10^2	6.07×10^2	6.29×10^2	6.68×10^2	6.00×10^2	6.00×10^2	6.49×10^2	6.05×10^2	6.33×10^2	6.00×10^2
	STD	8.13	2.91	1.18×10^1	1.18×10^1	7.57×10^{-4}	5.78×10^{-1}	5.30	4.62	6.01	1.49×10^{-4}
F ₇	MEAN	8.93×10^2	8.79×10^2	8.72×10^2	1.17×10^3	8.57×10^2	8.34×10^2	1.12×10^3	8.75×10^2	1.00×10^3	8.30×10^2
	STD	5.34×10^1	5.07×10^1	8.58×10^1	8.58×10^1	2.66×10^1	2.53×10^1	4.30×10^1	4.29×10^1	5.62×10^1	1.66×10^1
F ₈	MEAN	8.99×10^2	8.89×10^2	9.13×10^2	1.06×10^3	9.18×10^2	8.68×10^2	1.05×10^3	9.05×10^2	9.42×10^2	8.77×10^2
	STD	2.01×10^1	1.66×10^1	3.27×10^1	3.27×10^1	2.72×10^1	1.68×10^1	1.91×10^1	2.96×10^1	4.33×10^1	1.02×10^1
F ₉	MEAN	1.91×10^3	1.72×10^3	2.58×10^3	7.33×10^3	2.56×10^3	1.14×10^3	5.03×10^3	1.73×10^3	4.62×10^3	9.45×10^2
	STD	5.42×10^2	6.55×10^2	1.38×10^3	1.38×10^3	9.51×10^2	1.51×10^2	1.00×10^3	7.32×10^2	1.45×10^3	1.29×10^2
F ₁₀	MEAN	4.24×10^3	4.04×10^3	4.86×10^3	7.47×10^3	3.91×10^3	6.78×10^3	8.11×10^3	4.30×10^3	4.68×10^3	4.63×10^3
	STD	7.01×10^2	6.01×10^2	5.90×10^2	5.90×10^2	4.87×10^2	1.16×10^3	2.89×10^2	4.41×10^2	5.80×10^2	5.66×10^2
F ₁₁	MEAN	1.17×10^3	2.11×10^3	1.26×10^3	2.19×10^3	1.20×10^3	1.17×10^3	2.06×10^3	1.26×10^3	1.21×10^3	1.17×10^3
	STD	3.60×10^1	9.24×10^2	5.79×10^2	5.79×10^2	3.76×10^1	3.62×10^1	2.40×10^2	7.45×10^1	3.78×10^1	2.97×10^1
F ₁₂	MEAN	3.33×10^4	4.70×10^7	2.01×10^6	1.19×10^9	1.22×10^6	2.57×10^4	1.24×10^9	2.04×10^4	2.24×10^6	2.71×10^4
	STD	1.74×10^4	4.54×10^7	1.06×10^9	1.06×10^9	1.08×10^6	1.12×10^4	3.78×10^8	2.20×10^4	9.98×10^5	1.65×10^4

Table 1. Cont.

No.	Algorithm	QIO	GWO	SSA	NRO	BSLO	BSA	SCA	SFOA	DO	AOBSA
F ₁₃	MEAN	1.25×10^4	1.07×10^6	8.06×10^4	1.52×10^8	3.30×10^4	1.45×10^4	3.53×10^8	2.59×10^3	6.96×10^4	1.53×10^3
	STD	1.34×10^4	4.35×10^6	1.28×10^8	1.28×10^8	2.43×10^4	1.25×10^4	1.27×10^8	1.83×10^3	2.66×10^4	3.98×10^2
F ₁₄	MEAN	2.19×10^3	2.24×10^5	5.46×10^3	4.94×10^4	4.00×10^4	1.48×10^3	1.33×10^5	1.63×10^3	1.64×10^4	1.44×10^3
	STD	1.60×10^3	4.26×10^5	3.60×10^4	3.60×10^4	2.77×10^4	3.77×10^1	5.49×10^4	9.21×10^1	1.25×10^4	6.39
F ₁₅	MEAN	3.63×10^3	3.26×10^5	5.90×10^4	3.98×10^5	1.34×10^4	5.72×10^3	1.10×10^7	1.76×10^3	2.12×10^4	1.55×10^3
	STD	2.68×10^3	8.01×10^5	8.12×10^5	8.12×10^5	1.39×10^4	7.21×10^3	9.15×10^6	1.55×10^2	1.18×10^4	2.29×10^1
F ₁₆	MEAN	2.57×10^3	2.34×10^3	2.59×10^3	3.49×10^3	2.76×10^3	2.37×10^3	3.56×10^3	2.26×10^3	2.55×10^3	2.19×10^3
	STD	2.86×10^2	2.75×10^2	3.27×10^2	3.27×10^2	2.65×10^2	1.85×10^2	2.23×10^2	2.60×10^2	2.93×10^2	1.48×10^2
F ₁₇	MEAN	2.11×10^3	1.92×10^3	1.99×10^3	2.43×10^3	2.26×10^3	1.86×10^3	2.38×10^3	2.01×10^3	2.15×10^3	1.87×10^3
	STD	2.10×10^2	1.08×10^2	2.27×10^2	2.27×10^2	2.11×10^2	9.25×10^1	1.53×10^2	1.56×10^2	1.73×10^2	6.90×10^1
F ₁₈	MEAN	1.35×10^4	7.88×10^5	1.72×10^5	1.05×10^6	2.70×10^5	2.35×10^4	2.48×10^6	4.20×10^3	1.72×10^5	2.12×10^3
	STD	9.68×10^3	1.04×10^6	7.99×10^5	7.99×10^5	2.75×10^5	5.72×10^4	1.42×10^6	8.99×10^3	1.25×10^5	1.89×10^2
F ₁₉	MEAN	6.87×10^3	8.62×10^5	2.58×10^5	4.91×10^6	1.54×10^4	2.80×10^3	2.60×10^7	2.05×10^3	2.71×10^4	1.92×10^3
	STD	5.19×10^3	1.92×10^6	3.51×10^6	3.51×10^6	1.61×10^4	2.50×10^3	1.68×10^7	7.92×10^1	2.25×10^4	4.04
F ₂₀	MEAN	2.32×10^3	2.36×10^3	2.41×10^3	2.66×10^3	2.50×10^3	2.58×10^3	2.58×10^3	2.25×10^3	2.54×10^3	2.28×10^3
	STD	1.71×10^2	1.26×10^2	1.57×10^2	1.57×10^2	1.95×10^2	3.21×10^2	1.38×10^2	1.44×10^2	1.72×10^2	1.54×10^2
F ₂₁	MEAN	2.40×10^3	2.39×10^3	2.41×10^3	2.58×10^3	2.44×10^3	2.36×10^3	2.55×10^3	2.41×10^3	2.46×10^3	2.36×10^3
	STD	3.05×10^1	2.16×10^1	4.32×10^1	4.32×10^1	2.48×10^1	1.42×10^1	2.32×10^1	3.10×10^1	4.24×10^1	1.07×10^1
F ₂₂	MEAN	2.30×10^3	4.48×10^3	4.78×10^3	5.27×10^3	4.34×10^3	6.64×10^3	8.35×10^3	3.44×10^3	5.73×10^3	5.32×10^3
	STD	1.07	1.50×10^3	1.92×10^3	1.92×10^3	1.51×10^3	2.27×10^3	2.32×10^3	1.65×10^3	1.69×10^3	1.89×10^3
F ₂₃	MEAN	2.79×10^3	2.75×10^3	2.76×10^3	3.00×10^3	2.77×10^3	2.74×10^3	2.98×10^3	2.78×10^3	2.86×10^3	2.72×10^3
	STD	4.36×10^1	2.85×10^1	4.30×10^1	4.30×10^1	2.19×10^1	5.10×10^1	2.49×10^1	4.87×10^1	4.16×10^1	1.02×10^1
F ₂₄	MEAN	2.98×10^3	2.92×10^3	2.91×10^3	3.16×10^3	3.07×10^3	2.93×10^3	3.16×10^3	2.95×10^3	3.09×10^3	2.88×10^3
	STD	5.20×10^1	5.23×10^1	5.65×10^1	5.65×10^1	7.18×10^1	7.38×10^1	3.08×10^1	4.98×10^1	5.89×10^1	1.70×10^1
F ₂₅	MEAN	2.89×10^3	2.98×10^3	2.89×10^3	3.31×10^3	2.89×10^3	2.89×10^3	3.20×10^3	2.89×10^3	2.89×10^3	2.89×10^3
	STD	1.51×10^1	3.28×10^1	1.42×10^2	1.42×10^2	1.17×10^1	1.40×10^1	5.76×10^1	1.12×10^1	1.67×10^1	1.78

Table 1. Cont.

No.	Algorithm	QIO	GWO	SSA	NRO	BSLO	BSA	SCA	SFOA	DO	AOBSA
F ₂₆	MEAN	4.95×10^3	4.43×10^3	4.63×10^3	6.92×10^3	4.74×10^3	4.68×10^3	6.94×10^3	4.89×10^3	5.61×10^3	4.35×10^3
	STD	1.33×10^3	4.20×10^2	1.15×10^3	1.15×10^3	9.00×10^2	4.40×10^2	4.93×10^2	1.09×10^3	8.74×10^2	4.27×10^2
F ₂₇	MEAN	3.26×10^3	3.25×10^3	3.24×10^3	3.40×10^3	3.22×10^3	3.30×10^3	3.41×10^3	3.25×10^3	3.27×10^3	3.21×10^3
	STD	2.90×10^1	1.64×10^1	9.16×10^1	9.16×10^1	1.72×10^1	9.01×10^1	4.62×10^1	3.23×10^1	3.81×10^1	1.33×10^1
F ₂₈	MEAN	3.15×10^3	3.42×10^3	3.19×10^3	3.97×10^3	3.20×10^3	3.89×10^3	3.82×10^3	3.15×10^3	3.19×10^3	3.14×10^3
	STD	6.25×10^1	8.70×10^1	3.18×10^2	3.18×10^2	3.61×10^1	1.82×10^3	8.81×10^1	6.32×10^1	3.98×10^1	6.02×10^1
F ₂₉	MEAN	3.76×10^3	3.74×10^3	3.94×10^3	4.61×10^3	3.75×10^3	3.93×10^3	4.66×10^3	3.77×10^3	4.03×10^3	3.62×10^3
	STD	2.21×10^2	1.75×10^2	3.83×10^2	3.83×10^2	1.85×10^2	5.93×10^2	2.11×10^2	2.04×10^2	2.45×10^2	1.38×10^2
F ₃₀	MEAN	7.43×10^3	3.63×10^6	9.37×10^5	2.24×10^7	1.40×10^4	6.83×10^3	8.08×10^7	6.53×10^3	1.36×10^5	5.23×10^3
	STD	1.49×10^3	2.54×10^6	1.55×10^7	1.55×10^7	5.11×10^3	2.01×10^3	2.87×10^7	2.73×10^3	1.02×10^5	3.48×10^2

Table 2. The experimental results of the AOBSA and the nine compared algorithms on 30 benchmark functions from the CEC 2017 test suite with 50 dimensions.

No.	Algorithm	QIO	GWO	SSA	NRO	BSLO	BSA	SCA	SFOA	DO	AOBSA
F ₁	MEAN	3.16×10^3	6.50×10^9	6.05×10^3	3.63×10^{10}	7.10×10^3	1.20×10^3	3.97×10^{10}	2.05×10^8	1.49×10^4	2.38×10^2
	STD	4.04×10^3	3.13×10^9	6.20×10^9	6.20×10^9	8.38×10^3	1.59×10^3	7.13×10^9	8.69×10^8	1.02×10^4	3.00×10^2
F ₂	MEAN	5.31×10^2	5.48×10^{51}	1.17×10^9	4.43×10^{62}	6.79×10^{12}	1.14×10^{15}	2.97×10^{65}	5.10×10^{20}	2.33×10^7	5.37×10^{13}
	STD	9.64×10^2	2.85×10^{52}	2.43×10^{63}	2.43×10^{63}	2.62×10^{13}	5.00×10^{15}	1.61×10^{66}	2.22×10^{21}	7.14×10^7	1.72×10^{14}
F ₃	MEAN	1.61×10^3	7.62×10^4	3.00×10^2	9.51×10^4	9.22×10^2	9.43×10^3	1.01×10^5	3.02×10^2	3.19×10^2	1.30×10^4
	STD	7.46×10^2	1.37×10^4	1.59×10^4	1.59×10^4	3.48×10^2	2.90×10^3	1.12×10^4	9.71	1.07×10^1	4.27×10^3
F ₄	MEAN	5.08×10^2	9.76×10^2	5.53×10^2	5.40×10^3	5.18×10^2	5.12×10^2	6.51×10^3	5.55×10^2	5.11×10^2	4.70×10^2
	STD	5.36×10^1	2.29×10^2	1.96×10^3	1.96×10^3	5.42×10^1	5.02×10^1	1.30×10^3	3.19×10^2	5.93×10^1	3.93×10^1
F ₅	MEAN	7.18×10^2	7.02×10^2	7.57×10^2	1.08×10^3	7.58×10^2	6.91×10^2	1.05×10^3	7.39×10^2	8.13×10^2	7.13×10^2
	STD	3.50×10^1	3.22×10^1	3.41×10^1	3.41×10^1	3.69×10^1	4.77×10^1	3.15×10^1	5.40×10^1	5.84×10^1	2.72×10^1
F ₆	MEAN	6.03×10^2	6.13×10^2	6.44×10^2	6.85×10^2	6.00×10^2	6.01×10^2	6.68×10^2	6.13×10^2	6.41×10^2	6.00×10^2
	STD	2.24	4.96	7.88	7.88	6.95×10^{-3}	9.21×10^{-1}	4.78	5.90	6.61	1.46×10^{-2}
F ₇	MEAN	1.04×10^3	1.02×10^3	1.04×10^3	1.69×10^3	1.00×10^3	1.01×10^3	1.60×10^3	1.05×10^3	1.30×10^3	1.01×10^3
	STD	7.70×10^1	5.70×10^1	1.33×10^2	1.33×10^2	3.42×10^1	5.24×10^1	6.80×10^1	6.34×10^1	9.82×10^1	2.99×10^1

Table 2. Cont.

No.	Algorithm	QIO	GWO	SSA	NRO	BSLO	BSA	SCA	SFOA	DO	AOBSA
F ₈	MEAN	1.02×10^3	1.01×10^3	1.06×10^3	1.40×10^3	1.04×10^3	9.75×10^2	1.35×10^3	1.05×10^3	1.12×10^3	1.01×10^3
	STD	4.06×10^1	3.60×10^1	5.13×10^1	5.13×10^1	4.85×10^1	3.31×10^1	2.84×10^1	5.22×10^1	5.36×10^1	2.57×10^1
F ₉	MEAN	4.40×10^3	6.13×10^3	9.66×10^3	2.37×10^4	1.02×10^4	2.55×10^3	2.21×10^4	7.73×10^3	1.23×10^4	1.22×10^3
	STD	1.35×10^3	2.52×10^3	4.06×10^3	4.06×10^3	2.82×10^3	6.35×10^2	3.19×10^3	6.69×10^3	3.09×10^3	3.42×10^2
F ₁₀	MEAN	6.70×10^3	6.69×10^3	7.46×10^3	1.30×10^4	6.14×10^3	1.17×10^4	1.44×10^4	7.64×10^3	7.35×10^3	8.31×10^3
	STD	8.14×10^2	1.58×10^3	9.33×10^2	9.33×10^2	6.07×10^2	1.79×10^3	3.88×10^2	7.62×10^2	8.22×10^2	8.07×10^2
F ₁₁	MEAN	1.23×10^3	3.88×10^3	1.36×10^3	6.21×10^3	1.32×10^3	1.22×10^3	6.01×10^3	1.36×10^3	1.26×10^3	1.24×10^3
	STD	3.58×10^1	1.32×10^3	1.68×10^3	1.68×10^3	5.78×10^1	3.86×10^1	8.72×10^2	9.00×10^1	3.62×10^1	3.99×10^1
F ₁₂	MEAN	6.59×10^5	2.96×10^8	1.05×10^7	7.94×10^9	6.95×10^6	3.94×10^5	1.05×10^{10}	1.80×10^5	9.28×10^6	9.34×10^4
	STD	4.23×10^5	3.97×10^8	3.23×10^9	3.23×10^9	3.61×10^6	2.51×10^5	2.03×10^9	1.69×10^5	3.22×10^6	4.71×10^4
F ₁₃	MEAN	4.64×10^3	7.70×10^7	1.32×10^5	1.59×10^9	1.97×10^4	3.26×10^3	2.42×10^9	8.53×10^3	7.93×10^4	5.39×10^3
	STD	4.05×10^3	1.06×10^8	1.10×10^9	1.10×10^9	1.34×10^4	1.92×10^3	7.27×10^8	6.83×10^3	3.62×10^4	4.10×10^3
F ₁₄	MEAN	6.07×10^3	4.95×10^5	4.13×10^4	7.74×10^5	9.91×10^4	2.67×10^3	2.02×10^6	1.81×10^3	6.60×10^4	1.50×10^3
	STD	3.65×10^3	4.87×10^5	4.31×10^5	4.31×10^5	5.60×10^4	1.59×10^3	6.56×10^5	1.28×10^2	4.90×10^4	2.36×10^1
F ₁₅	MEAN	5.23×10^3	8.50×10^6	5.45×10^4	9.31×10^7	1.51×10^4	7.53×10^3	3.50×10^8	2.12×10^3	3.79×10^4	1.68×10^3
	STD	4.09×10^3	1.92×10^7	7.68×10^7	7.68×10^7	8.56×10^3	5.89×10^3	1.21×10^8	3.00×10^2	1.74×10^4	9.90×10^1
F ₁₆	MEAN	3.03×10^3	2.94×10^3	3.22×10^3	5.20×10^3	3.85×10^3	2.71×10^3	5.46×10^3	2.89×10^3	3.74×10^3	2.90×10^3
	STD	4.31×10^2	3.21×10^2	4.71×10^2	4.71×10^2	4.08×10^2	3.12×10^2	2.87×10^2	2.64×10^2	4.42×10^2	2.51×10^2
F ₁₇	MEAN	2.86×10^3	2.74×10^3	3.11×10^3	4.02×10^3	3.19×10^3	2.52×10^3	4.16×10^3	2.62×10^3	3.19×10^3	2.60×10^3
	STD	3.17×10^2	2.54×10^2	3.44×10^2	3.44×10^2	3.16×10^2	1.66×10^2	3.02×10^2	2.98×10^2	3.51×10^2	1.26×10^2
F ₁₈	MEAN	4.35×10^4	2.70×10^6	2.93×10^5	6.00×10^6	8.00×10^5	3.86×10^4	1.29×10^7	2.39×10^3	5.18×10^5	5.77×10^3
	STD	2.19×10^4	2.17×10^6	5.22×10^6	5.22×10^6	5.05×10^5	3.09×10^4	6.38×10^6	4.37×10^2	3.10×10^5	3.67×10^3
F ₁₉	MEAN	1.67×10^4	8.84×10^5	6.17×10^5	3.03×10^7	1.97×10^4	1.19×10^4	2.52×10^8	3.36×10^3	5.98×10^4	1.96×10^3
	STD	6.51×10^3	9.97×10^5	3.29×10^7	3.29×10^7	1.74×10^4	7.42×10^3	9.64×10^7	6.34×10^3	2.10×10^4	1.77×10^1
F ₂₀	MEAN	2.73×10^3	2.68×10^3	2.96×10^3	3.55×10^3	3.14×10^3	2.93×10^3	3.86×10^3	2.66×10^3	3.23×10^3	2.78×10^3
	STD	2.75×10^2	1.85×10^2	3.31×10^2	3.31×10^2	3.20×10^2	4.42×10^2	1.65×10^2	2.52×10^2	3.17×10^2	1.78×10^2

Table 2. Cont.

No.	Algorithm	QIO	GWO	SSA	NRO	BSLO	BSA	SCA	SFOA	DO	AOBSA
F ₂₁	MEAN	2.49×10^3	2.51×10^3	2.53×10^3	2.88×10^3	2.58×10^3	2.44×10^3	2.86×10^3	2.54×10^3	2.63×10^3	2.49×10^3
	STD	3.00×10^1	6.84×10^1	4.67×10^1	4.67×10^1	3.53×10^1	2.41×10^1	3.04×10^1	5.39×10^1	6.04×10^1	2.46×10^1
F ₂₂	MEAN	7.83×10^3	8.37×10^3	8.53×10^3	1.47×10^4	8.41×10^3	1.35×10^4	1.59×10^4	7.28×10^3	9.30×10^3	1.03×10^4
	STD	2.61×10^3	1.52×10^3	7.05×10^2	7.05×10^2	8.15×10^2	2.69×10^3	4.85×10^2	2.63×10^3	9.87×10^2	8.82×10^2
F ₂₃	MEAN	2.99×10^3	2.94×10^3	2.97×10^3	3.45×10^3	3.03×10^3	2.89×10^3	3.49×10^3	3.04×10^3	3.22×10^3	2.92×10^3
	STD	5.37×10^1	5.68×10^1	8.36×10^1	8.36×10^1	5.66×10^1	4.11×10^1	5.31×10^1	7.23×10^1	9.29×10^1	2.86×10^1
F ₂₄	MEAN	3.18×10^3	3.14×10^3	3.10×10^3	3.61×10^3	3.38×10^3	3.07×10^3	3.67×10^3	3.18×10^3	3.44×10^3	3.08×10^3
	STD	6.49×10^1	1.03×10^2	7.01×10^1	7.01×10^1	1.31×10^2	5.64×10^1	5.32×10^1	8.51×10^1	9.36×10^1	3.26×10^1
F ₂₅	MEAN	3.06×10^3	3.53×10^3	3.03×10^3	5.66×10^3	3.05×10^3	3.08×10^3	6.16×10^3	3.02×10^3	3.06×10^3	3.03×10^3
	STD	4.03×10^1	2.03×10^2	6.78×10^2	6.78×10^2	3.39×10^1	2.70×10^1	6.00×10^2	4.20×10^1	4.55×10^1	4.13×10^1
F ₂₆	MEAN	7.00×10^3	6.25×10^3	4.86×10^3	1.16×10^4	5.06×10^3	6.27×10^3	1.15×10^4	5.89×10^3	9.01×10^3	5.80×10^3
	STD	2.43×10^3	6.25×10^2	1.76×10^3	1.76×10^3	2.74×10^3	7.20×10^2	5.69×10^2	1.92×10^3	8.28×10^2	2.64×10^2
F ₂₇	MEAN	3.53×10^3	3.53×10^3	3.41×10^3	4.13×10^3	3.41×10^3	3.68×10^3	4.33×10^3	3.50×10^3	3.70×10^3	3.39×10^3
	STD	1.05×10^2	7.24×10^1	2.43×10^2	2.43×10^2	1.03×10^2	1.82×10^2	1.44×10^2	1.80×10^2	1.21×10^2	1.13×10^2
F ₂₈	MEAN	3.30×10^3	4.19×10^3	3.30×10^3	6.14×10^3	3.30×10^3	4.75×10^3	6.41×10^3	3.99×10^3	3.32×10^3	3.42×10^3
	STD	1.43×10^1	3.73×10^2	5.53×10^2	5.53×10^2	1.83×10^1	2.72×10^3	3.75×10^2	1.84×10^3	3.04×10^1	6.38×10^2
F ₂₉	MEAN	4.23×10^3	4.27×10^3	4.76×10^3	6.74×10^3	4.20×10^3	4.18×10^3	7.11×10^3	4.11×10^3	4.57×10^3	3.99×10^3
	STD	3.29×10^2	2.75×10^2	5.51×10^2	5.51×10^2	3.03×10^2	8.58×10^2	5.71×10^2	3.93×10^2	3.99×10^2	1.55×10^2
F ₃₀	MEAN	8.97×10^5	8.65×10^7	2.21×10^7	2.96×10^8	1.43×10^6	7.07×10^5	5.83×10^8	1.28×10^6	3.12×10^6	6.46×10^5
	STD	1.32×10^5	3.29×10^7	1.24×10^8	1.24×10^8	5.05×10^5	1.06×10^5	1.47×10^8	7.00×10^5	8.94×10^5	6.54×10^4

Tables 3 and 4 display the Friedman average ranking results produced by the AOBSA and the compared algorithms in solving the CEC 2017 test suite with 30 and 50 dimensions, respectively. In the two tables, MEAN and STD denote the mean solution and standard deviation from 30 independent repeated experiments, respectively. Looking at Table 3, the AOBSA obtained the best ranking in $F_1, F_4, F_5, F_6, F_7, F_9, F_{13}, F_{14}, F_{15}, F_{16}, F_{18}, F_{19}, F_{23}, F_{24}, F_{25}, F_{26}, F_{27}, F_{28}, F_{29}$, and F_{30} . QIO was the best of all algorithms in F_2, F_{11} , and F_{22} . BSLO obtained the best result in F_{10} . The BSA showed better performance than other algorithms in F_8, F_{17} , and F_{21} . The SFOA outperformed other algorithms in F_3, F_4, F_{12} , and F_{20} . GWO, SSA, NRO, SCA, and DO did not achieve the best ranking in any test functions. According to the average ranking presented in the last line, all algorithms can be ranked from best to worst as follows: AOBSA, SFOA, BSA, QIO, BSLO, SSA, GWO, DO, NRO, and SCA. Clearly, the AOBSA is the best of all algorithms. From Table 4, AOBSA obtained the best results in 11 test functions, i.e., $F_1, F_4, F_6, F_9, F_{12}, F_{14}, F_{15}, F_{19}, F_{27}, F_{29}$, and F_{30} . In addition, the AOBSA achieved the second ranking in six test functions, i.e., $F_5, F_{16}, F_{17}, F_{23}, F_{24}$, and F_{25} . This indicates the excellent global search capability of the AOBSA. Moreover, the BSA also shows excellent performance, beating the other algorithms in nine test functions, i.e., $F_5, F_8, F_{11}, F_{13}, F_{16}, F_{17}, F_{21}, F_{23}$, and F_{24} . QIO, SSA, BSLO, and SFOA show the best performance in one test function (i.e., F_2), two test functions (i.e., F_3 and F_{26}), two test functions (i.e., F_7 and F_{10}), and five test functions (i.e., $F_{18}, F_{20}, F_{22}, F_{25}$, and F_{28}), respectively. GWO, NRO, SCA, and DO did not get the best results in any test functions. From the last line of Table 5, all algorithms can be ranked from best to worst as follows: AOBSA, BSA, SFOA, QIO, BSLO, SSA, GWO, DO, NRO, and SCA. Obviously, the AOBSA is the best of all algorithms. According to the Friedman average ranking results, the AOBSA demonstrates better performance than the compared algorithms in solving the CEC 2017 test suite with 30 and 50 dimensions.

Table 3. The experimental results produced by the average ranking of the applied algorithms in the CEC 2017 test suite with 30 dimensions.

No.	QIO	GWO	SSA	NRO	BSLO	BSA	SCA	SFOA	DO	AOBSA
F_1	4.83	7.97	5.47	9.40	5.67	3.00	9.53	1.70	5.53	1.90
F_2	1.97	8.03	3.03	9.23	4.73	6.27	9.53	3.00	4.53	4.67
F_3	3.03	9.10	2.00	8.90	6.90	5.73	9.00	1.00	5.03	4.30
F_4	3.80	7.83	5.37	9.43	5.47	4.77	9.57	1.85	5.07	1.85
F_5	4.97	3.50	5.47	9.77	5.40	2.67	9.23	4.90	7.20	1.90
F_6	4.97	5.33	7.37	9.97	2.00	3.20	8.97	4.57	7.63	1.00
F_7	5.53	4.90	4.67	9.60	3.93	2.30	9.27	4.70	7.83	2.27
F_8	4.53	4.20	5.43	9.73	5.80	1.87	9.23	5.03	6.77	2.40
F_9	5.00	4.13	5.80	9.87	5.83	2.43	8.53	3.97	8.27	1.17
F_{10}	3.33	2.97	5.67	8.90	2.60	8.30	9.60	3.63	5.00	5.00
F_{11}	2.67	8.60	5.70	9.17	4.03	2.83	9.10	5.47	4.53	2.90
F_{12}	2.90	7.93	6.07	9.40	5.60	2.67	9.60	1.70	6.40	2.73
F_{13}	3.70	7.07	6.43	9.17	4.90	3.77	9.83	2.13	6.80	1.20
F_{14}	3.73	8.03	5.27	7.83	7.60	2.03	9.67	3.30	6.37	1.17
F_{15}	3.37	7.40	7.80	8.07	4.87	3.83	9.97	2.43	6.23	1.03
F_{16}	5.27	3.50	5.67	9.47	6.57	3.87	9.50	3.23	5.40	2.53
F_{17}	5.83	3.27	4.37	8.97	7.50	2.43	8.70	4.70	6.33	2.90
F_{18}	3.63	7.87	6.10	8.53	6.63	3.33	9.67	1.67	6.00	1.57
F_{19}	4.37	7.47	7.53	8.93	4.90	2.83	9.90	2.47	5.60	1.00
F_{20}	3.70	4.57	5.30	8.43	5.93	6.87	7.40	2.67	6.87	3.27
F_{21}	4.17	3.83	5.27	9.67	6.50	1.63	9.23	5.33	7.37	2.00
F_{22}	2.03	5.00	5.43	5.63	4.80	7.42	8.80	3.30	6.83	5.75
F_{23}	5.63	3.90	4.23	9.63	5.07	2.77	9.37	5.03	7.63	1.73
F_{24}	5.40	3.30	3.10	9.23	7.37	3.43	9.13	4.60	7.77	1.67

Table 3. Cont.

No.	QIO	GWO	SSA	NRO	BSLO	BSA	SCA	SFOA	DO	AOBSA
F ₂₅	4.57	7.97	4.47	9.73	3.97	4.90	9.27	4.00	3.80	2.33
F ₂₆	5.47	3.03	4.00	9.10	5.10	3.83	9.40	5.27	7.17	2.63
F ₂₇	5.50	5.57	3.73	9.10	2.67	6.27	9.40	4.83	6.10	1.83
F ₂₈	3.47	7.87	4.53	9.57	5.03	5.80	9.17	2.83	4.10	2.63
F ₂₉	4.23	3.93	6.00	9.03	4.33	4.57	9.27	4.40	6.50	2.73
F ₃₀	3.57	7.97	7.03	9.03	4.77	2.77	9.97	2.67	6.00	1.23
Avg.	4.17	5.87	5.28	9.08	5.22	3.95	9.29	3.55	6.22	2.38

Table 4. The experimental results produced by the average ranking of the applied algorithms in the CEC 2017 test suite with 50 dimensions.

No.	QIO	GWO	SSA	NRO	BSLO	BSA	SCA	SFOA	DO	AOBSA
F ₁	4.07	8.00	4.60	9.40	5.10	3.10	9.60	3.13	6.27	1.73
F ₂	1.17	8.00	2.60	9.07	4.20	5.77	9.93	6.33	2.67	5.27
F ₃	4.77	8.20	1.00	9.30	4.23	6.23	9.50	2.03	2.97	6.77
F ₄	3.90	7.93	5.50	9.33	4.13	4.30	9.67	3.33	4.23	2.67
F ₅	3.70	2.87	5.37	9.73	5.63	2.70	9.27	4.83	7.27	3.63
F ₆	3.97	5.47	7.63	9.93	1.97	3.13	9.03	5.43	7.40	1.03
F ₇	4.40	3.63	4.97	9.67	2.97	3.60	9.33	4.83	7.87	3.73
F ₈	4.03	3.33	5.60	9.87	4.93	1.83	9.13	4.97	7.40	3.90
F ₉	3.53	4.50	6.33	9.60	6.30	2.23	9.20	4.77	7.47	1.07
F ₁₀	3.10	2.70	4.60	8.83	2.20	8.17	9.83	5.00	4.30	6.27
F ₁₁	2.43	8.30	5.97	9.27	5.40	2.00	9.43	5.83	3.63	2.73
F ₁₂	3.50	8.00	6.27	9.33	5.73	3.10	9.67	1.97	6.00	1.43
F ₁₃	2.30	7.67	6.97	9.20	4.33	1.93	9.80	3.67	6.27	2.87
F ₁₄	3.93	7.93	5.43	8.87	6.70	2.90	9.93	2.30	6.00	1.00
F ₁₅	3.30	7.37	6.90	8.97	4.63	3.77	10.00	2.40	6.43	1.23
F ₁₆	4.30	3.87	5.03	9.27	7.23	2.17	9.73	3.27	6.77	3.37
F ₁₇	4.80	3.83	6.10	9.20	6.50	2.17	9.60	3.30	6.60	2.90
F ₁₈	3.70	7.93	5.43	9.03	6.80	3.27	9.70	1.00	6.10	2.03
F ₁₉	4.23	7.53	7.47	9.03	3.90	3.73	9.97	2.20	5.93	1.00
F ₂₀	3.53	3.20	5.30	8.50	6.23	5.23	9.77	2.70	6.80	3.73
F ₂₁	3.30	3.80	4.80	9.63	6.33	1.27	9.37	5.27	7.50	3.73
F ₂₂	3.83	3.13	4.07	8.80	3.27	8.07	9.77	2.97	4.77	6.33
F ₂₃	4.93	3.17	4.00	9.27	5.77	1.93	9.73	5.80	7.93	2.47
F ₂₄	5.00	3.63	2.97	9.07	7.07	2.17	9.73	4.97	7.80	2.60
F ₂₅	4.40	8.00	3.30	9.23	4.03	5.90	9.77	2.43	4.57	3.37
F ₂₆	5.53	4.27	3.17	9.27	3.90	4.13	9.40	4.20	7.90	3.23
F ₂₇	5.13	5.23	2.67	9.13	3.00	6.60	9.73	4.07	7.07	2.37
F ₂₈	3.63	7.60	3.80	8.97	3.50	6.60	9.27	3.40	4.63	3.60
F ₂₉	4.33	4.73	6.87	9.23	4.17	3.27	9.70	3.53	6.30	2.87
F ₃₀	3.23	8.00	7.00	9.07	4.50	1.87	9.93	4.07	5.93	1.40
Avg.	3.87	5.73	5.06	9.24	4.82	3.77	9.62	3.80	6.09	3.01

Tables 5 and 6 show the Wilcoxon signed-rank test results produced by the AOBSA and the compared algorithms in solving the CEC 2017 test suite with 30 and 50 dimensions, respectively. The significance level is set to 0.05. In the two tables, “R(+)” is the sum of the positive rank, “R(−)” is the sum of the negative rank, “S” is the obtained symbol, and “p-value” is the test P value. When “S” is marked by “+”, the AOBSA outperforms the compared algorithm in this test function; when “S” is marked by “−”, the AOBSA cannot compete with the compared algorithm in this test function; when “S” is marked by “=”, the AOBSA has the same performance as the compared algorithm in this test function. Looking

at Table 5, QIO outperforms the AOBSA in four test functions (i.e., F_2 , F_3 , F_{10} , and F_{22}) and has the same performance as the AOBSA in four test functions (i.e., F_{11} , F_{12} , F_{20} , and F_{28}), while it cannot compete with the AOBSA in the remaining 22 test functions. GWO is superior to the AOBSA in F_{10} and F_{22} , while it is inferior to the AOBSA in more than 80% of test functions. The SSA can beat the AOBSA in F_2 and F_3 , while it is inferior to the AOBSA in more than 85% of test functions. NRO can only get the same performance as AOBSA in F_{22} , and it has no advantage compared to AOBSA in all 29 test functions. BSLO outperforms the AOBSA in three test functions (i.e., F_2 , F_{10} , and F_{22}) but is inferior to the AOBSA in the remaining 27 test functions. The BSA can only beat the AOBSA in F_8 and F_{21} , but it is inferior to the AOBSA in more than three-quarters of the test functions. Compared to the AOBSA, the SCA has no advantage in any test functions. The SFOA can outperform or have the same performance as the AOBSA in ten test functions, but it is inferior to the AOBSA in two-thirds of the test functions. DO can only beat the AOBSA in F_2 , while the AOBSA outperforms DO in more than 85% of test functions. According to Table 6, QIO is superior to the AOBSA in F_2 , F_3 , F_{10} , and F_{22} , while it cannot compete with the AOBSA in more than half of test functions. GWO and SSA can beat or have the same performance as the AOBSA in nine test functions, while they are inferior to the AOBSA in 70% of test functions. NRO and SCA cannot beat the AOBSA in any test functions. Although BSLO and DO can obtain better performance than the AOBSA in F_2 , F_3 , F_{10} , and F_{22} , they are inferior to the AOBSA in 22 test functions and 24 test functions, respectively. The BSA shows excellent performance, able to beat the AOBSA in eight test functions, i.e., F_3 , F_5 , F_8 , F_{13} , F_{16} , F_{17} , F_{21} , and F_{23} . However, the AOBSA can still outperform the BSA in half of the test functions. DO is not inferior to the AOBSA in 40% of test functions, but it is inferior to the AOBSA in 60% of its functions. As discussed above, when solving the CEC 2017 test suite with 30 and 50 dimensions, the AOBSA outperforms the compared algorithms in at least half of the test functions.

Table 5. The Wilcoxon signed-rank test results of the applied algorithms in the CEC 2017 test suite with 30 dimensions.

No.	AOBSA vs.																						
	QIO				GWO				SSA				NRO				BSLO						
	R(+)	R(−)	p-Value	S	R(+)	R(−)	p-Value	S	R(+)	R(−)	p-Value	S	R(+)	R(−)	p-Value	S	R(+)	R(−)	p-Value	S			
F ₁	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₂	36	429	5.31×10^{-5}	−	465	0	1.73×10^{-6}	+	51	414	1.89×10^{-4}	−	465	0	1.73×10^{-6}	+	122	343	2.30×10^{-2}	−			
F ₃	1	464	1.92×10^{-6}	−	465	0	1.73×10^{-6}	+	0	465	1.73×10^{-6}	−	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₄	427	38	6.32×10^{-5}	+	465	0	1.73×10^{-6}	+	453	12	5.75×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₅	446	19	1.13×10^{-5}	+	395	70	8.31×10^{-4}	+	454	11	5.22×10^{-6}	+	465	0	1.73×10^{-6}	+	462	3	2.35×10^{-6}	+			
F ₆	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₇	462	3	2.35×10^{-6}	+	428	37	5.79×10^{-5}	+	448	17	9.32×10^{-6}	+	465	0	1.73×10^{-6}	+	413	52	2.05×10^{-4}	+			
F ₈	416	49	1.60×10^{-4}	+	397	68	7.16×10^{-4}	+	439	26	2.16×10^{-5}	+	465	0	1.73×10^{-6}	+	458	7	3.52×10^{-6}	+			
F ₉	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	464	1	1.92×10^{-6}	+			
F ₁₀	120	345	2.07×10^{-2}	−	62	403	4.53×10^{-4}	−	313	152	9.78×10^{-2}	=	465	0	1.73×10^{-6}	+	16	449	8.47×10^{-6}	−			
F ₁₁	239	226	8.94×10^{-1}	=	465	0	1.73×10^{-6}	+	461	4	2.60×10^{-6}	+	465	0	1.73×10^{-6}	+	372	93	4.11×10^{-3}	+			
F ₁₂	274	191	3.93×10^{-1}	=	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₁₃	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	464	1	1.92×10^{-6}	+			
F ₁₄	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₁₅	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₁₆	440	25	1.97×10^{-5}	+	325	140	5.71×10^{-2}	=	448	17	9.32×10^{-6}	+	465	0	1.73×10^{-6}	+	462	3	2.35×10^{-6}	+			
F ₁₇	435	30	3.11×10^{-5}	+	301	164	1.59×10^{-1}	=	395	70	8.31×10^{-4}	+	465	0	1.73×10^{-6}	+	460	5	2.88×10^{-6}	+			
F ₁₈	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₁₉	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₂₀	271	194	4.28×10^{-1}	=	347	118	1.85×10^{-2}	+	376	89	3.16×10^{-3}	+	455	10	4.73×10^{-6}	+	393	72	9.63×10^{-4}	+			
F ₂₁	435	30	3.11×10^{-5}	+	445	20	1.24×10^{-5}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₂₂	23	442	1.64×10^{-5}	−	108	357	1.04×10^{-2}	−	182	283	2.99×10^{-1}	=	222	243	8.29×10^{-1}	=	115	350	1.57×10^{-2}	−			
F ₂₃	455	10	4.73×10^{-6}	+	458	7	3.52×10^{-6}	+	452	13	6.34×10^{-6}	+	465	0	1.73×10^{-6}	+	464	1	1.92×10^{-6}	+			
F ₂₄	465	0	1.73×10^{-6}	+	395	70	8.31×10^{-4}	+	416	49	1.60×10^{-4}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₂₅	378	87	2.77×10^{-3}	+	465	0	1.73×10^{-6}	+	423	42	8.92×10^{-5}	+	465	0	1.73×10^{-6}	+	359	106	9.27×10^{-3}	+			
F ₂₆	339	126	2.85×10^{-2}	+	289	176	2.45×10^{-1}	=	331	134	4.28×10^{-2}	+	465	0	1.73×10^{-6}	+	351	114	1.48×10^{-2}	+			
F ₂₇	457	8	3.88×10^{-6}	+	462	3	2.35×10^{-6}	+	444	21	1.36×10^{-5}	+	465	0	1.73×10^{-6}	+	343	122	2.30×10^{-2}	+			
F ₂₈	278	187	3.49×10^{-1}	=	465	0	1.73×10^{-6}	+	377	88	2.96×10^{-3}	+	465	0	1.73×10^{-6}	+	423	42	8.92×10^{-5}	+			
F ₂₉	357	108	1.04×10^{-2}	+	346	119	1.96×10^{-2}	+	440	25	1.97×10^{-5}	+	465	0	1.73×10^{-6}	+	374	91	3.61×10^{-3}	+			
F ₃₀	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
+ / = / −				22/4/4				25/3/2				26/2/2				29/1/0				27/0/3			

Table 5. Cont.

No.	AOBSA vs.																
	BSA				SCA				SFOA				DO				
	R(+)	R(−)	<i>p</i> -Value	S	R(+)	R(−)	<i>p</i> -Value	S	R(+)	R(−)	<i>p</i> -Value	S	R(+)	R(−)	<i>p</i> -Value	S	
F ₁	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	99	366	3.89×10^{-3}	+	465	0	1.73×10^{-6}	+	
F ₂	322	143	6.56×10^{-2}	=	465	0	1.73×10^{-6}	+	137	328	4.95×10^{-2}	−	97	368	5.32×10^{-3}	−	
F ₃	441	24	1.80×10^{-5}	+	465	0	1.73×10^{-6}	+	0	465	1.73×10^{-6}	−	390	75	1.20×10^{-3}	+	
F ₄	454	11	5.22×10^{-6}	+	465	0	1.73×10^{-6}	+	309	156	9.81×10^{-2}	=	454	11	5.22×10^{-6}	+	
F ₅	333	132	3.87×10^{-2}	+	465	0	1.73×10^{-6}	+	462	3	2.35×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₆	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₇	247	218	7.66×10^{-1}	=	465	0	1.73×10^{-6}	+	427	38	6.32×10^{-5}	+	465	0	1.73×10^{-6}	+	
F ₈	121	344	2.18×10^{-2}	−	465	0	1.73×10^{-6}	+	431	34	4.45×10^{-5}	+	465	0	1.73×10^{-6}	+	
F ₉	440	25	1.97×10^{-5}	+	465	0	1.73×10^{-6}	+	455	10	4.73×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₁₀	460	5	2.88×10^{-6}	+	465	0	1.73×10^{-6}	+	108	357	1.04×10^{-2}	−	261	204	5.58×10^{-1}	=	
F ₁₁	232	233	9.92×10^{-1}	=	465	0	1.73×10^{-6}	+	436	29	2.84×10^{-5}	+	404	61	4.20×10^{-4}	+	
F ₁₂	217	248	7.50×10^{-1}	=	465	0	1.73×10^{-6}	+	149	316	8.59×10^{-2}	=	465	0	1.73×10^{-6}	+	
F ₁₃	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	424	41	8.19×10^{-5}	+	465	0	1.73×10^{-6}	+	
F ₁₄	433	32	3.72×10^{-5}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₁₅	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	464	1	1.92×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₁₆	389	76	1.29×10^{-3}	+	465	0	1.73×10^{-6}	+	293	172	2.13×10^{-1}	=	445	20	1.24×10^{-5}	+	
F ₁₇	216	249	7.34×10^{-1}	=	465	0	1.73×10^{-6}	+	424	41	8.19×10^{-5}	+	461	4	2.60×10^{-6}	+	
F ₁₈	457	8	3.88×10^{-6}	+	465	0	1.73×10^{-6}	+	253	212	6.73×10^{-1}	=	465	0	1.73×10^{-6}	+	
F ₁₉	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₂₀	424	41	8.19×10^{-5}	+	453	12	5.75×10^{-6}	+	202	263	5.30×10^{-1}	=	435	30	3.11×10^{-5}	+	
F ₂₁	128	337	3.16×10^{-2}	−	465	0	1.73×10^{-6}	+	464	1	1.92×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₂₂	350	115	1.11×10^{-2}	+	432	33	4.07×10^{-5}	+	83.5	381.5	6.91×10^{-4}	−	271	194	4.28×10^{-1}	=	
F ₂₃	332	133	4.07×10^{-2}	+	465	0	1.73×10^{-6}	+	462	3	2.35×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₂₄	425	40	7.51×10^{-5}	+	465	0	1.73×10^{-6}	+	450	15	7.69×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₂₅	423	42	8.92×10^{-5}	+	465	0	1.73×10^{-6}	+	364	101	6.84×10^{-3}	+	309	156	1.16×10^{-1}	=	
F ₂₆	376	89	3.16×10^{-3}	+	465	0	1.73×10^{-6}	+	337	128	2.38×10^{-2}	+	439	26	2.16×10^{-5}	+	
F ₂₇	459	6	3.18×10^{-6}	+	465	0	1.73×10^{-6}	+	430	35	4.86×10^{-5}	+	464	1	1.92×10^{-6}	+	
F ₂₈	429	36	5.31×10^{-5}	+	465	0	1.73×10^{-6}	+	240.5	224.5	8.55×10^{-1}	=	365	100	6.42×10^{-3}	+	
F ₂₉	340	125	2.70×10^{-2}	+	465	0	1.73×10^{-6}	+	382	83	2.11×10^{-3}	+	459	6	3.18×10^{-6}	+	
F ₃₀	431	34	4.45×10^{-5}	+	465	0	1.73×10^{-6}	+	423	42	8.92×10^{-5}	+	465	0	1.73×10^{-6}	+	
+ / = / −																	
				23/5/2				30/0/0				20/6/4				26/3/1	

Table 6. The Wilcoxon signed-rank test results of the applied algorithms in the CEC 2017 test suite with 50 dimensions.

No.	AOBSA vs.																						
	QIO				GWO				SSA				NRO				BSLO						
	R(+)	R(−)	p-Value	S	R(+)	R(−)	p-Value	S	R(+)	R(−)	p-Value	S	R(+)	R(−)	p-Value	S	R(+)	R(−)	p-Value	S			
F ₁	446	19	1.13×10^{-5}	+	465	0	1.73×10^{-6}	+	450	15	7.69×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₂	0	465	1.73×10^{-6}	−	465	0	1.73×10^{-6}	+	0	465	1.73×10^{-6}	−	465	0	1.73×10^{-6}	+	138	327	5.19×10^{-2}	−			
F ₃	0	465	1.73×10^{-6}	−	465	0	1.73×10^{-6}	+	0	465	1.73×10^{-6}	−	465	0	1.73×10^{-6}	+	0	465	1.73×10^{-6}	−			
F ₄	368	97	5.32×10^{-3}	+	465	0	1.73×10^{-6}	+	449	16	8.47×10^{-6}	+	465	0	1.73×10^{-6}	+	386	79	1.59×10^{-3}	+			
F ₅	260	205	5.72×10^{-1}	=	146	319	7.52×10^{-2}	=	406	59	3.59×10^{-4}	+	465	0	1.73×10^{-6}	+	426	39	6.89×10^{-5}	+			
F ₆	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	435	30	3.11×10^{-5}	+			
F ₇	280	185	3.29×10^{-1}	=	214	251	7.04×10^{-1}	=	344	121	2.18×10^{-2}	+	465	0	1.73×10^{-6}	+	157	308	1.20×10^{-1}	=			
F ₈	281	184	3.18×10^{-1}	=	198	267	4.78×10^{-1}	=	408	57	3.06×10^{-4}	+	465	0	1.73×10^{-6}	+	357	108	1.04×10^{-2}	+			
F ₉	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₁₀	1	464	1.92×10^{-6}	−	33	432	4.07×10^{-5}	−	83	382	2.11×10^{-3}	−	465	0	1.73×10^{-6}	+	1	464	1.92×10^{-6}	−			
F ₁₁	178	287	2.62×10^{-1}	=	465	0	1.73×10^{-6}	+	463	2	2.13×10^{-6}	+	465	0	1.73×10^{-6}	+	431	34	4.45×10^{-5}	+			
F ₁₂	459	6	3.18×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₁₃	183	282	3.09×10^{-1}	=	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	426	39	6.89×10^{-5}	+			
F ₁₄	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₁₅	454	11	5.22×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	464	1	1.92×10^{-6}	+			
F ₁₆	302	163	1.53×10^{-1}	=	262	203	5.44×10^{-1}	=	401	64	5.29×10^{-4}	+	465	0	1.73×10^{-6}	+	463	2	2.13×10^{-6}	+			
F ₁₇	397	68	7.16×10^{-4}	+	337	128	3.16×10^{-2}	+	449	16	8.47×10^{-6}	+	465	0	1.73×10^{-6}	+	463	2	2.13×10^{-6}	+			
F ₁₈	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₁₉	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₂₀	196	269	4.53×10^{-1}	=	135	330	4.49×10^{-2}	−	380	85	2.41×10^{-3}	+	465	0	1.73×10^{-6}	+	430	35	4.86×10^{-5}	+			
F ₂₁	204	261	5.58×10^{-1}	=	242	223	8.45×10^{-1}	=	358	107	9.84×10^{-3}	+	465	0	1.73×10^{-6}	+	464	1	1.92×10^{-6}	+			
F ₂₂	23	442	1.64×10^{-5}	−	35	430	4.86×10^{-5}	−	58	407	3.32×10^{-4}	−	465	0	1.73×10^{-6}	+	0	465	1.73×10^{-6}	−			
F ₂₃	453	12	5.75×10^{-6}	+	294	171	2.06×10^{-1}	=	398	67	6.64×10^{-4}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₂₄	456	9	4.29×10^{-6}	+	349	116	1.66×10^{-2}	+	289	176	2.45×10^{-1}	=	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+			
F ₂₅	338	127	3.00×10^{-2}	+	465	0	1.73×10^{-6}	+	229	236	9.43×10^{-1}	=	465	0	1.73×10^{-6}	+	337	128	3.16×10^{-2}	+			
F ₂₆	335	130	3.50×10^{-2}	+	388	77	1.38×10^{-3}	+	129	336	3.33×10^{-2}	−	465	0	1.73×10^{-6}	+	158	307	1.25×10^{-1}	=			
F ₂₇	430	35	4.86×10^{-5}	+	448	17	9.32×10^{-6}	+	290	175	2.37×10^{-1}	=	465	0	1.73×10^{-6}	+	286	179	2.71×10^{-1}	=			
F ₂₈	241	224	8.61×10^{-1}	=	435	30	3.11×10^{-5}	+	249	216	7.34×10^{-1}	=	464	1	1.92×10^{-6}	+	212	253	6.73×10^{-1}	=			
F ₂₉	376	89	3.16×10^{-3}	+	423	42	8.92×10^{-5}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	363	102	7.27×10^{-3}	+			
F ₃₀	463	2	2.13×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	463	2	2.13×10^{-6}	+			
+ / = / −				17/9/4				21/6/3				21/4/5				30/0/0				22/4/4			

Table 6. Cont.

No.	AOBSA vs.																
	BSA			SCA				SFOA				DO					
	R(+)	R(−)	<i>p</i> -Value	S	R(+)	R(−)	<i>p</i> -Value	S	R(+)	R(−)	<i>p</i> -Value	S	R(+)	R(−)	<i>p</i> -Value	S	
F ₁	397	68	7.16×10^{-4}	+	465	0	1.73×10^{-6}	+	396	69	7.71×10^{-4}	+	465	0	1.73×10^{-6}	+	
F ₂	292	173	2.21×10^{-1}	=	465	0	1.73×10^{-6}	+	427	38	6.32×10^{-5}	+	0	465	1.73×10^{-6}	−	
F ₃	75	390	1.20×10^{-3}	−	465	0	1.73×10^{-6}	+	0	465	1.73×10^{-6}	−	0	465	1.73×10^{-6}	−	
F ₄	379	86	2.58×10^{-3}	+	465	0	1.73×10^{-6}	+	300	165	1.65×10^{-1}	=	349	116	1.66×10^{-2}	+	
F ₅	132	333	3.87×10^{-2}	−	465	0	1.73×10^{-6}	+	338	127	3.00×10^{-2}	+	458	7	3.52×10^{-6}	+	
F ₆	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₇	188	277	3.60×10^{-1}	=	465	0	1.73×10^{-6}	+	377	88	2.96×10^{-3}	+	465	0	1.73×10^{-6}	+	
F ₈	44	421	1.06×10^{-4}	−	465	0	1.73×10^{-6}	+	369	96	4.99×10^{-3}	+	464	1	1.92×10^{-6}	+	
F ₉	462	3	2.35×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₁₀	458	7	3.52×10^{-6}	+	465	0	1.73×10^{-6}	+	107	358	9.84×10^{-3}	−	54	411	2.41×10^{-4}	−	
F ₁₁	149	316	8.59×10^{-2}	=	465	0	1.73×10^{-6}	+	463	2	2.13×10^{-6}	+	312	153	1.02×10^{-1}	=	
F ₁₂	465	0	1.73×10^{-6}	=	465	0	1.73×10^{-6}	+	358	107	9.84×10^{-3}	+	465	0	1.73×10^{-6}	+	
F ₁₃	116	349	1.66×10^{-2}	−	465	0	1.73×10^{-6}	+	328	137	4.95×10^{-2}	+	465	0	1.73×10^{-6}	+	
F ₁₄	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₁₅	460	5	2.88×10^{-6}	+	465	0	1.73×10^{-6}	+	452	13	6.34×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₁₆	110	355	1.17×10^{-2}	−	465	0	1.73×10^{-6}	+	223	242	8.45×10^{-1}	=	458	7	3.52×10^{-6}	+	
F ₁₇	126	339	2.85×10^{-2}	−	465	0	1.73×10^{-6}	+	242	223	8.45×10^{-1}	=	458	7	3.52×10^{-6}	+	
F ₁₈	456	9	4.29×10^{-6}	+	465	0	1.73×10^{-6}	+	0	465	1.73×10^{-6}	−	465	0	1.73×10^{-6}	+	
F ₁₉	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₂₀	309	156	1.16×10^{-1}	=	465	0	1.73×10^{-6}	+	129	336	3.33×10^{-2}	−	445	20	1.24×10^{-5}	+	
F ₂₁	10	455	4.73×10^{-6}	−	465	0	1.73×10^{-6}	+	396	69	7.71×10^{-4}	+	465	0	1.73×10^{-6}	+	
F ₂₂	430	35	4.86×10^{-5}	+	465	0	1.73×10^{-6}	+	2	463	2.13×10^{-6}	−	66	399	6.16×10^{-4}	−	
F ₂₃	115	350	1.57×10^{-2}	−	465	0	1.73×10^{-6}	+	452	13	6.34×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₂₄	170	295	1.99×10^{-1}	=	465	0	1.73×10^{-6}	+	448	17	9.32×10^{-6}	+	465	0	1.73×10^{-6}	+	
F ₂₅	434	31	3.41×10^{-5}	+	465	0	1.73×10^{-6}	+	157	308	1.20×10^{-1}	=	343	122	2.30×10^{-2}	+	
F ₂₆	375	90	3.38×10^{-3}	+	465	0	1.73×10^{-6}	+	253	212	6.73×10^{-1}	=	465	0	1.73×10^{-6}	+	
F ₂₇	455	10	4.73×10^{-6}	+	465	0	1.73×10^{-6}	+	376	89	3.16×10^{-3}	+	459	6	3.18×10^{-6}	+	
F ₂₈	396	69	7.71×10^{-4}	+	464	1	1.92×10^{-6}	+	211	254	6.58×10^{-1}	=	316	149	8.59×10^{-2}	=	
F ₂₉	221	244	8.13×10^{-1}	=	465	0	1.73×10^{-6}	+	294	171	2.06×10^{-1}	=	446	19	1.13×10^{-5}	+	
F ₃₀	339	126	2.85×10^{-2}	+	465	0	1.73×10^{-6}	+	464	1	1.92×10^{-6}	+	465	0	1.73×10^{-6}	+	
+ / = / −				15/7/8				30/0/0				18/7/5				24/2/4	

Figures 1 and 2 depict the convergence curves of the AOBSA and comparative algorithms in some typical test functions. The characteristics of these curves can be summarized as follows: (1) for most convergence curves, the AOBSA can converge to a better solution with fewer iterations compared to other algorithms; (2) for a few convergence curves, although the AOBSA has a slower convergence speed in the early stage, it can still search for better solutions than other algorithms as the number of iterations increases; and (3) for a few convergence curves, the convergence curves of all algorithms are relatively similar, but the solutions found by the AOBSA are still better than those found by the compared algorithms. Therefore, in terms of the characteristics of these convergence curves, the AOBSA has excellent global search ability and the ability to escape from local optima, meaning that it still has the potential to search for satisfactory solutions when solving complex optimization problems.

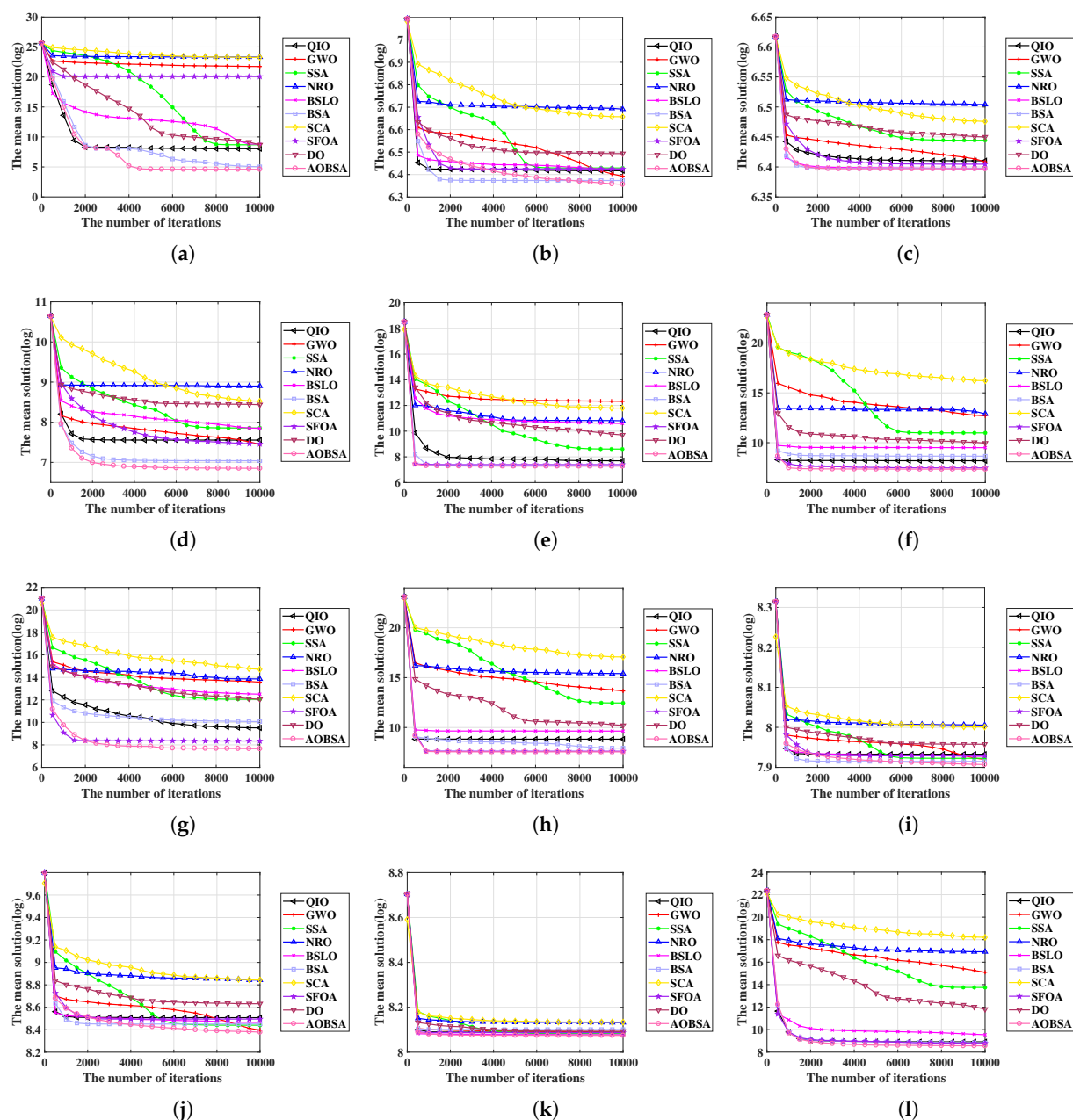


Figure 1. Some typical convergence curves produced by the applied algorithms in the CEC 2017 test suite with 30 dimensions: (a) F_1 ; (b) F_5 ; (c) F_6 ; (d) F_9 ; (e) F_{14} ; (f) F_{15} ; (g) F_{18} ; (h) F_{19} ; (i) F_{23} ; (j) F_{26} ; (k) F_{27} ; (l) F_{30} .

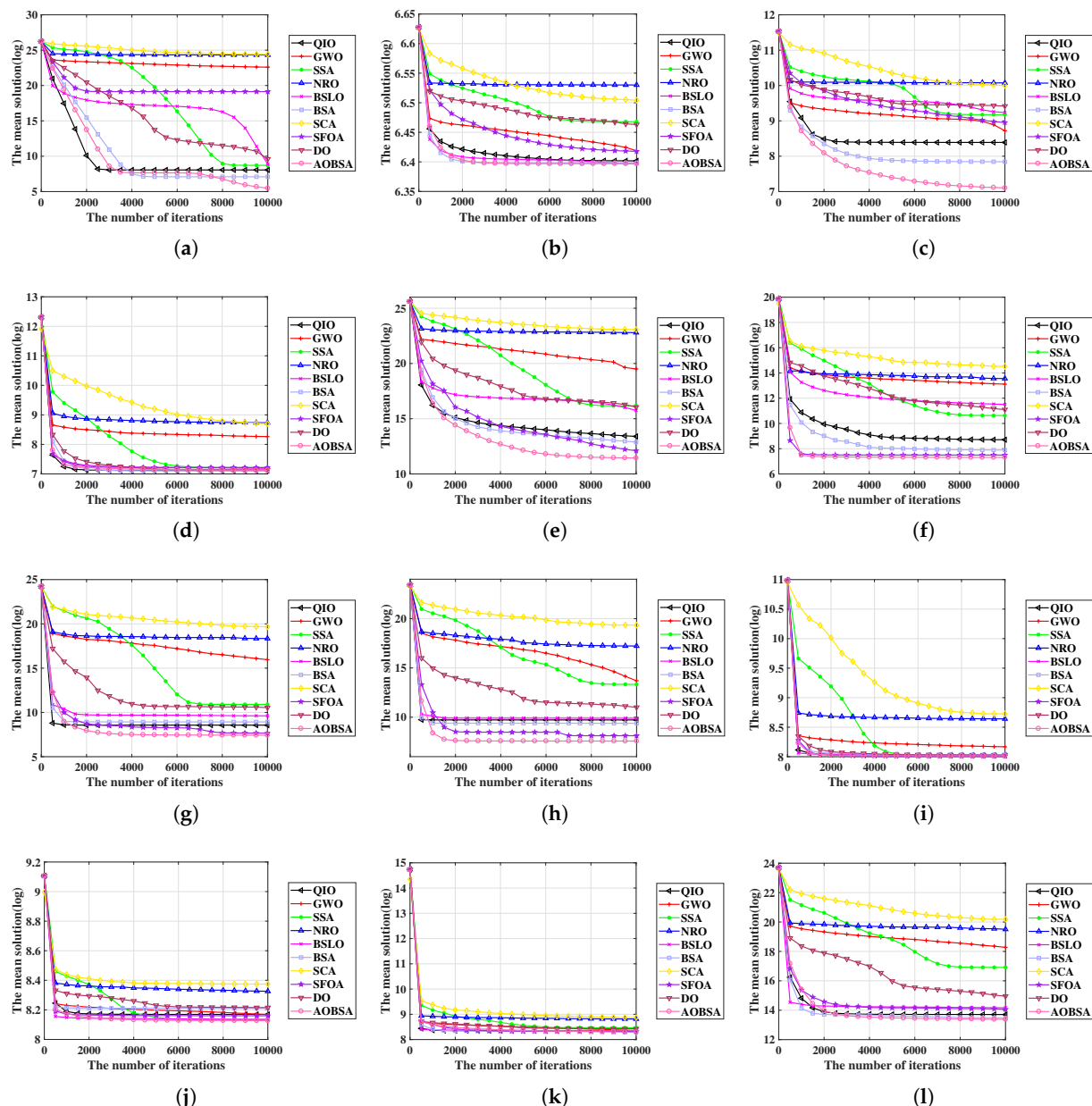


Figure 2. Some typical convergence curves produced by the applied algorithms in the CEC 2017 test suite with 50 dimensions: (a) F_1 ; (b) F_6 ; (c) F_{10} ; (d) F_{11} ; (e) F_{12} ; (f) F_{14} ; (g) F_{15} ; (h) F_{19} ; (i) F_{25} ; (j) F_{27} ; (k) F_{29} ; (l) F_{30} .

According to the results presented in Tables 1–6, the AOBSA can demonstrate better performance advantages than the comparison algorithms on most functions, but it does not show performance advantages on some functions, such as F_2 , F_{10} , and F_{22} . This phenomenon fully conforms to the NFL theory [23] and can be explained as follows: due to the different complexity of the objective function and the characteristics of the search space for each function, the AOBSA can demonstrate better comprehensive performance than the compared algorithms in solving the CEC 2017 test suite, but it cannot outperform the compared algorithms in every specific function. This phenomenon also appears in other literature that uses metaheuristic algorithms to solve optimization problems, such as [36–38]. In this section, we mainly discuss the performance of the AOBSA and the compared algorithms in solving the CEC2017 test suite when the population size is 30 with a problem dimension of 30, and when the population size is 50 with a problem dimension

of 50. To more comprehensively observe the impact of population size on algorithm performance, in the Appendix, we also present some experimental results obtained by the AOBSA and the comparison algorithm when the population size is 30 and the problem dimension is 50. These results, presented in Tables A1 and A2, also demonstrate the superior performance of the AOBSA.

5. Experimental Results in Engineering Optimization

This section aims to verify the performance of the AOBSA in solving practical engineering problems. The considered problems include the pressure vessel design problem and speed reducer design problem. The compared algorithms are nine metaheuristic algorithms: Harris Hawks Optimization (HHO) [39], Whale Optimization Algorithm (WOA) [40], Dung Beetle Optimizer (DBO) [41], Young's Double-Slit Experiment Optimizer (YDSE) [42], Great Wall Construction Algorithm (GWCA) [43], Newton–Raphson-Based Optimizer (NRO) [31], Beluga Whale Optimization [44], Escape Algorithm (ESC) [45], and Red-billed Blue Magpie Optimizer (RBMO) [46]. When applying the AOBSA to solve these two problems, the population size was set to 30, the maximum number of iterations was set to 500, and the number of independent repeated experiments for each problem was set to 30. Due to the complex constraints involved in these two engineering problems, this paper adopts the penalty function method to deal with them, as in [47]. In addition, the parameters and the experiential results of the compared algorithms were extracted from [48]. The experimental results are shown in Tables 7 and 8. In these two tables, WORST, BEST, MEAN, MEDIAN, and STD represent the worst solution, best solution, average solution, median solution, and standard deviation obtained from 30 independent repeated experiments, respectively. In addition, it should be noted that the statistical data obtained by the AOBSA involves many decimals. For ease of presentation, only some of the decimals are shown in Tables 7 and 8.

5.1. The Pressure Vessel Design Problem

The objective function of the pressure vessel design problem can be summarized as follows:

$$f(x) = 0.6224x_1x_3x_4 + 1.7881x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \quad (21)$$

The constraints of the pressure vessel design problem can be summarized as follows:

$$g_1(x) = -x_1 + 0.0193x_3 \leq 0, \quad (22)$$

$$g_2(x) = -x_2 + 0.0193x_3 \leq 0, \quad (23)$$

$$g_3(x) = -\pi x_3^2x_4 - \frac{4}{3}x_3^3 + 129600 \leq 0, \quad (24)$$

$$g_4(x) = x_4 - 240 \leq 0, \quad (25)$$

where $0 \leq x_1 \leq 99$, $0 \leq x_2 \leq 99$, $10 \leq x_3 \leq 200$, and $10 \leq x_4 \leq 200$.

The experimental results of the AOBSA and the compared algorithms in solving the pressure vessel design problem are presented in Table 7, where the AOBSA shows better performance than the compared algorithms in terms of WORST, BEST, MEAN, and MEDIAN. In addition, in terms of STD, the AOBSA is only inferior to YDSE and ERBMO, while outperforming the other eight algorithms. This indicates that, in terms of solution accuracy and stability, the AOBSA outperforms the comparison algorithms when solving this problem.

Table 7. The experimental results of the AOBSA and the compared algorithms in solving the pressure vessel design problem.

Algorithm	WORST	BEST	MEAN	MEDIAN	STD
HHO	1.277×10^4	6.246×10^3	7.185×10^3	6.901×10^3	1.143×10^3
WOA	3.118×10^4	6.854×10^3	1.001×10^4	8.823×10^3	4.479×10^3
DBO	7.544×10^3	6.060×10^3	6.491×10^3	6.357×10^3	3.972×10^2
YDSE	6.122×10^3	6.063×10^3	6.079×10^3	6.073×10^3	1.486×10^1
GWCA	7.544×10^3	6.060×10^3	6.646×10^3	6.413×10^3	5.257×10^2
NRO	7.876×10^3	6.091×10^3	6.711×10^3	6.529×10^3	5.056×10^2
BWO	7.821×10^3	6.354×10^3	7.031×10^3	7.052×10^3	3.620×10^2
ESC	7.497×10^3	6.115×10^3	6.641×10^3	6.572×10^3	3.528×10^2
RBMO	7.333×10^3	6.060×10^3	6.296×10^3	6.060×10^3	4.728×10^2
ERBMO	6.371×10^3	6.060×10^3	6.077×10^3	6.060×10^3	5.693×10^1
AOBSA	6.199×10^3	5.943×10^3	6.057×10^3	6.052×10^3	7.206×10^1

5.2. The Speed Reducer Design Problem

The objective function of the speed reducer design problem can be summarized as follows:

$$f_{\text{cost}}(\mathbf{x}) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2) \quad (26)$$

The constraints of the speed reducer design problem can be summarized as follows:

$$g_1(\mathbf{x}) = \frac{27}{x_1x_2x_3} - 1 \leq 0. \quad (27)$$

$$g_2(\mathbf{x}) = \frac{397.5}{x_1x_2^2x_3} - 1 \leq 0. \quad (28)$$

$$g_3(\mathbf{x}) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \leq 0. \quad (29)$$

$$g_4(\mathbf{x}) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \leq 0. \quad (30)$$

$$g_5(\mathbf{x}) = \sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + \frac{16.9 \times 10^6}{110x_6^3}} - 1 \leq 0. \quad (31)$$

$$g_6(\mathbf{x}) = \sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2 + \frac{157.5 \times 10^6}{85x_7^3}} - 1 \leq 0. \quad (32)$$

$$g_7(\mathbf{x}) = \frac{x_2x_3}{40} - 1 \leq 0. \quad (33)$$

$$g_8(\mathbf{x}) = \frac{5x_2}{x_1} - 1 \leq 0. \quad (34)$$

$$g_9(\mathbf{x}) = \frac{x_1}{12x_2} - 1 \leq 0. \quad (35)$$

$$g_{10}(\mathbf{x}) = \frac{1.56x_6 + 1.9}{x_4} - 1 \leq 0. \quad (36)$$

$$g_{11}(\mathbf{x}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0. \quad (37)$$

where $2.6 \leq x_1 \leq 3.6$, $0.7 \leq x_2 \leq 0.8$, $17 \leq x_3 \leq 28$, $7.3 \leq x_4 \leq 8.3$, $7.8 \leq x_5 \leq 8.3$, $2.9 \leq x_6 \leq 3.9$, and $5.0 \leq x_7 \leq 5.5$.

The experimental results of the AOBSA and the compared algorithms in solving the speed reducer design problem are shown in Table 8, where the AOBSA shows better performance than the compared algorithms in terms of WORST, BEST, MEAN, MEDIAN, and STD. This indicates that the comprehensive optimization performance of AOBSA in solving this problem is significantly superior to that of the comparison algorithms.

Table 8. The experimental results of the AOBSA and the compared algorithms in solving the speed reducer design problem.

Algorithm	WORST	BEST	MEAN	MEDIAN	STD
HHO	4.989×10^3	3.012×10^3	3.625×10^3	3.366×10^3	5.914×10^2
WOA	4.975×10^3	3.032×10^3	3.440×10^3	3.180×10^3	4.966×10^2
DBO	3.222×10^3	2.994×10^3	3.046×10^3	3.034×10^3	5.981×10^1
YDSE	3.008×10^3	2.996×10^3	2.998×10^3	2.997×10^3	2.625
GWCA	2.994×10^3	2.994×10^3	2.994×10^3	2.994×10^3	1.155×10^{-9}
NRO	3.177×10^3	3.003×10^3	3.063×10^3	3.052×10^3	4.914×10^1
BWO	3.266×10^3	3.035×10^3	3.111×10^3	3.104×10^3	5.493×10^1
ESC	2.995×10^3	2.994×10^3	2.995×10^3	2.994×10^3	7.256×10^{-2}
RBMO	2.994×10^3	2.994×10^3	2.994×10^3	2.994×10^3	7.820×10^{-6}
ERBMO	2.994×10^3	2.993×10^3	2.994×10^3	2.994×10^3	2.828×10^{-1}
AOBSA	2.983×10^3	2.983×10^3	2.983×10^3	2.983×10^3	9.394×10^{-3}

6. Conclusions

This paper proposes a new technique for improving the performance of population-based optimization algorithms, called the average operator. In this paper, we define a set of average operators including global average position, upper average position, lower average position, random global average position, random upper average position, and random lower average position. The defined average operators can reflect the local or overall average position of the population. This paper applies the defined average operator to the BSA and subsequently proposes the AOBSA. Thanks to the simple structure of the defined average operator and the absence of complex calculations, compared to the BSA, the computational complexity of the AOBSA is not significantly increased. To verify the performance of the AOBSA, it was employed to solve the complex CEC 2017 test suite with 30 and 50 dimensions. The experimental results demonstrate that the AOBSA exhibits better optimization performance compared to the compared QIO, GWO, SSA, NRO, BSLO, BSA, SCA, SFOA, and DO. In addition, the AOBSA also outperforms the comparison algorithms in terms of solving practical engineering problems, including the pressure vessel design problem and the speed reducer design problem.

Future research will focus on two directions: First, extending the application scope of the defined set of average operators by further applying them to other population-based optimization algorithms. Second, developing more efficient improvement techniques tailored to population-based optimization algorithms.

Funding: This project was supported in part by the Special Fund of the Taishan Scholar Program under Grant tsqn202408187, and in part by the Shandong Provincial Natural Science Foundation under Grant ZR2025QC1548.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the author.

Conflicts of Interest: The author declares no conflicts of interest.

Appendix A

This section mainly presents some experimental results obtained by the AOBSA and the comparison algorithms when the population size is 30 and the problem dimension is 50.

Table A1. The experimental results of AOBSA and the compared algorithms in solving the CEC 2017 test suite with 50-dimensional (population size is set to 30).

No.	Algorithm	QIO	GWO	SSA	NRO	BSLO	BSA	SCA	SFOA	DO	AOBSA
F ₁	MEAN	2.94×10^3	9.13×10^9	6.31×10^3	4.51×10^{10}	9.70×10^3	2.20×10^3	4.44×10^{10}	2.68×10^9	3.33×10^4	1.73×10^3
	STD	3.96×10^3	4.32×10^9	6.67×10^3	7.68×10^9	8.63×10^3	3.11×10^3	5.87×10^9	3.29×10^9	2.23×10^4	2.19×10^3
F ₂	MEAN	2.79×10^5	2.13×10^{50}	1.23×10^{11}	2.32×10^{63}	2.41×10^{14}	1.74×10^{22}	3.55×10^{66}	3.89×10^{32}	1.99×10^{11}	1.01×10^{16}
	STD	1.51×10^6	8.96×10^{50}	3.80×10^{11}	1.27×10^{64}	6.46×10^{14}	9.50×10^{22}	1.94×10^{67}	2.12×10^{33}	1.08×10^{12}	3.19×10^{16}
F ₃	MEAN	3.01×10^3	8.31×10^4	3.00×10^2	1.07×10^5	2.90×10^3	1.60×10^4	1.11×10^5	3.29×10^2	4.13×10^2	1.64×10^4
	STD	1.52×10^3	1.61×10^4	8.84×10^{-1}	1.36×10^4	1.38×10^3	6.99×10^3	1.54×10^4	1.36×10^2	8.16×10^1	4.41×10^3
F ₄	MEAN	5.23×10^2	1.38×10^3	5.42×10^2	6.64×10^3	5.32×10^2	5.35×10^2	7.34×10^3	7.45×10^2	5.51×10^2	5.00×10^2
	STD	5.84×10^1	5.95×10^2	5.28×10^1	2.57×10^3	6.73×10^1	5.66×10^1	1.26×10^3	5.32×10^2	5.93×10^1	6.17×10^1
F ₅	MEAN	7.60×10^2	7.15×10^2	7.73×10^2	1.09×10^3	7.79×10^2	7.07×10^2	1.06×10^3	7.76×10^2	8.12×10^2	7.41×10^2
	STD	3.41×10^1	4.16×10^1	5.45×10^1	4.20×10^1	4.48×10^1	4.24×10^1	3.22×10^1	4.46×10^1	4.53×10^1	2.92×10^1
F ₆	MEAN	6.13×10^2	6.17×10^2	6.42×10^2	6.86×10^2	6.00×10^2	6.04×10^2	6.69×10^2	6.27×10^2	6.45×10^2	6.00×10^2
	STD	5.92	4.59	6.83	8.67	3.22×10^{-2}	2.62	5.69	8.84	5.50	5.16×10^{-2}
F ₇	MEAN	1.15×10^3	1.09×10^3	1.07×10^3	1.71×10^3	1.00×10^3	1.05×10^3	1.63×10^3	1.13×10^3	1.35×10^3	1.04×10^3
	STD	9.43×10^1	7.19×10^1	9.46×10^1	1.56×10^2	4.86×10^1	9.05×10^1	8.97×10^1	1.17×10^2	8.85×10^1	3.07×10^1
F ₈	MEAN	1.04×10^3	1.02×10^3	1.06×10^3	1.41×10^3	1.07×10^3	1.01×10^3	1.37×10^3	1.09×10^3	1.14×10^3	1.04×10^3
	STD	4.23×10^1	3.91×10^1	5.39×10^1	4.69×10^1	4.67×10^1	3.42×10^1	2.61×10^1	5.08×10^1	6.89×10^1	2.86×10^1
F ₉	MEAN	5.56×10^3	6.17×10^3	9.22×10^3	2.67×10^4	1.06×10^4	3.48×10^3	2.32×10^4	1.00×10^4	1.35×10^4	2.49×10^3
	STD	1.22×10^3	3.10×10^3	2.85×10^3	4.10×10^3	2.47×10^3	9.55×10^2	2.65×10^3	6.24×10^3	4.12×10^3	1.16×10^3
F ₁₀	MEAN	7.18×10^3	6.85×10^3	7.74×10^3	1.36×10^4	6.38×10^3	1.19×10^4	1.44×10^4	7.22×10^3	7.74×10^3	8.86×10^3
	STD	8.14×10^2	7.81×10^2	9.31×10^2	7.31×10^2	6.66×10^2	1.70×10^3	3.64×10^2	7.95×10^2	1.00×10^3	9.37×10^2
F ₁₁	MEAN	1.24×10^3	4.89×10^3	1.38×10^3	8.18×10^3	1.32×10^3	1.26×10^3	6.28×10^3	1.40×10^3	1.27×10^3	1.29×10^3
	STD	4.18×10^1	2.06×10^3	6.98×10^1	2.93×10^3	6.15×10^1	3.36×10^1	1.12×10^3	1.02×10^2	3.16×10^1	5.52×10^1
F ₁₂	MEAN	1.03×10^6	9.04×10^8	1.47×10^7	1.12×10^{10}	8.91×10^6	7.35×10^5	1.19×10^{10}	9.27×10^8	1.12×10^7	1.49×10^5
	STD	7.17×10^5	8.77×10^8	8.34×10^6	5.92×10^9	5.32×10^6	4.57×10^5	2.59×10^9	3.05×10^9	5.05×10^6	8.13×10^4
F ₁₃	MEAN	6.89×10^3	1.15×10^8	1.09×10^5	2.75×10^9	1.66×10^4	8.16×10^7	3.26×10^9	1.32×10^8	8.26×10^4	8.06×10^3
	STD	7.78×10^3	1.68×10^8	7.35×10^4	2.14×10^9	1.22×10^4	4.47×10^8	1.83×10^9	7.21×10^8	4.16×10^4	8.75×10^3
F ₁₄	MEAN	1.38×10^4	4.87×10^5	5.08×10^4	1.15×10^6	1.14×10^5	4.73×10^3	2.60×10^6	2.19×10^3	8.12×10^4	1.57×10^3
	STD	9.08×10^3	4.91×10^5	4.02×10^4	7.91×10^5	6.31×10^4	2.95×10^3	1.12×10^6	1.30×10^3	7.15×10^4	7.91×10^1
F ₁₅	MEAN	6.83×10^3	1.57×10^7	6.02×10^4	1.91×10^8	1.64×10^4	6.83×10^3	3.72×10^8	1.33×10^8	3.97×10^4	4.81×10^3
	STD	5.17×10^3	2.77×10^7	4.08×10^4	2.15×10^8	7.01×10^3	6.70×10^3	1.90×10^8	6.15×10^8	1.50×10^4	5.54×10^3
F ₁₆	MEAN	3.38×10^3	2.91×10^3	3.22×10^3	5.22×10^3	3.88×10^3	3.00×10^3	5.57×10^3	3.04×10^3	3.69×10^3	2.98×10^3
	STD	4.97×10^2	3.10×10^2	4.46×10^2	6.61×10^2	3.72×10^2	3.60×10^2	2.62×10^2	3.62×10^2	4.47×10^2	2.45×10^2
F ₁₇	MEAN	2.92×10^3	2.72×10^3	3.11×10^3	4.20×10^3	3.17×10^3	2.69×10^3	4.40×10^3	2.90×10^3	3.32×10^3	2.61×10^3
	STD	3.15×10^2	2.45×10^2	2.66×10^2	4.05×10^2	3.21×10^2	2.58×10^2	2.97×10^2	3.10×10^2	3.41×10^2	1.63×10^2
F ₁₈	MEAN	8.70×10^4	7.40×10^6	3.42×10^5	8.05×10^6	1.64×10^6	2.95×10^5	1.75×10^7	6.77×10^3	1.02×10^6	1.57×10^4
	STD	5.99×10^4	1.15×10^7	1.53×10^5	7.21×10^6	1.14×10^6	1.25×10^6	1.09×10^7	5.40×10^3	6.94×10^5	1.09×10^4
F ₁₉	MEAN	1.55×10^4	1.14×10^7	7.32×10^5	7.14×10^7	1.48×10^4	1.21×10^4	2.82×10^8	3.17×10^7	6.41×10^4	2.36×10^3
	STD	7.94×10^3	4.67×10^7	3.50×10^5	8.73×10^7	1.61×10^4	9.38×10^3	1.21×10^8	1.74×10^8	2.99×10^4	1.42×10^3
F ₂₀	MEAN	2.85×10^3	2.78×10^3	3.07×10^3	3.67×10^3	3.06×10^3	2.92×10^3	3.82×10^3	2.82×10^3	3.26×10^3	2.81×10^3
	STD	2.38×10^2	1.95×10^2	2.88×10^2	2.39×10^2	3.40×10^2	4.84×10^2	1.75×10^2	2.43×10^2	3.04×10^2	1.66×10^2
F ₂₁	MEAN	2.52×10^3	2.52×10^3	2.52×10^3	2.91×10^3	2.60×10^3	2.47×10^3	2.89×10^3	2.60×10^3	2.62×10^3	2.51×10^3
	STD	4.90×10^1	5.49×10^1	4.51×10^1	5.47×10^1	6.79×10^1	3.46×10^1	4.22×10^1	6.14×10^1	5.90×10^1	2.94×10^1
F ₂₂	MEAN	8.38×10^3	8.03×10^3	9.01×10^3	1.46×10^4	8.45×10^3	1.35×10^4	1.61×10^4	7.86×10^3	9.61×10^3	1.08×10^4
	STD	2.16×10^3	1.17×10^3	1.61×10^3	2.33×10^3	7.80×10^2	1.81×10^3	4.72×10^2	2.95×10^3	8.65×10^2	9.22×10^2
F ₂₃	MEAN	3.07×10^3	2.98×10^3	2.99×10^3	3.46×10^3	3.03×10^3	2.93×10^3	3.52×10^3	3.13×10^3	3.22×10^3	2.95×10^3
	STD	7.61×10^1	4.87×10^1	6.54×10^1	7.80×10^1	6.14×10^1	5.21×10^1	5.31×10^1	1.04×10^2	9.22×10^1	1.71×10^1
F ₂₄	MEAN	3.25×10^3	3.16×10^3	3.12×10^3	3.62×10^3	3.39×10^3	3.18×10^3	3.70×10^3	3.33×10^3	3.49×10^3	3.11×10^3
	STD	7.31×10^1	7.28×10^1	5.47×10^1	9.14×10^1	1.22×10^2	1.52×10^2	5.95×10^1	1.42×10^2	1.13×10^2	2.92×10^1
F ₂₅	MEAN	3.06×10^3	3.63×10^3	3.04×10^3	6.30×10^3	3.06×10^3	3.08×10^3	6.14×10^3	3.14×10^3	3.06×10^3	3.03×10^3
	STD	3.55×10^1	2.18×10^2	3.20×10^1	8.40×10^2						

Table A2. The ranking results of the AOBSA and the compared algorithms in solving the CEC 2017 test suite with 50 dimensions (population size is set to 30).

Algorithm	QIO	GWO	SSA	NRO	BSLO	BSA	SCA	SFOA	DO	AOBSA
F ₁	2.93	7.90	3.57	9.50	4.90	2.63	9.50	5.60	6.17	2.30
F ₂	1.07	8.00	2.77	9.17	4.00	6.00	9.83	6.53	2.73	4.90
F ₃	4.60	8.27	1.10	9.37	4.40	6.43	9.37	1.93	2.97	6.57
F ₄	3.80	7.83	4.43	9.23	4.20	4.07	9.77	4.07	4.57	3.03
F ₅	4.70	2.23	5.20	9.73	5.60	2.37	9.27	5.40	6.73	3.77
F ₆	4.23	4.93	7.33	9.93	1.90	3.07	9.07	5.87	7.57	1.10
F ₇	5.80	4.53	3.90	9.67	2.20	3.43	9.30	4.93	7.90	3.33
F ₈	3.87	2.67	4.53	9.80	5.10	2.30	9.20	6.10	7.43	4.00
F ₉	3.73	3.80	5.63	9.70	6.47	2.23	9.07	5.83	7.20	1.33
F ₁₀	3.73	2.93	4.60	9.07	1.83	8.07	9.73	3.80	4.87	6.37
F ₁₁	2.23	8.40	5.63	9.63	4.57	2.63	8.97	6.03	3.13	3.77
F ₁₂	3.23	7.87	6.20	9.33	5.43	3.00	9.60	3.27	5.80	1.27
F ₁₃	2.37	7.63	6.73	9.27	3.57	2.63	9.63	4.47	6.37	2.33
F ₁₄	4.13	8.00	5.20	8.80	6.73	3.10	9.93	2.07	6.00	1.03
F ₁₅	3.00	7.23	6.87	9.03	4.53	2.70	9.80	3.30	6.57	1.97
F ₁₆	5.13	2.90	4.30	9.13	7.37	3.13	9.73	3.53	6.53	3.23
F ₁₇	4.50	3.27	5.73	9.17	6.07	3.10	9.70	4.47	6.73	2.27
F ₁₈	3.63	8.17	5.13	8.80	6.77	3.40	9.70	1.20	6.30	1.90
F ₁₉	4.13	7.47	7.47	8.93	3.67	3.53	9.93	2.57	5.97	1.33
F ₂₀	3.93	3.60	5.43	9.00	5.33	4.03	9.53	3.67	6.93	3.53
F ₂₁	3.83	3.47	4.00	9.70	6.27	1.83	9.30	6.53	6.83	3.23
F ₂₂	3.70	2.47	4.10	8.30	2.83	8.13	9.70	3.83	5.23	6.70
F ₂₃	5.70	3.43	3.37	9.17	4.80	1.93	9.77	6.63	7.67	2.53
F ₂₄	4.77	3.60	2.33	8.97	6.63	3.13	9.60	6.23	7.57	2.17
F ₂₅	3.97	7.93	3.03	9.63	4.37	5.17	9.37	4.47	4.33	2.73
F ₂₆	5.70	4.40	3.57	9.33	2.73	4.00	9.43	4.70	7.43	3.70
F ₂₇	5.03	4.50	2.47	8.63	2.70	6.83	9.83	5.40	6.97	2.63
F ₂₈	3.30	7.57	3.43	9.23	3.53	7.07	9.10	4.07	4.60	3.10
F ₂₉	4.70	5.27	6.50	9.30	3.77	3.80	9.70	5.00	5.17	1.80
F _{b30}	3.30	7.97	7.03	9.10	3.80	2.43	9.90	4.07	6.00	1.40
Avg.	3.96	5.61	4.72	9.25	4.54	3.87	9.54	4.52	6.01	2.98

References

1. Liu, X.F.; Fang, Y.; Zhan, Z.H.; Zhang, J. Strength Learning Particle Swarm Optimization for Multiobjective Multirobot Task Scheduling. *IEEE Trans. Syst. Man, Cybern. Syst.* **2023**, *53*, 4052–4063.
2. Wang, X.; Yao, H.; Mai, T.; Guo, S.; Liu, Y. Reinforcement Learning-Based Particle Swarm Optimization for End-to-End Traffic Scheduling in TSN-5G Networks. *IEEE/ACM Trans. Netw.* **2023**, *31*, 3254–3268.
3. Yang, X.; Feng, J.; Liu, L.; Pei, Q. Optimizing Resource Utilization in Consumer Electronics Networks Through an Enhanced Grey Wolf Optimization Algorithm With UAV Collaboration. *IEEE Trans. Consum. Electron.* **2025**, *71*, 7376–7386.
4. Jyothi, K.K.; Borra, S.R.; Srilakshmi, K.; Balachandran, P.K.; Reddy, G.P.; Colak, I.; Dhanamjayulu, C.; Chinthaginjala, R.; Khan, B. A novel optimized neural network model for cyber attack detection using enhanced whale optimization algorithm. *Sci. Rep.* **2024**, *14*, 5590.
5. Yu, F.; Guan, J.; Wu, H.; Wang, H.; Ma, B. Multi-population differential evolution approach for feature selection with mutual information ranking. *Expert Syst. Appl.* **2025**, *260*, 125404.
6. Dubey, P.; Saudagar, A.K.J.; Poonia, R.C.; Asati, M. Robust and imperceptible image watermarking using chaotic map-integrated quantum-inspired multi-objective cuckoo search optimization. *Sci. Rep.* **2025**, *15*, 41749.
7. Liu, M.; Yuan, Y.; Xu, A.; Deng, T.; Jian, L. A learning-based artificial bee colony algorithm for operation optimization in gas pipelines. *Inf. Sci.* **2025**, *690*, 121593.
8. Majumdar, P.; Mitra, S. Salp swarm algorithm using lens opposition-based learning and local search for constrained optimization problems. *Iran J. Comput. Sci.* **2025**, *8*, 659–686.
9. Fu, Y.; Liu, D.; Chen, J.; He, L. Secretary bird optimization algorithm: A new metaheuristic for solving global optimization problems. *Artif. Intell. Rev.* **2024**, *57*, 123.
10. Pan, J.S.; Zhang, L.G.; Wang, R.B.; Snášel, V.; Chu, S.C. Gannet optimization algorithm: A new metaheuristic algorithm for solving engineering optimization problems. *Math. Comput. Simul.* **2022**, *202*, 343–373.

11. Dehghani, M.; Montazeri, Z.; Trojovská, E.; Trojovský, P. Coati Optimization Algorithm: A new bio-inspired metaheuristic algorithm for solving optimization problems. *Knowl.-Based Syst.* **2023**, *259*, 110011.
12. Abdel-Basset, M.; Mohamed, R.; Azeem, S.A.A.; Jameel, M.; Abouhawwash, M. Kepler optimization algorithm: A new metaheuristic algorithm inspired by Kepler's laws of planetary motion. *Knowl.-Based Syst.* **2023**, *268*, 110454.
13. Trojovský, P.; Dehghani, M. A new bio-inspired metaheuristic algorithm for solving optimization problems based on walrus behavior. *Sci. Rep.* **2023**, *13*, 8775.
14. Zhang, Y.; Jin, Z.; Mirjalili, S. Generalized normal distribution optimization and its applications in parameter extraction of photovoltaic models. *Energy Convers. Manag.* **2020**, *224*, 113301.
15. Azizi, M.; Aickelin, U.; A. Khorshidi, H.; Baghalzadeh Shishehgharkhaneh, M. Energy valley optimizer: A novel metaheuristic algorithm for global and engineering optimization. *Sci. Rep.* **2023**, *13*, 226.
16. Deng, L.; Liu, S. Snow ablation optimizer: A novel metaheuristic technique for numerical optimization and engineering design. *Expert Syst. Appl.* **2023**, *225*, 120069.
17. Civicioglu, P. Backtracking search optimization algorithm for numerical optimization problems. *Appl. Math. Comput.* **2013**, *219*, 8121–8144.
18. Wu, D.; Foong, L.K.; Lyu, Z. Two neural-metaheuristic techniques based on vortex search and backtracking search algorithms for predicting the heating load of residential buildings. *Eng. Comput.* **2022**, *38*, 647–660.
19. Hu, Z.; Dai, C.; Su, Q. Adaptive backtracking search optimization algorithm with a dual-learning strategy for dynamic economic dispatch with valve-point effects. *Energy* **2022**, *248*, 123558.
20. Zhang, Y.; Zhou, G.; Hang, P.; Huang, C.; Huang, H. An enhanced backtracking search algorithm for the flight planning of a multi-drones-assisted commercial parcel delivery system. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 11396–11409.
21. Dai, C.; Hu, Z.; Su, Q. An adaptive hybrid backtracking search optimization algorithm for dynamic economic dispatch with valve-point effects. *Energy* **2022**, *239*, 122461.
22. Emami, H.; Emami, S.; Rezaverdinejad, V. A backtracking search-based extreme gradient boosting algorithm for soil moisture prediction using meteorological variables. *Earth Sci. Inform.* **2025**, *18*, 181.
23. Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **2002**, *1*, 67–82.
24. Awad, N.; Ali, M.; Liang, J.; Qu, B.; Suganthan, P. *Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization*; Technical Report; National University of Defense Technology: Changsha, China; China and Kyungpook National University: Daegu, Republic of Korea; Nanyang Technological University: Singapore. 2016.
25. R.V. Rao.; Savsani, V.; Vakharia, D. Teaching—Learning-based optimization: A novel method for constrained mechanical design optimization problems. *Comput.-Aided Des.* **2011**, *43*, 303–315.
26. Zhang, Y.; Jin, Z. Group teaching optimization algorithm: A novel metaheuristic method for solving global optimization problems. *Expert Syst. Appl.* **2020**, *148*, 113246.
27. Majumdar, P.; Mitra, S. Enhanced honey badger algorithm based on nonlinear adaptive weight and golden sine operator. *Neural Comput. Appl.* **2025**, *37*, 367–386.
28. Zhao, W.; Wang, L.; Zhang, Z.; Mirjalili, S.; Khodadadi, N.; Ge, Q. Quadratic Interpolation Optimization (QIO): A new optimization algorithm based on generalized quadratic interpolation and its applications to real-world engineering problems. *Comput. Methods Appl. Mech. Eng.* **2023**, *417*, 116446.
29. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61.
30. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191.
31. Sowmya, R.; Premkumar, M.; Jangir, P. Newton-Raphson-based optimizer: A new population-based metaheuristic algorithm for continuous optimization problems. *Eng. Appl. Artif. Intell.* **2024**, *128*, 107532.
32. Bai, J.; Nguyen-Xuan, H.; Atroshchenko, E.; Kosec, G.; Wang, L.; Abdel Wahab, M. Blood-sucking leech optimizer. *Adv. Eng. Softw.* **2024**, *195*, 103696.
33. Mirjalili, S. SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowl.-Based Syst.* **2016**, *96*, 120–133.
34. Zhong, C.; Li, G.; Meng, Z.; Li, H.; Yildiz, A.R.; Mirjalili, S. Starfish optimization algorithm(SFOA): A bio-inspired metaheuristic algorithm for global optimization compared with 100 optimizers. *Neural Comput. Appl.* **2025**, *37*, 3641–3683.
35. Zhao, S.; Zhang, T.; Ma, S.; Chen, M. Dandelion Optimizer: A nature-inspired metaheuristic algorithm for engineering applications. *Eng. Appl. Artif. Intell.* **2022**, *114*, 105075.
36. Pu, C.; Jia, Y.; Zhang, Z.; Zhou, H.; Liu, L.; Qian, P.; Iqbal, N.; Emzir, M.F. A fuzzy adaptive particle swarm optimization algorithm with Gaussian mutation for constrained engineering problems. *Appl. Soft Comput.* **2025**, *185*, 113908.
37. Hussein, N.K.; Qaraad, M.; El Najjar, A.M.; Farag, M.; Elhosseini, M.A.; Mirjalili, S.; Guinovart, D. Schrödinger optimizer: A quantum duality-driven metaheuristic for stochastic optimization and engineering challenges. *Knowl.-Based Syst.* **2025**, *328*, 114273.

38. Saad, M.R.; Emam, M.M.; Hosney, M.E.; Samee, N.A.; Alkanhel, R.I.; Houssein, E.H. Fourier transform optimizer: A novel physics-inspired metaheuristic algorithm for optimization problems. *Knowl.-Based Syst.* **2026**, *340*, 115651.
39. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H. Harris hawks optimization: Algorithm and applications. *Future Gener. Comput. Syst.* **2019**, *97*, 849–872.
40. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67.
41. Xue, J.; Shen, B. Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. *J. Supercomput.* **2023**, *79*, 7305–7336.
42. Abdel-Basset, M.; El-Shahat, D.; Jameel, M.; Abouhawwash, M. Young’s double-slit experiment optimizer: A novel metaheuristic optimization algorithm for global and constraint optimization problems. *Comput. Methods Appl. Mech. Eng.* **2023**, *403*, 115652.
43. Ziyu, G.; Changjiang, R.; Jingtai, N.; Peixi, W.; Yizi, S. Great Wall Construction Algorithm: A novel meta-heuristic algorithm for engineer problems. *Expert Syst. Appl.* **2023**, *233*, 120905.
44. Zhong, C.; Li, G.; Meng, Z. Beluga whale optimization: A novel nature-inspired metaheuristic algorithm. *Knowl.-Based Syst.* **2022**, *251*, 109215.
45. Ouyang, K.; Fu, S.; Chen, Y.; Cai, Q.; Heidari, A.A.; Chen, H. Escape: An optimization method based on crowd evacuation behaviors. *Artif. Intell. Rev.* **2024**, *58*, 19.
46. Fu, S.; Li, K.; Huang, H.; Ma, C.; Fan, Q.; Zhu, Y. Red-billed blue magpie optimizer: A novel metaheuristic algorithm for 2D/3D UAV path planning and engineering design problems. *Artif. Intell. Rev.* **2024**, *57*, 134.
47. She, Y.X. *Engineering Optimization: An Introduction with Metaheuristic Applications*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 2010.
48. Wang, H.; Xin, Z.; Qi, X.; Zhou, Q.; Liu, J.; Li, Z. Enhanced Red-billed Blue Magpie Optimizer for engineering optimization problems. *Sci. Rep.* **2026**, *16*, 10619.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.