A Hybrid Algorithm Framework with Learning and Complementary Fusion Features for Whale Optimization Algorithm

Wangyu Tong

1School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430070, Hubei, China
2School of Computer Science, Hubei University of Technology, Wuhan 430068, Hubei, China

Correspondence should be addressed to Wangyu Tong; ruyu319@hbut.edu.cn

Received 29 October 2019; Accepted 16 January 2020; Published 18 February 2020

It has been observed that the structure of whale optimization algorithm (WOA) is good at exploiting capability, but it easily suffers from premature convergence. Hybrid metaheuristics are of the most interesting recent trends for improving the performance of WOA. In this paper, a hybrid algorithm framework with learning and complementary fusion features for WOA is designed, called hWOA liar. First, WOA is integrated with complementary feature operators to enhance exploration capability. Second, the proposed algorithm framework adopts a learning parameter $l_p$ according to adaptive adjustment operator to replace the random parameter $p$. To further verify the efficiency of the hWOA liar, the DE/rand/1 operator of differential evolution (DE) and the mutate operator of backtracking search optimization algorithm (BSA) are embedded into WOA, respectively, to form two new algorithms called WOA-DE and WOA-BSA under the proposed framework. Twenty-three benchmark functions and six engineering design problems are employed to test the performance of WOA-DE and WOA-BSA. Experimental results show that WOA-DE and WOA-BSA are competitive compared with some state-of-the-art algorithms.

1. Introduction

Nowadays, most function optimization problems and engineering design problems are turning out to be complicated, which prevents traditional methods from providing accurate or even satisfactory solutions. However, the emergence of the metaheuristic optimization algorithms breaks this situation and attracts the attention of many researchers because of their flexibility, derivation-free mechanisms, and local optima avoidance. Generally, metaheuristic algorithms can be divided into two main categories. The first category is evolutionary algorithms (EAs) that are a new family of optimization algorithms inspired from the evolutionary procedure or intelligent behaviors of organisms such as genetic algorithm (GA) [1], differential evolution (DE) algorithm [2], biogeography-based algorithm (BBO) [3], backtracking search optimization algorithm (BSA) [4], grey prediction evolution algorithm (GPEA) [5], and so on. The second category is swarm intelligence algorithms (SIs) which mimic the social behaviors of groups of animals. The most popular SI is particle swarm optimization (PSO) [6]. Other popular algorithms are artificial bee colony (ABC) [7], cuckoo search (CS) [8], grey wolf optimizer (GWO) [9], and so on.

Whale optimization algorithm, proposed by Mirjalili et al. in 2016 [10], is one of the most recent SIs. Although it has strong local search capability due to its bubble-net attacking method, it is easy to get into local optimization and is weak in exploration. Hybrid metaheuristic algorithms, as one of the most efficient methods for enhancing the performance of algorithm, have been testified to be more efficient than the method that uses solely an algorithm. Since WOA was proposed, it has been widely promoted in various fields by combining with other algorithms. One of the most widely used strategies to construct hybrid WOA is by simply hybridizing two algorithms according to a certain way. For example,
Bentouati et al. [11] designed a hybrid whale optimization algorithm and pattern search technique called WOA-PS and tested it in optimal power flow problem. Xiong et al. [12] put forward a hybrid algorithm called DE/WOA, which combined the exploration of DE with the exploitation of WOA for extracting the accurate parameters of PV models. Luo et al. [13] proposed a MDE-WOA algorithm which combines modified DEGL algorithm and WOA to solve global optimization problems. Mafarja et al. [14] introduced a hybrid algorithm with whale optimization algorithm and simulated annealing, in which two hybridization models are considered for solving the feature selection problem.

Another strategy to construct hybrid WOA is by combining operators with complementary fusion features into WOA to form a superior algorithm. Trivedi et al. [15] hybridized particle swarm optimization with whale optimization algorithm called PSO- WOA to handle global numerical function optimization. The main idea is to combine best characteristic of both algorithms to form a novel mutation operator. Korashy et al. [16] hybridized whale optimization algorithm and grey wolf optimizer algorithm called WOA-GWO for optimal coordination of direction overcurrent relays. Xu et al. [17] proposed a memetic WOA (MWOA) through incorporating a chaotic local search into WOA to enhance its exploitation ability. Kaveh et al. [18] attempted to enhance the original formulation of the WOA by hybridizing it with some concepts of the colliding bodies optimization (CBO). The new method, known as WOA-CBO algorithm, is applied to building site layout planning problem. Khalilpourazari et al. [19] introduced a novel hybrid algorithm termed as sine-cosine WOA (SCWOA) for parameter optimization problem of multipass milling process to decrease total production time. Singh and Hachimi [20] proposed a new hybrid whale optimization algorithm with mean grey wolf optimizer (WOA-MGWO) for global optimization. Abdel-Basset et al. [21] proposed a new algorithm which combines the WOA with a local search strategy to defeat the permutation flow shop scheduling problem. Laskar et al. [22] proposed a new population-based hybrid metaheuristic algorithm called whale-PSO algorithm (HWPSO) to manage complex optimization problems.

In this paper, a hybrid algorithm framework named hWOAIf is proposed by combining learning and complementary fusion features into WOA. Specifically, the proposed algorithm framework adopts a learning parameter \( l_p \) according to adaptive adjustment operator instead of the random parameter \( p \), namely, the selection of search equation depends on the process of \( l_p \) learning from the subpopulation. Additionally, to improve the exploration of WOA, operators with complementary fusion features are employed to balance exploration and exploitation capacity. This paper adopts the DE/rand/1 operator of differential evolution (DE) and the mutation operator of backtracking search optimization algorithm (BSA) to form two new algorithms, respectively, called WOA-DE and WOA-BSA. Then, twenty-three benchmark functions and six engineering design problems are employed to test the performance of WOA-DE and WOA-BSA. Experimental results show that WOA-DE and WOA-BSA are competitive compared with some state-of-the-art algorithms.

The main contributions of this paper can be summarized as follows:

(i) Proposing a hybrid algorithm framework for WOA, named hWOAIf. First, designing a learning parameter \( l_p \) according to adaptive adjustment operator. Second, combining complementary fusion feature operators into WOA.

(ii) Forming two new algorithms WOA-DE and WOA-BSA by complementary fusion features, in which the DE/rand/1 operator of DE and the mutation operator of BSA are incorporated to WOA, respectively.

The rest of the paper is organized as follows. Section 2 gives a brief review of WOA, DE, and BSA. The proposed hybrid algorithm framework is presented in Section 3. In Section 4, the experimental results and analysis are given. Finally, the conclusions are shown in Section 5.

2. Preliminary

2.1. Whale Optimization Algorithm (WOA). Whale optimization algorithm is a new swarm-based metaheuristic optimization algorithm, proposed by Mirjalili and Lewis in 2016 [10], which mimics the intelligent foraging behavior of humpback whales. The algorithm is inspired by bubble-net hunting strategy. WOA includes three operators to simulate the search for prey, encircling prey, and bubble-net foraging behavior of humpback whales. Among them, the encircling prey and spiral bubble-net attacking method is the exploitation phase, while searching for prey is the exploration phase. The mathematical model of the two phase is presented below.

2.1.1. Stage 1: Exploitation Phase (Encircling Prey/Bubble-Net Attacking Method). During the exploitation phase, humpback whales update their position according to two mechanisms: shrinking encircling mechanism (encircling prey) and spiral updating position (spiral bubble-net attacking method). Shrinking encircling mechanism is represented by the following equations:

\[
\bar{X}(t + 1) = X^*\left(t\right) - \bar{A} \cdot \text{Dis}, \quad \text{Dis} = \left| \bar{C} \cdot X^\tau\left(t\right) - \bar{X}\left(t\right) \right|,
\]

where \( t \) represents the current iteration, \( X^\tau\left(t\right) \) is the position vector of the best solution obtained so far, and \( \bar{X}\left(t\right) \) is the position vector. \( \bar{A} \) and \( \bar{C} \) are coefficient vectors and the calculation methods are as follows:

\[
\bar{A} = 2 \bar{a} \cdot \bar{r} - \bar{a},
\]

\[
\bar{C} = 2 \cdot \bar{r},
\]

where \( \bar{a} \) is linearly decreased from 2 to 0 over the course of iterations (in both exploration and exploitation phases) and \( \bar{r} \) is a random vector in \([0, 1]\).

Then, the spiral updating position is mathematically expressed by the following equation:
\[ \bar{X}(t + 1) = \text{Dis} \cdot e^{\text{bl}} \cdot \cos(2\pi l) + X^*(t), \]
\[ \text{Dis} = |\bar{X}(t) - \bar{X}(t)|, \]  
(3)

where \( \text{Dis} \) represents the distance of the \( i \)th humpback whale to best solution obtained so far, \( b \) is a constant defining the logarithmic spiral shape, \( l \) is a random number in \([-1, 1]\), and \( \cdot \) is the element-by-element multiplication.

When whales capture their prey, it is worth noting that the two mechanisms mentioned above, that is, shrinking encircling mechanism and spiral updating position, are executed simultaneously. In order to imitate this behavior, a probability of 50% is assumed to choose between them. The mathematical model is as follows:
\[ \bar{X}(t + 1) = \begin{cases} \bar{X}^*(t) - A \cdot \text{Dis}, & \text{if } p < 0.5, \\ \text{Dis} \cdot e^{\text{bl}} \cdot \cos(2\pi l) + X^*(t), & \text{if } p \geq 0.5, \end{cases} \]
(4)

where \( p \) is a random number in \([0, 1]\).

2.1.2. Stage 2: Exploration Phase (Searching for Prey). In this stage, in order to enhance the exploration capability in WOA, we update the position of whale according to a randomly chosen whale instead of the best whale found so far. Therefore, a coefficient vector \( A \) with the random values greater than 1 or less than -1 is used to force the whale to move far away from the best known whale. The model can be mathematically expressed as follows:
\[ \bar{X}(t + 1) = \bar{X}_\text{rand}(t) - A \cdot \text{Dis}, \]
\[ \text{Dis} = |C \cdot \bar{X}_\text{rand}(t) - \bar{X}(t)|, \]  
(5)

where \( \bar{X}_\text{rand} \) is a random position vector selected from the current population.

2.2. Differential Evolution (DE). Differential evolution (DE) algorithm is a population-based approach firstly proposed by Storn and Price in 1997 [2] and proven to be one of the most promising global search methods. After generating initial population, the population is updated by looping mutation, crossover, and selection operations. This paper adopts a classic mutation operation, called DE/rand/1. In brief, DE procedures can be summarized in the following four steps.

2.2.1. Step 1: Initialization. At the beginning, an initial population is generated by means of random number distribution. We can initialize the \( j \)th dimension of the \( i \)th individual according to
\[ x_{i,j} = L_j + \text{rand}(0, 1) \cdot (U_j - L_j), \quad i = 1, 2, \ldots, N; \]
\[ j = 1, 2, \ldots, D, \]  
(6)

where \( N \) is the population size, \( D \) is the dimensional of individual, \( \text{rand}(0, 1) \) denotes an uniformly distributed random number in \([0, 1]\) range, and \( U_j \) and \( L_j \) denote the upper and lower bounds of the \( j \)th dimension, respectively.

2.2.2. Step 2: Mutation. After the initialization, DE enters its main loop. Each target individual \( x_i \) in the population is used to generate a mutant individual \( v_i \) via mutation operators (DE/rand/1), where the generated \( v_i \) can be represented as
\[ v_i = x_{i_1} + F \cdot (x_{i_2} - x_{i_3}) + F \cdot (x_{i_4} - x_{i_5}), \quad r_1 \neq r_2 \neq r_3 \neq i, \]
(7)

where \( r_1, r_2, \) and \( r_3 \) are randomly selected from the current population and \( F \) is the mutation control parameter to scale the difference vector. Similarly, we give another five frequently used mutation operators as follows:

1) "DE/rand/2"
\[ v_i = x_{i_1} + F \cdot (x_{i_2} - x_{i_3}) + F \cdot (x_{i_4} - x_{i_5}) + F \cdot (x_{i_6} - x_{i_7}), \quad r_1 \neq r_2 \neq r_3 \neq i. \]
(8)

2) "DE/best/1"
\[ v_i = x_{i_1} + F \cdot (x_{i_2} - x_{i_3}), \quad r_1 \neq r_2 \neq i. \]
(9)

3) "DE/best/2"
\[ v_i = x_{i_1} + F \cdot (x_{i_2} - x_{i_3}) + F \cdot (x_{i_4} - x_{i_5}), \quad r_1 \neq r_2 \neq r_3 \neq i. \]
(10)

4) "DE/rand-to-best/1"
\[ v_i = x_{i_1} + F \cdot (x_{i_2} - x_{i_3}) + F \cdot (x_{i_4} - x_{i_5}), \quad r_1 \neq r_2 \neq i. \]
(11)

5) "DE/current-to-best/1"
\[ v_i = x_{i_1} + F \cdot (x_{i_2} - x_{i_3}) + F \cdot (x_{i_4} - x_{i_5}), \quad r_1 \neq r_2 \neq r_3 \neq i. \]
(12)

where \( x_{i_1} \) is the individual that has the best fitness function value.

2.2.3. Step 3: Crossover. Then, a binomial crossover operation is applied to the target individual \( x_i \) and mutant individual \( v_i \) to generate a trial individual \( u_i \) as follows:
\[ u_{i,j} = \begin{cases} v_{i,j}, & \text{if } \text{rand}(0, 1) \leq \text{CR} \text{ or } j = j_{\text{rand}}, \\ x_{i,j}, & \text{otherwise}, \end{cases} \]
(13)

where \( j_{\text{rand}} \) is a randomly chosen integer in \([1, D]\) and \( \text{CR} \in [0, 1] \) is called the crossover control parameter.
2.2.4. Step 4: Selection. Finally, the greedy selection operation is used to select the better one from the target individual \( x_i \) and crossover individual \( u_i \) into the next generation. This operation is mainly based on the comparison of the fitness value and is performed as shown below:

\[
x_{t+1}^{i} = \begin{cases} 
    u_i, & \text{if fitness}(u_i) < \text{fitness}(x_i), \\
    x_i, & \text{otherwise},
\end{cases}
\]

where fitness is the fitness function.

2.3. Backtracking Search Optimization Algorithm (BSA). Backtracking search optimization algorithm (BSA) is a population-based metaheuristic algorithm, proposed by Pinar Civicioglu in 2013 [4]. Similar to most metaheuristic algorithms, the algorithm achieves the purpose of optimization through the mutation, crossover, and selection of population. What makes the BSA unique is its ability to remember historical populations, which enables it to benefit from previous generation populations by mining historical information. There are five steps contained in original BSA called initialization, selection-I, mutation, crossover, and selection-II. The five steps of BSA are simply introduced as follows.

2.3.1. Step 1: Initialization. BSA initializes the population \( P \) and the historical population \( \text{old } P \) with the following formula, respectively:

\[
P_{i,j} \sim U(\text{low}_j, \text{up}_j),
\]

\[
\text{old } P_{i,j} \sim U(\text{low}_j, \text{up}_j),
\]

where \( i = 1, 2, \ldots, N \), \( j = 1, 2, \ldots, D \). \( N \) and \( D \) represent the population size and the population dimension, respectively. \( U \) is the uniform distribution. \( \text{low}_j \) and \( \text{up}_j \) are the lower and upper boundaries of variables.

2.3.2. Step 2: Selection-I. Firstly, the historical population \( \text{old } P \) is updated according to equation (17). Then, the locations of individuals in \( \text{old } P \) are randomly changed as shown in equation (18).

\[
\text{if } a < b(a, b \sim U(0, 1)), \text{then old } P = P,
\]

\[
\text{old } P = \text{permuting}(\text{old } P),
\]

where permuting(\( \cdot \)) performs a random permutation of the integers from 1 to \( N \).

2.3.3. Step 3: Mutation. The mutation operator of BSA can generate the initial trial population which takes advantage of the historical information and the current information. The mutation operation can be expressed as follows:

\[
M_{i,j} = P_{i,j} + F \cdot (\text{old } P_{i,j} - P_{i,j}),
\]

where \( F \) is a control parameter and \( F = 3 \cdot \text{rand } n(\text{rand } n \sim U(0, 1)) \). This operation gives the algorithm powerful global search ability.

2.3.4. Step 4: Crossover. BSA’s crossover process has two steps. The first step generates a binary integer-valued matrix map of size \( N \times D \). The second step is to determine the positions of crossover individual elements in population \( P \) according to the generated matrix map and then exchange such individual elements in \( P \) with the corresponding position elements in population \( M \) to obtain the final trial population \( T \). The crossover operation can be expressed as follows:

\[
V_{i,j} = \begin{cases} 
    P_{i,j}, & \text{if map}_{i,j} = 1, \\
    M_{i,j}, & \text{otherwise}.
\end{cases}
\]

After crossover operation, some individuals of the trial population \( T \) obtained at the end of BSA’s crossover process may overflow the allowed search space limits. At this point, the individuals beyond the boundary will be regenerated according to equation (15).

2.3.5. Step 5: Selection-II. The greedy selection mechanism is employed to maintain the most promising trial individuals into the next generation. Compare the fitness values of the target individuals and trial individuals; if the fitness value of trial individual is less than that of the target individual, then the trial individual is accepted for the next generation; otherwise, the target individual is retained in the population. The selection operator is defined as

\[
P_{i} = \begin{cases} 
    T_{i}, & \text{if fitness}(T_{i}) < \text{fitness}(P_{i}), \\
    P_{i}, & \text{otherwise},
\end{cases}
\]

where \( f(\cdot) \) is the objective function value of an individual.

3. Proposed Hybrid Framework for WOA: hWOA

WOA which imitates the hunting behavior of humpback whales is one of the most recent SIs. It includes three stages: searching for prey, encircling prey, and hunting prey. The basic WOA has the disadvantage of weak exploration capability. This paper proposes a hybrid algorithm framework (hWOA) to improve the global search capability of WOA. Operators with complementary fusion features, DE/rand/1 operator of DE and mutation operator BSA, are embedded into WOA to form two new algorithms which include WOA-DE and WOA-BSA, respectively. In addition, a learning parameter \( lp \) adjusted by adaptive adjustment operator will replace the random parameter \( p \) of the original WOA, which can continuously balance exploration and exploitation through learning mechanism. The proposed framework consists of two main parts, hybrid framework and learning parameter mechanism, explained below.
3.1. Proposed Hybrid Framework for WOA. The mutation operators of equations (1) and (3) in WOA have strong exploitation capability, but only the mutation operator of equation (5) has a strong exploration capability. This makes WOA more capable of exploitation and less capable of exploration. In order to balance the exploration and exploitation capabilities of WOA, a generic framework flowchart was developed and is shown in Figure 1. With this framework, BSA mutation operator (equation (7)) and DE/rand/1 mutation operator (equation (19)) are used to improve the exploration capability of WOA.

In order to have a clearer understanding of our proposed hWOAlf, we summarize the four mutation operators used in this framework as follows:

(i) Operator 1: if \( p < r \) and \(|A| < 1\)
\[
\bar{T}_i = \bar{X}_i - \overrightarrow{A} \cdot \left| \overrightarrow{C} \cdot \bar{X}_i - \bar{X}_i \right|, \tag{22}
\]

(ii) Operator 2: if \( p < r \) and \(|A| \geq 1\)
\[
\bar{T}_i = \bar{X}_{\text{other}} - \overrightarrow{A} \cdot \left| \overrightarrow{C} \cdot \bar{X}_{\text{other}} - \bar{X}_i \right|, \tag{23}
\]

(iii) Operator 3: if \( p \geq r \) and \(|A| < 1\)
\[
\bar{T}_i = \left| \bar{X}_i - \bar{X}_r \right| \cdot e^{\overrightarrow{i} \cdot \cos (2\pi l)} + \bar{X}_i, \tag{24}
\]

(iv) Operator 4: if \( p \geq r \) and \(|A| \geq 1\)
\[
\bar{T}_i = \bar{X}_i + F_1 \cdot \left( \text{old} \bar{X}_i - \bar{X}_r \right) \text{or} \bar{T}_i = \bar{X}_{x1} + F_2 \cdot \left( \bar{X}_{x2} - \bar{X}_{x3} \right), \tag{25}
\]

where \( T_i \) is a test individual.

3.2. Proposed Learning Parameter Mechanism. In basic WOA, \( p \) is a random number. In this paper, we design \( lp \) as a function with learning strategy. When the population is fully evolved in the current iteration, the following four numbers are recorded: the number of individuals mutated by operator 1 and operator 2 is \( n1 \), and the number of individuals mutated by these two operators that can successfully enter the next generation is \( s1 \); similarly, the number of individuals mutated by operator 3 and operator 4 is \( n2 \). The number of individuals better than their parents after the mutation of these two operators is \( s2 \). The learning parameter \( lp \) is designed as follows:
\[
lp = \frac{1 + (s1/n1)}{2 + (s1/n1) + (s2/n2)} \tag{26}
\]

The initial \( lp \) value is 0.5 and needs to be updated after each iteration. In this paper, \( lp \) is designed to control the execution frequency of the search operators in equations (22) and (23) and those in equations (24) and (25). In the process of algorithm evolution, when \( lp \) is less than a random number, operator 1 and operator 2 are used to mutate; otherwise, operator 3 and operator 4 are selected to mutate. This continuous adaptive process will enable the algorithm to choose the most suitable learning strategy in the evolution process.

3.3. The Pseudocode of hWOAlf. For clarity, the pseudocode of hWOAlf is given in Algorithm 1.

4. Experimental Results and Discussion

In this section, to verify the validity of the proposed hybrid algorithm framework, WOA-DE and WOA-BSA are demonstrated on twenty-three benchmark functions and six engineering design problems. Experimental results and comparisons with other representative algorithms show that the proposed hWOAlf is effective and efficient. All experiments are implemented in the software of MATLAB 2012a. The computer system running the program is Intel(R) Core(TM) i5-3230M CPU @ 2.60 GHz and 4 GB RAM.

4.1. Twenty-Three Benchmark Functions. In order to evaluate the performance of the proposed algorithm framework, WOA-DE and WOA-BSA are compared with some EAs and SIs, i.e., DE, BBO, BSA, PSO, HS, GSA, and WOA on twenty-three benchmark functions listed in Tables 1–3. These functions can be divided into three groups: unimodal, multimodal, and fixed-dimension multimodal. The unimodal functions \( F_1 \) to \( F_7 \) have only one global optimum, so they are used to investigate the exploitation capability, while the remaining functions \( F_8 \) to \( F_{23} \) with more than two local optima are utilized to evaluate the exploration ability. In addition, in the tables, \( D \) denotes the number of function dimensions, Range is the boundary of the function search space, and \( f_{\text{min}} \) is the theoretical optimum. In order to have a fair comparison, each experiment is independently run 30 times for all the compared algorithms, and the results of every run are recorded. The maximal iteration number (Maxiter) is used as the termination condition for all algorithms, which is set to 1000, and population size is set to 30.

4.2. Comparison Results of Twenty-Three Benchmark Functions. Tables 4 and 5 tabulate the Best, Mean, and Std results of twenty-three benchmark functions for WOA-DE, WOA-BSA, and other seven compared algorithms. The results of WOA-DE and WOA-BSA are shown in bold.

4.2.1. Exploitation Capability. Functions \( F_1 \) to \( F_7 \) can be used to evaluate the exploitation capability of the proposed algorithms, since each of them has only one global optimal value. The results from Table 4 show that with respect to the best, WOA-DE and WOA-BSA rank first and second, respectively, for functions \( F_1 \) to \( F_7 \) among all compared algorithms, which means that the results obtained by WOA-DE
and WOA-BSA are closer to the theoretical optimal solutions on these functions and WOA-DE and WOA-BSA have very strong exploitation capacity compared with other algorithms. In terms of Mean and Std index, WOA-BSA outperforms the other algorithms, while WOA-DE is not the best but it is better than the other six algorithms.

### 4.2.2. Exploration Capability

Functions $F_8$–$F_{23}$ are multimodal functions. They include many local optima, and the number of local optima increases exponentially with the increase of dimension $D$. So, these functions are considered to be very useful in evaluating the exploration capabilities of algorithms. The results of $F_8$–$F_{23}$ test functions for WOA-DE and WOA-BSA are shown in Table 5, where $F_8$–$F_{13}$ are multimodal functions and $F_{14}$–$F_{23}$ are fixed-dimension multimodal functions. The results from Table 5 show that WOA-DE and WOA-BSA have no advantage in terms of Best, Mean, and Std compared to the other six algorithms (such as DE, BBO, BSA, PSO, HS, and GSA). However, both of them are better than the original algorithm WOA, which indicates that our proposed hybrid algorithm framework can effectively improve the performance of the original algorithm WOA.

Figure 2 represents six convergence graphs obtained by the nine algorithms for $F_1$, $F_3$, and $F_7$ (unimodal functions), $F_{10}$ and $F_{11}$ (multimodal functions), and $F_{12}$ (hybrid composition function). As can be seen from Figure 2, WOA-BSA has faster convergence speed for $F_1$, $F_3$, $F_7$, $F_{10}$, $F_{11}$, and $F_{12}$; meanwhile, WOA-DE also has faster speed. Based on the above results, we can see that WOA-DE and WOA-BSA have competitive performance, although they are not better than the compared algorithms in all aspects.

### 4.3. Engineering Design Problems

In order to further examine the proposed hybrid framework, it is applied to six well-known engineering design problems mixed the variables taken from the optimization literatures. The description of these problems including three-bar truss, pressure vessel, speed reducer, gear train, cantilever beam, and I-beam is given in Appendix A. Table 6 shows the parameter settings of six engineering problems according reference [23, 24], where $N$, $T$, and $D$ represent the population size, the maximum number of iterations, and the dimension of optimization problems, respectively. However, there are a number of complex constraints in each problem, which make it challenging to find the optimal solution satisfying all the constraints. So, a constraint handling method, such as penalty functions, feasibility and dominance rules, stochastic ranking, multiobjective concepts, ε-constrained methods, and Deb’s heuristic constraints processing method [25, 26],

![Flowchart of the proposed hWOAlf.](image-url)
This paper adopts Deb’s heuristic constraints processing method. Each experiment is independently run 30 times and the results obtained include the worst value (Worst), the mean value (Mean), the best function value (Best), the standard deviation (Std), and the function evaluations (FEs).

### 4.3.1 Three-Bar Truss Design Problem

Three-bar truss design (as shown in Figure 3) is to minimize the weight of a truss with three bars, subject to stress constraints. The problem has two design variables and three nonlinear inequality constraints. The formulation for this problem is shown in Appendix A.1.

---

**Algorithm 1: The pseudocode of hWOAlf.**

```plaintext
Input: ObjFun, N, D, T_{max}, lb_{1:D}, ub_{1:D}
Output: Leaderminimum, X^*

(1) //rand ~ U (0, 1), randperm (N) ~ random permutation of the integers 1 : N, normrand (u, d) ~ random number chosen from a normal distribution with mean u and standard deviation d

(2) **Initialization for** i = 1 to N **do**
(3) for j = 1 to D **do**
(4) x_{ij} = lb_{j} + rand * ub_{j}, % Generate an initial whales population P = \{X_1, \ldots, X_N\}
(5) end
(6) end
(7) fitness = ObjFun(P) % Calculate the fitness of each search agent
(8) X^* = the best search agent
(9) Leaderminimum = the fitness of the best search agent
(10) t = 1 and Ip = 0.5
(11) while t ≤ T_{max} **do**
(12) **Mutation**
(13) % Update the Position of search agents
(14) for i = 1 to N **do**
(15) Update a, A, C and l
(16) if Ip < rand then
(17) if |A| < 1 then
(18) the individual is mutated by equation (22)
(19) else
(20) Select a random individual X_{rand}; the individual is mutated by equation (23)
(21) end
(22) else
(23) if |A| < 1 then
(24) the individual is mutated by equation (24)
(25) else
(26) the individual is mutated by equation (25)
(27) end
(28) end
(29) % Checking allowable range
(30) for i = 1 to N **do**
(31) for j = 1 to D **do**
(32) if x_{ij} < lb_{j} or x_{ij} > ub_{j} then
(33) x_{ij} = lb_{j} + rand * ub_{j}
(34) end
(35) end
(36) end
(37) % Update the leader
(38) for i = 1 to N **do**
(39) fitness(i) = ObjFun(X_i) % Calculate objective function values of T
(40) if fitness(i) < Leaderminimum then
(41) Leaderminimum = fitness(i)
(42) X^* = X_i
(43) end
(44) end
(45) % Update the learning parameter Ip
(46) if the Ip is updated by equation (26)
(47) t = t + 1
(48) end
(49) end
```
The results of WOA-DE and WOA-BSA are compared with WOA, dynamic stochastic selection (DEDS) [27], hybrid evolutionary algorithm (HEAA) [28], hybrid particle swarm optimization with differential evolution (PSO-DE) [29], differential evolution with level comparison (DELC) [30], mine blast algorithm (MBA) [31], and crow search algorithm (CSA) [32]. The best solutions obtained from these algorithms are presented in Table 7. From Table 7, within the allowable error range of the constraint function value, the function value of the best solutions of WOA-DE and WOA-BSA is the smallest among all the comparison algorithms, equaling to 263.895711. As can be seen from Table 8, although the FEs (equaling to 10000) obtained by DELC are the smallest, WOA-DE and WOA-BSA find out the current best solution of this problem. In addition, Figure 4 shows the convergence figure of WOA-DE, WOA-BSA, and WOA on three-bar truss design problem, which indicates the convergence speed of WOA-DE, and WOA-BSA is faster than that of WOA. That is to say, the proposed hybrid framework is effective.

### 4.3.2. Pressure Vessel Design Problem

This problem aims to minimize the total cost of a cylindrical vessel subject to constraints on the material, forming, and welding as shown in Figure 5. From this picture, we can see that when the head is hemispherical and both ends of the cylindrical container are sealed by hemispherical heads. Four optimization variables are the thickness of the shell $T_s (x_1)$, thickness of the head $T_h (x_2)$, inner radius $R (x_3)$, and cylindrical section length $L (x_4)$, where $T_s$ and $T_h$ need to be integer multiple of 0.0625, while $R (x_3)$ and $L (x_4)$ are continuous variables. In addition, the formulation of this problem is shown in Appendix A.2.

The approaches applied to this problem include GA based on coevolution model (CGA) [33], GA based on a dominance-based tour tournament selection (DGA) [34], coevolutionary PSO (CPSO) [35], hybrid PSO (HPSO) [36], coevolutionary differential evolution (CDE) [37], CSA, and WOA. WOA-DE and WOA-BSA are compared with the seven algorithms listed above, and the best solutions obtained by these methods are shown in Table 9. From Table 9, we can see that the objective function values of the best solutions of WOA-DE (equaling to 6059.708057) and WOA-BSA (equaling to 6059.708025) are superior to those of comparison algorithms. Moreover, as can be seen from Table 10, the FEs obtained by the proposed WOA-DE and WOA-BSA are minimal, equaling to 30240 and 30140, respectively. This suggests that WOA-DE and WOA-BSA can find out the current best solution of this problem with the smallest computational overhead. Figure 6 shows that WOA-DE and WOA-BSA have a faster convergence rate than the original WOA on pressure vessel design issues.

### 4.3.3. Speed Reducer Design Problem

The speed reducer design problem, shown in Figure 7, is a practical design problem that has been often used as a benchmark problem for testing the performance of different optimization algorithms. The objective is to find the minimum weight of speed reducer under the constraints of bending stress of the gear teeth, surface stress, transverse deflections of the shafts, and stresses in the shafts. There are seven design variables, namely, the face width $b (x_1)$, the tooth module $m (x_2)$, the number of teeth in pinion $z (x_3)$, the length of the first shaft between bearings $l_1 (x_4)$, the length of the second shaft between bearings $l_2 (x_5)$, and the diameter of the first shaft $d_1 (x_6)$ and the second shaft $d_2 (x_7)$. The mathematical formulation of the problem is shown in Appendix A.3.

Tables 11 and 12 show the best solutions and statistical results of MDE, DEDS, DELC, HEAA, PSO-DE, MBA, and WOA. From Table 11, it can be seen that the function values of the current best feasible solutions of WOA-DE and WOA-BSA are equal to 2994.469448 and 2994.469656, respectively. The statistical results are given in Table 12. From the results of this table, we can see that the FEs of WOA-DE and WOA-BSA equaling to 6300 are better than those of the other algorithms except MBA equaling to 6300. This indicates that WOA-DE and WOA-BSA in the framework, compared to other algorithms, can obtain a relatively better solution by sacrificing a little computational overhead on this problem. Moreover, Figure 8 presents the convergence figure of WOA-DE, WOA-BSA, and WOA on this problem. From Figure 8, the convergence speeds of WOA-DE and WOA-BSA are faster than WOA, and their solutions are closer to the optimal solution.

### 4.3.4. Gear Train Design Problem

Gear train design aims to minimize the cost of the gear ratio of the gear train. It has only boundary constraints in parameters. Variables to be optimized are in discrete form since each gear has to have an integral number of teeth. In the problem, the design variables are $n_a (x_1)$, $n_b (x_2)$, $n_d (x_3)$, and $n_f (x_4)$, which are discrete and must be integers. The gear ratio is defined as $n_b / n_d / n_a$. The formulation of this problem is shown in Appendix A.4. Figure 9 shows the speed reducer and its parameters.
| Function | $D$ | Range | $f_{\text{min}}$ |
|----------|----|-------|----------------|
| $F_k(x) = \sum_{i=1}^{D} - x_i \sin(\sqrt[3]{x_i})$ | 30 | $[-500, 500]$ | -418.9829 x 5 |
| $F_8(x) = \sum_{i=1}^{D} (x_i^2 - 10 \cos(2\pi x_i) + 10)$ | 30 | $[-5.12, 5.12]$ | 0 |
| $F_{10}(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D} \sum_{i=1}^{D} \cos(2\pi x_i)\right) + 20 + e$ | 30 | $[-32, 32]$ | 0 |
| $F_{11}(x) = (1/4000) \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos(x_i / \sqrt{D}) + 1$ | 30 | $[-600, 600]$ | 0 |
| $F_{12}(x) = \pi / D \left[ 10 \sin(\pi y_1) + \sum_{i=1}^{D} (y_i - 1)^2 \left[ 1 + 10 \sin^2(\pi y_{i+1}) \right] + (y_D - 1)^2 \right] + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$ | 30 | $[-50, 50]$ | 0 |
| $y_i = 1 + x_i + 1/4, \quad u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a < x_i < a \\ k(-x_i - a)^m, & x_i < a \end{cases}$ | | | |
| $F_{13}(x) = 0.1 \left[ \sin^2(3\pi x_1) + \sum_{i=1}^{D} (x_i - 1)^2 \left[ 1 + \sin^2(3\pi x_{i+1} + 1) \right] + (x_D - 1)^2 \left[ 1 + \sin^2(2\pi x_D) \right] \right] + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$ | 30 | $[-50, 50]$ | 0 |
Table 3: Description of fixed-dimension multimodal benchmark functions.

| Function | $D$ | Range   | $f_{\text{min}}$ |
|----------|-----|---------|-----------------|
| $F_{14}(x) = ((1/500) + \sum_{j=1}^{25} (1/j + \sum_{i=1}^{j} (x_i - a_{ij})^{6}))^{-1}$ | 2   | $[-65, 65]$ | 1               |
| $F_{15}(x) = \sum_{i=1}^{11} |a_i - (x_1 (b_i^{2} + b_i x_2) b_i^{2} + b_i x_3 + x_4)|$ | 4   | $[-5, 5]$  | 0.00030         |
| $F_{16}(x) = \sum_{i=1}^{10} |x_1 - (5/4 \pi x_1^{2} + (5/\pi x_1 - 6)^{2} + 10 (1 - (1/8 \pi) \cos x_1) + 10 |$ | 2   | $[-5, 5]$  | -1.0316         |
| $F_{17}(x) = \sum_{i=1}^{10} |x_1 - (5/4 \pi x_1^{2} + (5/\pi x_1 - 6)^{2} + 10 (1 - (1/8 \pi) \cos x_1) + 10 |$ | 2   | $[-2, 2]$  | 0.398           |
| $F_{18}(x) = \sum_{i=1}^{10} |x_1 - (5/4 \pi x_1^{2} + (5/\pi x_1 - 6)^{2} + 10 (1 - (1/8 \pi) \cos x_1) + 10 |$ | 3   | $[1, 3]$   | -3.86           |
| $F_{19}(x) = \sum_{i=1}^{10} |x_1 - (5/4 \pi x_1^{2} + (5/\pi x_1 - 6)^{2} + 10 (1 - (1/8 \pi) \cos x_1) + 10 |$ | 6   | $[0, 1]$   | -3.32           |
| $F_{20}(x) = \sum_{i=1}^{10} |x_1 - (5/4 \pi x_1^{2} + (5/\pi x_1 - 6)^{2} + 10 (1 - (1/8 \pi) \cos x_1) + 10 |$ | 4   | $[0, 10]$  | -10.1232        |
| $F_{21}(x) = \sum_{i=1}^{10} |x_1 - (5/4 \pi x_1^{2} + (5/\pi x_1 - 6)^{2} + 10 (1 - (1/8 \pi) \cos x_1) + 10 |$ | 4   | $[0, 10]$  | -10.4028        |
| $F_{22}(x) = \sum_{i=1}^{10} |x_1 - (5/4 \pi x_1^{2} + (5/\pi x_1 - 6)^{2} + 10 (1 - (1/8 \pi) \cos x_1) + 10 |$ | 4   | $[0, 10]$  | -10.5363        |
| AL | $f^{\text{rank}}$ | DE | BBO | BSA | PSO | HS | GSA | WOA | WOA-DE | WOA-BSA |
|----|-----------------|----|-----|-----|-----|----|-----|-----|--------|---------|
| $F_1$ | Best | 2.90E-12 | 0.454643 | 0.000908 | 1.04E-11 | 0.042385 | 4.64E-17 | 2.76E-167 | 6.92E-190 | 1.62E-187 |
| | Mean | 1.33E-11 | 1.013223 | 0.006257 | 9.73E-09 | 0.063236 | 1.01E-16 | 1.46E-153 | 1.07E-168 | 1.45E-172 |
| | Std | 9.98E-12 | 0.311571 | 0.010917 | 4.23E-08 | 0.010899 | 3.76E-17 | 7.91E-153 | 0 | 0 |
| $F_2$ | Best | 3.99E-08 | 0.271733 | 0.006726 | 1.00E-05 | 0.575724 | 3.50E-08 | 3.31E-114 | 2.46E-124 | 2.01E-124 |
| | Mean | 6.80E-08 | 0.324236 | 0.020418 | 0.001313 | 0.727765 | 5.20E-08 | 9.88E-105 | 8.47E-113 | 1.27E-112 |
| | Std | 1.95E-08 | 0.041092 | 0.014974 | 0.004947 | 0.05865 | 1.17E-08 | 4.94E-104 | 4.21E-112 | 5.75E-112 |
| $F_3$ | Best | 10808.03597 | 83.765444 | 394.952176 | 3.208562 | 3084.50285 | 195.538197 | 1677.30974 | 175.10162 | 532.871248 |
| | Mean | 15723.1259 | 135.934273 | 878.22206 | 14.904551 | 5228.201445 | 9735.512836 | 10045.18144 | 7621.728202 | 4199.038749 |
| | Std | 2683.155994 | 34.026326 | 315.589525 | 7.677907 | 1591.552755 | 129.835093 | 3898.890192 | 5986.48178 | 6.519626 |
| $F_4$ | Best | 1.080885 | 0.742244 | 3.262876 | 0.311111 | 4.216935 | 1.35E-08 | 0.019063 | 5.68E-10 | 4.30E-19 |
| | Mean | 1.575045 | 0.972736 | 5.969996 | 0.563087 | 6.113542 | 1.264362 | 17.704825 | 14.259819 | 1.562264 |
| | Std | 0.38751 | 0.123737 | 1.579686 | 0.153909 | 1.246667 | 1.284382 | 14.58112 | 17.560992 | 6.519626 |
| $F_5$ | Best | 25.398273 | 39.474199 | 43.67835 | 11.716821 | 91.050677 | 25.861836 | 26.607341 | 25.886812 | 26.835697 |
| | Mean | 38.31135 | 149.884822 | 114.032052 | 60.388778 | 342.973593 | 37.244847 | 27.1471 | 27.512334 | 27.496506 |
| | Std | 21.417578 | 35.092327 | 35.216402 | 43.241123 | 492.14478 | 24.882941 | 0.485147 | 0.547343 | 0.539586 |
| $F_6$ | Best | 2.88E-12 | 0.62279 | 0.000764 | 2.51E-11 | 0.041828 | 4.50E-17 | 0.007148 | 0.012986 | 0.015406 |
| | Mean | 8.90E-12 | 1.071114 | 0.007884 | 9.68E-09 | 0.070002 | 1.03E-16 | 0.057661 | 0.263633 | 0.201257 |
| | Std | 5.67E-12 | 0.279422 | 0.008319 | 3.00E-08 | 0.011199 | 4.46E-17 | 0.078559 | 0.224346 | 0.192754 |
| $F_7$ | Best | 0.016252 | 0.002011 | 0.012074 | 0.037104 | 0.035229 | 0.016733 | 0.000115 | 0.00016 | 0.0001 |
| | Mean | 0.027322 | 0.006396 | 0.030676 | 0.078177 | 0.055369 | 0.049762 | 0.001935 | 0.001579 | 0.001579 |
| | Std | 0.005743 | 0.002154 | 0.012799 | 0.023769 | 0.011801 | 0.021587 | 0.001959 | 0.001692 | 0.001761 |
| Rank | Best | 6 | 8 | 7 | 6 | 4 | 8 | 2 | 3 | 1 |
| | Mean | 5 | 7 | 6 | 4 | 8 | 2 | 3 | 3 | 1 |
| | Std | 1 | 5 | 6 | 4 | 7 | 1 | 2 | 3 | 1 |
| AL | F<sup>n</sup> | DE | BBO | BSA | PSO | HS | GSA | WOA | WOA-DE | WOA-BSA |
|----|------------|----|-----|-----|-----|----|-----|-----|--------|---------|
| 12 | −12569.4866 | −9288.59115 | −12398.8552 | −8562.80236 | −12569.47154 | −3257.715639 | −8674.343572 | −8985.661303 | −10577.83435 | −1183224502 |
| 66 | 21.623781 | 497.301943 | 222.385063 | 1574.830603 | 0.990111 | 355.9370954 | 863.9712672 | 861.1801078 | 1183224502 |
| 12569.4866 | −292.44538 | 15.69597 | 14.9929932 | 17.9498293 | 14.9243812 | 0 | 0 | 0 | 0 |
| 537.843758 | 54.3792857 | 25.786533 | 46.538306 | 25.4645625 | 25.8025875 | 0 | 0 | 0 | 0 |
| 1265.2567 | 0.7051995 | 0.0113335 | 0.3174959 | 0.4253235 | 1.50e−09 | 0 | 0 | 0 | 0 |
| 1265.2567 | 6.2664864 | 14.3334357 | 4.03673 | 14.2792972 | 4.7726843 | 6.63041 | 0 | 0 | 0 |
| 1265.2567 | 4.37E−07 | 0.226971 | 0.0075303 | 3.00E−04 | 1.0809697 | 5.45E−09 | 8.88E−16 | 8.88E−16 | 0 |
| 1265.2567 | 7.0E−07 | 0.0294424 | 0.083237 | 2.0663489 | 7.70E−09 | 4.44E−15 | 4.20E−15 | 3.97E−15 | 0 |
| 1265.2567 | 1.52E−07 | 0.0705995 | 0.0113335 | 0.3174959 | 0.4253235 | 1.50e−09 | 0 | 0 | 0 |
| 1265.2567 | 2.36E−07 | 0.205077 | 0.083237 | 2.0663489 | 7.70E−09 | 4.44E−15 | 4.20E−15 | 3.97E−15 | 0 |
| 1265.2567 | −1.0316281 | −1.0316281 | −1.0316281 | −1.0316281 | −1.0316281 | −1.0316281 | −1.0316281 | −1.0316281 | −1.0316281 |
| 1265.2567 | 6.71E−16 | 0.0207068 | 6.12E−16 | 6.78E−16 | 5.35E−08 | 5.13E−16 | 3.86E−11 | 6.71E−16 | 5.98E−16 |
| 1265.2567 | 0.3978871 | 0.3978871 | 0.3978871 | 0.3978871 | 0.3978871 | 0.3978871 | 0.3978871 | 0.3978871 | 0.3978871 |
| 1265.2567 | 9.33E−18 | 9.33E−18 | 9.33E−18 | 9.33E−18 | 9.33E−18 | 9.33E−18 | 9.33E−18 | 9.33E−18 | 9.33E−18 |
| 1265.2567 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 1265.2567 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 1265.2567 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 1265.2567 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 1265.2567 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 1265.2567 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 1265.2567 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 1265.2567 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| AL | F_{rank} | DE         | BBO        | BSA        | PSO        | HS         | GSA        | WOA        | WOA-DE     | WOA-BSA    |
|----|----------|------------|------------|------------|------------|------------|------------|------------|------------|------------|
|    | Best     | -10.402941 | -10.402941 | -10.402941 | -10.402941 | -10.402941 | -10.402941 | -10.402941 | -10.402941 | -10.402941 |
|    | Mean     | -10.211001 | -5.529595  | -10.402941 | -9.521178  | -6.545124  | -10.402941 | -10.402941 | -8.448955  | -7.763089  |
|    | Std      | 1.045029   | 3.343034   | 1.69E-14   | 2.005406   | 3.686937   | 1.32E-15   | 2.836906   | 3.128103   | 2.86345    |
| F_{22} | Best     | -10.53641  | -10.53641  | -10.53641  | -10.53641  | -10.53641  | -10.53641  | -10.53641  | -10.53641  | -10.53641  |
|      | Mean     | -10.536383 | -5.855753  | -10.53641  | -9.16093   | -5.180275  | -10.175835 | -9.03007   | -7.990287  | -8.647563  |
|      | Std      | 0.000108   | 3.655594   | 1.11E-09   | 2.570428   | 3.329307   | 1.485436   | 2.591415   | 3.225805   | 2.733822   |

| Rank | Best | 1 | 8 | 3 | 2 | 7 | 5 | 6 | 4 | 2 |
|------|------|---|---|---|---|---|---|---|---|---|
| Mean | 2    | 9 | 1 | 6 | 8 | 7 | 5 | 3 | 4 | 4 |
| Std  | 1    | 9 | 2 | 6 | 8 | 3 | 7 | 4 | 5 | 5 |
Table 13 shows the comparison of best solutions for artificial bee colony algorithm (ABC) [39], MBA, CSA, WOA, WOA-DE, and WOA-BSA. From the table, we can see that the best solutions obtained by the six algorithms are the same (equaling to $2.7e-12$). In addition, the statistical results of all six algorithms are shown in Table 14. WOA-DE
with FEs of 940 and WOA-BSA with FEs of 1900 rank second and fourth, respectively, among all the six compared algorithms. The FEs of the top one are 60 belonging to ABC. In addition, WOA-DE and WOA-BSA have faster convergence speeds than WOA from the function convergence graphs of WOA-DE, WOA-BSA, and WOA (see Figure 10).

4.3.5. Cantilever Beam Design Problem. The objective of cantilever beam design problem is to minimize the weight of a cantilever beam consisting of five hollow sections with fixed diameters as shown in Figure 11. The design variables are the heights of all beam elements, and there is a constraint that should not be violated by the final optimization design. The formulation of this problem is shown in Appendix A.5.

Table 15 provides the comparison results between WOA-DE, WOA-BSA, WOA, MMA [40], GCAI [40], GCAII [40], cuckoo search algorithm (CS) [41], symbiotic organisms search (SOS) [42], and moth-flame optimization algorithm (MOF) [43]. The results in Table 15 show that WOA-DE and WOA-BSA can also effectively solve such problems. Convergence plot using WOA-DE, WOA-BSA, and WOA for
Table 9: Comparison of best solutions of the pressure vessel design problem.

| Method     | Worst        | Mean          | Best          | Std            | FEs   |
|------------|--------------|---------------|---------------|----------------|-------|
| CGA [33]   | 6308.497000  | 6293.843200   | 6288.744500   | 7.413300       | 900,000 |
| DGA [34]   | 6469.322000  | 6177.253300   | 6059.946300   | 130.929700     | 80,000  |
| CPSO [35]  | 6363.804100  | 6147.133200   | 6061.077700   | 86.450000      | 240,000 |
| HPSO [36]  | 6288.677000  | 6099.932300   | 6059.714300   | 86.200000      | 81,000  |
| CDE [37]   | 6371.045500  | 6085.230300   | 6059.734000   | 43.013000      | 204,800 |
| CSA [32]   | 7332.841621  | 6342.499106   | 6059.714363   | 384.945416     | 250,000 |
| WOA        | 8564.987629  | 7124.412382   | 6059.211997   | 715.900580     | 99,940  |
| WOA-DE     | 44828.494260 | 8946.493319   | 6059.708057   | 7565.793120    | 30,240  |
| WOA-BSA    | 12956.012602 | 7114.983582   | 6059.708025   | 1679.555214    | 30,140  |

Table 10: Comparison of statistical results of the pressure vessel design problem.

| Method     | Worst        | Mean          | Best          | Std            | FEs   |
|------------|--------------|---------------|---------------|----------------|-------|
| CGA [33]   | 6308.497000  | 6293.843200   | 6288.744500   | 7.413300       | 900,000 |
| DGA [34]   | 6469.322000  | 6177.253300   | 6059.946300   | 130.929700     | 80,000  |
| CPSO [35]  | 6363.804100  | 6147.133200   | 6061.077700   | 86.450000      | 240,000 |
| HPSO [36]  | 6288.677000  | 6099.932300   | 6059.714300   | 86.200000      | 81,000  |
| CDE [37]   | 6371.045500  | 6085.230300   | 6059.734000   | 43.013000      | 204,800 |
| CSA [32]   | 7332.841621  | 6342.499106   | 6059.714363   | 384.945416     | 250,000 |
| WOA        | 8564.987629  | 7124.412382   | 6059.211997   | 715.900580     | 99,940  |
| WOA-DE     | 44828.494260 | 8946.493319   | 6059.708057   | 7565.793120    | 30,240  |
| WOA-BSA    | 12956.012602 | 7114.983582   | 6059.708025   | 1679.555214    | 30,140  |
Figure 6: Convergence plots using WOA, WOA-DE, and WOA-BSA for pressure vessel design problem.

![Convergence plots using WOA, WOA-DE, and WOA-BSA for pressure vessel design problem.](image)

Figure 7: Speed reducer design problem.

![Speed reducer design problem.](image)

Table 11: Comparison of best solutions for the speed reducer design problem.

| Method      | X_1    | X_2    | X_3    | X_4    | X_5    | X_6    | X_7    |
|-------------|--------|--------|--------|--------|--------|--------|--------|
| Method      | 3.500010 | 0.700000 | 17.000000 | 7.300156 | 7.800027 | 3.350221 | 5.286685 |
| MDE [38]    | 3.500000 | 0.700000 | 17.000000 | 7.300000 | 7.715320 | 3.350215 | 5.286654 |
| DEDS [27]   | 3.500000 | 0.700000 | 17.000000 | 7.300000 | 7.715320 | 3.350215 | 5.286654 |
| DELC [30]   | 3.500023 | 0.700000 | 17.000013 | 7.300428 | 7.715377 | 3.350231 | 5.286664 |
| HEAA [28]   | 3.500000 | 0.700000 | 17.000000 | 7.300000 | 7.715772 | 3.350215 | 5.286683 |
| PSO-DE [29] | 3.500000 | 0.700000 | 17.000000 | 7.300000 | 7.942261 | 3.350218 | 5.286654 |
| MBA [31]    | 3.500000 | 0.700000 | 17.000000 | 7.300000 | 7.919344 | 3.350218 | 5.286654 |
| WOA         | 3.500000 | 0.700000 | 17.000000 | 7.300000 | 7.715310 | 3.350218 | 5.286654 |
| WOA-DE      | 3.500000 | 0.700000 | 17.000000 | 7.300000 | 7.715310 | 3.350218 | 5.286654 |
| WOA-BSA     | 3.500000 | 0.700000 | 17.000000 | 7.300000 | 7.715310 | 3.350218 | 5.286654 |

f(X) values:

- MDE [38]: 2996.356689
- DEDS [27]: 2994.471066
- DELC [30]: 2994.471066
- HEAA [28]: 2994.471066
- PSO-DE [29]: 2994.471066
- MBA [31]: 2994.471066
- WOA: 2994.471066
- WOA-DE: 2994.471066
- WOA-BSA: 2994.471066

Table 12: Comparison of statistical results for the speed reducer design problem.

| Method      | Worst   | Mean    | Best    | Std     | FEs     |
|-------------|---------|---------|---------|---------|---------|
| MDE [38]    | 2996.390137 | 2996.367220 | 2996.356689 | 8.2e-03 | 24,000  |
| DEDS [27]   | 2994.471066 | 2994.471066 | 2994.471066 | 3.58e-12 | 30,000  |
| DELC [30]   | 2994.471066 | 2994.471066 | 2994.471066 | 1.9e-12  | 30,000  |
| HEAA [28]   | 2994.752311 | 2994.613368 | 2994.499107 | 7.0e-02  | 40,000  |
| PSO-DE [29] | 2996.348204 | 2996.348174 | 2996.348167 | 6.4e-06  | 54,350  |
| MBA [31]    | 2999.652444 | 2996.769019 | 2994.482453 | 1.560000 | 6300    |
| WOA         | 3561.958300 | 3065.730259 | 3005.192607 | 108.307891 | 15,000  |
| WOA-DE      | 3013.442478 | 2998.271376 | 2994.469448 | 4.750316  | 15,000  |
| WOA-BSA     | 3007.155564 | 2997.610011 | 2994.469656 | 3.292733  | 15,000  |
Figure 8: Convergence plots using WOA, WOA-DE, and WOA-BSA for speed reducer design problem.

Figure 9: Gear train design problem.

Table 13: Comparison of best solutions of the gear train design problem.

| Method   | ABC [39] | MBA [31] | CSA [32] | WOA  | WOA-DE | WOA-BSA |
|----------|----------|----------|----------|------|--------|---------|
| $X_1$    | 49.00000 | 43.00000 | 49.00000 | 43.000000 | 49.000000 | 43.000000 |
| $X_2$    | 16.00000 | 16.00000 | 19.00000 | 16.000000 | 19.000000 | 19.000000 |
| $X_3$    | 19.00000 | 19.00000 | 16.00000 | 19.000000 | 16.000000 | 16.000000 |
| $X_4$    | 43.00000 | 49.00000 | 43.00000 | 49.000000 | 43.000000 | 49.000000 |
| $f(X)$   | 2.7e−12  | 2.7e−12  | 2.7e−12  | 2.7e−12  | 2.7e−12  | 2.7e−12  |

Table 14: Comparison of statistical results of the gear train design problem.

| Method   | Worst   | Mean     | Best     | Std        | FEs   |
|----------|---------|----------|----------|------------|-------|
| ABC [39] | N.A     | 3.6e−10  | 2.7e−12  | 5.52E−10   | 60    |
| MBA [31] | 2.1e−08 | 2.5e−09  | 2.7e−12  | 3.94E−09   | 1120  |
| CSA [32] | 3.2e−08 | 2.06e−09 | 2.7e−12  | 5.1e−09    | 100,000 |
| WOA     | 2.73E−08| 3.26E−09 | 2.7E−12  | 5.97E−09   | 6800  |
| WOA-DE  | 4.47E−08| 4.80E−09 | 2.7E−12  | 9.51E−09   | 940   |
| WOA-BSA | 3.98E−08| 4.03E−09 | 2.7E−12  | 7.40E−09   | 1900  |
cantilever beam design problem (as shown in Figure 12) indicates that WOA-DE and WOA-BSA have faster rates of convergence, which are close to the optimal solution.

4.3.6. I-Beam Design Problem. In this problem, the goal is to minimize the vertical deflection of the beam shown in Figure 13. There are four design variables for this problem: length \((x_1)\), height \((x_2)\), and two thicknesses \((x_3, x_4)\). The problem includes only one optimization constraint and is formulated in Appendix A.6.

Table 16 compares the best solutions of seven state-of-the-art methods including adaptive response surface method (ARSM) [44], improved ARSM (IARSM) [44], CS, SOS, WOA, WOA-DE, and WOA-BSA for I-beam design problem. From the results of this table, we can see that the optimal values obtained by SOS, WOA-DE, and WOA-BSA are the same. The convergence plots of the three algorithms (WOA, WOA-DE, and WOA-BSA) are shown in Figure 14. We can see that WOA-DE and WOA-BSA have faster convergence speed.

4.4. Analysis and Discussion of Experimental Results. Taking the experimental results on twenty-three benchmark functions and six engineering design problems into consideration simultaneously, the comprehensive performance of the proposed hybrid framework is discussed based on WOA-DE and WOA-BSA from three aspects, including the solution accuracy, convergence rate, and algorithmic robustness.

4.4.1. Discussion of the Solution Accuracy. The Best values in the above tables can reflect the solution accuracy of an algorithm in solving twenty-three benchmark functions and six engineering design problems. The smaller the Best values, the better the solution accuracy. On the one hand, compared with eight algorithms on twenty-three benchmark functions, WOA-DE ranks first among seven unimodal functions; however, it ranks fourth among multimodal functions, which means that WOA-DE is not competitive enough for more complex functions. WOA-BSA ranks second for both unimodal functions (except WOA-DE) and multimodal functions (except DE). This shows that WOA-BSA is competitive in solving function optimization problems. On the other hand, compared with some state-of-the-art algorithms on engineering design problems, WOA-DE can find out the current best optimal solutions in four problems, including three-bar truss design problem, speed reducer design problem, gear train design problem, and I-beam design problem. WOA-BSA can obtain the current best feasible solutions in four engineering problems. All the above experimental results indicate that the proposed hybrid framework has excellent performance with respect to solution accuracy.

4.4.2. Discussion of the Convergence Rate. The FE values reflect the computational overhead of an algorithm when finding out the best solution of an optimization problem. The smaller the computational overhead of an algorithm, the faster the convergence rate. On the one hand, when solving the six
4.4.3 Discussion of the Algorithmic Robustness. The Mean and Std values in the above tables can reflect the robustness of an algorithm. The smaller the Mean and Std values, the higher the robustness of an algorithm. On the one hand, on twenty-three benchmark functions, the Mean and Std values of WOA-DE rank third in unimodal functions and rank third and fourth in multimodal functions, respectively, while the ranking of the Mean and Std values obtained by WOA-BSA is fourth and fifth for unimodal functions, respectively. For multimodal functions, the ranking of WOA-BSA is the first for the Mean and Std values, which indicates that the algorithm has strong robustness. On the other hand, on six engineering problems, unfortunately, the Mean and Std values obtained by WOA-DE are mediocre, while the performance of WOA-BSA on Mean and Std values of 30 run times is also not competitive enough. All in all, the WOA-DE and WOA-BSA have better overall performance in terms of robustness, but there is still much room for improvement.

From the analysis of results based on the above two sets of experiments on WOA-DE and WOA-BSA, we can provide insight into the following two characteristics about the proposed algorithm framework. Firstly, the proposed framework is an effective and generic technique for function optimization and engineering constrained problems. Secondly, the overall performance of the framework is outstanding, but there is still space to improve its robustness.
5. Conclusion

The original WOA is easy to fall into local optimum when it carries out the operation of prey encircling. To address this issue, a hybrid algorithm framework with learning and complementary fusion features for WOA called hWOAlf is proposed in this paper. First, a learning parameter \( l_p \) according to adaptive adjustment operator is proposed to balance exploration and exploitation capability. Furthermore, the proposed hybrid algorithm framework called hWOAlf takes advantage of the complementary fusion feature operator to enhance the exploration capability of WOA. Two new algorithms, WOA-DE and WOA-BSA, are formed by embedding the DE/rand/1 operator of DE and the mutate operator into WOA, respectively. The effectiveness of the proposed hWOAlf framework is exhaustively demonstrated through comparing WOA-DE and WOA-BSA with some state-of-the-art optimization algorithms on twenty-three continuous benchmark functions and six engineering design problems.

In the future, there are two further studies to be done; on the one hand, better parameter tuning strategies can be developed to balance the exploration and exploitation capabilities of WOA; on the other hand, more metaheuristics are expected to be added to the proposed hybrid framework.

### Appendix

#### A. Engineering Design Problems

**A.1. Three-Bar Truss Design Problem.**

\[
\begin{align*}
\min f(x) &= \left(2\sqrt{2}x_1 + x_2\right) \times l, \\
\text{subject to:} & \\
& \quad \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0, \\
& \quad \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \leq 0, \\
& \quad \frac{1}{\sqrt{2}x_2 + x_1} P - \sigma \leq 0, \\
& \quad 0 \leq x_i \leq 1, \quad i = 1, 2, \\
& \quad l = 100 \text{ cm}, \\
& \quad P = 2 \text{ kN/cm}^2, \\
& \quad \sigma = 2 \text{ kN/cm}^2.
\end{align*}
\]

---

**Table 16: Comparison results for I-beam design problem.**

| Method   | \(x_1\)   | \(x_2\)   | \(x_3\)   | \(x_4\)   | \(f(X)\)   |
|----------|------------|------------|------------|------------|-------------|
| ARSM [44]| 37.050000  | 80.000000  | 1.710000   | 2.310000   | 0.015700    |
| IARSM [44]| 48.420000  | 79.990000  | 0.900000   | 2.400000   | 0.131000    |
| CS [41]  | 50.000000  | 80.000000  | 0.900000   | 2.321675   | 0.013075    |
| SOS [42] | 50.000000  | 80.000000  | 0.900000   | 2.321792   | 0.013074    |
| WOA      | 44.000000  | 80.000000  | 0.900815   | 2.644304   | 0.013173    |
| WOA-DE   | 50.000000  | 80.000000  | 0.900000   | 2.321792   | 0.013074    |
| WOA-BSA  | 50.000000  | 80.000000  | 0.900000   | 2.321792   | 0.013074    |

---

**Figure 14:** Convergence plots using WOA, WOA-DE, and WOA-BSA for I-beam design problem.
A.2. Pressure Vessel Design Problem.

\[
\begin{align*}
\min f(x) &= 0.6224x_1x_3x_4 + 1.7781x_2^2 \\
&\quad + 3.1661x_1^2x_4 + 19.84x_1^2x_3,
\end{align*}
\]
subject to:

\[
\begin{align*}
g_1(x) &= -x_1 + 0.0193x_3 \leq 0, \\
g_2(x) &= -x_2 + 0.00954x_3 \leq 0, \\
g_3(x) &= -\pi x_1^2 x_4 - (4/3)\pi x_2^2 + 1.296,000 \leq 0, \\
g_4(x) &= x_4 - 240 \leq 0, \\
0 \leq x_i \leq 100, & \quad i = 1, 2, \\
10 \leq x_i \leq 200, & \quad i = 3, 4.
\end{align*}
\]

A.3. Speed Reducer Design Problem.

\[
\begin{align*}
\min f(x) &= 0.7854x_1x_2^2(3.3333x_2^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_2^2 + x_3^2) \\
&\quad + 7.4777(x_3^3 + x_2^3) + 0.7854(x_4x_2^2 + x_3x_3^2),
\end{align*}
\]
subject to:

\[
\begin{align*}
g_1(x) &= \frac{27}{x_1x_2^2x_3} - 1 \leq 0, \\
g_2(x) &= \frac{397.5}{x_1x_2^2x_3^2} - 1 \leq 0, \\
g_3(x) &= \frac{1.93x_4^3}{x_2x_3^3x_4^2} - 1 \leq 0, \\
g_4(x) &= \frac{1.93x_4^3}{x_2x_3^3x_4^2} - 1 \leq 0, \\
g_5(x) &= \left[\left(\frac{745x_1}{x_2x_3}\right)^2 + 16.9 \times 10^6\right]^{1/2} - 1 \leq 0, \\
g_6(x) &= \left[\left(\frac{745x_1}{x_2x_3}\right)^2 + 157.5 \times 10^6\right]^{1/2} - 1 \leq 0, \\
g_7(x) &= \frac{x_2x_3}{40} - 1 \leq 0, \\
g_8(x) &= \frac{5x_2}{x_1} - 1 \leq 0, \\
g_9(x) &= \frac{x_1}{12x_2} - 1 \leq 0, \\
g_{10}(x) &= \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0, \\
g_{11}(x) &= \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0,
\end{align*}
\]
where

\[
2.6 \leq x_1 \leq 3.6, \\
0.7 \leq x_2 \leq 0.8, \\
17 \leq x_3 \leq 28, \\
7.3 \leq x_4, x_5 \leq 8.3, \\
2.9 \leq x_6 \leq 3.9, \\
5.0 \leq x_7 \leq 5.5.
\]

(A.4)

A.4. Gear Train Design Problem.

\[
\min f(x) = \left( \frac{1}{6.931} - \frac{x_3 x_2}{x_1 x_4} \right)^2,
\]

subject to:

\[
12 \leq x_1, x_2, x_3, x_4 \leq 60.
\]

(A.5)

A.5. Cantilever Beam Design Problem.

\[
\min f(x) = 0.6224 \left( x_1 + x_2 + x_3 + x_4 + x_5 \right),
\]

subject to:

\[
g_1(x) = \frac{61}{x_1} + \frac{27}{x_2} + \frac{19}{x_3} + \frac{7}{x_4} + \frac{1}{x_5} - 1 \leq 0,
\]

\[
0.01 \leq x_1, x_2, x_3, x_4, x_5 \leq 100.
\]

(A.6)

A.6. I-Beam Design Problem.

\[
\min f(x) = \frac{5000}{\left( x_3 \left( x_2 - 2x_4 \right)^3/12 \right) + \left( \left( x_1 x_2 \right)/6 \right) + 2x_1 x_4 \left( \left( x_2 - x_4 \right)/2 \right)^2},
\]

subject to:

\[
g_1(x) = 2x_1 x_3 + x_3 \left( x_2 - 2x_4 \right) \leq 300,
\]

\[
g_2(x) = \frac{180,000 x_2}{x_3 \left( x_2 - 2x_4 \right)^3 + 2x_1 x_4 \left( 4x_3^4 + 3x_3 \left( x_2 - 2x_4 \right) \right) + \frac{15,000 x_1}{\left( x_2 - 2x_4 \right) x_3^3 + 2x_3 x_4^3} \leq 6,
\]

\[
10 \leq x_1 \leq 50, 10 \leq x_2 \leq 80, 0.9 \leq x_3 \leq 5, 0.9 \leq x_4 \leq 5.
\]

Data Availability

The data used to support the findings of this study are included within the article. In addition, in order to better share the research results, the LaTeX codes related to the data are available from the corresponding author upon request.

Conflicts of Interest

The author declares that there are no conflicts of interest.

References

[1] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, MIT Press, Cambridge, MA, USA, 1992.

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

[3] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.

[4] P. Civicioglu, “Backtracking search optimization algorithm for numerical optimization problems,” *Applied Mathematics and Computation*, vol. 219, no. 15, pp. 8121–8144, 2013.

[5] Z. Hu, X. Xu, Q. Su, H. Zhu, and J. Guo, “Grey prediction evolution algorithm for global optimization,” *Applied Mathematical Modelling*, vol. 79, no. 79, pp. 145–160, 2020.

[6] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the ICNN'95 - International Conference on Neural Networks*, IEEE, Perth, Australia, November 1995.

[7] D. Karaboga, *An Idea Based on Honey Bee Swarm for Numerical Optimization*, Technical Report-tr06, Erciyes University, Engineering Faculty, Computer Engineering Department, Kayseri, Turkey, 2005.

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

[9] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[10] S. Mirjalili and A. Lewis, “The whale optimization algorithm,” *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

[11] B. Bentouati, L. Chaib, and S. Chettih, “A hybrid whale algorithm and pattern search technique for optimal power flow problem,” in *Proceedings of the International Conference on*
H. Wang, Z. Hu, Y. Sun, Q. Su, and X. Xia, “A novel hybrid whale optimization algorithm based on modified differential evolution for global optimization problems,” Applied Intelligence, vol. 49, no. 5, pp. 1982–2000, 2019.

M. M. Mafarja and S. Mirjalili, “Hybrid whale optimization algorithm with simulated annealing for feature selection,” Neurocomputing, vol. 260, pp. 302–312, 2017.

I. N. Trivedi, P. Jangir, A. Kumar, N. Jangir, and R. Totlani, “A novel hybrid PSO-WOA algorithm for global numerical functions optimization,” Advances in Computer and Computational Sciences, pp. 53–60, 2018.

A. N. Jadhav and N. Komathi, “WGC: hybridization of exponential grey wolf optimizer with whale optimization for data clustering,” Alexandria Engineering Journal, vol. 57, no. 3, pp. 1569–1584, 2018.

Z. Xu, Y. Yu, H. Yachi, J. Ji, Y. Todo, and S. Gao, “A novel memetic whale optimization algorithm for optimization,” Lecture Notes in Computer Science, pp. 384–396, 2018.

A. Kaveh and M. Rastegar Moghadam, “A hybrid WOA-CBO algorithm for construction site layout planning problem,” Scientia Iranica, vol. 25, no. 3, pp. 1094–1104, 2018.

S. Khalilpourazari and S. Khalilpourazary, “SCWOA: an efficient hybrid algorithm for parameter optimization of multi-pass milling process,” Journal of Industrial and Production Engineering, vol. 35, no. 3, pp. 135–147, 2018.

N. Singh and H. Hachimi, “A new hybrid whale optimizer algorithm with mean strategy of grey wolf optimizer for global optimization,” Mathematical and Computational Applications, vol. 23, no. 1, p. 14, 2018.

M. Abdel-Basset, G. Manogaran, D. El-Shahat, and S. Mirjalili, “A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem,” Future Generation Computer Systems, vol. 85, pp. 129–145, 2018.

N. M. Laskar, K. Guha, I. Chatterjee, S. Chanda, K. L. Baishnab, and P. K. Paul, “HWPSO: a new hybrid whale-particle swarm optimization algorithm and its application in electronic design optimization problems,” Applied Intelligence, vol. 49, no. 1, pp. 265–291, 2019.

H. Wang, Z. Hu, Y. Sun et al., “Modified backtracking search optimization algorithm inspired by simulated annealing for constrained engineering optimization problems,” Computational Intelligence and Neuroscience, vol. 2018, Article ID 9167414, 27 pages, 2018.

Z. Li, Z. Hu, Y. Miao et al., “Deep-mining backtracking search optimization algorithm guided by collective wisdom,” Mathematical Problems in Engineering, vol. 2019, Article ID 2540102, 2019.

H. Wang, Z. Hu, Y. Sun, Q. Su, and X. Xia, “A novel modified BSA inspired by species evolution rule and simulated annealing principle for constrained engineering optimization problems,” Neural Computing and Applications, vol. 31, no. 8, pp. 4157–4184, 2019.

Z. Hu, Q. Su, X. Yang, and Z. Xiong, “Not guaranteeing convergence of differential evolution on a class of multimodal functions,” Applied Soft Computing, vol. 41, pp. 479–487, 2016.
[43] S. Mirjalili, “Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm,” Knowledge-Based Systems, vol. 89, pp. 228–249, 2015.

[44] G. G. Wang, “Adaptive response surface method using inherited Latin hypercube design points,” Journal of Mechanical Design, vol. 125, no. 2, pp. 210–220, 2003.