mathematics

Article

A Comprehensive Multi-Strategy Enhanced Biogeography-Based Optimization Algorithm for High-Dimensional Optimization and Engineering Design Problems

Chenyang Gao [1], Teng Li [1], Yuelin Gao [1,2,*] and Ziyu Zhang [2]

1 School of Computer Science and Engineering, Xidian University, Xi'an 710071, China; ghygao2008@gmail.com (C.G.); tengli@xidian.edu.cn (T.L.)
2 Ningxia Province Key Laboratory of Intelligent Information and Data Processing, North Minzu University, Yinchuan 750021, China; algorithmzzy@126.com
* Correspondence: gaoyuelin@263.net

Abstract: The biogeography-based optimization (BBO) algorithm is known for its simplicity and low computational overhead, but it often struggles with falling into local optima and slow convergence speed. Against this background, this work presents a multi-strategy enhanced BBO variant, named MSBBO. Firstly, the example chasing strategy is proposed to eliminate the destruction of the inferior solutions to superior solutions. Secondly, the heuristic crossover strategy is designed to enhance the search ability of the population. Finally, the prey search–attack strategy is used to balance the exploration and exploitation. To verify the performance of MSBBO, we compare it with standard BBO, seven BBO variants (PRBBO, BBOSB, HGBBO, FABBO, BLEHO, MPBBO and BBOIMAM) and seven meta-heuristic algorithms (GWO, WOA, SSA, ChOA, MPA, GJO and BWO) on multiple dimensions of 24 benchmark functions. It concludes that MSBBO significantly outperforms all competitors both on convergence accuracy, speed and stability, and MSBBO basically converges to the same results on 10,000 dimensions as on 1000 dimensions. Further, MSBBO is applied to six real-world engineering design problems. The experimental results show that our work is still more competitive than other latest optimization techniques (COA, EDO, OMA, SHO and SCSO) on constrained optimization problems.

Keywords: biogeography-based optimization; heuristic crossover; prey search–attack; high-dimensional optimization; engineering design

MSC: 68W50

check for updates

Citation: Gao, C.; Li, T.; Gao, Y.; Zhang, Z. A Comprehensive Multi-Strategy Enhanced Biogeography-Based Optimization Algorithm for High-Dimensional Optimization and Engineering Design Problems. *Mathematics* **2024**, *12*, 435. https://doi.org/10.3390/math12030435

Academic Editors: Tien Anh Tran, Roman Rodriguez Aguilar, Gerhard-Wilhelm Weber, Igor Litvinchev, Joshua Thomas and Pandian Vasant

Received: 15 December 2023
Revised: 19 January 2024
Accepted: 23 January 2024
Published: 29 January 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the development of engineering technology and science, the optimization problem has been widely existent in all aspects of social production. In order to solve different optimization problems, various optimization techniques have been designed. They are all involved in problems where there are optimal solutions, such as robot path planning [1], vehicle routing [2], portfolio optimization [3], job-shop scheduling [4], array antenna optimization [5], etc. The purpose of optimization is to reduce cost consumption, improve economic returns, enhance system efficiency, save time, etc. At present, a relatively complete optimization system has been formed, which mainly uses mathematical methods to provide solutions to various problems. These methods are mainly divided into two categories: traditional optimization technology and meta-heuristic algorithm. Traditional optimization methods rely more on the known information of the problem to solve the deterministic problem effectively, such as the branch and bound algorithm [6], conjugate gradient method [7], steepest descent method [8], etc. Unlike these techniques, meta-heuristic algorithms obtain new optimization models by simulating certain natural phenomena or animal behaviors. Meta-heuristic algorithms can not only solve *NP*-hard problems but also find the optimal solution in finite time, and do not need the gradient information of the problem, so they are more suitable for nonlinear and complex optimization problems. They meet the requirements of current optimization problems and become mainstream optimization methods.

As shown in Figure 1, meta-heuristic algorithms can be roughly divided into five categories according to different classification standards: evolution-based optimization algorithms, such as the genetic algorithm (GA) [9], differential evolution (DE) [10] and immune algorithm (IA) [11]; population-based optimization algorithms, such as particle swarm optimization (PSO) [12], whale optimization algorithm (WOA) [13] and sparrow search algorithm (SSA) [14]; nature-based optimization algorithms, such as biogeography-based optimization (BBO) [15], invasive weed optimization (IWO) [16] and tree seed algorithm (TSA) [17]; physics-based optimization algorithms, such as the Archimedes optimization algorithm (AOA) [18], Kepler optimization algorithm (KOA) [19] and Young's double-slit experiment optimizer (YDSE) [20]; and human-based optimization algorithms, such as student psychology-based optimization (SPBO) [21], teaching and learning optimization (TLO) [22] and human behavior-based optimization (HBBO) [23]. Different meta-heuristic algorithms have different performance in solving different types of optimization problems. These existing meta-heuristic algorithms have been widely used in various fields of manufacturing. However, with the rapid development of the information age, the complexity of the problems faced by human beings is different from the past, and gradually tends to be high dimensional and large scale. According to the investigation, problems with more than 100 dimensions are considered high-dimensional optimization problems [24–26]. But the vast majority of meta-heuristic algorithms have not been used to solve optimization problems with more than 100 dimensions. Therefore, in this context, it is still necessary to improve the existing meta-heuristic algorithm to solve the optimization problem of higher dimensions.

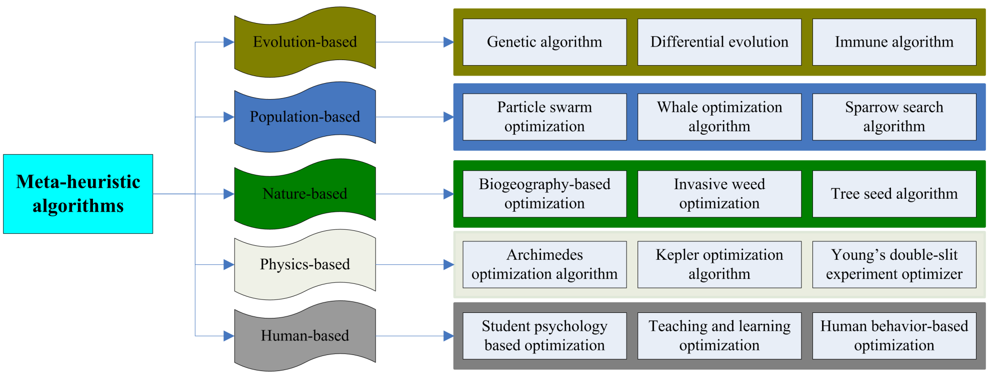


Figure 1. Meta-heuristic algorithm classification.

The BBO algorithm is a novel meta-heuristic algorithm proposed by Simon in "*IEEE TEVC*" in 2008 [15]. The BBO algorithm simulates the movement of species between different habitats in nature. Due to its simple principle and few parameters, BBO attracted many scholars once it was proposed and has been widely used in various fields [27–30]. Compared with other meta-heuristic algorithms, BBO has the following advantages: (1) BBO uses the migration operator to complete the variable exchange between candidate solutions, which is more ergodic than the crossover operator. When the dimension of the optimization problem is high, it can still search for the optimal direction in each dimension. (2) BBO does not need to adjust parameters, as the only two parameters are fixed. In other words, when solving high-dimensional optimization problems, the parameters will not affect the algorithm performance of BBO. This is more convenient than PSO, DE and other algorithms. (3) BBO can take advantage of the useful information carried by the current population. Unlike other meta-heuristic algorithms, the BBO candidate solution variables come from all population individuals. This enables the BBO to search adequately, even in high-dimensional environments. Based on the above discussion, we can find that BBO has more obvious advantages than other meta-heuristic algorithms in solving high-dimensional optimization problems. Therefore, based on the unique advantages of BBO, we choose to use BBO to challenge the efficient solution of high-dimensional optimization problems. This is also the motivation of choosing BBO in this paper.

However, according to the “no free lunch” principle, the BBO algorithm is not perfect. Similar to many meta-heuristic algorithms, BBO has some shortcomings, such as slow convergence speed, premature convergence, and falling into local optima. In recent years, researchers have proposed various BBO variants to improve its performance and prevent premature convergence [31]. For example, Ergezer [32] designed the OBBO algorithm based on the opposition-based learning (OBL) strategy for the first time. Experimental results show that the probability of obtaining the optimal solution of the problem is much better than that of the standard BBO algorithm. Wang et al. [33] designed a biogeography-based krill herd (BBKH) algorithm to solve complex optimization problems. The main improvement is to introduce a new krill migration operator to deal with nonlinear problems more efficiently. Lohokare et al. [34] adopted an improved mutation function to embed the neighborhood mutation of DE into BBO, thus accelerating the convergence of BBO. In their study, when the authors evaluated the performance of their proposed approach on benchmark test suite and economical load scheduling problems, their improved BBO outperformed the standard BBO. In order to reduce the dependence of BBO on the coordinate system of the optimization problem, Chen et al. [35] developed a variant of BBO based on covariance matrix migration. Experiments show that this method is superior to previous BBO variants. Then, Sang et al. [36] proposed DCGBBO by a hierarchical tissue-like P system with triggering ablation rules, making use of the evolution rule and communication rule to achieve migration and mutation, which reduces the computational complexity. Recently, to enhance the overall performance of BBO algorithm, [37] designed a novel BBO variant with hybrid migration operator and feedback differential evolution mechanism, referred to as HFBBO. It is a “living algorithm” that can self-regulate the mutation mode. The HFBBO feedback differential evolution mechanism is designed to replace the random mutation operator, so the population can select the mutation mode intelligently to avoid falling into the local optima.

We investigated articles on the BBO algorithm in some well-known journals as shown in Figure 2. As you can see, BBO already exists in a certain number of variants, which makes its performance also improve. Unfortunately, these variants do not make the BBO algorithm suitable for high-dimensional optimization environments. BBO is still not effective in solving high-dimensional global optimization problems. Why not try to improve the BBO algorithm to solve them? Therefore, in order to make a breakthrough in this field, this paper proposes a simple and efficient multi-strategy-enhanced BBO variant based on the example chasing strategy, heuristic crossover strategy and prey search–attack strategy, referred to as MSBBO. It is “simple” because our algorithm has lower computational complexity, simpler steps and fewer parameters than BBO. It is “efficient” because our method can effectively solve large-scale global optimization problems up to 10,000 dimensions. The primary contributions of this paper are summarized as follows:

(1) A novel framework of BBO is proposed, which is simpler and more efficient than the original BBO algorithm. Meanwhile, MSBBO has lower computational complexity than BBO.
(2) MSBBO uses the example chasing strategy to eliminate the misguidance of bad information in the population. Then, the heuristic crossover and prey search–attack strategies are used to balance the exploration and exploitation of the population. MSBBO makes BBO suitable for high-dimensional optimization environments.
(3) MSBBO successfully challenges the 10,000-dimensional numerical optimization problem. Compared with other meta-heuristic algorithms, its convergence performance is basically not affected by dimensions and has good ductility.

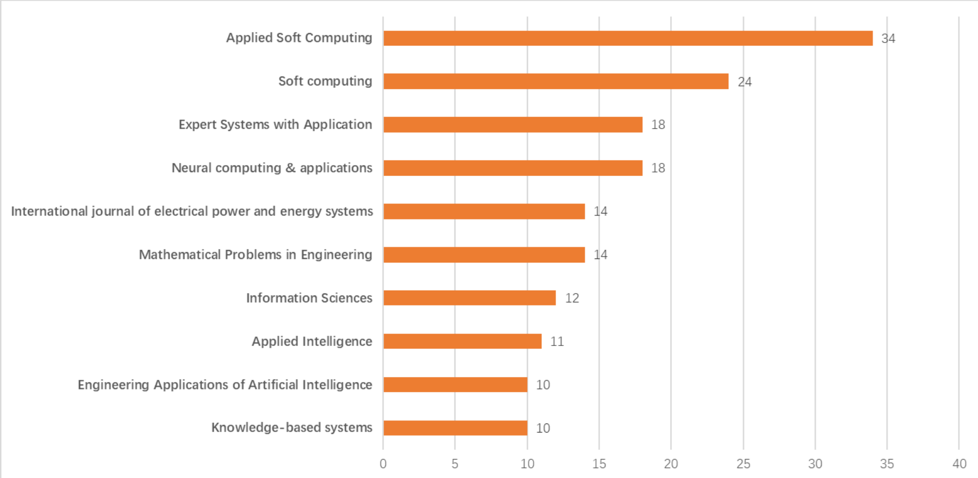


Figure 2. BBO articles' source statistics.

The organization of the rest of this paper is as follows: Section 2 introduces the standard BBO algorithm. Section 3 describes the three improvement strategies of MSBBO in this paper. Then, Section 4 analyzes the complexity of the proposed MSBBO. Section 5 is the numerical experiment and analysis. The last section, Section 6, is the conclusion. The graphical abstract of this paper is shown in Figure 3.

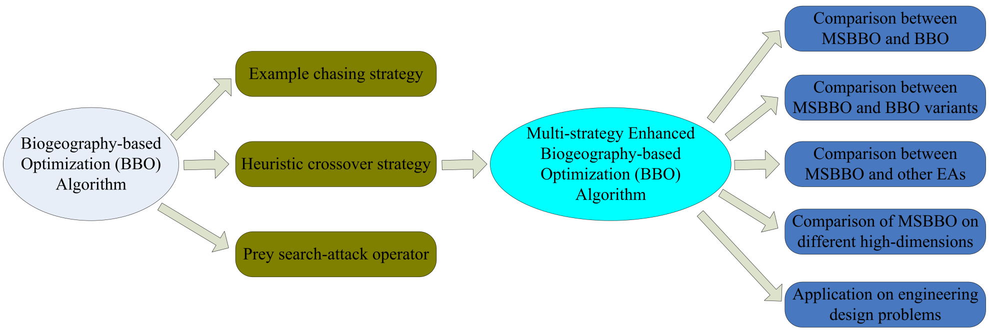


Figure 3. Graphical abstract of this paper.

2. Standard BBO

In BBO, every habitat contains some characteristic variables. They determine how many species a habitat can hold and are called suitability index variables ($SIVs$). A habitat that is suitable for living species is interpreted as having a high habitat suitability index (HSI). So, the habitat with higher HSI is more likely to emigrate species, while the habitat with lower HSI is more likely to immigrate species. This is the main idea of BBO [15]. Table 1 shows the correspondence.

Table 1. Correspondence among biogeography theory and BBO algorithm.

Biogeography Theory	Biogeography-Based Optimization Algorithm
Habitats (Islands)	Individuals (candidate solutions)
Habitat suitability index (HSI)	Objective function value (fitness)
Suitability index variables ($SIVs$)	Characteristic variables of solutions
Catastrophic events destroyed the habitat	Mutation
The number of habitats	Population size (the number of solutions)
Habitats with low HSI immigrate species	Inferior solutions accept variables
Habitats with high HSI emigrate species	Superior solutions share variables

In the first stage, BBO uses Equation (1) to generate N habitats as the initial population, and each habitat contains D variables:

$$x_{ij} = lb_j + rand(0,1) \cdot \left(ub_j - lb_j\right) \tag{1}$$

where $i = 1,2,\ldots,N$; $j = 1,2,\ldots,D$. x_i=(x_{i1},x_{i2},...,x_{iD}). ub_j and lb_j are the upper and lower limits of the j-th variable, respectively. Then, the HSI of each habitat is calculated based on the fitness function, and the population is sorted by the HSI. Specifically, if x_{ij} is the best individual in the population, then $i = 1$; if x_{ij} has the second highest fitness in the population, then $i = 2$, and so on. In other words, the subscript i of x_{ij} indicates its fitness ranking in the population. So, each x_i is assigned a new i, and the species number S_i of x_i is calculated according to Equation (2):

$$S_i = S_{max} - 2 \cdot i, i = 1, 2, \cdots, N \tag{2}$$

where S_{max} is the maximum species number, which is usually set to $2 \cdot N$. In this paper, we adopt the cosine migration model to calculate the immigration rate λ_i and emigration rate μ_i:

$$\lambda_i = \frac{I}{2}\left(1 + \cos\frac{\pi \cdot S_i}{S_{max}}\right), \mu_i = \frac{E}{2}\left(1 - \cos\frac{\pi \cdot S_i}{S_{max}}\right). \tag{3}$$

where I is the maximum immigration rate and E is the maximum emigration rate. They are usually set to 1.

In the second stage, the specific operation of migration operator is to generate a random number between [0, 1] for each variable of x_i. If it is smaller than λ_i, in the remaining N-1 habitats, the x_k to be emigrated is determined according to μ_k. Then, the variable of x_k is used to replace the corresponding variable of x_i.

In the third stage, it is the mutation operator. The species probability P_i of each habitat is calculated through Equation (4):

$$P_i = \begin{cases} -(\lambda_i + \mu_i)P_i + \mu_{i+1}P_{i+1}, & S_i = 0 \\ -(\lambda_i + \mu_i)P_i + \lambda_{i-1}P_{i-1} + \mu_{i+1}P_{i+1}, & 1 \le S_i \le S_{max} - 1 \\ -(\lambda_i + \mu_i)P_i + \lambda_{i-1}P_{i-1}, & S_i = S_{max} \end{cases} \tag{4}$$

The mutation rate of a habitat is inversely proportional to its species probability. Therefore, the mutation rate m_i of each habitat is as follows:

$$m_i = \left(1 - \frac{P_i}{P_{max}}\right) \cdot m_{max}, P_{max} = max\left(\{P_i\}_{i=1}^{N}\right) \tag{5}$$

where, m_{max} is the maximum mutation rate. For each habitat x_i, a number between [0, 1] is randomly generated, and if it is smaller than m_i, x_i needs to be mutated. Then, for each variable of x_i, a random number in the range of upper and lower bounds is generated to replace the original variable. Algorithm 1 shows the computation pseudo-code of BBO.

Algorithm 1 Pseudo-code of the BBO.

```
initialize parameters: S_max, I, E, N, and m_max
initialize the population by Equation (1)
for t = 1 to T do
   calculate the HSI and sort from best to worst
   calculate the S_i by Equation (2), the λ_i and μ_i by Equation (3)
   calculate the P_i by Equation (4), the m_i by Equation (5)
   for i = 1 to N do
      % Migration
      for j = 1 to D do
         if rand(0,1) < λ_i do
            select the x_k according to {μ_k}_{k=1}^N
            x_ij = x_kj
         end if
      end for
      % Mutation
      if rand(0,1) < m_i do
         for j = 1 to D do
            x_ij = lb_j + rand(0,1) · (ub_j − lb_j)
         end for
      end if
   end for
end for
output the optimal solution
```

3. Proposed Algorithm: MSBBO

3.1. Motivation

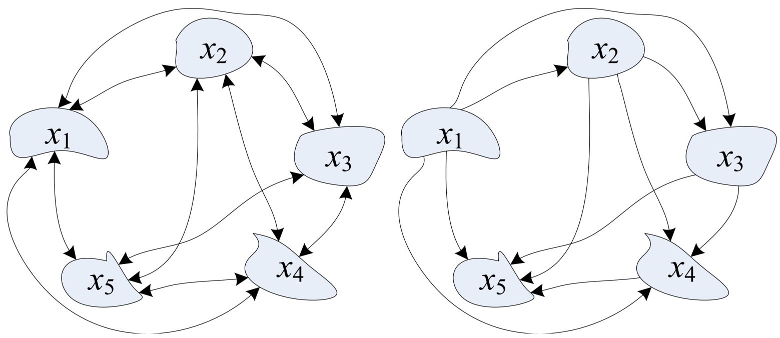
BBO uses roulette to select habitats to be emigrated, which can cause damage to the habitats with high HSI. At the same time, BBO replicates a single variable of another individual in the form of direct migration. However, a candidate solution performs well because the whole vector is closer to the optimal solution in the problem space, rather than because a single variable value performs well. In addition, the BBO random mutation operator cannot effectively help the algorithm to escape from the local optima. In fact, this mutation is blind and cannot maintain the population diversity. In addition, BBO relies only on the replacement of variables between different habitats to search for new solutions, which cannot balance the exploration and exploitation well and has a slow convergence speed. These are the main reasons why BBO cannot effectively solve high-dimensional numerical problems [38].

According to the literature reviews, we can observe that many studies have not completely overcome these shortcomings. Some variants change the mode of direct migration but do not eliminate the damage of inferior solutions to superior solutions, such as LxBBO [39], TDBBO [40], IWO/BBO [41], etc. Although HGBBO [42] and HFBBO [37] eliminate the damage of the inferior solutions to the superior solutions, the candidate solution changes only a certain part of variables in the migration. When solving high-dimensional optimization problems, the updating of candidate solutions cannot traverse every dimension. EMBBO [43] and NBBO [44] directly delete the random mutation operator but do not design more effective strategies to avoid the population falling into the local optima. In addition, the evolutionary mechanism of BBO itself determines that its performance ceiling is not high. Therefore, other heuristic strategies need to be considered to improve the convergence performance.

In view of the above shortcomings, this section proposes three improvement strategies to obtain a new variant of BBO with excellent performance. We firstly propose the example chasing strategy to eliminate the destruction of the inferior solutions to the superior solutions, thus effectively maintaining population diversity. Secondly, the heuristic crossover strategy is designed to enhance the search ability of the algorithm. The algorithm can search more adequately in the vicinity of superior individuals. Then, the prey search–attack strategy is designed to balance between the exploration and exploitation. In the evolution of the new algorithm population, the search emphasis is different in different stages. Meanwhile, to balance the computational complexity, the random mutation operator in the original BBO algorithm is deleted. The details are as follows.

3.2. Example Chasing Strategy

BBO randomly selects the candidate solution to emigrate, which is easy to damage the solutions with high fitness. For instance, x_i is the immigration individual, and x_j is the emigration individual selected by roulette. There is a good chance that j is greater than i, which means that islands with lower HSI immigrate to islands with higher HSI, and the variables of the bad candidate solution replace the variables of the better one. It not only reduces the population diversity but also causes the population to deviate from the optimal solution. In fact, candidate solutions with high fitness tend not to accept variables from candidate solutions with low fitness. Therefore, to avoid the inferior individuals destroying the superior individuals, the example chasing strategy is designed. We set examples based on the ranking of each individual. For individual x_i, it ranks i-th in the population, and the fitness of other individuals better than x_i can only be $x_1, x_2, ..., x_{i-1}$. These individuals rank higher than x_i, so they become the examples of x_i, and x_i becomes the chaser. Everyone has a natural tendency to chase the examples, so the chasers achieve better fitness by chasing their examples. The reason why human beings can progress is they continue to learn from the best and surpass them. To intuitively explain the principle of the example chasing strategy, Figure 4 is plotted.



(**a**) Random migration operator (**b**) Migration with example chasing rule

Figure 4. Random migration vs. the example chasing strategy.

As shown in Figure 4a, every two-candidate solution can migrate to each other. x_1 can be replaced by any lower-fitness solution, while x_5 can emigrate variable values to any better candidate solution. So, random migration will cause the inferior individuals to destroy the superior individuals, thus reducing the population diversity. Figure 4b is the example chasing strategy. It can be observed that only unidirectional migration can be carried out between candidate solutions. That is, the poor individuals can only accept features from the better individuals, and the poor individuals cannot affect the better individuals. For instance, only x_1 is ranked higher than x_2, so x_2 can only accept the variables from x_1, while individuals ranked lower than x_2 can accept the variables from x_2 but cannot emigrate to it.

During the migration of x_i, the example x_k of x_i can be selected by Equation (6):

$$k = \ round\ (ceil\ (1, i-1)), i = 2, 3, \cdots, N. \tag{6}$$

where $ceil(*)$ is a random number between 1 and i, and $round(*)$ is an integer function.

Therefore, the example chasing strategy avoids the bad influence of poor individuals on good individuals, effectively maintains the population diversity, and speeds up the search of the population to the optimal solution. In addition, it does not need to calculate the emigration rate of each individual, which reduces the calculation amount.

3.3. Heuristic Crossover Strategy

In the natural evolution of organisms, two homologous chromosomes through mating and recombination will form a new chromosome, thus giving rise to a new individual or species [45]. New individuals often absorb the advantages from their parents and thus better adapt to the current living environment. If it is used in the evolutionary algorithm, the search efficiency of the population in the search space can be improved. Inspired by this idea, to overcome the defects of the direct migration mode in the BBO algorithm, this paper designs a dynamic random heuristic crossover strategy as shown in Equation (7):

$$x_i^{t+1} = x_k^t + \alpha * (x_{best}^t - x_i^t) \tag{7}$$

where t is the current iteration number, k is the example individual selected by Equation (6), and x_{best}^t is the optimal individual of the current population. It should be noted that the heuristic crossover we designed is carried out for the whole candidate solution vector, rather than including only some variables like the standard BBO algorithm. Therefore, when solving a high-dimensional global optimization problem, the search of the problem space for candidate solutions can traverse every dimension.

Equation (7) has an important impact on the performance of MSBBO, and is expected to improve the search ability and convergence speed of MSBBO in the iterative process. It consists of a basis vector and a difference vector. The former is used to determine the search center, and the latter is used to control the search scope and direction. In the MSBBO framework, to make full use of the promising information provided by elite individuals and their guiding effect on other individuals, we use example individuals as the base vector. We have reason to believe the better solution is closer to the optimal solution, so the population can search in more valuable areas. The role of examples is crucial to the growth of an individual. It is because we have role models that we can become better people. We enhance our own abilities by mimicking the behaviors of examples or by being influenced by their personalities. Poor candidate solutions can also improve their competitiveness by absorbing the characteristics of good solutions. In addition, the optimal solution x_{best}^t of the current population is used to generate the difference vector to ensure a preferable searching direction. Through this strategy, individuals can exploit the area around an example; meanwhile, they can be attracted by the x_{best}^t.

The dynamic parameter α is a random number that varies nonlinearly with the current iteration number t, which is given as Equation (8):

$$\alpha = \frac{1}{2} * \left(\sin(2\pi * freq * t) * \frac{t}{T} + 1 \right) \tag{8}$$

The main idea of the dynamic parameter α is to design a novel formula which not only permits the adjustment of parameter values but also permits the adjustment of its direction. Such a possibility is well offered by the sine function. By transforming the sine function, the value of a given parameter increases and decreases periodically. This is accurately what we need, with some flexibility in the search direction when changing the parameter. $freq$ is used to control the fluctuation frequency of parameter α. After a lot of experiments, we suggest that the best value of $freq$ is 0.25.

3.4. Prey Search–Attack Operator

The BBO random mutation operator easily generates low-quality habitats and reduces the population diversity. It cannot effectively help the algorithm to escape from the local optima, and the calculation of species probability consumes much CPU time. Therefore, in MSBBO, we delete the random mutation operator, which further reduces the computational complexity. Meanwhile, inspired by the searching and attacking behavior of predators [46], we put forward the prey search–attack operator. It supposes the best solution of the current population x_{best}^t is the prey p^t, and the rest N-1 solutions are predators. Then, the searching and attacking behavior is defined as the following:

$$p^t = \omega_1 \cdot x_i^t + \omega_2 \cdot (x_{best}^t - x_i^t) \tag{9}$$

$$x_i^{t+1} = |x_{best}^t - p^t| \tag{10}$$

Equation (9) is used for searching the prey x_{best}^t, and Equation (10) is used for attacking the prey x_{best}^t. ω_1 and ω_2 are two self-adjusting parameters, which can be calculated by the following:

$$\omega_1 = (rand + 1) \cdot (1 - t/T) \tag{11}$$

$$\omega_2 = 2 \cdot (1 - t/T) \cdot (rand - 0.5) \tag{12}$$

where ω_1 and ω_2 are used to better balance between the exploration and exploitation over the whole evolution. The former is responsible for the global search, that is, exploration; the latter is responsible for the local search, that is, exploitation. ω_1 makes x_i search spaciously in the entire solution space where the prey x_{best}^t lives. On the contrary, ω_2 makes x_i search around a small limited small area of the prey x_{best}^t. The values of ω_1 and ω_2 are both large at the beginning of the evolution because in the early stage, it is necessary to maintain the population diversity. Later in the iteration, the population has already closed to the optimal solution, and it is not recommended to search in a large range. Instead, a more refined exploit should be tried near the optimal solution, so the values of ω_1 and ω_2 are all small. Therefore, the parameters $\omega_1$and ω_2 guarantee a dynamic balance of the exploration and exploitation.

To sum up, this section proposes a multi-strategy enhanced BBO variant based on the example chasing strategy, heuristic crossover strategy and prey search–attack operator. Algorithm 2 shows the calculation flow of MSBBO.

Algorithm 2 Pseudo-code of the MSBBO.

```
initialize parameters: S_max, I, N, freq
initialize the population by Equation (1)
calculate the S_i by Equation (2), the λ_i by Equation (3)
calculate the HSI and sort from best to worst
for t = 1 to T do
    for i = 1 to N do
        if rand(0,1) < λ_i do
            select the x_k according to Equation (6)
            heuristic crossover of x_i by Equations (7) and (8)
        end if
        calculate the ω_1 by Equation (11), the ω_2 by Equation (12)
        search the prey by Equation (9)
        attack the prey by Equation (10)
    end for
    calculate the HSI and sort from best to worst
end for
output the optimal solution
```

---

4. Complexity Analysis

In this section, the computational complexity of MSBBO and BBO is compared. The comparison between Algorithms 1 and 2 shows that MSBBO moves the calculation of the immigration rate out of the loop. Because it is based on the ranking, there is no need to calculate again. In BBO, the N individuals' immigration rate and emigrate rate are calculated in each iteration. So, the total calculation complexity of BBO is $O(2 \cdot T \cdot N)$. While MSBBO adopts the example chasing strategy to select the emigration individual, there is no need to calculate the emigration rate, so the calculation time is $O(N)$. The original migration operator of BBO requires each dimension of each individual to be judged and computed separately, so the total computation is $O(T \cdot N \cdot D)$. While the heuristic crossover of MSBBO migrates the whole candidate solution vector and reduces one "for" loop, the computational complexity is $O(T \cdot N)$. Then, in the mutation operator, according to Equations (4) and (5), its calculation times of each generation are $O(2 \cdot N)$. So the total calculation times of the BBO mutation operator is at least $O(2 \cdot T \cdot N)$. MSBBO uses prey search–attack operator to replace the mutation operator, which adds two calculation times in each iteration. So, the total calculation complexity is also $O(2 \cdot T \cdot N)$. However, MSBBO has fewer judgments than BBO in each iteration because the execution of the prey search–attack operator does not need to generate random numbers for judgment. So, BBO has $O(T \cdot N)$ more computation times than MSBBO in generating random numbers.

According to the above analysis, the computational complexity of BBO is $O(T \cdot N(D+5))$, while that of MSBBO is $O(N(3 \cdot T + 1))$. The MSBBO designed in this paper can effectively reduce the calculation cost of the BBO algorithm and reduce the calculation amount. We will further verify this by experiments later on.

5. Experimental Results and Analysis

5.1. Experiment Preparation

In order to fully test the comprehensive performance and competitiveness of MSBBO, in this section, a series of comparative experiments are made on a set of classic benchmark functions and several engineering design optimization problems. We first compare the MSBBO algorithm with the standard BBO algorithm to verify the effectiveness of the three improvement strategies. We then compare MSBBO to seven excellent BBO variants to verify the superiority of MSBBO in the same class of algorithms. After that, in order to verify the advancement of MSBBO in different algorithms, we choose seven new meta-heuristic algorithms to compare with it. Finally, we apply the proposed algorithm to six practical engineering problems, and select five advanced optimization techniques as competitors to demonstrate the value and development potential of MSBBO at the application level. The used well-known benchmark functions are shown in Table 2. They contain complex problems such as unimodal, multimodal, irregular, compound and nonlinear, which can fully test the comprehensive performance of the algorithms. The number of independent runs is 51, and the population size (N) of each compared algorithm is 50. We choose the Wilcoxon rank-sum test to analyze and evaluate all experimental results [47]. Then, the development environment is MATLAB R2022a.

Table 2. Twenty-four benchmark functions.

Function	Search Space	$f(x^*)$
$f1(x) = \sum_{i=1}^{D} ix_i^2$	$[-10, 10]^D$	0
$f2(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	$[-10, 10]^D$	0
$f3(x) = \sum_{i=1}^{D} x_i^2 + \left(\sum_{i=1}^{D} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{D} 0.5ix_i\right)^4$	$[-5, 10]^D$	0
$f4(x) = \max_{i=1}^{D}\{\lvert x_i \rvert\}$	$[-100, 100]^D$	0
$f5(x) = \sum_{i=1}^{D}\left(\sum_{j=1}^{i} x_j\right)^2 \times (1 + 0.4\lvert N(0,1)\rvert)$	$[-100, 100]^D$	0
$f6(x) = \sum_{i=1}^{D} ix_i^4 + rand$	$[-100, 100]^D$	0
$f7(x) = \sum_{i=1}^{D} z_i^2 - 450, z = x - o$	$[-100, 100]^D$	-450
$f8(x) = \sum_{i=1}^{D} \lvert x_i \rvert^{i+1}$	$[-1, 1]^D$	0
$f9(x) = \exp\left(0.5 \sum_{i=1}^{D} \lvert x_i \rvert\right) - 1$	$[-1.28, 1.28]^D$	0
$f10(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} x_i^2$	$[-100, 100]^D$	0
$f11(x) = \sum_{i=1}^{D} \lfloor x_i + 0.5 \rfloor^2$	$[-100, 100]^D$	0

Table 2. *Cont.*

Function	Search Space	$f(x^*)$
$f12(x) = \sum_{i=1}^{D}\left(\sum_{j=1}^{i} x_j\right)^2$	$[-100, 100]^D$	0
$f13(x) = \sum_{i=1}^{D}\left[z_i^2 - 10\cos(2\pi z_i) + 10\right] - 330, z = x - o$	$[-5.12, 5.12]^D$	-330
$f14(x) = \sum_{i=1}^{D}\left[z_i^2 - 10\cos(2\pi z_i) + 10\right]$, $z_i = \begin{cases} x_i, \lvert x_i\rvert < 0.5 \\ \text{round } (2x_i)/2, \text{ else} \end{cases}$	$[-5.12, 5.12]^D$	0
$f15(x) = -20\exp\left(-0.2\sqrt{\sum_{i=1}^{D}\frac{z_i^2}{D}}\right) - \exp\left[\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)\right] + e - 120, z = x - o$	$[-32, 32]^D$	-140
$f16(x) = \sum_{i=1}^{D}\lvert x_i \sin(x_i) + 0.1x_i\rvert$	$[-10, 10]^D$	0
$f17(x) = \frac{1}{4000}\left[\sum_{i=1}^{D}(z_i - 100)^2\right] - \left[\prod_{i=1}^{D}\cos\left(\frac{z_i - 100}{\sqrt{i}}\right)\right] - 179, z = x - o$	$[-600, 600]^D$	-180
$f18(x) = -\cos\left(2\pi\sqrt{\sum_i^D x_i^2}\right) + 0.1 \times \sqrt{\sum_i^D x_i^2} + 1$	$[-100, 100]^D$	0
$f19(x) = \sum_{i=1}^{D}\left\{\sum_{k=0}^{k_{\max}}\left[a^k\cos\left(2\pi b^k(x_i + 0.5)\right)\right]\right\} - D\sum_{k=0}^{k_{\max}}\left[a^k\cos\left(2\pi b^k \times 0.5\right)\right]$, $a = 0.5, b = 3, k_{\max} = 20$	$[-0.5, 0.5]^D$	0
$f20(x) = \frac{\pi}{D}\{10\sin^2(\pi y_i) + \sum_{i=1}^{D-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_D - 1)^2\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$ $y_i = 0.25(x_i + 1) + 1, u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, x_i > a \\ 0, = -a \le x_i \le a \\ k(-x_i - a)^m, x_i < -a \end{cases}$	$[-50, 50]^D$	0
$f21(x) = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{D-1}(x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})] + (x_D - 1)[1 + \sin^2(2\pi x_D)]\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, x_i > a \\ 0, -a \le x_i \le a \\ k(-x_i - a)^m, x_i < -a \end{cases}$	$[-50, 50]^D$	0
$f22(x) = \sum_{i=1}^{n-1} F(x_i, x_{i+1}) + F(x_n, x_1)$, $F(x, y) = (x^2 + y^2)^{0.25} \cdot \left[\sin^2\left(50(x^2 + y^2)^{0.1}\right) + 1\right]$	$[-100, 100]^D$	0
$f23(x) = \sum_{i=1}^{D-1}\left[x_i^2 + 2x_{i+1}^2 - 0.3\cos(3\pi x_i)\cos(3\pi x_{i+1}) + 0.3\right]$	$[-100, 100]^D$	0
$f24(x) = \sum_{i=1}^{D/4}[(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4]$	$[-4, 5]^D$	0

5.2. Comparison between MSBBO and Standard BBO

In order to test the effectiveness of MSBBO, we compare MSBBO with BBO first in this subsection. The standard BBO is not suitable for large-scale optimization problems, so we compare the performance of the two algorithms on 30, 50 and 100 dimensions, respectively. Our work does not add additional function evaluations, that is, the maximum iteration number (T) of MSBBO should be equal to that of BBO. We set $T = 1000$. The mean and standard deviation of the 51 errors are summarized in Table 3, and the last line is the results of the Wilcoxon rank-sum test. The representative meaning of "$(w/t/l)$" is w(+: win)/t($\approx$: tie)/l(-: lose), where "-" means that the performance of the compared algorithm is worse than MSBBO, "+" means the compared algorithm is superior, and "$\approx$" means the compared algorithm is similar to MSBBO. The bolded data represents the best value in both algorithms.

Table 3. Comparison results of MSBBO and BBO ($D = 30, 50, 100$).

F	BBO (*D* = 30)		MSBBO (*D* = 30)		BBO (*D* = 50)		MSBBO (*D* = 50)		BBO (*D* = 100)		MSBBO (*D* = 100)	
	Mean	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**
$f1$	1.74E+00	4.83E-01	**0.00E+00**	**0.00E+00**	1.47E+01	1.80E+01	**0.00E+00**	**0.00E+00**	3.56E+02	3.33E+03	**0.00E+00**	**0.00E+00**
$f2$	1.24E+00	6.40E-02	**0.00E+00**	**0.00E+00**	3.49E+00	2.53E-01	**0.00E+00**	**0.00E+00**	1.65E+01	2.82E+00	**0.00E+00**	**0.00E+00**
$f3$	6.89E+01	2.99E+02	**0.00E+00**	**0.00E+00**	2.12E+02	1.55E+03	**0.00E+00**	**0.00E+00**	7.91E+02	7.05E+03	**0.00E+00**	**0.00E+00**
$f4$	9.75E+00	3.53E+00	**0.00E+00**	**0.00E+00**	2.01E+01	5.39E+00	**0.00E+00**	**0.00E+00**	4.12E+01	6.70E+00	**0.00E+00**	**0.00E+00**
$f5$	1.88E+04	2.09E+07	**0.00E+00**	**0.00E+00**	5.51E+04	1.37E+08	**0.00E+00**	**0.00E+00**	2.11E+05	9.67E+08	**0.00E+00**	**0.00E+00**
$f6$	4.61E+02	1.88E+05	**2.42E-05**	**2.47E-05**	1.40E+04	1.73E+08	**3.03E-05**	**4.05E-05**	2.25E+06	8.81E+11	**4.52E-05**	**4.41E-05**
$f7$	1.29E+01	2.25E+01	**0.00E+00**	**0.00E+00**	6.61E+01	2.46E+02	**0.00E+00**	**0.00E+00**	8.55E+02	1.73E+04	**0.00E+00**	**0.00E+00**
$f8$	8.86E-06	2.13E-10	**0.00E+00**	**0.00E+00**	1.60E-05	5.24E-10	**0.00E+00**	**0.00E+00**	2.26E-05	1.56E-09	**0.00E+00**	**0.00E+00**
$f9$	8.93E-02	2.46E-04	**0.00E+00**	**0.00E+00**	2.78E-01	1.72E-03	**0.00E+00**	**0.00E+00**	2.30E+00	1.07E-01	**0.00E+00**	**0.00E+00**
$f10$	3.70E+05	8.62E+10	**0.00E+00**	**0.00E+00**	1.36E+06	6.38E+11	**0.00E+00**	**0.00E+00**	9.03E+06	9.35E+12	**0.00E+00**	**0.00E+00**
$f11$	1.29E+01	2.50E+01	**0.00E+00**	**0.00E+00**	7.63E+01	4.23E+02	**0.00E+00**	**0.00E+00**	8.49E+02	2.05E+04	**0.00E+00**	**0.00E+00**
$f12$	1.77E+03	5.33E+05	**0.00E+00**	**0.00E+00**	2.40E+04	6.00E+07	**0.00E+00**	**0.00E+00**	9.86E+05	4.18E+10	**0.00E+00**	**0.00E+00**
$f13$	4.76E+00	1.70E+00	**0.00E+00**	**0.00E+00**	1.73E+01	7.91E+00	**0.00E+00**	**0.00E+00**	8.17E+01	5.71E+01	**0.00E+00**	**0.00E+00**
$f14$	4.66E+00	2.02E+00	**0.00E+00**	**0.00E+00**	1.62E+01	6.40E+00	**0.00E+00**	**0.00E+00**	6.10E+01	1.37E+01	**0.00E+00**	**0.00E+00**
$f15$	1.85E+00	9.78E-02	**4.44E-16**	**0.00E+00**	2.93E+00	5.39E-02	**4.44E-16**	**0.00E+00**	4.98E+00	8.09E-02	**4.44E-16**	**0.00E+00**
$f16$	8.14E-02	9.15E-04	**0.00E+00**	**0.00E+00**	4.06E-01	8.68E-03	**0.00E+00**	**0.00E+00**	3.63E+00	2.76E-01	**0.00E+00**	**0.00E+00**
$f17$	1.10E+00	1.07E-03	**0.00E+00**	**0.00E+00**	1.60E+00	2.64E-02	**0.00E+00**	**0.00E+00**	8.29E+00	1.44E+00	**0.00E+00**	**0.00E+00**
$f18$	2.07E+00	5.86E-02	**0.00E+00**	**0.00E+00**	3.79E+00	1.73E-01	**0.00E+00**	**0.00E+00**	8.91E+00	4.75E-01	**0.00E+00**	**0.00E+00**
$f19$	3.22E+00	1.41E-01	**0.00E+00**	**0.00E+00**	7.51E+00	3.37E-01	**0.00E+00**	**0.00E+00**	2.56E+01	3.23E+00	**0.00E+00**	**0.00E+00**
$f20$	5.18E-01	1.38E-01	**7.07E-02**	**2.90E-03**	6.54E-01	1.32E-01	**2.93E-01**	**2.07E-02**	3.04E+00	4.73E-01	**8.21E-01**	**9.38E-02**
$f21$	3.54E+00	6.57E-01	**5.95E-01**	**6.07E-02**	6.62E+00	2.17E+00	**3.54E+00**	**7.36E-01**	3.81E+02	4.84E+05	**1.53E+01**	**1.37E+00**
$f22$	4.11E+01	2.09E+01	**0.00E+00**	**0.00E+00**	8.52E+01	6.65E+01	**0.00E+00**	**0.00E+00**	2.66E+02	1.65E+02	**0.00E+00**	**0.00E+00**
$f23$	4.43E+01	2.40E+02	**0.00E+00**	**0.00E+00**	2.23E+02	3.63E+03	**0.00E+00**	**0.00E+00**	2.53E+03	1.93E+05	**0.00E+00**	**0.00E+00**
$f24$	3.86E+00	6.75E+00	**0.00E+00**	**0.00E+00**	1.61E+01	7.63E+01	**0.00E+00**	**0.00E+00**	1.55E+02	2.71E+03	**0.00E+00**	**0.00E+00**
w/t/l	0/0/24		-		0/0/24		-		0/0/24		-	

It can be seen from Table 3 that the numerical calculation results of the proposed MSBBO algorithm on 24 benchmark functions (D = 30, 50 and 100) are significantly better than those of the BBO algorithm. This shows that our work has obviously improved the convergence accuracy of BBO. Through careful observation, it can be seen that the results of MSBBO on 30, 50 and 100 dimensions are basically the same, indicating that its performance is almost unaffected by dimensional changes on the lower dimensions. According to this, MSBBO has good malleability and can challenge higher dimensions. When $D = 100$, MSBBO also converges to the optimal value with zero error on 20 benchmark functions. This shows that MSBBO acquires a better exploration ability and still preserves the exploitation ability. This is because the example chasing strategy proposed in this paper blocks the transmission of bad information of the inferior solutions to the superior solutions, thus successfully maintaining the population diversity. The heuristic crossover strategy can enhance the random search ability of the algorithm. It helps the population exploit more fully in the vicinity of a superior candidate solution, thereby efficiently improving the convergence accuracy. At the same time, the prey search–attack operator helps the population to switch freely between global search and local search so as to quickly find the optimal evolutionary direction. Therefore, the improvement strategies in this paper effectively enhance the overall performance of the original BBO algorithm so that it can achieve the balance between exploration and exploitation. The three strategies complement each other, without excess, and work together.

Then, to provide an intuitive comparison [48,49], some convergence curves and boxplots of MSBBO and BBO are plotted as shown in Figure 5. The convergence curve is used to observe the convergence rate and the evolution state of the population, and the boxplot is used to evaluate the stability of the algorithm. From Figure 5, MSBBO converges much faster than BBO on all selected functions (D = 30, 50 and 100). Especially on the functions $f9$, $f11$, $f14$ and $f19$, the convergence curve of MSBBO decreases rapidly, which indicates that its population is rapidly concentrated towards the optimal solution. This shows that the strategies in our work can accelerate the convergence speed of the original algorithm by a large margin. Further, all the convergence trends of MSBBO on the three low-dimensions are basically the same, and there is no obvious difference. This also proves that the convergence performance of the proposed approach is not affected by dimensional changes on low dimensions. Meanwhile, by carefully observing the boxplots, it also shows that MSBBO is more stable and has stronger robustness than BBO. The results of 51 runs of the BBO algorithm fluctuate greatly, and there are outliers, such as functions $f2$, $f6$, $f8$ and $f24$, etc. MSBBO does not have any outliers, and the boxplot on all functions is almost a straight line. This means that each search of MSBBO converges to almost the same optimal value, providing superior consistency. In addition, both BBO and MSBBO were run 51 times on each benchmark function ($D = 100$), so we calculated their respective average run times on each function as shown in Figure 6. It can be intuitively found that the average calculation time of MSBBO on each function is much smaller than that of BBO, which not only improves the convergence accuracy but also saves time consumption. It also verifies that the complexity analysis in Section 4 is correct, and MSBBO has a simpler framework that reduces the unnecessary calculation of the immigration rate and the judgment steps for each dimension.

According to the above experiments and discussions, the performance of MSBBO is fully better than that of BBO. Therefore, the improvement strategies in our work successfully enhance the optimization capacity of BBO. In particular, MSBBO improves the malleability of the algorithm to the problem dimension.

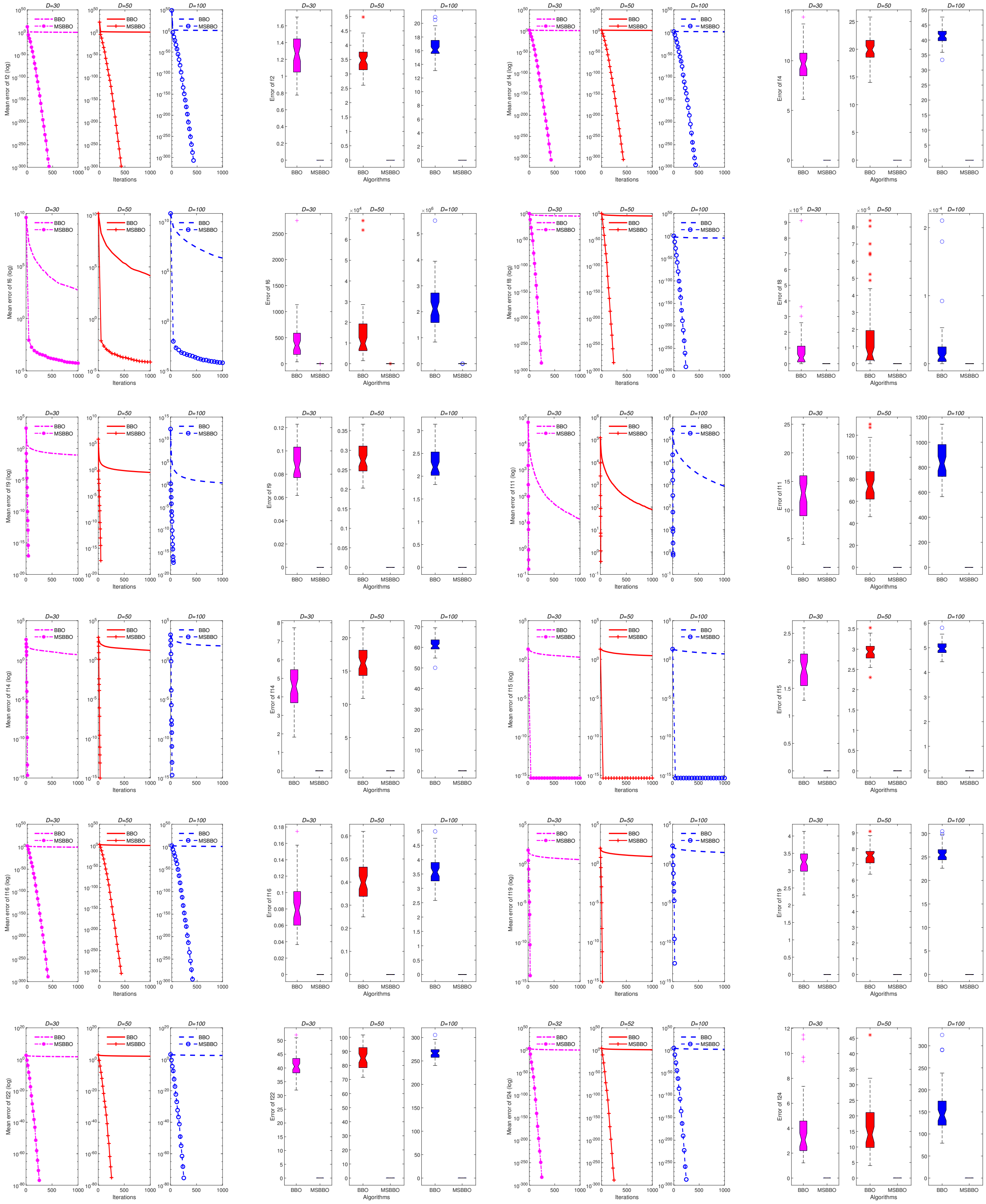
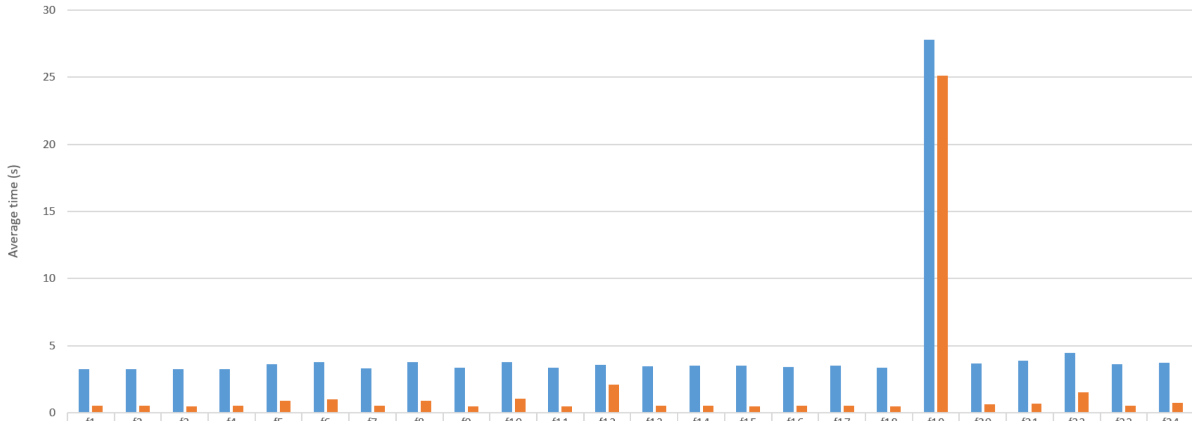


Figure 5. Convergence curves and boxplots of MSBBO and BBO on different benchmark functions ($D = 30, 50, 100$).



	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	f16	f17	f18	f19	f20	f21	f22	f23	f24
BBO	3.25141	3.25231	3.22849	3.25325	3.60505	3.75821	3.31913	3.79041	3.35443	3.7801	3.3718	3.54374	3.46376	3.51959	3.50572	3.42414	3.51198	3.34608	27.819	3.64946	3.85265	4.45553	3.61324	3.70765
MSBBO	0.53456	0.51992	0.48149	0.52023	0.88351	0.98492	0.49268	0.88405	0.44094	1.02719	0.48071	2.1099	0.49191	0.49985	0.47429	0.51628	0.51135	0.44684	25.1022	0.63483	0.69576	1.52248	0.49316	0.72837

Figure 6. Average running seconds of BBO and MSBBO on each function ($D = 100$).

5.3. Comparison between MSBBO and BBO Variants

In this subsection, we compare MSBBO with seven excellent BBO variants: PRBBO [50], BBOSB [51], HGBBO [42], FABBO [52], BLEHO [53], MPBBO [54] and BBOIMAM [55]. We compare their performance on 200 dimensions. Consistent with Section 5.2, eight algorithms search the optimal values on 24 benchmark functions, and the mean and standard deviation of 51 errors are used as evaluation indexes. Table 4 shows the comparison results. Among them, the mean error is the focus of comparison, and the bold data represent the optimal value.

From Table 4, MSBBO outperforms all BBO variants more than 90% of the problems according to the Wilcoxon rank-sum test. The experimental results of PRBBO and HGBBO are the same, their convergence results on $f20$ and $f21$ are better than MSBBO, and they converge to the same optimal value on f11, but the convergence results on the remaining 22 functions are worse than MSBBO. BBOSB performs better than MSBBO only on $f21$ and worse on all other functions. In contrast, FABBO is the least competitive, with significantly worse experimental results than MSBBO on all functions. In the high-dimensional environment of 200 dimensions, MSBBO can still converge to the theoretical optimal value on 20 benchmark functions, which is basically consistent with the results in Table 3. This shows that the performance of the MSBBO algorithm does not decrease significantly with the increase in dimension, and can adapt well to the high-dimensional optimization environment. In addition, MSBBO performs better on 91.7% of the benchmark problems than other algorithms of the same class, demonstrating the superiority of this paper's work in the BBO variants. It achieves more trustworthy results with a simpler algorithmic structure and complexity. Also, for better evaluation, the convergence curves and boxplots of them on different functions are shown in Figure 7. Obviously, MSBBO converges the fastest on all functions and does not fall into local optima. Especially on functions $f13$, $f14$, $f19$ and $f23$, the convergence curves of MSBBO are almost perpendicular to the horizontal axis; the convergence speed is fast. The example chasing strategy prevents population degradation, so the convergence curves of MSBBO will not fluctuate and will always converge. Then, the heuristic crossover strategy ensures that the population search can traverse every dimension, so MSBBO can quickly find the evolutionary direction in high-dimensional environments. Finally, the prey search–attack strategy ensures the dynamic balance between global exploration and local exploitation of the population, and successfully improves the convergence rate. In addition, a closer look at the boxplots shows that MSBBO has almost no boxplot and no outliers on all benchmark functions. In contrast, other BBO variants have erratic performance on some functions, and their algorithmic generality is relatively low. Therefore, the improvement strategies designed in this paper effectively improve the comprehensive performance of the original algorithm and have strong competitiveness among BBO variants.

Table 4. Comparison results of MSBBO and BBO variants ($D = 200$).

F	PRBBO (2017)		BBOSB (2018)		HGBBO (2020)		FABBO (2021)		BLEHO (2022)		MPBBO (2022)		BBOIMAM (2022)		MSBBO	
	Mean	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**
$f1$	1.43E-02	2.09E-05	2.01E+03	6.06E+04	2.13E-15	1.23E-30	1.47E-13	1.95E-27	1.80E+02	2.65E+03	7.93E-04	1.63E-07	1.29E+02	4.84E+02	**0.00E+00**	**0.00E+00**
$f2$	2.98E-02	3.18E-05	3.42E+01	1.25E+01	4.19E-11	1.56E-22	3.21E-07	6.91E-16	2.66E+01	2.54E+01	3.81E+02	7.19E+04	1.11E+01	5.55E-01	**0.00E+00**	**0.00E+00**
$f3$	2.08E+03	5.83E+04	2.97E+03	6.08E+04	3.78E+03	3.27E+04	2.54E+03	6.29E+04	1.46E+05	2.34E+10	2.61E+02	5.32E+03	1.55E+03	2.19E+04	**0.00E+00**	**0.00E+00**
$f4$	3.01E+01	1.99E+01	2.98E+01	7.53E+00	2.39E-02	7.06E-05	6.89E+01	1.20E+02	2.09E-02	1.67E-04	1.18E+01	3.09E+00	2.12E+01	3.17E+00	**0.00E+00**	**0.00E+00**
$f5$	4.05E+05	5.84E+09	5.09E+05	2.89E+09	6.95E+05	7.22E+10	4.68E+05	4.62E+09	1.87E+05	2.02E+09	2.18E+05	4.41E+09	1.78E+05	6.67E+08	**0.00E+00**	**0.00E+00**
$f6$	4.61E+01	1.68E+03	4.07E+03	1.46E+06	5.15E-02	1.35E-04	2.50E-01	3.86E-03	7.06E+03	4.18E+06	8.82E-01	3.27E-01	3.86E+04	8.48E+07	**4.81E-05**	**4.41E-05**
$f7$	1.98E-02	5.78E-05	2.20E+01	4.68E+00	1.28E-16	6.78E-33	1.02E-13	3.38E-28	3.04E+01	1.59E+01	3.51E-04	8.51E-08	1.41E+02	2.83E+02	**0.00E+00**	**0.00E+00**
$f8$	2.35E-21	1.17E-41	2.29E-05	6.36E-10	1.24E-109	2.68E-218	8.64E-01	3.33E-01	1.91E-23	7.38E-46	1.86E-19	8.30E-38	1.60E-06	4.63E-12	**0.00E+00**	**0.00E+00**
$f9$	1.40E-03	7.94E-08	2.31E+05	2.39E+10	4.94E-12	3.90E-24	1.99E-08	2.94E-18	4.69E+00	3.14E+00	1.65E-04	1.47E-09	1.74E+00	3.74E-02	**0.00E+00**	**0.00E+00**
$f10$	2.32E+01	9.02E+01	3.98E+05	6.59E+09	3.59E-10	2.85E-20	7.12E+01	6.03E+03	1.02E+07	5.47E+12	2.77E+04	1.42E+08	1.01E+06	6.47E+10	**0.00E+00**	**0.00E+00**
$f11$	**0.00E+00**	**0.00E+00**	4.75E+01	4.72E+01	**0.00E+00**	**0.00E+00**	4.54E+01	1.82E+02	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.52E+02	2.70E+02	**0.00E+00**	**0.00E+00**
$f12$	7.56E+01	7.15E+02	3.30E+05	1.82E+09	1.22E-11	8.06E-23	3.76E-09	4.10E-18	2.06E+06	6.25E+10	3.59E+01	6.74E+01	5.17E+05	2.37E+09	**0.00E+00**	**0.00E+00**
$f13$	6.63E+02	6.61E+02	7.12E+02	2.79E+03	1.31E-11	2.21E-22	1.14E+03	1.11E+04	3.34E+02	4.49E+03	3.99E+02	4.80E+03	1.02E+02	1.64E+02	**0.00E+00**	**0.00E+00**
$f14$	5.08E+02	7.19E+02	5.15E+02	2.58E+03	4.40E-04	2.96E-07	1.28E+03	1.76E+04	2.65E+02	2.27E+04	4.85E+02	3.67E+03	8.06E+01	3.05E+01	**0.00E+00**	**0.00E+00**
$f15$	1.26E-02	3.66E-06	3.11E+00	4.65E-02	1.99E+01	6.88E-04	1.60E+00	4.80E-01	3.63E+00	5.71E-02	6.84E+00	6.53E+01	2.26E+00	1.79E-02	**4.44E-16**	**0.00E+00**
$f16$	9.36E-01	6.22E-01	3.81E+01	1.74E+01	4.32E-02	3.99E-04	6.42E+00	8.08E+01	2.44E+01	2.45E+01	9.14E+00	8.52E+00	3.73E+00	5.73E-01	**0.00E+00**	**0.00E+00**
$f17$	5.28E-03	3.29E-06	1.78E-01	4.85E-04	1.11E-16	0.00E+00	3.84E-03	2.21E-05	1.26E+00	1.28E-03	2.79E-03	1.35E-05	2.31E+00	2.83E-02	**0.00E+00**	**0.00E+00**
$f18$	2.81E+00	3.55E-02	4.71E+00	8.88E-02	4.01E-01	3.98E-03	2.12E+00	2.09E-01	4.19E+00	7.28E-02	1.42E+00	2.51E-02	5.67E+00	4.96E-02	**0.00E+00**	**0.00E+00**
$f19$	9.38E-01	8.32E-03	1.88E+02	2.05E+01	6.87E-10	4.51E-20	6.35E+01	3.29E+01	1.26E+02	1.36E+02	8.27E-01	9.15E-03	4.10E+01	1.62E+00	**0.00E+00**	**0.00E+00**
$f20$	2.06E-02	3.31E-04	1.76E+00	2.49E-01	**7.96E-06**	**3.26E-12**	7.82E+00	1.06E+01	1.14E+01	4.55E+00	3.49E+00	2.35E+00	6.89E-02	1.59E-04	9.47E-01	7.06E-02
$f21$	2.73E-01	1.79E-02	1.84E+01	2.05E+01	**8.71E-04**	**1.11E-07**	1.09E+02	8.51E+03	5.11E+01	2.38E+02	2.52E-02	6.21E-04	6.59E+00	6.89E-01	3.21E+01	2.05E+00
$f22$	1.17E+02	1.02E+02	6.11E+02	2.38E+03	2.42E-03	5.70E-07	1.25E+03	7.29E+03	1.09E+03	5.35E+03	4.06E+02	7.15E+04	3.13E+02	1.75E+02	**0.00E+00**	**0.00E+00**
$f23$	7.21E-01	1.03E-01	1.22E+02	8.72E+01	4.34E-15	1.81E-29	4.16E+00	7.08E+00	1.70E+02	1.61E+02	2.11E-02	2.94E-04	4.70E+02	4.07E+03	**0.00E+00**	**0.00E+00**
$f24$	8.75E+00	8.98E+00	1.85E+03	8.66E+04	8.30E-01	5.30E-02	1.90E-01	5.15E-03	2.65E+01	4.97E+01	2.95E+00	2.06E+00	2.15E+01	3.55E+01	**0.00E+00**	**0.00E+00**
w/t/l	2/1/21		1/0/23		2/1/21		0/0/24		0/1/23		1/1/22		2/0/22		-	

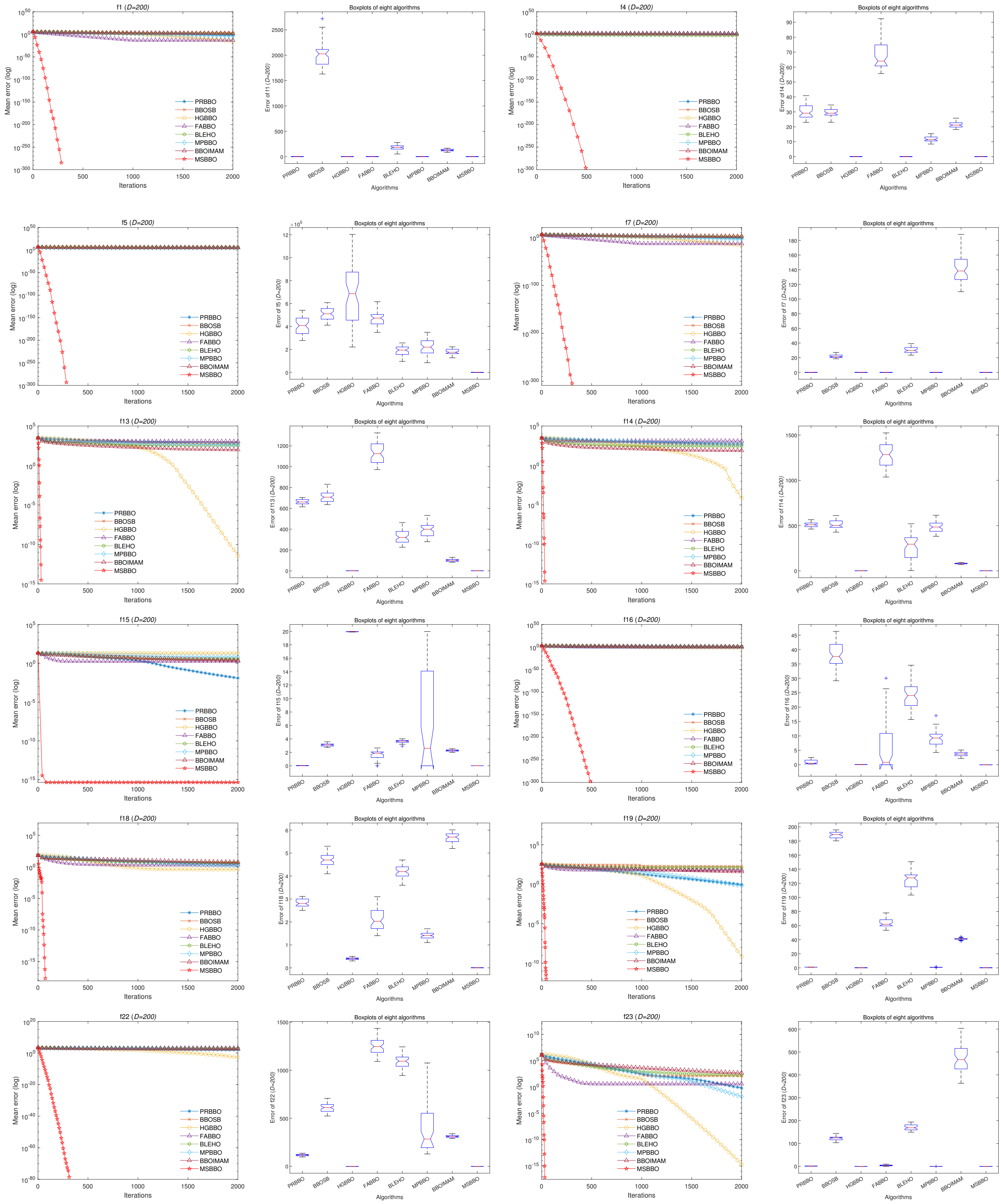


Figure 7. Convergence curves and boxplots of MSBBO and BBO variants on different benchmark functions ($D = 200$).

To sum up, the performance of MSBBO is better than that of PRBBO, BBOSB, HGBBO, FABBO, BLEHO, MPBBO and BBOIMAM. Compared to the same type of algorithm, MSBBO has outstanding ductility and is more suitable for high-dimensional optimization problems.

5.4. Comparison between MSBBO and Other Meta-Heuristic Algorithms

To further verify the superiority of MSBBO for solving large-scale optimization problems, we compare it with seven meta-heuristic algorithms proposed in the past few years: GWO [56], WOA [13], SSA [57], ChOA [58], MPA [59], GJO [60] and BWO [61]. Among them, GWO, WOA, SSA, ChOA and MPA are highly cited algorithms. GJO and BWO are the novel algorithms proposed in the last years which are competitive. Therefore, MSBBO can further verify its advancement by comparing with these outstanding algorithms. We compare their performance on 500 dimensions. Similarly, the mean and standard deviation of 51 errors are used as evaluation indexes. Table 5 shows the experimental results, and the bold data represent the optimal value.

According to Table 5, MSBBO still has the best overall performance among the eight new meta-heuristic algorithms, with the minimum mean error obtained on 22 functions and zero error on 20 functions. GWO converges to the theoretical optimal value on three functions ($f11$, $f13$ and $f23$), while the results on the remaining 21 functions are inferior to MSBBO. WOA converges to the theoretical optimal value only on the function $f9$, and the experimental results on the remaining functions are inferior to MSBBO. SSA and BWO converge to relatively better results among the eight compared algorithms on $f20$ and $f21$, the results on $f11$ are equal to MSBBO, and the results on the other functions are inferior to MSBBO. However, ChOA and GJO do not show some competitiveness, and the experimental results on 24 benchmark functions are inferior to MSBBO. In contrast, SSA and MPA are more competitive. MPA outperforms MSBBO on two functions ($f20$ and $f21$) and converges to the optimal solution on eight functions ($f8$, $f9$, $f11$, $f13$, $f14$, $f17$, $f19$ and $f23$). Although these novel algorithms show excellent performance on low-dimensional problems, their precision obviously decreases when solving high-dimensional optimization problems. On the contrary, even when $D = 500$, MSBBO converges error-free to the optimal value of the objective function on 20 problems. In other words, our work is still clearly competitive among different types of meta-heuristic algorithms, so it is advanced. A careful comparison between Tables 4 and 5 shows that the experimental results of MSBBO on 200 dimensions are almost the same as on 500 dimensions. Therefore, the improvement strategies in our work enable the population to search in a high-dimensional environment, and the algorithm performance has good ductility.

For a better evaluation of MSBBO and the seven new meta-heuristic algorithms, Figure 8 shows the convergence curves and boxplots of them on different problems. It can be found that the MSBBO algorithm converges much faster than other competitors on different benchmark functions, saving at least 1500 iterations and not falling into the local optima. Especially on functions $f13$, $f14$, $f15$, $f19$ and $f23$, MSBBO converges rapidly, and the convergence curve is almost invisible. Further, even if $D = 500$, the boxplot of MSBBO on different functions is almost invisible, and the algorithm performance remains stable. This shows that MSBBO can continue to challenge higher-dimensional problems.

Table 5. Comparison results of MSBBO and other meta-heuristic algorithms ($D = 500$).

F	GWO (2014)		WOA (2016)		SSA (2019)		ChOA (2019)		MPA (2020)		GJO (2022)		BWO (2022)		MSBBO	
	Mean	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**	**Mean**	**Std**
$f1$	6.10E-56	5.02E-56	4.90E-103	2.03E-102	5.73E-05	1.32E-04	1.06E-12	9.06E-13	2.92E-125	7.86E-125	1.31E-16	1.49E-16	1.89E-04	1.51E-04	**0.00E+00**	**0.00E+00**
$f2$	1.00E-33	6.29E-34	1.15E-108	3.19E-108	6.69E-03	5.18E-03	2.18E-09	1.27E-09	1.63E-06	1.15E-05	4.44E-140	5.38E-140	1.83E-02	7.94E-03	**0.00E+00**	**0.00E+00**
$f3$	7.02E+03	5.32E+03	7.97E+03	3.52E+02	1.22E+00	3.08E+00	6.57E+02	4.06E+02	2.90E-02	2.01E-02	1.05E+04	1.05E+04	1.39E+00	1.26E+00	**0.00E+00**	**0.00E+00**
$f4$	9.94E+01	7.07E+01	9.89E+01	3.91E-01	1.05E-04	9.49E-05	2.63E+02	2.13E+02	6.28E-42	9.13E-42	2.13E+02	1.76E+02	3.50E-04	1.57E-04	**0.00E+00**	**0.00E+00**
$f5$	4.82E+05	2.99E+05	4.32E+07	2.83E+07	2.89E+00	2.67E+00	1.31E+06	6.79E+05	3.30E+02	6.46E+02	1.80E+07	1.12E+07	9.73E+00	7.54E+00	**0.00E+00**	**0.00E+00**
$f6$	6.84E-03	3.73E-03	2.00E+02	2.81E+02	3.52E-03	3.15E-03	6.51E-03	3.31E-03	2.59E-04	1.11E-04	2.74E+05	1.82E+05	2.31E-03	1.40E-03	**4.98E-05**	**5.99E-05**
$f7$	3.83E-56	3.73E-56	1.73E-102	1.13E-101	1.00E-05	1.75E-05	2.64E-13	1.89E-13	6.28E-126	1.02E-125	1.60E-15	2.02E-15	7.46E-05	6.95E-05	**0.00E+00**	**0.00E+00**
$f8$	5.65E-24	6.55E-24	1.40E-05	2.08E-05	2.60E-12	6.30E-12	2.61E+00	1.92E+00	**0.00E+00**	**0.00E+00**	2.15E-06	2.35E-06	1.20E-12	1.40E-12	**0.00E+00**	**0.00E+00**
$f9$	3.89E-15	3.31E-15	**0.00E+00**	**0.00E+00**	4.19E-04	3.59E-04	8.62E-11	5.33E-11	**0.00E+00**	**0.00E+00**	4.66E-16	2.54E-16	1.26E-03	4.90E-04	**0.00E+00**	**0.00E+00**
$f10$	2.56E-53	1.85E-53	2.51E-102	1.55E-101	1.05E+00	1.71E+00	6.21E-10	5.63E-10	1.01E-121	3.56E-121	4.16E-26	5.50E-26	5.85E+00	5.03E+00	**0.00E+00**	**0.00E+00**
$f11$	**0.00E+00**	**0.00E+00**	2.81E+01	3.49E+01	**0.00E+00**	**0.00E+00**	1.40E-01	3.51E-01	**0.00E+00**	**0.00E+00**	5.64E+02	4.66E+02	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
$f12$	4.23E-52	2.49E-52	1.34E-98	1.68E-98	1.39E+00	1.16E+00	8.24E-09	6.54E-09	8.73E-122	1.61E-121	9.39E-15	1.11E-14	2.95E+00	1.60E+00	**0.00E+00**	**0.00E+00**
$f13$	**0.00E+00**	**0.00E+00**	1.25E+02	5.98E+02	7.46E-06	1.48E-05	3.88E-06	4.23E-06	**0.00E+00**	**0.00E+00**	1.17E+03	1.50E+03	4.42E-05	4.06E-05	**0.00E+00**	**0.00E+00**
$f14$	1.24E+00	1.48E+00	7.29E+01	1.49E+02	1.34E-05	2.52E-05	2.82E-01	2.85E-01	**0.00E+00**	**0.00E+00**	8.99E+01	8.57E+01	3.69E-05	2.41E-05	**0.00E+00**	**0.00E+00**
$f15$	9.05E-14	7.37E-14	3.25E+00	7.11E+00	1.64E-04	1.58E-04	5.13E+01	4.48E+01	4.44E-15	0.00E+00	1.08E-10	1.21E-10	4.77E-04	2.40E-04	**4.44E-16**	**0.00E+00**
$f16$	2.01E-32	2.16E-32	3.13E+01	5.94E+01	2.32E-04	2.24E-04	1.83E-07	1.93E-07	2.77E-75	7.32E-75	2.55E+01	2.76E+01	1.85E-03	6.85E-04	**0.00E+00**	**0.00E+00**
$f17$	6.05E-04	2.99E-03	1.22E-03	5.39E-03	4.19E-06	1.12E-05	1.14E-02	1.34E-02	**0.00E+00**	**0.00E+00**	1.81E-02	2.12E-02	1.55E-05	1.48E-05	**0.00E+00**	**0.00E+00**
$f18$	2.69E-01	1.39E-01	1.22E+00	5.44E-01	6.06E-04	1.01E-03	2.20E-01	1.34E-01	1.60E-01	4.95E-02	4.27E+00	2.85E+00	1.93E-02	1.86E-02	**0.00E+00**	**0.00E+00**
$f19$	5.24E-14	7.35E-14	6.75E-14	8.46E-14	2.17E-01	1.51E-01	5.22E-08	4.40E-08	**0.00E+00**	**0.00E+00**	2.06E-13	1.69E-13	5.17E-01	2.59E-01	**0.00E+00**	**0.00E+00**
$f20$	1.22E+00	1.04E+00	2.34E+05	1.58E+05	**1.19E-09**	**2.14E-09**	2.45E+00	1.99E+00	2.66E-02	3.30E-03	4.80E+04	2.68E+04	1.86E-04	2.52E-04	1.09E+00	3.05E-02
$f21$	8.61E+01	5.98E+01	1.47E+05	9.36E+04	**4.57E-07**	**1.31E-06**	9.03E+01	4.26E+01	3.84E+01	1.08E+00	9.46E+04	6.91E+04	4.24E-06	3.10E-06	8.30E+01	4.75E+00
$f22$	2.27E-15	9.95E-16	4.51E-65	6.96E-65	6.88E+00	4.09E+00	3.72E-03	2.28E-03	1.63E-43	4.36E-43	9.25E-82	1.04E-81	1.57E+01	2.86E+00	**0.00E+00**	**0.00E+00**
$f23$	**0.00E+00**	**0.00E+00**	1.40E-16	4.57E-16	2.77E-04	5.38E-04	4.36E-11	4.60E-11	**0.00E+00**	**0.00E+00**	4.77E-15	3.19E-15	1.96E-03	1.86E-03	**0.00E+00**	**0.00E+00**
$f24$	4.10E-06	4.43E-06	1.60E-05	4.39E-05	7.55E-07	1.22E-06	1.50E-06	1.45E-06	2.27E-124	1.15E-123	1.55E-01	1.04E-01	4.66E-06	4.38E-06	**0.00E+00**	**0.00E+00**
w/t/l	0/3/21		0/1/23		2/1/21		0/0/24		2/8/14		0/0/24		2/1/21		-	

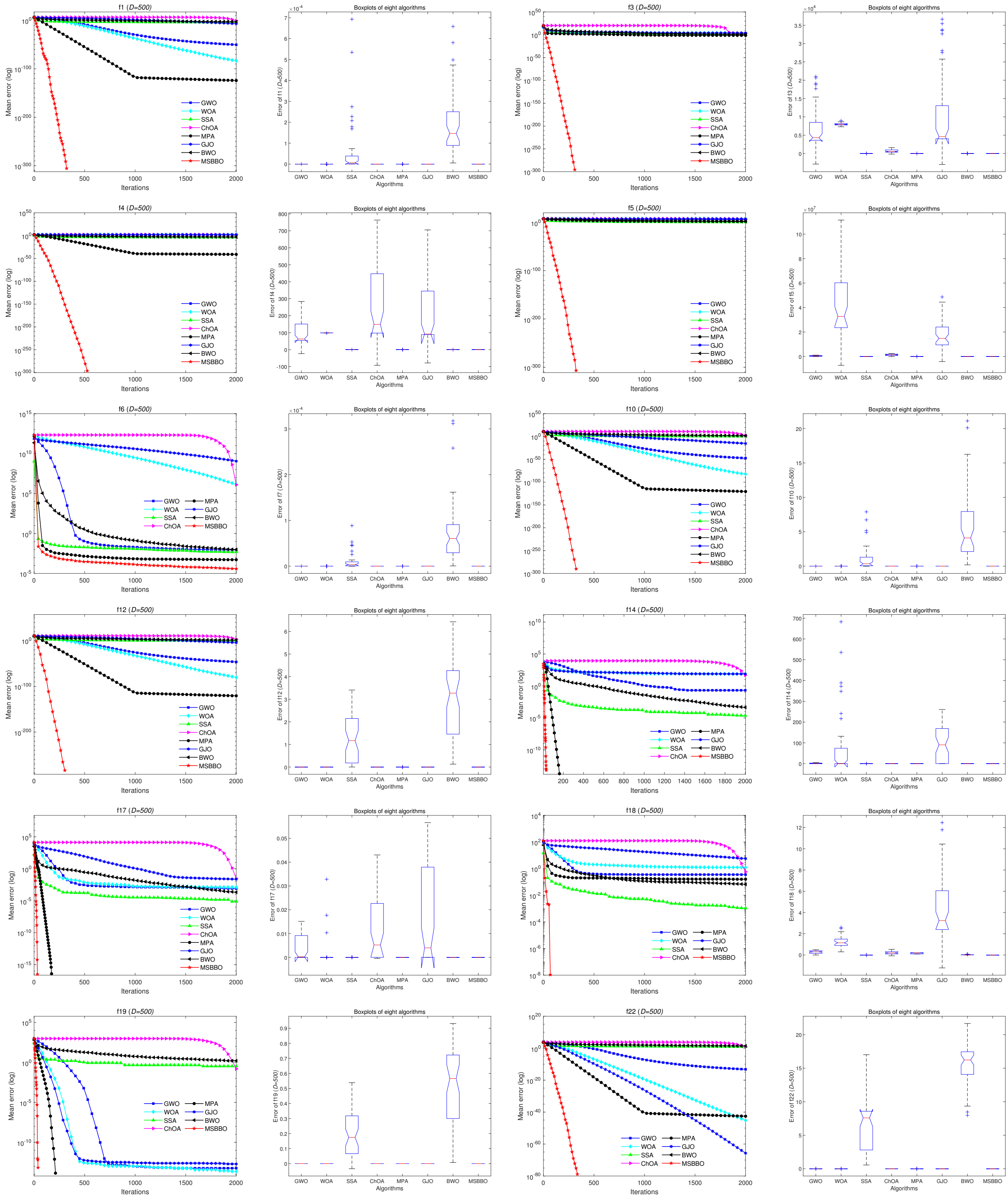


Figure 8. Convergence curves and boxplots of MSBBO and other meta-heuristic algorithms on different benchmark functions ($D = 500$).

According to the above analysis, for large-scale optimization problems, the performance of MSBBO is significantly better than that of GWO, WOA, SSA, ChOA, MPA, GJO and BWO in both solution quality and convergence speed. In addition, the convergence performance of MSBBO on 500 dimensions is not overtly different from that on low dimensions, so the performance is maintained well.

5.5. Comparison of MSBBO on Different High Dimensions

Although problems with more than 100 dimensions are defined as high-dimensional optimization problems. But in fact, many large-scale optimization problems in human society go far beyond 100 dimensions, and even beyond thousands of dimensions. In this context, the ability of the algorithm to adapt to high-dimensional optimization environment needs to be as strong as possible. In order to conduct a more thorough examination and testing, we compare the performance of MSBBO on the dimensions of 500, 1000, 2000, 5000 and 10,000, respectively. The mean error and standard deviation of the results are summarized in Table 6. As shown in Table 6, except for $f20$ and $f21$, MSBBO basically converges to the same results on 10,000 dimensions as on 500 dimensions. Though the accuracy is reduced on $f6$, it does not exceed two exponential levels. Therefore, in the search space below 10,000 dimensions, the convergence accuracy of MSBBO is basically not affected by the dimensions. Then, to fully demonstrate the advantages of MSBBO in high-dimensional environments, some convergence curves of MSBBO on different dimensions are plotted and shown in Figure 9.

From Figure 9, it is not difficult to conclude that with the increase in dimensions, the convergence curves of MSBBO are basically the same. There is no obvious separation of the convergence curves on different dimensions. In other words, there is basically no difference in the domain structure of populations on different dimensions, and they can also gather quickly in a high-dimensional space. The performance of most meta-heuristic algorithms decreases significantly with the increase in dimension, but the performance of MSBBO is relatively little affected by the change of dimension. It is fair to say that we have taken on a very challenging job, as there are very few algorithms that can plot the effects of Figure 9. Our algorithm has great advantages in solving high-dimensional optimization problems.

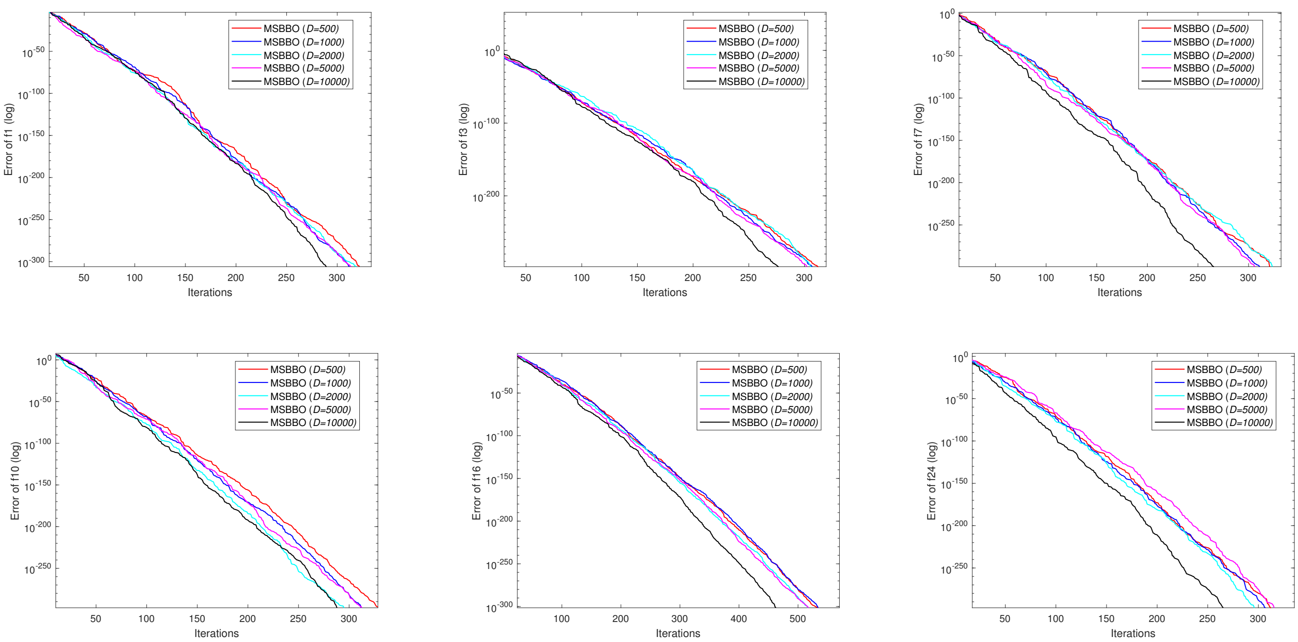


Figure 9. Convergence curves of MSBBO on $f1, f3, f7, f10, f16$ and $f24$ ($D = 500, 1000, 2000, 5000$ and 10,000).

Table 6. Results obtained by MSBBO on 24 benchmark functions ($D = 500, 1000, 2000, 5000$ and 10,000).

F	MSBBO (D = 500)		MSBBO (D = 1000)		MSBBO (D = 2000)		MSBBO (D = 5000)		MSBBO (D = 10,000)	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
$f1$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f2$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f3$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f4$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f5$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f6$	4.98E-05	5.99E-05	5.10E-05	3.81E-05	5.37E-05	9.76E-05	5.87E-05	6.03E-05	6.02E-05	5.94E-05
$f7$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f8$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f9$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f10$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f11$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f12$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f13$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f14$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f15$	4.44E-16	0.00E+00	4.44E-16	0.00E+00	4.44E-16	0.00E+00	4.44E-16	0.00E+00	4.44E-16	0.00E+00
$f16$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f17$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f18$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f19$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f20$	1.09E+00	3.05E-02	1.13E+00	1.79E-02	1.16E+00	6.15E-02	1.16E+00	6.31E-02	1.17E+00	6.43E-02
$f21$	8.30E+01	4.75E+00	1.67E+02	9.49E+00	3.32E+02	1.97E+01	8.46E+02	3.09E+01	1.90E+03	4.12E+01
$f22$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f23$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$f24$	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00

5.6. Application on Engineering Design Problems

Finally, we are also concerned about the application value of MSBBO in practical problems. To simply verify the usefulness of MSBBO, we apply it to the following six real-world engineering optimization problems: pressure vessel design, tension/compression spring design, welded beam design, speed reducer design, step-cone pulley problem and robot gripper problem. At the same time, these problems are also solved by some new advanced optimization techniques. Therefore, we selected six optimization methods (COA [62], EDO [63], OMA [64], SHO [65] and SCSO [66]) just proposed in 2023 to compare the results to fully verify the superiority and competitiveness of MSBBO. The population size of all algorithms is 50, and the maximum number of iterations is 1000.

The six engineering design problems are all constrained optimization problems. When using meta-heuristic algorithms to solve constraint optimization problems, besides the performance of the algorithm, the processing technology of constraint conditions is also very important. If the treatment of constraints is not applicable, even the algorithm with superior performance cannot search for the optimal solution. In order to make the six engineering problems meet the use conditions of the meta-heuristic algorithm and the experimental results more dependent on the search ability of the algorithm, we choose the penalty function method in literature [67] to deal with the constraints of these problems:

$$\begin{aligned}
&\text{Minimize} \\
&F(\vec{x}) = f(\vec{x}) \pm \left(\sum_{i=1}^{p} a_i G_i(\vec{x}) + \sum_{j=1}^{q} b_j H_j(\vec{x}) \right) \\
&G_i(\vec{x}) = \max\big(0, g_i(\vec{x})\big)^{\eta} \\
&H_j(\vec{x}) = \left| h_j(\vec{x}) \right|^{\lambda}
\end{aligned} \tag{13}$$

$G_i(\vec{x})$is the inequality constraint, $H_j(\vec{x})$ is the equality constraint, p is the number of inequality constraints, q is the number of equality constraints, a_i and b_j are constant, η and λ is equal to 1 or 2. For this penalty method, when the candidate solution violates any constraint, the value of the objective function increases, pushing the population into the feasible region from the infeasible solution [67].

5.6.1. Pressure Vessel Design

The goal of the pressure vessel design problem is to minimize the cost of fabrication [68]. As shown in Figure 10, L is the section length of the cylinder part without considering the head, R is the inner wall radius of the cylinder part, T_s and T_h are the wall thicknesses of the cylinder part and the head, respectively [68]. Therefore, T_s, T_h, R and L are the four optimization variables. The mathematical formulation and four constraint functions are shown in Equation (14). The optimal results of the six comparison algorithms are summarized in Table 7, and the convergence curves of the objective function on this problem are shown in Figure 11.

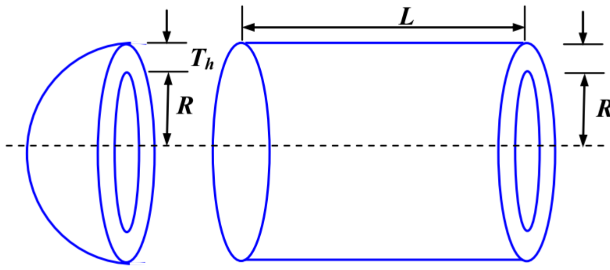


Figure 10. Pressure vessel design problem.

$$
\begin{aligned}
& X=[x_1,x_2,x_3,x_4]=[T_s,T_h,R,L] \\
& \text{minimize } f(X)=0.6224x_1x_3x_4+1.7781x_2x_3^2 \\
& +3.1661x_1^2x_4+19.84x_1^2x_3 \\
& \text{s.t. } g_1(X)=-x_1+0.0193x_3\le 0 \\
& g_2(X)=-x_2+0.00954x_3\le 0 \\
& g_3(X)=-\pi x_3^2x_4-4\pi x_3^3/3+1296000\le 0 \\
& g_4(X)=x_4-240\le 0 \\
& \text{where } 0\le x_i\le 100, i=1,2; 10\le x_i\le 200, i=3,4
\end{aligned} \tag{14}
$$

Table 7. Results for pressure vessel design problem.

Algorithm	Optimal Values for Variables				Optimal Cost
	T_s	T_h	R	L	
COA	0.78425309	0.38785440	40.63463345	195.95592683	5902.85656647
EDO	0.77997855	0.38558510	40.38924403	199.85289900	5909.44930611
OMA	0.77827133	0.38470014	40.32491200	199.92634333	5885.51298728
SHO	0.78255188	0.38700919	40.54668188	196.86310662	5893.43974421
SCSO	0.79522133	0.38960618	40.66736096	195.21548366	5976.11395352
MSBBO	**0.77816864**	**0.38464916**	**40.31961873**	**200**	**5885.33277894**

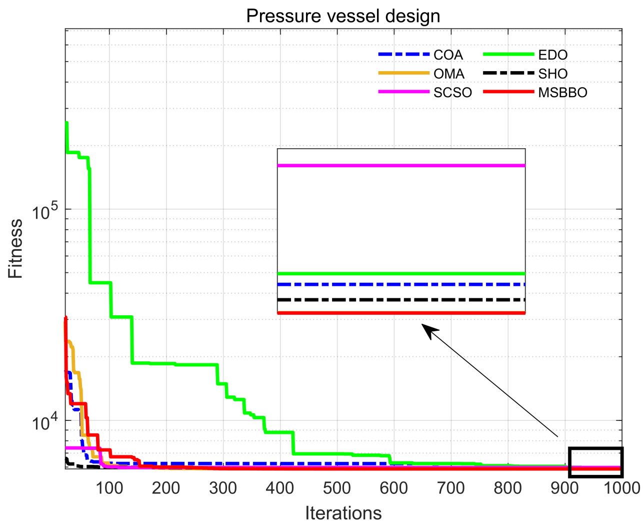


Figure 11. Optimal convergence curves on pressure vessel design problem.

As can be seen from Table 7 and Figure 11, our work achieves a smaller objective function value (5885.33277894) than the other five new technologies, which further optimizes the pressure vessel design problem. Under the condition of satisfying various constraints, MSBBO gives a new and better solution: [T_s, T_h, R, L] = [0.77816864, 0.38464916, 40.31961873, 200].

5.6.2. Tension/Compression Spring Design

The tension/compression spring design problem is to minimize the weight of the spring while meeting the constraints of minimum deflection, vibration frequency, and shear stress [69]. As shown in Figure 12, it consists of three variables: the wire diameter (d), the mean coil diameter (D) and the number of active coils (P). The mathematical model is given in Equation (15). The results are summarized in Table 8, and the convergence curves are shown in Figure 13. It is not difficult to find that MSBBO finds the value of the objective function (0.01266959) with higher precision when all variables conform to the constraint.

In contrast, the results of COA, EDO and SHO are also very close to our algorithm. As new optimization techniques, they also provide reliable solutions.

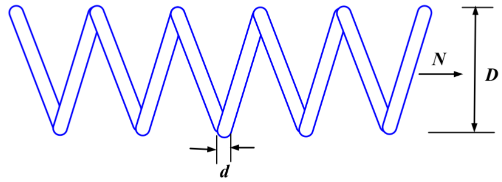


Figure 12. Tension/compression spring design problem.

$$
\begin{aligned}
& X = [x_1, x_2, x_3] = [d, D, P] \\
& \text{minimize } f(X) = (x_3 + 2)x_2 x_1^2 \\
& \text{s.t. } g_1(X) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0 \\
& g_2(X) = \frac{4x_2^2 - x_1 x_2}{12566\left(x_2 x_1^3 - x_1^4\right)} + \frac{1}{5108 x_1^2 - 1} \leq 0 \\
& g_3(X) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0 \\
& g_4(X) = \frac{x_1 + x_2}{1.5} - 1 \leq 0 \\
& \text{where } 0.05 \leq x_1 \leq 2, 0.25 \leq x_2 \leq 1.3, 2 \leq x_3 \leq 15.
\end{aligned}
\tag{15}
$$

Table 8. Results for tension/compression spring design problem.

Algorithm	Optimal Values for Variables			Optimal Cost
	d	***D***	***P***	
COA	0.05122526	0.34565821	12.14215544	0.01269823
EDO	0.05190445	0.36182344	11.22500450	0.01267212
OMA	0.05356063	0.40339534	8.53973118	0.01272961
SHO	0.05058048	0.33062883	13.22719241	0.01268814
SCSO	0.05718457	0.50390409	5.53685327	0.01318243
MSBBO	**0.05120413**	**0.34516305**	**12.26044071**	**0.01266959**

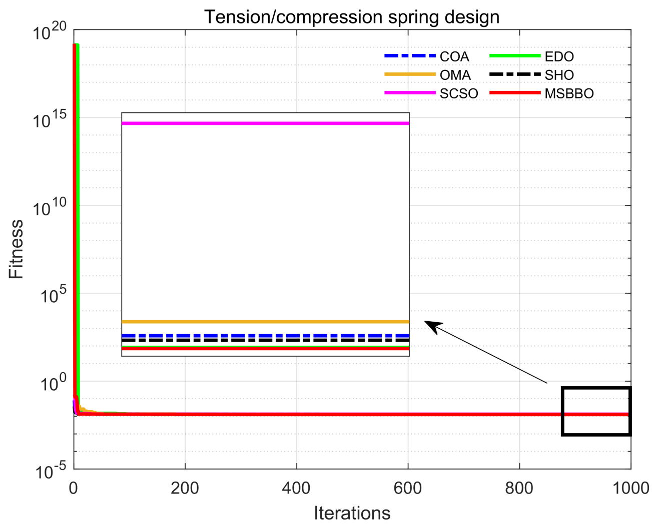


Figure 13. Optimal convergence curves on tension/compression spring design problem.

5.6.3. Welded Beam Design

The objective of the welding beam design problem is to obtain the minimum manufacturing cost [70]. As shown in Figure 14, this optimization problem has four variables that need to be calculated: the thickness of the weld (h), the length of the attached part of the bar (l), the height of the bar (t), and the thickness of the bar (b). Then, there are seven constraints that need to be satisfied in this optimization design [70]. These constraints include shear stress (τ), bending stress in beam (σ), deflection of beam end (δ) and buckling load of bar (P_b). The mathematical model of this problem is given in Equation (16). Similarly, Table 9 shows the optimal results of the six comparison algorithms, and Figure 15 shows the convergence curves of the objective function on the welded beam design problem.

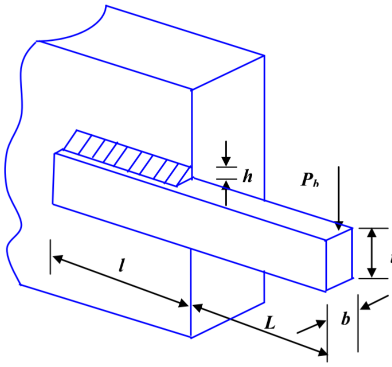


Figure 14. Welded beam design problem.

$$
\begin{aligned}
&\vec{x} = [x_1 x_2 x_3 x_4] = [h, l, t, b] \\
&\text{minimize } f(\vec{x}) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2), \\
&g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \leqslant 0, \\
&g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \leqslant 0, \\
&g_3(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \leqslant 0, \\
&g_4(\vec{x}) = x_1 - x_4 \leqslant 0, \\
&g_5(\vec{x}) = P - P_c(\vec{x}) \leqslant 0, \\
&g_6(\vec{x}) = 0.125 - x_1 \leqslant 0, \\
&g_7(\vec{x}) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2) - 5.0 \leqslant 0 \\
&0.1 \leqslant x_1 \leqslant 2, \\
&0.1 \leqslant x_2 \leqslant 10, \\
&0.1 \leqslant x_3 \leqslant 10 \\
&0.1 \leqslant x_4 \leqslant 2 \\
&\tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau'' \frac{x_2}{2R} + (\tau'')^2}, \\
&\tau' = \frac{p}{\sqrt{2} x_1 x_2}, \tau'' = \frac{MR}{J}, M = P\left(L + \frac{x_2}{2}\right), \\
&R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \\
&J = 2\left\{\sqrt{2} x_1 x_2 \left[\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}, \\
&\sigma(\vec{x}) = \frac{6PL}{x_4 x_3^2}, \delta(\vec{x}) = \frac{6PL^3}{E x_3^2 x_4} \\
&P_c(\vec{x}) = \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), \\
&P = 6000 \text{ lb}, L = 14 \text{ in.}, \quad \delta_{\max} = 0.25 \text{ in.}, \\
&E = 30 \times 10^6 \text{ psi}, \ G = 12 \times 10^6 \text{ psi}, \\
&\tau_{\max} = 13{,}600 \text{ psi}, \ \sigma_{\max} = 30{,}000 \text{ psi}
\end{aligned}
\tag{16}
$$

Table 9. Results for welded beam design problem.

Algorithm	Optimal Values for Variables				Optimal Cost
	h	*l*	*t*	*b*	
COA	0.18479628	3.68381216	9.22577702	0.19867497	1.69837299
EDO	0.19793629	3.36058378	9.18941514	0.19902434	1.67299413
OMA	**0.19883231**	**3.33736530**	**9.19202432**	**0.19883231**	**1.67021773**
SHO	0.17440082	3.86323716	9.20290796	0.19878156	1.70196639
SCSO	0.17898507	3.68017384	9.47591891	0.19754191	1.72245954
MSBBO	**0.19883231**	**3.33736530**	**9.19202432**	**0.19883231**	**1.67021773**

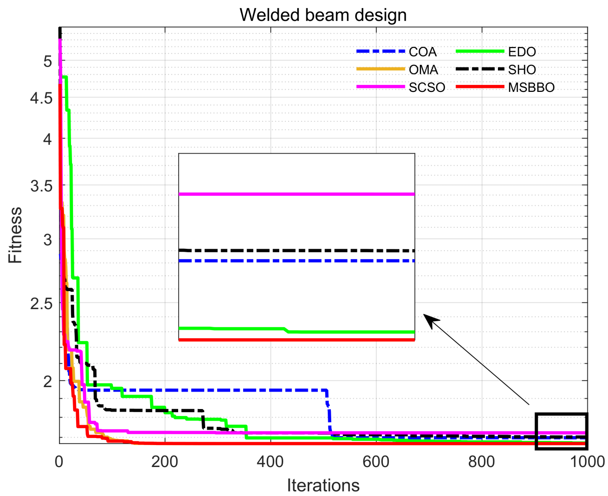


Figure 15. Optimal convergence curves on welded beam design problem.

According to the experimental results, MSBBO and OMA search for the same optimal solution: $[h, l, t, b]$ = [0.19883231, 3.33736530, 9.19202432, 0.19883231]. The values of the variables obtained by the two algorithms are exactly equal (1.67021773). This shows that the performance of the two methods is not significantly different on the welded beam design problem. As a newly proposed optimization technique, OMA has been fully tested theoretically. Therefore, it can be considered that the solving ability of MSBBO on this problem has reached the level of advanced optimization technology.

5.6.4. Speed Reducer Design

The purpose of the speed reducer design problem is to minimize the cost when the 11 constraints are met [71]. The problem consists of seven decision variables as shown in Figure 16. Its mathematical model is shown in Equation (17). Similarly, six optimization methods are used to search for optimal decision variables and objective function values for this problem. The experimental results and convergence curves are shown in Table 10 and Figure 17, respectively. It can be found that MSBBO also obtains the optimal objective function value of the six algorithms. In addition, the EDO, OMA and SHO calculations are also very close to our work, and the solutions they provide are also worth referring to.

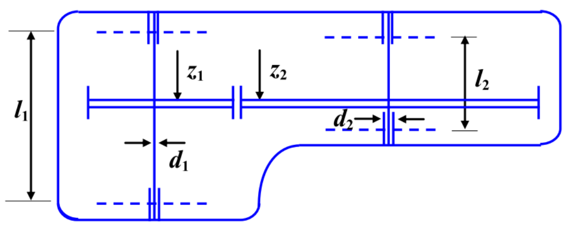


Figure 16. Speed reducer design problem.

$$
\begin{aligned}
&\vec{x} = [x_1, x_2, x_3, x_4, x_5, x_6, x_7] = [b, m, z, l_1, l_2, d_1, d_2] \\
&\text{minimize} \quad \begin{array}{l} f_4(\vec{x}) = 0.7854 x_1 x_2^2 \left(3.3333 x_3^2 + 14.9334 x_3 - 43.0934\right) \\ -1.508 x_1 \left(x_6^2 + x_7^2\right) + 0.7854 x_1 \left(x_4 x_6^2 - x_5 x_7^2\right). \end{array} \\
&g_1(\vec{x}) = \frac{27}{x_1 x_2^2 x_3} - 1 \leq 0, \\
&g_2(\vec{x}) = \frac{397.5}{x_1 x_2^2 x_3^2} - 1 \leq 0, \\
&g_3(\vec{x}) = \frac{1.93 x_4^2}{x_2 x_6^4 x_3} - 1 \leq 0, \\
&g_4(\vec{x}) = \frac{1.93 x_5^2}{x_2 x_7^4 x_3} - 1 \leq 0, \\
&g_5(\vec{x}) = \frac{\left(\left(\frac{74 x_4}{x_2 x_3}\right)^2 + 16.9 \times 10^6\right)^{0.5}}{110 x_6^3} - 1 \leq 0, \\
&g_6(\vec{x}) = \frac{\left(\left(\frac{745 x_5}{x_2 x_3}\right)^2 + 157.5 \times 10^6\right)^{0.5}}{85 x_7^3} - 1 \leq 0, \\
&g_7(\vec{x}) = \frac{x_2 x_3}{40} - 1 \leq 0, \\
&g_8(\vec{x}) = \frac{5 x_2}{x_1} - 1 \leq 0, \\
&g_9(\vec{x}) = \frac{x_1}{12 x_2} - 1 \leq 0, \\
&g_{10}(\vec{x}) = \frac{1.5 x_6 + 1.9}{x_4} - 1 \leq 0, \\
&g_{11}(\vec{x}) = \frac{1.1 x_7 + 1.9}{x_5} - 1 \leq 0. \\
&\text{where } 2.6 \leq x_1 \leq 3.6, 0.7 \leq x_2 \leq 0.8, 17 \leq x_3 \leq 28, \\
&7.3 \leq x_4 \leq 8.3, 7.3 \leq x_5 \leq 8.3, \\
&2.9 \leq x_6 \leq 3.9, 5.0 \leq x_7 \leq 5.5.
\end{aligned} \tag{17}
$$

Table 10. Results for the speed reducer design problem.

Algorithm	**Optimal Values for Variables**							**Optimal Cost**
	b	m	z	l_1	l_2	d_1	d_2	
COA	3.50000068	0.7	17	7.3	8.007338125	3.354940375	5.286998297	3002.176768192
EDO	3.50025780	0.70000066	17	7.3	7.723941215	3.350658886	5.286670403	2994.758133972
OMA	3.5	0.7	17	7.3	7.715319911	3.350540949	5.286654465	2994.424465758
SHO	3.50004682	0.7	17	7.30178682	7.715591607	3.350558567	5.286654691	2994.469209053
SCSO	3.51205842	0.7	17	7.3	7.766370701	3.351151796	5.286672013	3000.448065947
MSBBO	**3.50000001**	**0.7**	**17**	**7.30000014**	**7.715320035**	**3.350540986**	**5.286654467**	**2994.424489954**

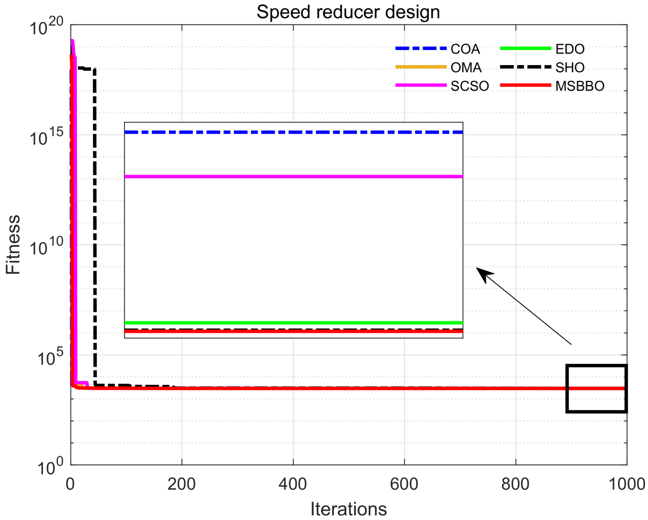


Figure 17. Optimal convergence curves on the speed reducer design problem.

5.6.5. Step-Cone Pulley Problem

This problem requires the design of a step-cone pulley based on the values of five decision variables [72]. Among them, the four design variables are the diameter of each step (d_1, d_2, d_3, d_4), and the last design variable is the width of the pulley (w) as shown in Figure 18. The problem consists of 11 constraint adjustments, of which 3 are equality constraints and 8 are inequality constraints, to ensure that all step sizes, tension ratios, and belt transfer power are the same. Its mathematical model is shown in Equation (18).

For the sake of discussion, Table 11 shows the optimal results of all comparison algorithms, and Figure 19 shows their convergence curves on this problem. According to the results in Table 11, it can be concluded that MSBBO can still obtain the ideal optimal solution in several advanced algorithms on the step-cone pulley problem. In addition, OMA also obtains the same precision as the MSBBO objective function value (8.18149598). In Figure 19, the MSBBO convergence curve is at the bottom and coincides with the OMA convergence curve. But they have different decision variable values. This shows that the optimal solution of the same problem is not unique, and our work provides new design parameter values for the step-cone pulley problem: [d_1, d_2, d_3, d_4, w] = [16.96572695, 28.25753037, 50.79673307, 84.49572374, 89.99999337].

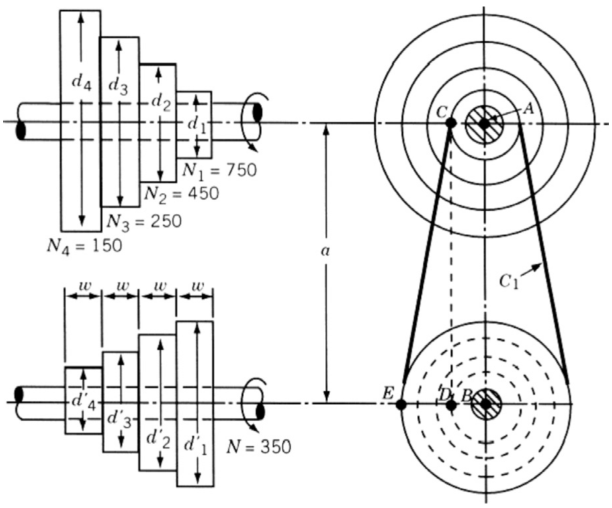


Figure 18. The step-cone pulley problem.

$$\begin{aligned}
&\text{minimize: } f(\bar{x}) = \rho\omega\left[d_1^2\left\{11+\left(\frac{N_1}{N}\right)^2\right\}+d_2^2\left\{1+\left(\frac{N_2}{N}\right)^2\right\}\right.\\
&\left.+d_3^2\left\{1+\left(\frac{N_3}{N}\right)^2\right\}+d_4^2\left\{1+\left(\frac{N_4}{N}\right)^2\right\}\right]\\
&h_1(\bar{x}) = C_1 - C_2 = 0,\\
&h_2(\bar{x}) = C_1 - C_3 = 0,\\
&h_3(\bar{x}) = C_1 - C_4 = 0\\
&g_{i=1,2,3.4}(\bar{x}) = -R_i \le 2,\\
&g_{i=1,2,3.4}(\bar{x}) = (0.75 \times 745.6998) - P_i \le 0\\
&C_i = \frac{\pi d_i}{2}\left(1+\frac{N_i}{N}\right)+\frac{\left(\frac{N_i}{N}-1\right)^2}{4a}+2a, i=(1,2,3,4)\\
&R_i = \exp\left(\mu\left\{\pi - 2\sin^{-1}\left\{\left(\frac{N_i}{N}-1\right)\frac{d_i}{2a}\right\}\right\}\right), i=(1,2,3,4)\\
&P_i = \mathrm{st}!(1-R_i)\frac{\pi d_i N_i}{60}, i=(1,2,3,4), t=8\text{ mm}\\
&s = 1.75\text{ MPa}, \mu=0.35, \rho=7200\text{ kg/m}^3, a=3\text{ mm}
\end{aligned} \tag{18}$$

Table 11. Results for step-cone pulley problem.

Algorithm	Optimal Values for Variables					Optimal Cost
	d_1	d_2	d_3	d_4	w	
COA	17.43376313	29.03743308	50.95003894	89.57133620	89.72893135	8.80527504
EDO	17.00069596	28.33533123	50.82602160	84.55984091	89.95618331	8.19564986
OMA	**16.96572313**	**28.25752810**	**50.79671071**	**84.49571607**	**90**	**8.18149598**
SHO	17.12626161	28.25787354	50.79728294	84.49688761	89.99901887	8.19717274
SCSO	18.25834926	28.68767069	51.89516285	88.88993899	88.65129296	8.75660602
MSBBO	**16.96572695**	**28.25753037**	**50.79673307**	**84.49572374**	**89.99999337**	**8.18149598**

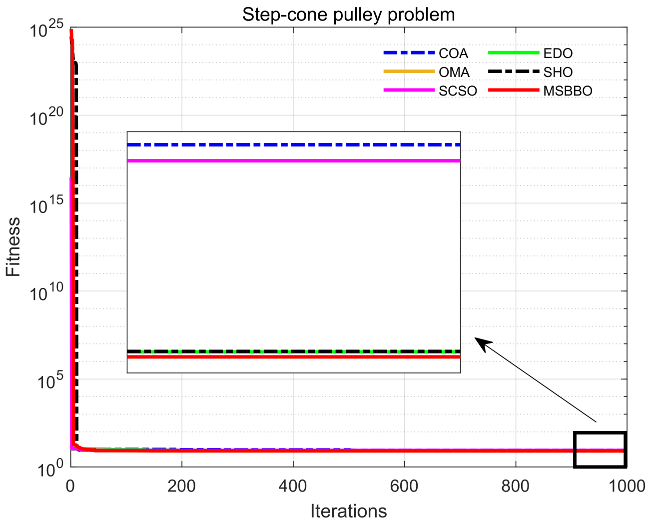


Figure 19. Optimal convergence curves on the step-cone pulley problem.

5.6.6. Robot Gripper Problem

The optimization goal of the robot gripper problem is to minimize the difference between the maximum and minimum force [73]. It is by the gripper end displacement of the range applied to the gripper. This problem consists of seven consecutive decision variables: a, b, c, d, e, f and δ as shown in Figure 20. The robot gripper problem needs to satisfy seven inequality constraints, which is complex. Its mathematical model is shown in Equation (19). As with the other problems, the results of the six comparison algorithms are summarized in Table 12, and Figure 21 shows their convergence curves. It is not difficult to see that MSBBO, as an enhanced variant of BBO, is significantly better than the other five new meta-heuristic algorithms when seven inequality constraints are met. This shows that our improvement strategies have effectively enhanced the application value of BBO. Therefore, MSBBO can also be widely used on engineering constrained optimization problems.

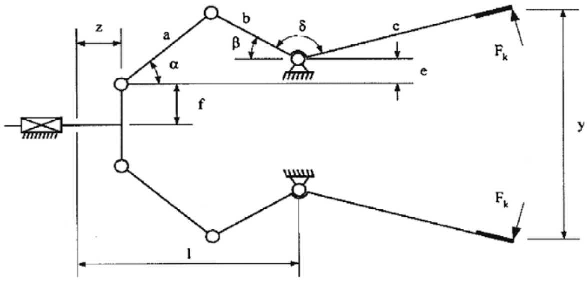


Figure 20. The robot gripper problem.

$$
\begin{aligned}
&\text{minimize} f(x) = \max_z F_k(x,z) - \min_z F_k(x,z) \\
&g_1(x) = Y_{\min} - y(x, Z_{\max}) \geq 0, \\
&g_2(x) = y(x, Z_{\max}) \geq 0, \quad g_3(x) = y(x,0) - Y_{\max} \geq 0, \\
&g_4(x) = Y_G - y(x,0) \geq 0, \quad g_5(x) = (a+b)^2 - l^2 - e^2 \geq 0, \\
&g_6(x) = (l - Z_{\max})^2 + (a-e)^2 - b^2 \geq 0, \\
&g_7(x) = l - Z_{\max} \geq 0, \\
&g = \sqrt{(l-z)^2 + e^2}, \quad \alpha = \arccos\left(\frac{a^2 + g^2 - b^2}{2ag}\right) + \phi \\
&\beta = \arccos\left(\frac{b^2 + g^2 - a^2}{2bg}\right) - \phi \\
&\phi = \arctan\left(\frac{e}{l-z}\right) + \phi, \quad F_k = \left(\frac{Pb\sin(\alpha+\beta)}{2c\cos(\alpha)}\right) \\
&y(x,z) = 2(e + f + c\sin(\beta+\delta)) \\
&Y_{\min} = 50, \quad Y_{\max} = 100, \quad P = 100 \\
&Y_G = 150, \quad Z_{\max} = 100, \quad P \leq 0 \leq c \leq 200 \\
&10 \leq a, b, f \leq 150, \quad 100 \leq c \leq 1 \leq \delta \leq 3.14 \\
&0 \leq e \leq 50, \quad 100 \leq l \leq 300, \quad 1 \leq \delta
\end{aligned}
\tag{19}
$$

Table 12. Results for the robot gripper problem.

Algorithm	Optimal Values for Variables							Optimal Cost
	a	*b*	*c*	*d*	*e*	*f*	*δ*	
COA	149.99754208	149.65008561	159.71207516	0.00198394	10.06515284	115.49379010	1.57052121	3.49048709
EDO	149.96736172	98.92864085	199.99158377	49.94588624	150	124.89353758	2.86707789	3.55150434
OMA	147.04217688	134.34472269	200	12.48032203	149.35062039	106.73574813	2.44307744	2.84065650
SHO	144.95148508	144.77644040	100.00021186	0.05032617	11.72681600	100.42656588	1.36167252	5.26168942
SCSO	149.14088858	148.88101175	148.11100725	0.04507554	60.56845351	108.40010554	1.99300044	3.63308379
MSBBO	**149.61527355**	**149.39619170**	**199.20669308**	**8.60E-16**	**149.63248094**	**108.80577404**	**2.42474600**	**2.69788610**

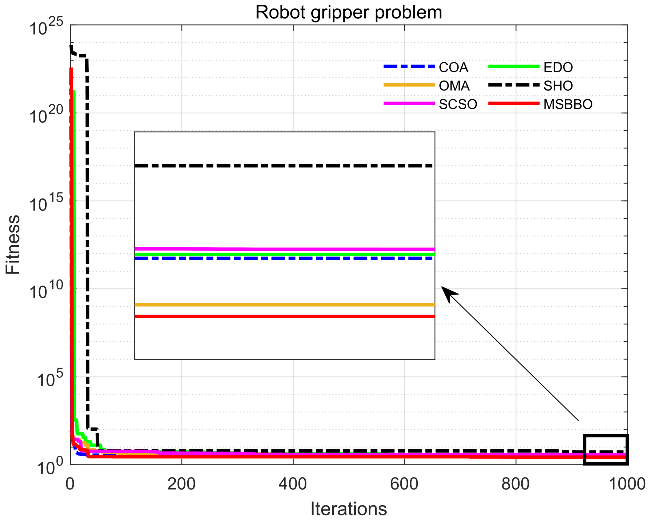


Figure 21. Optimal convergence curves on the robot gripper problem.

6. Conclusions

This paper proposes a comprehensive multi-strategy enhanced BBO variant. Firstly, the example chasing strategy is proposed to effectively maintain the population diversity. Then, the heuristic crossover strategy is designed to enhance the search ability of the algorithm. Finally, the prey search–attack strategy is designed to balance the exploration and exploitation of the algorithm, which improves the convergence accuracy and speed effectively. MSBBO makes BBO suitable for high-dimensional optimization environments. Compared to other BBO variants (PRBBO, BBOSB, HGBBO, FABBO, BLEHO, MPBBO and BBOIMAM), MSBBO uses only three improvement strategies to greatly improve the BBO convergence performance, which enables it to effectively solve high-dimensional optimization problems and has simpler computational complexity than the original BBO. At the same time, MSBBO is compared with seven latest meta-heuristic algorithms (GWO, WOA, SSA, ChOA, MPA, GJO and BWO). Experimental results show that MSBBO has better performance than all compared algorithms and is more suitable for solving large-scale optimization problems. Further, MSBBO and five new optimization techniques (COA, EDO, OMA, SHO and SCSO) are used to solve six engineering problems, and the results show that our work is also more competitive than these new optimization techniques.

Because the algorithm can effectively balance exploration and exploitation, and can adapt to high-dimensional optimization environments, it is expected to perform well in multi-objective problems. In addition, in the case of hardware equipment, we can try to implement this algorithm in a distributed way and solve more complex practical problems. Therefore, we plan to implement it in a distributed manner in future studies. Moreover, it can also be combined with other techniques in the search area of optimization problems, such as local or global search methods, to improve its performance. Meanwhile, it can be used to solve many complex problems, such as feature selection, image segmentation, job-shop scheduling and vehicle routing problems, as well as optimization node positioning problems in systems. Therefore, we will pay more attention to the application of the MSBBO algorithm for complex optimization problems in the future.

Author Contributions: C.G.: Investigation, Methodology, Experiment, Writing—original draft. T.L.: Supervision, Funding acquisition. Y.G.: Funding acquisition, Writing—review and editing. Z.Z.: Data curation, Writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Key Project of Ningxia Natural Science Foundation [2022AAC02043]. the National Natural Science Foundation of China under Grant [61902291], the Construction Project of First-class Subjects in Ningxia Higher Education [NXYLXK2017B09], and the Major Proprietary Funded Project of North Minzu University [ZDZX201901].

Data Availability Statement: All the data in Section 5 were obtained under the same experimental environment. Then, all the source programs of the compared BBO variants in Section 5.3 are coded according to their original references. We solemnly declare that all data in this paper are true and valid.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper. All authors guarantee that the paper is legitimate and belongs to their own scientific research results.

References

1. Qian, K.; Liu, Y.; Tian, L.; Bao, J. Robot path planning optimization method based on heuristic multi-directional rapidly-exploring tree. *Comput. Electr. Eng.* **2020**, *85*, 106688. [CrossRef]
2. Islam, M.A.; Gajpal, Y.; ElMekkawy, T.Y. Hybrid particle swarm optimization algorithm for solving the clustered vehicle routing problem. *Appl. Soft Comput.* **2021**, *110*, 107655. [CrossRef]
3. Ertenlice, O.; Kalayci, C.B. A survey of swarm intelligence for portfolio optimization: Algorithms and applications. *Swarm Evol. Comput.* **2018**, *39*, 36–52. [CrossRef]
4. Ding, H.; Gu, X. Hybrid of human learning optimization algorithm and particle swarm optimization algorithm with scheduling strategies for the flexible job-shop scheduling problem. *Neurocomputing* **2020**, *414*, 313–332. [CrossRef]
5. Darvish, A.; Ebrahimzadeh, A. Improved fruit-fly optimization algorithm and its applications in antenna arrays synthesis. *IEEE Trans. Antennas Propag.* **2018**, *66*, 1756–1766. [CrossRef]
6. Liu, Y.; Jin, S.; Zhou, J.; Hu, Q. A branch-and-bound algorithm for the unit-capacity resource constrained project scheduling problem with transfer times. *Comput. Oper. Res.* **2023**, *151*, 106097. [CrossRef]
7. Babaie-Kafaki, S. A survey on the Dai–Liao family of nonlinear conjugate gradient methods. *Rairo-Oper. Res.* **2023**, *57*, 43–58. [CrossRef]
8. Mittal, G.; Giri, A.K. A modified steepest descent method for solving non-smooth inverse problems. *J. Comput. Appl. Math.* **2023**, *424*, 114997. [CrossRef]
9. Holland, J. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Application to Biology, Control and Artificial Intelligence*; University of Michigan Press: Ann Arbor, MI, USA, 1975.
10. Storn, R.; Price, K. Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341. [CrossRef]
11. Dasgupta, D. An artificial immune system as a multi-agent decision support system. In Proceedings of the SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218), San Diego, CA, USA, 14 October 1998; Volume 4, pp. 3816–3820. [CrossRef]
12. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95 International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948. [CrossRef]
13. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng.* **2016**, *95*, 51–67. [CrossRef]
14. Xue, J.; Shen, B. A novel swarm intelligence optimization approach: Sparrow search algorithm. *Syst. Sci. Control. Eng.* **2020**, *8*, 22–34. [CrossRef]
15. Simon, D. Biogeography-based optimization. *IEEE Trans. Evol. Comput.* **2008**, *12*, 702–713. [CrossRef]
16. Xing, B.; Gao, W.J.; Xing, B.; Gao, W.J. Invasive weed optimization algorithm. In *Innovative Computational Intelligence: A Rough Guide to 134 Clever Algorithms*; Springer: Cham, Switzerland, 2014; pp. 177–181. [CrossRef]
17. Kiran, M.S. TSA: Tree-seed algorithm for continuous optimization. *Expert Syst. Appl.* **2015**, *42*, 6686–6698. [CrossRef]
18. Hashim, F.A.; Hussain, K.; Houssein, E.H.; Mabrouk, M.S.; Al-Atabany, W. Archimedes optimization algorithm: A new metaheuristic algorithm for solving optimization problems. *Appl. Intell.* **2021**, *51*, 1531–1551. [CrossRef]
19. 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. [CrossRef]
20. 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. [CrossRef]
21. Das, B.; Mukherjee, V.; Das, D. Student psychology based optimization algorithm: A new population based optimization algorithm for solving optimization problems. *Adv. Eng. Softw.* **2020**, *146*, 102804. [CrossRef]
22. Rao, R.V.; Savsani, V.J.; Vakharia, D.P. Teaching–learning-based optimization: An optimization method for continuous non-linear large scale problems. *Inf. Sci.* **2012**, *183*, 1–15. [CrossRef]
23. Ahmadi, S.A. Human behavior-based optimization: A novel metaheuristic approach to solve complex optimization problems. *Neural Comput. Appl.* **2017**, *28* (Suppl. S1), 233–244. [CrossRef]
24. Chakraborty, S.; Saha, A.K.; Chakraborty, R.; Saha, M. An enhanced whale optimization algorithm for large scale optimization problems. *Knowl.-Based Syst.* **2021**, *233*, 107543. [CrossRef]

25. Rivera, M.M.; Guerrero-Mendez, C.; Lopez-Betancur, D.; Saucedo-Anaya, T. Dynamical Sphere Regrouping Particle Swarm Optimization: A Proposed Algorithm for Dealing with PSO Premature Convergence in Large-Scale Global Optimization. *Mathematics* **2023**, *11*, 4339. [CrossRef]
26. Long, W.; Wu, T.; Liang, X.; Xu, S. Solving high-dimensional global optimization problems using an improved sine cosine algorithm. *Expert Syst. Appl.* **2019**, *123*, 108–126. [CrossRef]
27. Goel, L. A novel approach for face recognition using biogeography based optimization with extinction and evolution. *Multimed. Tools Appl.* **2022**, *81*, 10561–10588. [CrossRef]
28. Jain, A.; Rai, S.; Srinivas, R.; Al-Raoush, R.I. Bioinspired modeling and biogeography-based optimization of electrocoagulation parameters for enhanced heavy metal removal. *J. Clean. Prod.* **2022**, *338*, 130622. [CrossRef]
29. Zhang, Z.; Gao, Y.; Zuo, W. A dual biogeography-based optimization algorithm for solving high-dimensional global optimization problems and engineering design problems. *IEEE Access* **2022**, *10*, 55988–56016. [CrossRef]
30. Zhang, Q.; Wei, L.; Yang, B. Research on Improved BBO Algorithm and Its Application in Optimal Scheduling of Micro-Grid. *Mathematics* **2022**, *10*, 2998. [CrossRef]
31. Ma, H.; Simon, D.; Siarry, P.; Yang, Z.; Fei, M. Biogeography-based optimization: A 10-year review. *IEEE Trans. Emerg. Top. Comput. Intell.* **2017**, *1*, 391–407. [CrossRef]
32. Ergezer, M.; Simon, D.; Du, D. Oppositional biogeography-based optimization. In Proceedings of the 2009 IEEE International Conference on Systems, Man and Cybernetics, San Antonio, TX, USA, 11–14 October 2009; pp. 1009–1014. [CrossRef]
33. Wang, G.G.; Gandomi, A.H.; Alavi, A.H. An effective krill herd algorithm with migration operator in biogeography-based optimization. *Appl. Math. Model.* **2014**, *38*, 2454–2462. [CrossRef]
34. Lohokare, M.R.; Panigrahi, B.K.; Pattnaik, S.S.; Devi, S.; Mohapatra, A. Neighborhood search-driven accelerated biogeography-based optimization for optimal load dispatch. *IEEE Trans. Syst. Man Cybern. Part Appl. Rev.* **2012**, *42*, 641–652. [CrossRef]
35. Chen, X.; Tianfield, H.; Du, W.; Liu, G. Biogeography-based optimization with covariance matrix based migration. *Appl. Soft Comput.* **2016**, *45*, 71–85. [CrossRef]
36. Sang, X.; Liu, X.; Zhang, Z.; Wang, L. Improved biogeography-based optimization algorithm by hierarchical tissue-like P system with triggering ablation rules. *Math. Probl. Eng.* **2021**, *2021*, 6655614. [CrossRef]
37. Zhang, Z.; Gao, Y.; Liu, Y.; Zuo, W. A hybrid biogeography-based optimization algorithm to solve high-dimensional optimization problems and real-world engineering problems. *Appl. Soft Comput.* **2023**, *144*, 110514. [CrossRef]
38. Zhang, Z.; Gao, Y.; Guo, E. A supercomputing method for large-scale optimization: A feedback biogeography-based optimization with steepest descent method. *J. Supercomput.* **2023**, *79*, 1318–1373. [CrossRef]
39. Garg, V.; Deep, K. Performance of Laplacian Biogeography-Based Optimization Algorithm on CEC 2014 continuous optimization benchmarks and camera calibration problem. *Swarm Evol. Comput.* **2016**, *27*, 132–144. [CrossRef]
40. Zhao, F.; Qin, S.; Zhang, Y.; Ma, W.; Zhang, C.; Song, H. A two-stage differential biogeography-based optimization algorithm and its performance analysis. *Expert Syst. Appl.* **2019**, *115*, 329–345. [CrossRef]
41. Khademi, G.; Mohammadi, H.; Simon, D. Hybrid invasive weed/biogeography-based optimization. *Eng. Appl. Artif. Intell.* **2017**, *64*, 213–231. [CrossRef]
42. Zhang, X.; Wang, D.; Fu, Z.; Liu, S.; Mao, W.; Liu, G.; Jiang, Y.; Li, S. Novel biogeography-based optimization algorithm with hybrid migration and global-best Gaussian mutation. *Appl. Math. Model.* **2020**, *86*, 74–91. [CrossRef]
43. Zhang, X.; Kang, Q.; Tu, Q.; Cheng, J.; Wang, X. Efficient and merged biogeography-based optimization algorithm for global optimization problems. *Soft Comput.* **2019**, *23*, 4483–4502. [CrossRef]
44. Reihanian, A.; Feizi-Derakhshi, M.R.; Aghdasi, H.S. NBBO: A new variant of biogeography-based optimization with a novel framework and a two-phase migration operator. *Inf. Sci.* **2019**, *504*, 178–201. [CrossRef]
45. Gerton, J.L.; Hawley, R.S. Homologous chromosome interactions in meiosis: Diversity amidst conservation. *Nat. Rev. Genet.* **2005**, *6*, 477–487. [CrossRef]
46. Dhiman, G.; Garg, M.; Nagar, A.; Kumar, V.; Dehghani, M. A novel algorithm for global optimization: Rat swarm optimizer. *J. Ambient. Intell. Humaniz. Comput.* **2021**, *12*, 8457–8482. [CrossRef]
47. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [CrossRef]
48. Atanassov, K.; Vassilev, P. Intuitionistic fuzzy sets and other fuzzy sets extensions representable by them. *J. Intell. Fuzzy Syst.* **2020**, *38*, 525–530. [CrossRef]
49. Atanassov, K. Intuitionistic fuzzy modal topological structure. *Mathematics* **2022**, *10*, 3313. [CrossRef]
50. Feng, Q.; Liu, S.; Zhang, J.; Yang, G.; Yong, L. Improved biogeography-based optimization with random ring topology and Powell's method. *Appl. Math. Model.* **2017**, *41*, 630–649. [CrossRef]
51. Xiong, G.; Shi, D. Hybrid biogeography-based optimization with brain storm optimization for non-convex dynamic economic dispatch with valve-point effects. *Energy* **2018**, *157*, 424–435. [CrossRef]
52. Farrokh Ghatte, H. A hybrid of firefly and biogeography-based optimization algorithms for optimal design of steel frames. *Arab. J. Sci. Eng.* **2021**, *46*, 4703–4717. [CrossRef]
53. Li, W.; Wang, G.G. Elephant herding optimization using dynamic topology and biogeography-based optimization based on learning for numerical optimization. *Eng. Comput.* **2022**, *38* (Suppl. S2), 1585–1613. [CrossRef]

54. Zhang, X.; Wen, S.; Wang, D. Multi-population biogeography-based optimization algorithm and its application to image segmentation. *Appl. Soft Comput.* **2022**, *124*, 109005. [CrossRef]
55. Liang, S.; Fang, Z.; Sun, G.; Qu, G. Biogeography-based optimization with adaptive migration and adaptive mutation with its application in sidelobe reduction of antenna arrays. *Appl. Soft Comput.* **2022**, *121*, 108772. [CrossRef]
56. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
57. Jain, M.; Singh, V.; Rani, A. A novel nature-inspired algorithm for optimization: Squirrel search algorithm. S*Warm Evol. Comput.* **2019**, *44*, 148–175. [CrossRef]
58. Khishe, M.; Mosavi, M.R. Chimp optimization algorithm. *Expert Syst. Appl.* **2020**, *149*, 113338. [CrossRef]
59. Faramarzi, A.; Heidarinejad, M.; Mirjalili, S.; Gandomi, A.H. Marine Predators Algorithm: A nature-inspired metaheuristic. *Expert Syst. Appl.* **2020**, *152*, 113377. [CrossRef]
60. Chopra, N.; Ansari, M.M. Golden jackal optimization: A novel nature-inspired optimizer for engineering applications. *Expert Syst. Appl.* **2022**, *198*, 116924. [CrossRef]
61. Zhong, C.; Li, G.; Meng, Z. Beluga whale optimization: A novel nature-inspired metaheuristic algorithm. *Knowl.-Based Syst.* **2022**, *251*, 109215. [CrossRef]
62. 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. [CrossRef]
63. Abdel-Basset, M.; El-Shahat, D.; Jameel, M.; Abouhawwash, M. Exponential distribution optimizer (EDO): A novel math-inspired algorithm for global optimization and engineering problems. *Artif. Intell. Rev.* **2023**, *56*, 9329–9400. [CrossRef]
64. Cheng, M.Y.; Sholeh, M.N. Optical microscope algorithm: A new metaheuristic inspired by microscope magnification for solving engineering optimization problems. *Knowl.-Based Syst.* **2023**, *279*, 110939. [CrossRef]
65. Zhao, S.; Zhang, T.; Ma, S.; Wang, M. Sea-horse optimizer: A novel nature-inspired meta-heuristic for global optimization problems. *Appl. Intell.* **2023**, *53*, 11833–11860. [CrossRef]
66. Seyyedabbasi, A.; Kiani, F. Sand Cat swarm optimization: A nature-inspired algorithm to solve global optimization problems. *Eng. Comput.* **2023**, *39*, 2627–2651. [CrossRef]
67. Wang, L.; Cao, Q.; Zhang, Z.; Mirjalili, S.; Zhao, W. Artificial rabbits optimization: A new bio-inspired meta-heuristic algorithm for solving engineering optimization problems. *Eng. Appl. Artif. Intell.* **2022**, *114*, 105082. [CrossRef]
68. Trojovská, E.; Dehghani, M.; Trojovský, P. Zebra optimization algorithm: A new bio-inspired optimization algorithm for solving optimization algorithm. *IEEE Access* **2022**, *10*, 49445–49473. [CrossRef]
69. Hashim, F.A.; Houssein, E.H.; Hussain, K.; Mabrouk, M.S.; Al-Atabany, W. Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems. *Math. Comput. Simul.* **2022**, *192*, 84–110. [CrossRef]
70. Yapici, H.; Cetinkaya, N. A new meta-heuristic optimizer: Pathfinder algorithm. *Appl. Soft Comput.* **2019**, *78*, 545–568. [CrossRef]
71. Zhao, W.; Zhang, Z.; Wang, L. Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications. *Eng. Appl. Artif. Intell.* **2020**, *87*, 103300. [CrossRef]
72. Savsani, P.; Savsani, V. Passing vehicle search (PVS): A novel metaheuristic algorithm. *Appl. Math. Model.* **2016**, *40*, 3951–3978. [CrossRef]
73. Krenich, S.; Osyczka, A. Optimization of Robot Gripper Parameters Using Genetic Algorithms. In *Romansy 13*; Springer: Vienna, Austria, 2000. [CrossRef]

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.