algorithms

MDPI

Article

Improved Brain Storm Optimization Algorithm Based on Flock Decision Mutation Strategy

Yanchi Zhao [1], Jianhua Cheng [1,*] and Jing Cai [2]

[1] College of Intelligent Systems Science and Engineering, Harbin Engineering University, Harbin 150001, China; zhaoyanchi12@hrbeu.edu.cn
[2] Beijing Institute of Space Long March Vehicle, Beijing 100081, China; sunny_93cj@sina.com
* Correspondence: ins_cheng@163.com

Abstract: To tackle the problem of the brain storm optimization (BSO) algorithm's suboptimal capability for avoiding local optima, which contributes to its inadequate optimization precision, we developed a flock decision mutation approach that substantially enhances the efficacy of the BSO algorithm. Furthermore, to solve the problem of insufficient BSO algorithm population diversity, we introduced a strategy that utilizes the good point set to enhance the initial population's quality. Simultaneously, we substituted the K-means clustering approach with spectral clustering to improve the clustering accuracy of the algorithm. This work introduced an enhanced version of the brain storm optimization algorithm founded on a flock decision mutation strategy (FDIBSO). The improved algorithm was compared against contemporary leading algorithms through the CEC2018. The experimental section additionally employs the AUV intelligence evaluation as an application case. It addresses the combined weight model under various dimensional settings to substantiate the efficacy of the FDIBSO algorithm further. The findings indicate that FDIBSO surpasses BSO and other enhanced algorithms for addressing intricate optimization challenges.

Keywords: brain storm optimization; flock decision mutation; good point set; spectral clustering; combined weight

Citation: Zhao, C.; Cheng, J.; Cai, J. Improved Brain Storm Optimization Algorithm Based on Flock Decision Mutation Strategy. *Algorithms* **2024**, *17*, 172. https://doi.org/10.3390/a17050172

Academic Editor: Roberto Montemanni

Received: 27 March 2024
Revised: 18 April 2024
Accepted: 19 April 2024
Published: 23 April 2024

1. Introduction

Autonomous underwater vehicles are gradually evolving from program-driven modes to intelligent modes of self-decision, self-learning, and self-adaptation [1,2]. The evaluation of intelligence can effectively reduce testing costs and provide strong guidance for the development of intelligence levels [3]. However, current AUV intelligence evaluation technology urgently needs in-depth exploration and research. Among this technology, the combined weight method is a key technology that ensures the realization of evaluation. The combined weight model is evolving with increasingly complex traits including higher dimensions, nonlinear behavior, and lack of differentiability. This model's pronounced nonlinearity and absence of differentiability contribute to the presence of numerous local optima, resulting in a multi-modal phenomenon. When an optimization problem becomes complex, it significantly challenges the efficacy of optimization methods.

Inspired by the human brainstorming conference, significant scholarly interest has focused on the brain storm optimization algorithm due to its remarkable effectiveness at addressing multi-model problems, as cited in [4]. This algorithm is structured around a four-phase process, which includes clustering, substitution, selection, and mutation. The clustering mechanism intrinsic to the BSO algorithm substantially enhances the population's diversity by distributing solutions into several subgroups. The practicality of the BSO algorithm is well-documented across a variety of applications, including route optimization [5], visual data analysis [6], networked sensor systems, and additional domains [7].

While the clustering technique employed in conventional brain storm optimization algorithms aids with augmenting the diversity of the population, there are still notable

shortcomings inherent to the traditional BSO approach. Compared to other enhanced intelligent algorithms, the classic BSO exhibits a slower convergence rate and often falls short of identifying the optimal solution. A hybrid self-adaptive sine cosine algorithm with opposition-based learning (MSCA) has been proposed [8]. The opposite population is created by applying opposite numbers, influenced by the perturbation rate, to escape the local optimum within MSCA. Experimental data indicate that MSCA is extremely competitive. Yang proposed a multi-strategy whale optimization algorithm (MSWOA) [9]. This algorithm employs a chaotic logistic map to create an initial population with high quality and incorporates a Lévy flight mechanism to preserve diversity within the population throughout each iteration. The experimental data demonstrate that the MSWOA is exceptionally effective at tackling complex challenges. A new ensemble algorithm called e-mPSOBSA has been proposed [10]. The algorithm incorporates BSA's exploratory capacity to enhance global exploration and local exploitation and to maintain a suitable balance throughout the search process. Santana proposed a novel version of the binary artificial bee colony algorithm (NBABC) [11]. Experimental data demonstrate the new algorithm significantly enhances the optimization precision and maintains superiority compared to several recently developed algorithms. Ali introduced an enhancement to the QANA algorithm by developing an advanced version of the binary quantum-based avian navigation optimizer, termed IBQANA [12], which exhibits further advancements to the algorithm's performance.

In light of the superior performance exhibited by these highly competitive algorithms, it has become evident that the efficacy of conventional BSO algorithms requires additional enhancement. Consequently, researchers have focused on refining the fundamental parameters, clustering techniques, and mutation approaches of the traditional BSO algorithm. Chen presented a version of brain storm optimization that incorporates agglomerative hierarchical clustering analysis [13], which yields favorable outcomes and ensures fast convergence. The introduction of agglomerative hierarchical clustering into BSO has been implemented, followed by an analysis of its effect on the efficacy of the creation operator. Although there is a marginal improvement in optimization accuracy compared to the original BSO, this modified algorithm still does not adequately address the issue of the BSO algorithm's propensity for getting stuck at the local optimum. A brain storm optimization algorithm enhanced with a reinforcement learning mechanism has been introduced [14]. Four distinct mutation tactics were devised to bolster the algorithm's searching ability at various phases. The results show that this algorithm surpasses additional improved BSO algorithms with regard to efficiency. However, the algorithm's performance is less competitive than other improved swarm intelligence algorithms. Shen proposed a brain storm optimization algorithm with an adaptive learning strategy [15]. The BSO-AL, as proposed, creates new individuals through exploration, imitation, or adaptive learning. However, the algorithm tested fewer functions and lacked some reference values. An enhanced BSO algorithm that utilizes a difference-mutation technique and leverages the global-best individual has been introduced [16]. This approach substitutes the conventional mutation step in BSO with a difference step, markedly accelerating the convergence process. The algorithm adopts a global-best mutation strategy, which substantially enhances optimization performance. Nevertheless, the algorithm still struggles with the issue of local optimum entrapment, which hinders its effectiveness at tackling intricate multi-modal challenges. Tuba innovatively integrated the principles of chaos theory into the BSO algorithm by applying chaotic maps [17], resulting in an enhanced version of the BSO algorithm. This modified algorithm exhibits a marginal performance improvement compared to its predecessor, though the benefits are not particularly significant. Nevertheless, the incorporation of chaotic maps presents a novel approach to tackle the issue of algorithms that are prone to premature convergence. A chaotic local search-enhanced memetic brain storm optimization algorithm has been introduced [18]. This study presents a novel method that combines the BSO algorithm and chaotic local search, aiming to address the propensity of the BSO algorithm to become stuck at local maxima. Despite this integration, the algorithm shows

only marginal gains in optimization precision. An enhanced BSO algorithm incorporating an advanced discussion mechanism has been proposed [19]. It integrates a difference step approach while streamlining the BSO's selection methodology. This innovation aims to bolster global search capabilities during the initial phase and to refine local search activities in subsequent stages, thereby elevating the precision of the algorithm's optimization results. Additionally, the implementation of the difference step approach improved the convergence velocity of the algorithm. Despite these advancements, its performance for optimizing high-dimensional multi-modal problems does not meet anticipated benchmarks, and the algorithm remains prone to entrapment in local optimum. A proposed global-best brain storm optimization algorithm incorporates a discussion mechanism and a difference step [20]. This algorithm melds a suite of enhancement tactics, each with distinct characteristics, resulting in superior convergence rates and optimization precision relative to its predecessors. Nonetheless, the algorithm exhibits a propensity to become ensnared in local optima when dealing with intricate optimization challenges, indicating a necessity for additional refinements.

In conclusion, the existing improved brain storm optimization algorithms suffer from several deficiencies, including sluggish rates of convergence, suboptimal optimization accuracy, and a significant tendency to become trapped in a local optimum. Slow convergence hampers the overall efficiency of the algorithm; specifically, achieving a predetermined level of accuracy requires more time when the convergence rate is lower, which diminishes the algorithm's practical utility. Optimization accuracy is a critical indicator of an algorithm's efficacy, and a lack of precision indicates substandard performance. Moreover, there is a risk that the algorithm could get caught in a local optimum, resulting in considerable time lost during iterative processes and affecting the ultimate optimization accuracy. Hence, refining the BSO algorithm in this study aims to enhance the rate of convergence and the precision of optimization beyond the current improvements and to augment the algorithm's capacity to escape local optima in multi-model problems. Furthermore, an improved clustering technique is required to address the high computational demands and low clustering accuracy caused by the K-means clustering method in the conventional BSO algorithm. Ultimately, the objective to enhance the precision of weight calculation in AUV performance evaluation can be realized.

Overall, this work proposes the flock decision mutation strategy and introduces the good point set and spectral clustering. The paper's primary innovations include the algorithms' exceptional capability to escape local optima during complex optimization tasks involving multiple models and high dimensions coupled with their enhanced rate of convergence and greater precision in optimization. Subsequently, a refined brain storm optimization algorithm that incorporates the flock decision mutation strategy has been introduced. This work: (1) designs the flock decision mutation strategy to improve the optimization accuracy; (2) introduces the good point set designed to establish the initial population to enhance the diversity of the population at the beginning of the iteration process; (3) replaces K-means clustering with spectral clustering to improve the clustering accuracy of the algorithm; (4) extensive experimentation and data analysis are conducted utilizing a benchmark test suite from the CEC2018 [21]; (5) multiple simulations based on the combined weight model in AUV intelligence evaluation further confirm the efficacy of the suggested algorithm.

2. BSO

The BSO algorithm draws its inspiration from human brainstorming conferences, effectively harnessing human intelligence traits to address problems. It offers more benefits over conventional swarm intelligence algorithms, particularly addressing issues involving multiple dimensions. The algorithm is structured around four key phases: clustering, substitution, selection, and mutation.

Initially, the population of n candidate solutions undergoing iteration is segmented into m groups using the K-means clustering technique. This approach is intended to mimic

the collaborative dynamics of human group discussions, thereby enhancing the algorithm's search efficiency.

Subsequently, a parameter $p_{replace}$ is designated alongside the generation of a random number r_1 within the interval [0, 1]. Should r_1 fall below $p_{replace}$, a fresh individual is created to supplant the chosen cluster center. An excessively high value of $p_{replace}$ can impede the algorithm's convergence efficacy and diminish the population's diversity. Conversely, an unduly low value might precipitate premature convergence of the algorithm.

In the third step, three probability parameters p_{one}, p_{one_center}, and p_{two_center} are established, with the concurrent generation of random numbers r_2, r_3, and r_4. If r_2 falls below p_{one}, a mutation is performed on a single individual within one cluster. If not, individuals from two clusters are merged and then mutated. During the mutation of an individual within a cluster, if r_3 is lower than p_{one_center}, then the mutation is applied to the central individual of the cluster. If r_3 is above the threshold, a non-central individual is randomly picked from the same cluster for mutation. Similarly, when mutating individuals from two different clusters, if r_4 is less than p_{two_center}, the central individuals of both clusters are chosen for mutation. If r_4 exceeds p_{two_center}, random non-central individuals from each cluster are selected and merged before applying the mutation.

Fourth, the selected individuals undergo fusion or mutation processes, after which they are evaluated against the original individuals based on their fitness levels. Individuals who exhibit superior performance will be preserved following these operations. The process of fusion is described as follows:

$$X_f = v \times X_1 + (1 - v) \times X_2, \tag{1}$$

where X_f represents the individual post-fusion, v is a number randomly chosen from the range 0 to 1, and X_1 and X_2 are two random individuals selected for merging. The mutation process proceeds as follows:

$$X_m = X_s + \xi \times n(\mu, \sigma), \tag{2}$$

where X_m denotes the individual post-mutation, X_s identifies the chosen individual for mutation, $n(\mu, \sigma)$ represents a Gaussian random number, and ξ serves as the mutation coefficient. The formula for this coefficient is as follows:

$$\xi = \log sig\left(\frac{0.5 \times g_{max} - g}{k}\right) \times rand(), \tag{3}$$

where k is the adjustment factor, g_{max} denotes the upper limit of iterations, and g represents the number of the current iteration.

3. FDIBSO

This section introduces enhancements to the BSO algorithm in three key areas to boost its performance. The procedures of the FDIBSO algorithm are detailed in Algorithm 1.

3.1. Initialization Based on the Good Point Set

The BSO algorithm's starting population is randomly created, a method that is simpler to execute but results in a wider and more erratic spread of initial positions, impacting the algorithm's efficiency at converging. Therefore, implementing a strategy to even out the initial population's spread and to boost its diversity becomes essential.

The good point set, introduced by the Chinese mathematician Hua Luogeng, is a method to make the distribution of random solutions more uniform and to improve the solution's quality [22]. Therefore, many scholars have used it in the population initialization step of intelligent algorithms to improve algorithm performance [23,24]. This paper introduces the method into the BSO algorithm to improve its performance. Initialization based on the good point set is as follows, where n_i denotes the ith individual:

First, establish the population size n and set the dimensionality to D then, calculate $r = [r_1, r_2, \ldots, r_D]$, where r_i is calculated as follows:

$$r_i = mod(2cos(\frac{2\pi i}{7})n_i, 1) \tag{4}$$

Second, construct the good point set $P_D(i) = [r_1 i_1, r_2 i_2, \ldots, r_D i_D]$.

Third, configure the population using the designated good point set as outlined below.

$$X_i = a + P_D(i)(b - a), \tag{5}$$

where a and b denote the lower and upper bounds, respectivcly, of the individual distribution space. Figure 1 illustrates the comparative impact of using the good point set with a population size of 100, depicting the good point set on the left and the randomly generated population set on the right, thereby confirming the efficacy of the good point set.

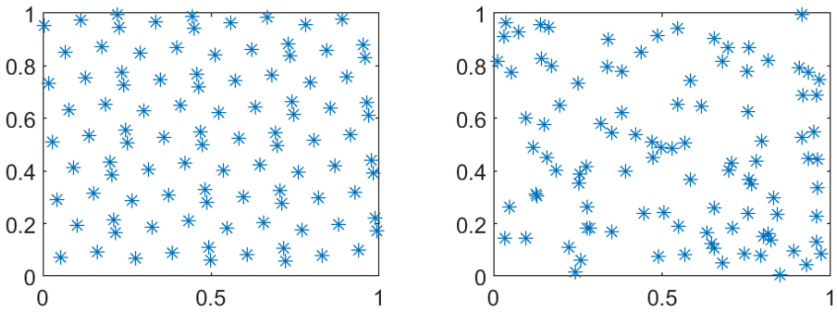


Figure 1. The comparison effect of the good point set.

3.2. Spectral Clustering Method

The BSO algorithm employs K-means clustering, which is valued for its straightforwardness and ease of implementation. However, a notable issue arises during the update of cluster centers, which can be substantially affected by outliers since the mean value calculation includes every individual. Furthermore, the K-means clustering method is prone to the problem of insufficient clustering accuracy, leading to a decrease in the algorithm's optimization accuracy when dealing with complex optimization problems such as multi-peak problems.

In recent years, spectral clustering has become one of the most popular modern clustering algorithms [25]. Spectral clustering techniques have surfaced as a structured simplification of the NP-hard normalized cut clustering issue and have been effectively utilized across various complex clustering contexts [26,27]. Compared to the K-means clustering approach, the spectral clustering method aligns better with various data distributions and enhances clustering outcomes. Consequently, this paper pioneers the integration of spectral clustering with the BSO algorithm, yielding enhanced optimization performance. The spectral clustering algorithm flow is as follows.

First, consider the population space as a network and then find the adjacency matrix W of the network graph. The degree matrix D is then computed from W.

Second, compute the Laplacian matrix L with the following expression (6):

$$L = D - W \tag{6}$$

Third, normalize the Laplacian matrix and then compute the first k eigenvectors of the normalized Laplacian matrix to form a new eigenmatrix P.

Fourth, K-means clustering is done on the eigenmatrix P to show the final clustering results. Figure 2 shows the comparative effectiveness of the two clustering methods for clustering when dealing with multi-peak functions, with the K-means clustering method on the left side and spectral clustering on the right side, thus confirming that the spectral

clustering method has higher clustering accuracy and achieves more accurate classification to enhance the local searching ability of each class at the later stage of the algorithm.

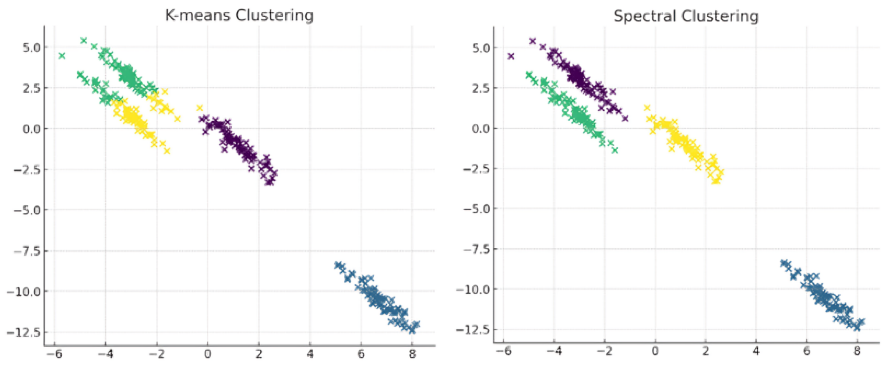


Figure 2. The comparison effect of the spectral clustering.

3.3. Flock Decision Mutation Strategy

During the initial stages of intelligent algorithm search iterations, it is crucial to enhance global search to increase population diversity and accelerate convergence. In the later stages, bolstering the local search becomes essential to refine optimization precision. To achieve the above objectives, methods such as the difference step strategy [28], global-best strategy [29,30], and elite mutation strategy [31,32] have been sequentially introduced to enhance the precision of the BSO algorithm's optimization.

However, the improved BSO algorithms using these strategies are still ineffective for addressing multi-model problems in multiple dimensions, which is mainly due to insufficient exploration of the search domain, i.e., the issue of insufficient population diversity, which causes the algorithms to quickly converge to a local optimum during the final phases of the iteration process, ultimately impacting the optimization precision of the algorithm. Therefore, this paper designed a flock decision mutation strategy, drawing on the concept of flock evolution, which sufficiently improves the population diversity during initial iterations and introduces a globally optimal individual to strengthen the local search towards the end of the iterations, significantly improving the performance of the algorithm. The basic principle of the flock decision mutation strategy is shown as follows:

First, the scope of an individual and the flock to which it belongs can be defined using Equation (7):

$$Dis_i(t) = \|X_i(t-1) - X_i(t)\|, \tag{7}$$

where $Dis_i(t)$ is the Euclidean distance between the current generation of individuals and the preceding cohort of individuals, $X_i(t-1)$ is the previous generation of individuals, and $X_i(t)$ is the current generation of individuals to be mutated.

Second, an individual is considered to belong to the flock of $X_i(t)$ if that individual satisfies the following conditions. Then, the flock of $X_i(t)$ is shown in Equation (8).

$$PF_i(t) = \left\{ X_j(t) \mid D_i\left(X_i(t), X_j(t)\right) \le Dis_i(t) \right\}, \tag{8}$$

where $PF_i(t)$ is the flock of $X_i(t)$, $X_j(t)$ is the individual from the whole population, and $D_i\left(X_i(t), X_j(t)\right)$ represents the Euclidean separation between $X_i(t)$ and $X_j(t)$.

Third, mutation on $X_i(t)$ is realized through the flock of individual $X_i(t)$, any member within the population, along with the globally optimal individual, as demonstrated in (9).

$$\begin{cases} X_{iF}(t) = X_i(t) + rand \times (X_{F1,i}(t) - X_1(t)) + rand \times (X_{F2,i}(t) - X_2(t)), t < 0.7g_{max} \\ X_{iF}(t) = X_i(t) + rand \times (X_{best} - X_1(t)), t \ge 0.7g_{max}, \end{cases} \tag{9}$$

where $X_{iF}(t)$ is the new individual after the flock decision mutation strategy, $X_1(t)$ and $X_2(t)$ are two randomly selected members from the entire population, $X_{F1,i}(t)$ and $X_{F2,i}(t)$

are two random individuals from the flock of individual $X_i(t)$, X_{best} represents the optimal member of the preceding population, and g_{max} denotes the upper limit of iterations.

The core idea of the flock decision mutation strategy is to make each individual in the population carry out a better mutation centered on itself during the initial phase of the algorithm's iteration, enhancing the caliber of the individuals and ensuring the population's diversity. During the advanced phases of the algorithm's iteration, its capacity to conduct local searches is enhanced by guiding the individuals to emulate the behavior of the the globally optimal individual. The pseudocode for this algorithm is presented in Algorithm 1 below.

Algorithm 1: The FDIBSO algorithm

Require: n, population size; m, total clusters; g_{max}, maximum iterations
for i=1 to n **do**
  initialize the population using the good point set and generate solution X_i
  assess the fitness of X_i
end for
while $g < g_{max}$ **do**
  partition n into m groups using spectral clustering techniques
  for i =1 to m **do**
    establish the solution possessing optimal fitness as the central point
  end for
  if $rand < p_{replace}$ **then**
    randomly create a member to substitute for the designated cluster center
  end if
  if $rand < p_{one}$ **then**
    select the individual from one cluster
    if $rand < p_{one\_center}$ **then**
      select the cluster center to mutate
    else
      arbitrarily choose a member from this cluster for mutation
    end if
  else
    choose the individual from two clusters
    if $rand < p_{two\_center}$ **then**
      the pair of cluster centers is merged and subsequently altered
    else
      two members from every chosen cluster
      are arbitrarily chosen for merging and subsequent mutation
    end if
  end if
  if $g < g_{max} * 0.7$ **then**
    $X_{iF}(g) = X_i(g) + rand \times (X_{F1,i}(g) - X_1(g)) + rand \times (X_{F2,i}(g) - X_2(g))$
  else
    $X_{iF}(g) = X_i(g) + rand \times (X_{best} - X_1(g)$
  end if
  evaluate the fitness of X_{iF} by friend decision mutation
  retain excellent individual
  $g = g + 1$
end while

4. Results

Trials were conducted under 30- and 50-dimensional settings using the CEC2018. Comparative simulation experiments were conducted on nine algorithms: BSO [4], ADMBSO [19], GDBSO [16], DDGBSO [20], MSWOA [9], HOA [33], WMFO [34], MFO-SFR [35], and

FDIBSO. Performance analyses of each algorithm were carried out, with each simulation set being independently run 30 times. In addition, based on the combined weight model in the AUV intelligence evaluation, we compared various algorithms and verified the advantages of the FDIBSO algorithm in engineering applications. The simulations were executed on the MATLAB 2018A platform.

4.1. Parameter Settings

The fundamental parameters for the BSO algorithm along with its enhanced version discussed in this document are cited from [4]. Settings for these parameters are specified as follows: population size $n = 100$, cluster count $m = 5$, adjustment factor $k = 20$, evaluation tally $F_{max} = 5 * 10^4$, probability parameter $P_{replace} = 0.1$, $P_{one} = 0.5$, $P_{one\_center} = 0.3$, and $P_{two\_center} = 0.2$. In addition, the FDIBSO algorithm presented in this paper is compared with four intelligent algorithms of different types, which are set up according to their parameters in their respective literature.

4.2. Simulation Results and Analysis

Tables 1–4 display the optimization performance of each algorithm when the dimension D is set to 30 and 50. For every benchmark function, 30 trials are conducted, yielding two statistical measures: the mean and the best value. The optimal average for each function is emphasized in boldface.

By examining the optimization accuracy of the CEC2018, various insights emerge. First, in assessing 30-dimensional challenges against other enhanced BSO variants, the FDIBSO algorithm significantly outperforms the basic BSO algorithm. Furthermore, the performance improvements are still notably substantial compared to the modestly enhanced versions. For the CEC2018 benchmark test suite, which comprises 29 functions, FDIBSO achieves optimization with greater accuracy on 16 of those functions.

Second, when juxtaposed with other enhanced BSO variants, the efficacy of the FDIBSO algorithm exhibits a decline in specific benchmark functions tailored for 50-dimensional issues. Table 2 illustrates that the FDIBSO algorithm outperforms in optimization accuracy for just 14 functions, with a slight decline in its lead.

Third, in the context of 30-dimensional challenges relative to other competitive intelligent algorithms, the performance of the FDIBSO algorithm is still superior. The FDIBSO and MFO-SFR algorithms exhibit comparable performance, optimizing 12 functions with greater accuracy. Their overall effectiveness significantly surpasses that of the three other algorithms compared, underscoring the principle that no single intelligent algorithm can perfectly solve every optimization challenge.

Fourth, under the 50-dimensional scenario, FDIBSO's performance diminishes relative to other intelligent algorithms, yet it still secures the lead in optimizing ten functions. Its overall performance is only outmatched by the MFO-SFR algorithm, but it remains superior to the three other algorithms it was compared with. In a word, these experimental outcomes collectively affirm the superior optimization performance of the FDIBSO algorithm.

To more effectively show the FDIBSO algorithm's ability in convergence speed, Figures 3 and 4 depict the convergence trajectories for various enhanced BSO algorithms across four benchmark functions from the CEC2018.

The blue curve in the figure represents the FDIBSO algorithm. Although the convergence speed of this algorithm is significantly improved compared to the traditional BSO algorithm, it is not much improved compared to the BSO variant algorithm. This problem is because the FDIBSO algorithm employs the good point set and the flock decision mutation strategy, significantly improving population diversity. As a result, in the early stages, the search is more comprehensive, and convergence is relatively slow. However, with iterations, the FDIBSO algorithm can escape local optima and achieve higher optimization accuracy.

Table 1. Comparison of BSO variants for 30-dimensional challenges.

BFs		BSO	ADMBSO	GDBSO	DDGBSO	FDIBSO
C1	Mean	1.18×10^{8}	3.10×10^{3}	1.41×10^{3}	1.74×10^{3}	$\mathbf{1.36 \times 10^{3}}$
	Min	1.31×10^{2}	1.68×10^{2}	1.31×10^{2}	2.31×10^{2}	1.30×10^{2}
C3	Mean	8.92×10^{4}	3.94×10^{4}	7.98×10^{4}	9.85×10^{3}	$\mathbf{1.70 \times 10^{3}}$
	Min	5.63×10^{4}	3.04×10^{4}	5.68×10^{4}	6.94×10^{3}	7.02×10^{2}
C4	Mean	5.88×10^{2}	4.98×10^{2}	4.90×10^{2}	5.02×10^{2}	$\mathbf{4.88 \times 10^{2}}$
	Min	4.82×10^{2}	4.85×10^{2}	4.65×10^{2}	4.10×10^{2}	4.69×10^{2}
C5	Mean	6.82×10^{2}	5.60×10^{2}	7.03×10^{2}	$\mathbf{5.19 \times 10^{2}}$	6.98×10^{2}
	Min	6.42×10^{2}	5.39×10^{2}	6.77×10^{2}	5.07×10^{2}	6.54×10^{2}
C6	Mean	6.53×10^{2}	$\mathbf{6.00 \times 10^{2}}$	6.01×10^{2}	6.01×10^{2}	6.46×10^{2}
	Min	6.41×10^{2}	6.00×10^{2}	6.00×10^{2}	6.01×10^{2}	6.34×10^{2}
C7	Mean	1.12×10^{3}	7.95×10^{2}	9.45×10^{2}	$\mathbf{7.29 \times 10^{2}}$	1.01×10^{3}
	Min	1.03×10^{3}	7.74×10^{2}	9.23×10^{2}	7.15×10^{2}	9.19×10^{2}
C8	Mean	9.34×10^{2}	$\mathbf{8.62 \times 10^{2}}$	1.01×10^{3}	1.07×10^{3}	9.36×10^{2}
	Min	8.86×10^{2}	8.29×10^{2}	9.89×10^{2}	1.04×10^{3}	9.09×10^{2}
C9	Mean	3.73×10^{3}	$\mathbf{9.00 \times 10^{2}}$	9.29×10^{2}	2.35×10^{3}	3.21×10^{3}
	Min	2.77×10^{3}	9.00×10^{2}	9.00×10^{2}	2.20×10^{3}	2.39×10^{3}
C10	Mean	$\mathbf{4.91 \times 10^{3}}$	5.09×10^{3}	8.59×10^{3}	8.34×10^{3}	5.54×10^{3}
	Min	3.86×10^{3}	4.20×10^{3}	2.52×10^{2}	7.60×10^{3}	4.83×10^{3}
C11	Mean	1.83×10^{3}	$\mathbf{1.17 \times 10^{3}}$	1.24×10^{3}	1.92×10^{3}	1.26×10^{3}
	Min	1.23×10^{3}	1.12×10^{3}	1.20×10^{3}	1.18×10^{3}	1.13×10^{3}
C12	Mean	1.92×10^{7}	9.14×10^{4}	8.91×10^{4}	5.96×10^{4}	$\mathbf{2.75 \times 10^{4}}$
	Min	4.72×10^{6}	2.24×10^{4}	2.58×10^{4}	2.30×10^{4}	1.12×10^{4}
C13	Mean	3.79×10^{4}	1.53×10^{4}	1.17×10^{4}	1.07×10^{4}	$\mathbf{3.72 \times 10^{3}}$
	Min	2.57×10^{4}	4.26×10^{3}	2.67×10^{3}	3.27×10^{3}	2.53×10^{3}
C14	Mean	1.81×10^{5}	1.34×10^{4}	1.41×10^{4}	1.52×10^{4}	$\mathbf{1.46 \times 10^{3}}$
	Min	2.15×10^{3}	2.07×10^{3}	6.00×10^{3}	1.63×10^{3}	1.45×10^{3}
C15	Mean	3.17×10^{4}	5.10×10^{3}	5.84×10^{3}	2.97×10^{3}	$\mathbf{1.17 \times 10^{3}}$
	Min	1.83×10^{4}	2.56×10^{3}	1.57×10^{3}	1.93×10^{3}	1.61×10^{3}
C16	Mean	3.18×10^{3}	2.76×10^{3}	3.25×10^{3}	4.14×10^{3}	$\mathbf{2.61 \times 10^{3}}$
	Min	2.84×10^{3}	2.19×10^{3}	2.95×10^{3}	3.56×10^{3}	2.09×10^{3}
C17	Mean	2.36×10^{3}	$\mathbf{2.05 \times 10^{3}}$	2.26×10^{3}	2.81×10^{3}	2.25×10^{3}
	Min	1.91×10^{3}	1.81×10^{3}	1.82×10^{3}	2.54×10^{3}	1.82×10^{3}
C18	Mean	4.01×10^{5}	2.10×10^{5}	1.33×10^{6}	1.53×10^{5}	$\mathbf{1.95 \times 10^{3}}$
	Min	9.90×10^{4}	7.40×10^{4}	4.19×10^{5}	2.26×10^{4}	1.87×10^{3}
C19	Mean	6.96×10^{5}	9.53×10^{3}	8.77×10^{3}	3.38×10^{3}	$\mathbf{1.98 \times 10^{3}}$
	Min	5.42×10^{4}	2.48×10^{3}	2.03×10^{3}	2.51×10^{3}	1.95×10^{3}
C20	Mean	2.70×10^{3}	2.43×10^{3}	2.63×10^{3}	$\mathbf{2.37 \times 10^{3}}$	2.48×10^{3}
	Min	2.42×10^{3}	2.26×10^{3}	2.35×10^{3}	2.20×10^{3}	2.36×10^{3}
C21	Mean	2.51×10^{3}	$\mathbf{2.36 \times 10^{3}}$	2.50×10^{3}	2.62×10^{3}	2.47×10^{3}
	Min	2.45×10^{3}	2.34×10^{3}	2.46×10^{3}	2.56×10^{3}	2.41×10^{3}
C22	Mean	6.72×10^{3}	2.86×10^{3}	3.06×10^{3}	3.41×10^{3}	$\mathbf{2.56 \times 10^{3}}$
	Min	4.02×10^{3}	2.30×10^{3}	2.30×10^{3}	2.53×10^{3}	2.30×10^{3}
C23	Mean	3.36×10^{3}	2.72×10^{3}	2.86×10^{3}	3.38×10^{3}	$\mathbf{2.59 \times 10^{3}}$
	Min	3.04×10^{3}	2.69×10^{3}	2.81×10^{3}	3.10×10^{3}	2.49×10^{3}
C24	Mean	3.54×10^{3}	2.89×10^{3}	3.02×10^{3}	2.81×10^{3}	$\mathbf{2.68 \times 10^{3}}$
	Min	3.41×10^{3}	2.87×10^{3}	2.98×10^{3}	2.79×10^{3}	2.62×10^{3}
C25	Mean	2.96×10^{3}	$\mathbf{2.89 \times 10^{3}}$	$\mathbf{2.89 \times 10^{3}}$	2.89×10^{3}	$\mathbf{2.89 \times 10^{3}}$
	Min	2.90×10^{3}	2.89×10^{3}	2.88×10^{3}	2.89×10^{3}	2.88×10^{3}
C26	Mean	8.25×10^{3}	4.50×10^{3}	5.54×10^{3}	9.81×10^{3}	$\mathbf{4.49 \times 10^{3}}$
	Min	6.78×10^{3}	4.19×10^{3}	4.73×10^{3}	9.11×10^{3}	3.77×10^{3}
C27	Mean	3.95×10^{3}	3.21×10^{3}	3.22×10^{3}	$\mathbf{3.16 \times 10^{3}}$	3.28×10^{3}
	Min	3.69×10^{3}	3.20×10^{3}	3.20×10^{3}	3.14×10^{3}	3.23×10^{3}
C28	Mean	3.45×10^{3}	3.20×10^{3}	3.21×10^{3}	3.28×10^{3}	$\mathbf{3.20 \times 10^{3}}$
	Min	3.35×10^{3}	3.14×10^{3}	3.12×10^{3}	3.18×10^{3}	3.12×10^{3}
C29	Mean	4.45×10^{3}	4.12×10^{3}	$\mathbf{4.02 \times 10^{3}}$	5.68×10^{3}	4.04×10^{3}
	Min	4.07×10^{3}	3.80×10^{3}	3.47×10^{3}	4.81×10^{3}	3.68×10^{3}
C30	Mean	3.32×10^{6}	$\mathbf{1.25 \times 10^{4}}$	1.26×10^{4}	3.39×10^{4}	1.40×10^{4}
	Min	4.54×10^{5}	5.95×10^{3}	6.39×10^{3}	3.51×10^{3}	5.96×10^{3}

Table 2. Comparison of BSO variants for 50-dimensional challenges.

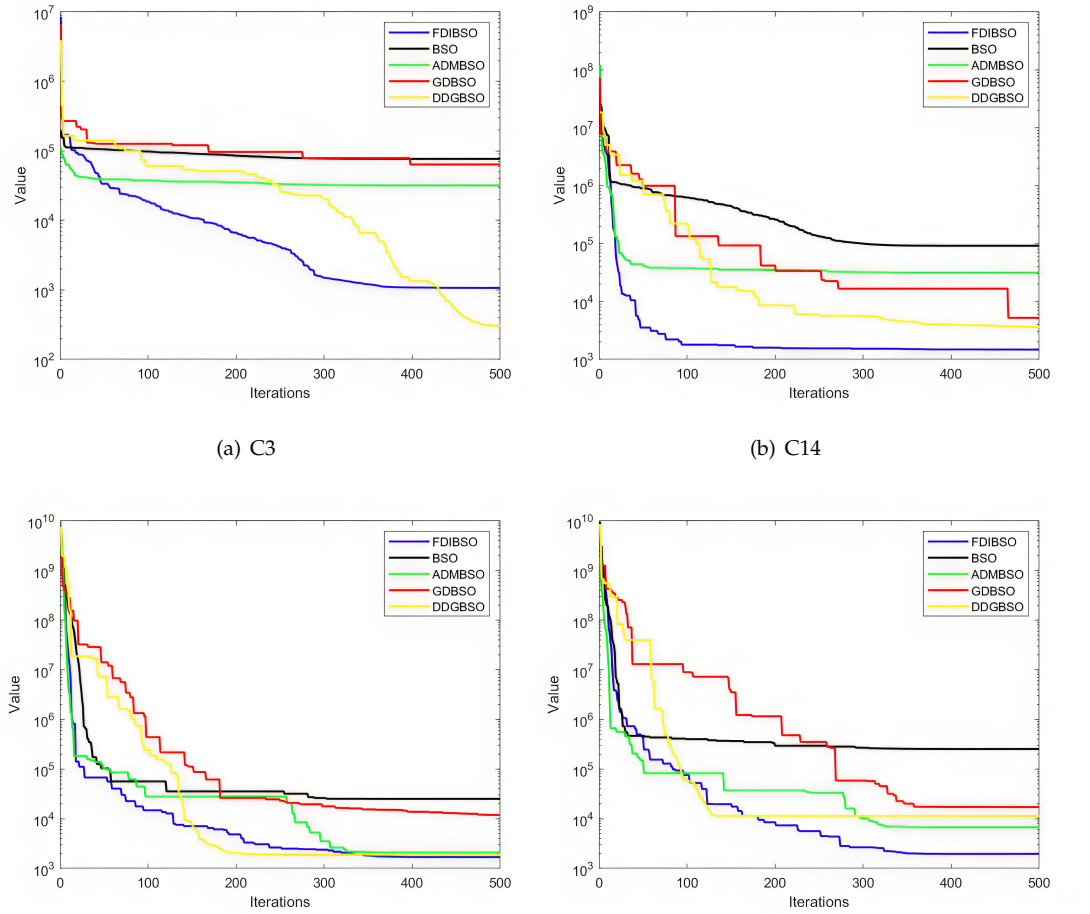
BFs		BSO	ADMBSO	GDBSO	DDGBSO	FDIBSO
C1	Mean	7.48×10^{9}	3.22×10^{3}	8.49×10^{3}	2.75×10^{3}	$\mathbf{1.06 \times 10^{3}}$
	Min	4.83×10^{9}	3.78×10^{2}	2.65×10^{3}	2.68×10^{3}	2.83×10^{2}
C3	Mean	1.97×10^{5}	1.23×10^{5}	2.16×10^{5}	1.58×10^{5}	$\mathbf{3.73 \times 10^{4}}$
	Min	1.58×10^{5}	9.89×10^{4}	1.73×10^{5}	1.02×10^{5}	1.74×10^{4}
C4	Mean	1.75×10^{3}	5.26×10^{2}	5.31×10^{2}	$\mathbf{2.45 \times 10^{2}}$	6.18×10^{2}
	Min	1.13×10^{3}	4.33×10^{2}	4.63×10^{2}	1.71×10^{2}	5.61×10^{2}
C5	Mean	8.16×10^{2}	$\mathbf{6.09 \times 10^{2}}$	9.33×10^{2}	1.10×10^{3}	8.41×10^{2}
	Min	7.78×10^{2}	5.83×10^{2}	9.16×10^{2}	1.06×10^{3}	8.22×10^{2}
C6	Mean	6.60×10^{2}	$\mathbf{6.01 \times 10^{2}}$	6.05×10^{2}	6.83×10^{2}	6.55×10^{2}
	Min	6.55×10^{2}	6.01×10^{2}	6.03×10^{2}	6.73×10^{2}	6.51×10^{2}
C7	Mean	1.68×10^{3}	$\mathbf{9.08 \times 10^{2}}$	1.23×10^{3}	1.95×10^{3}	1.54×10^{3}
	Min	1.53×10^{3}	8.85×10^{2}	1.19×10^{3}	1.84×10^{3}	1.33×10^{3}
C8	Mean	1.12×10^{3}	$\mathbf{9.27 \times 10^{2}}$	1.23×10^{3}	1.43×10^{3}	1.10×10^{3}
	Min	1.04×10^{3}	8.91×10^{2}	1.19×10^{3}	1.37×10^{3}	1.07×10^{3}
C9	Mean	1.15×10^{4}	$\mathbf{1.18 \times 10^{3}}$	1.99×10^{3}	3.00×10^{4}	9.31×10^{3}
	Min	8.24×10^{3}	9.13×10^{2}	1.10×10^{3}	2.31×10^{4}	6.43×10^{3}
C10	Mean	8.28×10^{3}	$\mathbf{7.94 \times 10^{3}}$	1.48×10^{4}	1.43×10^{4}	8.23×10^{3}
	Min	7.01×10^{3}	6.48×10^{3}	1.37×10^{4}	1.26×10^{4}	6.83×10^{3}
C11	Mean	4.98×10^{3}	$\mathbf{1.32 \times 10^{3}}$	1.49×10^{3}	2.29×10^{4}	1.44×10^{3}
	Min	2.40×10^{3}	1.21×10^{3}	1.42×10^{3}	1.23×10^{4}	1.29×10^{3}
C12	Mean	3.51×10^{8}	2.19×10^{6}	$\mathbf{1.62 \times 10^{6}}$	4.74×10^{6}	2.28×10^{5}
	Min	2.01×10^{7}	2.57×10^{5}	4.93×10^{5}	2.55×10^{5}	1.03×10^{5}
C13	Mean	7.28×10^{4}	$\mathbf{5.69 \times 10^{3}}$	6.07×10^{3}	1.79×10^{4}	9.74×10^{4}
	Min	2.67×10^{4}	2.23×10^{3}	1.73×10^{3}	7.89×10^{3}	2.47×10^{4}
C14	Mean	7.82×10^{5}	1.39×10^{5}	2.15×10^{5}	2.12×10^{7}	$\mathbf{1.62 \times 10^{3}}$
	Min	3.69×10^{4}	4.34×10^{4}	4.42×10^{4}	5.88×10^{6}	1.56×10^{3}
C15	Mean	2.57×10^{4}	7.29×10^{3}	9.65×10^{3}	4.03×10^{9}	$\mathbf{2.46 \times 10^{3}}$
	Min	1.80×10^{4}	1.73×10^{3}	2.00×10^{3}	1.59×10^{9}	1.73×10^{3}
C16	Mean	3.86×10^{3}	3.63×10^{3}	4.96×10^{3}	6.87×10^{3}	$\mathbf{3.58 \times 10^{3}}$
	Min	3.24×10^{3}	3.32×10^{3}	4.38×10^{3}	5.93×10^{3}	3.04×10^{3}
C17	Mean	3.47×10^{3}	$\mathbf{3.13 \times 10^{3}}$	3.81×10^{3}	4.82×10^{3}	3.21×10^{3}
	Min	2.74×10^{3}	2.72×10^{3}	3.43×10^{3}	3.65×10^{3}	2.92×10^{3}
C18	Mean	2.36×10^{6}	1.70×10^{6}	3.35×10^{6}	4.50×10^{7}	$\mathbf{7.29 \times 10^{3}}$
	Min	8.73×10^{5}	3.75×10^{5}	1.36×10^{6}	1.97×10^{7}	2.51×10^{3}
C19	Mean	1.60×10^{6}	1.47×10^{4}	1.32×10^{4}	1.23×10^{9}	$\mathbf{2.63 \times 10^{3}}$
	Min	9.59×10^{4}	2.13×10^{3}	2.18×10^{3}	3.74×10^{8}	2.17×10^{3}
C20	Mean	3.47×10^{3}	3.31×10^{3}	4.02×10^{3}	3.82×10^{3}	$\mathbf{3.22 \times 10^{3}}$
	Min	2.90×10^{3}	2.89×10^{3}	3.66×10^{3}	3.32×10^{3}	2.69×10^{3}
C21	Mean	2.73×10^{3}	2.43×10^{3}	2.72×10^{3}	3.01×10^{3}	$\mathbf{2.43 \times 10^{3}}$
	Min	2.61×10^{3}	2.38×10^{3}	2.70×10^{3}	2.91×10^{3}	2.32×10^{3}
C22	Mean	1.05×10^{4}	9.55×10^{3}	1.65×10^{4}	1.61×10^{4}	$\mathbf{1.02 \times 10^{4}}$
	Min	9.52×10^{3}	7.86×10^{3}	1.58×10^{4}	1.46×10^{4}	9.43×10^{3}
C23	Mean	4.04×10^{3}	$\mathbf{2.88 \times 10^{3}}$	3.17×10^{3}	4.19×10^{3}	2.47×10^{3}
	Min	3.74×10^{3}	2.82×10^{3}	3.11×10^{3}	3.89×10^{3}	2.44×10^{3}
C24	Mean	4.27×10^{3}	3.05×10^{3}	3.33×10^{3}	4.61×10^{3}	$\mathbf{2.91 \times 10^{3}}$
	Min	4.05×10^{3}	2.99×10^{3}	3.30×10^{3}	4.27×10^{3}	2.85×10^{3}
C25	Mean	3.94×10^{3}	3.03×10^{3}	3.03×10^{3}	1.30×10^{4}	$\mathbf{3.03 \times 10^{3}}$
	Min	3.43×10^{3}	2.96×10^{3}	2.99×10^{3}	8.02×10^{3}	2.83×10^{3}
C26	Mean	1.35×10^{4}	5.50×10^{3}	7.88×10^{3}	1.67×10^{4}	$\mathbf{5.08 \times 10^{3}}$
	Min	1.23×10^{4}	4.85×10^{3}	7.28×10^{3}	1.50×10^{4}	4.24×10^{3}
C27	Mean	5.98×10^{3}	$\mathbf{3.35 \times 10^{3}}$	3.37×10^{3}	6.37×10^{3}	3.81×10^{3}
	Min	5.29×10^{3}	3.22×10^{3}	3.23×10^{3}	5.58×10^{3}	3.45×10^{3}
C28	Mean	5.01×10^{3}	3.29×10^{3}	3.31×10^{3}	$\mathbf{3.08 \times 10^{3}}$	3.30×10^{3}
	Min	4.26×10^{3}	3.26×10^{3}	3.27×10^{3}	3.22×10^{3}	3.22×10^{3}
C29	Mean	6.64×10^{3}	4.84×10^{3}	5.18×10^{3}	1.73×10^{4}	$\mathbf{4.05 \times 10^{3}}$
	Min	6.03×10^{3}	4.01×10^{3}	4.56×10^{3}	1.18×10^{4}	3.84×10^{3}
C30	Mean	1.26×10^{8}	1.12×10^{6}	$\mathbf{9.56 \times 10^{5}}$	2.22×10^{9}	7.54×10^{6}
	Min	8.35×10^{7}	6.79×10^{5}	6.91×10^{5}	8.61×10^{8}	4.94×10^{6}

Table 3. Comparison of FDIBSO with latest competitive algorithms on 30-D problems.

BFs		MSWOA	HOA	WMFO	MFO-SFR	FDIBSO
C1	Mean	9.50×10^{3}	3.03×10^{9}	3.82×10^{3}	1.79×10^{3}	$\mathbf{1.36 \times 10^{3}}$
	Min	3.90×10^{3}	2.20×10^{9}	2.10×10^{4}	1.01×10^{2}	1.02×10^{2}
C3	Mean	9.10×10^{3}	3.08×10^{4}	$\mathbf{3.91 \times 10^{2}}$	1.31×10^{4}	1.70×10^{3}
	Min	6.65×10^{3}	2.03×10^{4}	3.01×10^{2}	7.51×10^{3}	7.02×10^{3}
C4	Mean	4.84×10^{2}	1.05×10^{3}	$\mathbf{4.81 \times 10^{2}}$	4.91×10^{2}	4.88×10^{2}
	Min	4.81×10^{2}	7.60×10^{2}	4.25×10^{2}	4.70×10^{2}	4.69×10^{2}
C5	Mean	6.08×10^{2}	7.94×10^{2}	6.74×10^{2}	$\mathbf{5.23 \times 10^{2}}$	6.98×10^{2}
	Min	5.93×10^{2}	7.61×10^{2}	6.23×10^{2}	5.11×10^{2}	6.54×10^{2}
C6	Mean	6.01×10^{2}	6.61×10^{2}	6.37×10^{2}	$\mathbf{6.00 \times 10^{2}}$	6.46×10^{2}
	Min	6.01×10^{2}	6.50×10^{2}	6.14×10^{2}	6.00×10^{2}	6.34×10^{2}
C7	Mean	8.08×10^{2}	$\mathbf{1.03 \times 10^{2}}$	1.06×10^{3}	7.67×10^{2}	1.01×10^{3}
	Min	7.82×10^{2}	9.98×10^{2}	6.25×10^{2}	7.46×10^{2}	9.19×10^{2}
C8	Mean	1.10×10^{3}	1.06×10^{3}	9.54×10^{2}	$\mathbf{8.21 \times 10^{2}}$	9.36×10^{2}
	Min	9.59×10^{2}	1.04×10^{3}	8.55×10^{2}	8.09×10^{2}	9.09×10^{2}
C9	Mean	9.93×10^{2}	4.29×10^{3}	4.54×10^{3}	$\mathbf{9.04 \times 10^{2}}$	3.21×10^{3}
	Min	8.38×10^{2}	2.67×10^{3}	1.68×10^{3}	9.01×10^{2}	2.39×10^{3}
C10	Mean	7.76×10^{3}	8.34×10^{3}	5.19×10^{3}	$\mathbf{4.06 \times 10^{3}}$	5.44×10^{3}
	Min	5.03×10^{3}	7.55×10^{3}	3.76×10^{3}	2.46×10^{3}	4.83×10^{3}
C11	Mean	$\mathbf{1.08 \times 10^{3}}$	1.80×10^{3}	1.25×10^{3}	1.14×10^{3}	1.26×10^{3}
	Min	1.05×10^{3}	1.70×10^{3}	1.17×10^{3}	1.11×10^{3}	1.13×10^{3}
C12	Mean	5.15×10^{4}	3.76×10^{8}	1.01×10^{5}	1.51×10^{5}	$\mathbf{2.75 \times 10^{4}}$
	Min	3.32×10^{4}	2.82×10^{8}	6.93×10^{3}	2.04×10^{4}	1.12×10^{4}
C13	Mean	1.98×10^{4}	1.05×10^{8}	6.66×10^{3}	6.41×10^{3}	$\mathbf{3.72 \times 10^{3}}$
	Min	1.52×10^{4}	3.30×10^{7}	1.40×10^{3}	1.69×10^{3}	2.53×10^{3}
C14	Mean	2.39×10^{4}	1.34×10^{5}	1.41×10^{4}	8.20×10^{3}	$\mathbf{1.46 \times 10^{3}}$
	Min	9.40×10^{3}	5.22×10^{4}	3.03×10^{3}	2.02×10^{3}	1.45×10^{3}
C15	Mean	5.18×10^{3}	3.14×10^{7}	1.16×10^{4}	5.61×10^{3}	$\mathbf{1.71 \times 10^{3}}$
	Min	4.54×10^{3}	8.14×10^{6}	1.61×10^{3}	1.52×10^{3}	1.61×10^{3}
C16	Mean	3.78×10^{3}	3.79×10^{3}	2.62×10^{3}	$\mathbf{1.86 \times 10^{3}}$	2.61×10^{3}
	Min	2.82×10^{3}	3.42×10^{3}	2.07×10^{3}	1.62×10^{3}	2.09×10^{3}
C17	Mean	2.89×10^{3}	2.36×10^{3}	2.73×10^{3}	1.75×10^{3}	$\mathbf{2.25 \times 10^{3}}$
	Min	2.42×10^{3}	2.10×10^{3}	1.96×10^{3}	1.73×10^{3}	1.82×10^{3}
C18	Mean	1.66×10^{5}	1.21×10^{6}	8.19×10^{4}	1.49×10^{5}	$\mathbf{1.95 \times 10^{3}}$
	Min	1.52×10^{5}	4.28×10^{5}	6.88×10^{3}	4.31×10^{4}	1.87×10^{3}
C19	Mean	4.92×10^{3}	4.43×10^{7}	1.56×10^{4}	6.53×10^{3}	$\mathbf{1.98 \times 10^{3}}$
	Min	3.86×10^{3}	1.77×10^{7}	2.31×10^{3}	1.91×10^{3}	1.95×10^{3}
C20	Mean	2.92×10^{3}	2.68×10^{3}	2.69×10^{3}	$\mathbf{2.09 \times 10^{3}}$	2.48×10^{3}
	Min	2.47×10^{3}	2.49×10^{3}	2.29×10^{3}	2.00×10^{3}	2.36×10^{3}
C21	Mean	2.62×10^{3}	2.58×10^{3}	2.46×10^{3}	$\mathbf{2.32 \times 10^{3}}$	2.47×10^{3}
	Min	2.34×10^{3}	2.54×10^{3}	2.39×10^{3}	2.31×10^{3}	2.41×10^{3}
C22	Mean	2.30×10^{3}	4.15×10^{3}	5.29×10^{3}	$\mathbf{2.30 \times 10^{3}}$	2.56×10^{3}
	Min	2.30×10^{3}	2.71×10^{3}	2.30×10^{3}	2.30×10^{3}	2.30×10^{3}
C23	Mean	2.62×10^{3}	3.13×10^{3}	2.86×10^{3}	2.67×10^{3}	$\mathbf{2.59 \times 10^{3}}$
	Min	2.36×10^{3}	3.06×10^{3}	2.76×10^{3}	2.65×10^{3}	2.49×10^{3}
C24	Mean	2.89×10^{3}	3.19×10^{3}	3.00×10^{3}	2.84×10^{3}	$\mathbf{2.68 \times 10^{3}}$
	Min	2.65×10^{3}	3.12×10^{3}	2.91×10^{3}	2.83×10^{3}	2.62×10^{3}
C25	Mean	2.89×10^{3}	3.14×10^{3}	2.90×10^{3}	2.89×10^{3}	$\mathbf{2.89 \times 10^{3}}$
	Min	2.88×10^{3}	3.07×10^{3}	2.88×10^{3}	2.89×10^{3}	2.88×10^{3}
C26	Mean	4.23×10^{3}	4.74×10^{3}	5.84×10^{3}	$\mathbf{3.90 \times 10^{3}}$	4.49×10^{3}
	Min	3.77×10^{3}	3.74×10^{3}	4.74×10^{3}	3.74×10^{3}	3.77×10^{3}
C27	Mean	$\mathbf{3.20 \times 10^{3}}$	3.72×10^{3}	3.28×10^{3}	3.22×10^{3}	3.28×10^{3}
	Min	3.15×10^{3}	3.62×10^{3}	3.22×10^{3}	3.21×10^{3}	3.23×10^{3}
C28	Mean	3.22×10^{3}	3.52×10^{3}	3.20×10^{3}	3.22×10^{3}	$\mathbf{3.20 \times 10^{3}}$
	Min	3.05×10^{3}	3.47×10^{3}	3.12×10^{3}	3.20×10^{3}	3.12×10^{3}
C29	Mean	3.50×10^{3}	4.73×10^{3}	4.07×10^{3}	$\mathbf{3.41 \times 10^{3}}$	4.04×10^{3}
	Min	3.35×10^{3}	4.45×10^{3}	3.55×10^{3}	3.32×10^{3}	3.68×10^{3}
C30	Mean	1.62×10^{4}	2.61×10^{7}	1.08×10^{4}	$\mathbf{7.84 \times 10^{3}}$	1.40×10^{4}
	Min	1.07×10^{3}	1.22×10^{7}	5.67×10^{3}	6.36×10^{3}	5.96×10^{3}

Table 4. Comparison of FDIBSO with latest competitive algorithms on 50-D problems.

BFs		MSWOA	HOA	WMFO	MFO-SFR	FDIBSO
C1	Mean	1.46×10^{8}	1.15×10^{10}	4.26×10^{3}	3.46×10^{4}	$\mathbf{1.06 \times 10^{3}}$
	Min	6.34×10^{7}	8.35×10^{9}	1.05×10^{2}	9.39×10^{3}	2.83×10^{2}
C3	Mean	2.03×10^{5}	8.23×10^{4}	$\mathbf{9.95 \times 10^{2}}$	5.50×10^{4}	3.37×10^{4}
	Min	9.34×10^{4}	6.99×10^{4}	3.22×10^{2}	4.22×10^{4}	1.74×10^{4}
C4	Mean	$\mathbf{5.06 \times 10^{2}}$	2.59×10^{3}	5.39×10^{2}	5.87×10^{2}	6.18×10^{2}
	Min	4.28×10^{2}	2.06×10^{3}	4.96×10^{2}	5.20×10^{2}	5.61×10^{2}
C5	Mean	$\mathbf{5.47 \times 10^{2}}$	1.05×10^{3}	8.61×10^{2}	5.62×10^{2}	8.41×10^{2}
	Min	5.18×10^{2}	1.01×10^{3}	7.21×10^{2}	5.32×10^{2}	8.22×10^{2}
C6	Mean	6.29×10^{2}	6.75×10^{2}	6.51×10^{2}	$\mathbf{6.00 \times 10^{2}}$	6.55×10^{3}
	Min	6.21×10^{2}	6.65×10^{2}	6.29×10^{2}	6.00×10^{2}	6.51×10^{2}
C7	Mean	$\mathbf{7.81 \times 10^{2}}$	1.34×10^{3}	1.46×10^{3}	8.68×10^{2}	1.54×10^{3}
	Min	7.54×10^{2}	1.29×10^{3}	1.20×10^{3}	8.10×10^{2}	1.33×10^{3}
C8	Mean	1.41×10^{3}	1.35×10^{3}	1.13×10^{3}	$\mathbf{8.61 \times 10^{2}}$	1.10×10^{3}
	Min	8.10×10^{2}	1.28×10^{3}	1.02×10^{3}	8.32×10^{3}	1.07×10^{3}
C9	Mean	1.46×10^{3}	2.15×10^{4}	1.19×10^{4}	$\mathbf{9.24 \times 10^{2}}$	9.31×10^{3}
	Min	1.11×10^{3}	1.42×10^{4}	5.50×10^{3}	9.07×10^{2}	6.43×10^{3}
C10	Mean	2.21×10^{3}	1.47×10^{4}	7.76×10^{3}	$\mathbf{6.53 \times 10^{3}}$	8.23×10^{3}
	Min	2.03×10^{3}	1.32×10^{4}	6.40×10^{3}	5.14×10^{3}	6.83×10^{3}
C11	Mean	1.26×10^{3}	3.78×10^{3}	1.30×10^{3}	$\mathbf{1.26 \times 10^{3}}$	1.44×10^{3}
	Min	1.16×10^{3}	3.24×10^{3}	1.20×10^{3}	1.15×10^{3}	1.29×10^{3}
C12	Mean	1.27×10^{7}	2.42×10^{9}	5.72×10^{5}	1.87×10^{6}	$\mathbf{2.28 \times 10^{5}}$
	Min	8.56×10^{6}	1.71×10^{9}	1.34×10^{5}	1.08×10^{6}	1.03×10^{5}
C13	Mean	7.87×10^{4}	5.70×10^{8}	8.90×10^{3}	$\mathbf{5.36 \times 10^{3}}$	9.74×10^{4}
	Min	4.86×10^{4}	4.31×10^{8}	2.61×10^{3}	1.75×10^{3}	2.47×10^{4}
C14	Mean	2.59×10^{3}	8.32×10^{5}	3.67×10^{4}	4.02×10^{4}	$\mathbf{1.62 \times 10^{3}}$
	Min	2.54×10^{3}	2.22×10^{5}	1.15×10^{4}	1.20×10^{4}	1.56×10^{3}
C15	Mean	3.21×10^{3}	2.30×10^{8}	7.07×10^{3}	2.96×10^{3}	$\mathbf{2.46 \times 10^{3}}$
	Min	2.81×10^{3}	9.91×10^{7}	1.94×10^{3}	1.53×10^{3}	1.73×10^{3}
C16	Mean	3.71×10^{3}	5.18×10^{3}	3.58×10^{3}	$\mathbf{2.61 \times 10^{3}}$	3.58×10^{3}
	Min	2.57×10^{3}	4.75×10^{3}	2.51×10^{3}	2.15×10^{3}	3.04×10^{3}
C17	Mean	3.81×10^{3}	3.89×10^{3}	3.48×10^{3}	$\mathbf{2.47 \times 10^{3}}$	3.21×10^{3}
	Min	2.77×10^{3}	3.18×10^{3}	2.83×10^{3}	2.02×10^{3}	2.92×10^{3}
C18	Mean	5.65×10^{4}	8.48×10^{6}	1.94×10^{5}	1.25×10^{6}	$\mathbf{7.29 \times 10^{3}}$
	Min	4.29×10^{4}	4.50×10^{6}	3.38×10^{4}	1.07×10^{5}	2.51×10^{3}
C19	Mean	1.68×10^{5}	7.92×10^{7}	1.54×10^{4}	1.28×10^{4}	$\mathbf{2.63 \times 10^{3}}$
	Min	9.57×10^{4}	2.98×10^{7}	2.17×10^{3}	2.45×10^{3}	2.17×10^{3}
C20	Mean	3.23×10^{3}	3.68×10^{3}	3.32×10^{3}	$\mathbf{2.49 \times 10^{3}}$	3.22×10^{3}
	Min	2.96×10^{3}	3.26×10^{3}	2.53×10^{3}	2.08×10^{3}	2.69×10^{3}
C21	Mean	2.85×10^{3}	2.85×10^{3}	2.64×10^{3}	$\mathbf{2.36 \times 10^{3}}$	2.43×10^{3}
	Min	2.56×10^{3}	2.77×10^{3}	2.52×10^{3}	2.34×10^{3}	2.32×10^{3}
C22	Mean	2.33×10^{4}	1.53×10^{4}	9.54×10^{3}	$\mathbf{8.34 \times 10^{3}}$	1.02×10^{4}
	Min	2.31×10^{3}	4.47×10^{3}	8.21×10^{3}	6.32×10^{3}	9.43×10^{3}
C23	Mean	2.64×10^{3}	3.75×10^{3}	3.18×10^{3}	2.79×10^{3}	$\mathbf{2.47 \times 10^{3}}$
	Min	2.49×10^{3}	3.53×10^{3}	3.06×10^{3}	2.76×10^{3}	2.44×10^{3}
C24	Mean	3.77×10^{3}	3.74×10^{3}	3.27×10^{3}	2.97×10^{3}	$\mathbf{2.91 \times 10^{3}}$
	Min	3.52×10^{3}	3.60×10^{3}	3.10×10^{3}	2.93×10^{3}	2.85×10^{3}
C25	Mean	3.99×10^{3}	4.33×10^{3}	3.06×10^{3}	3.07×10^{3}	$\mathbf{3.03 \times 10^{3}}$
	Min	3.73×10^{3}	4.00×10^{3}	3.02×10^{3}	2.99×10^{3}	2.83×10^{3}
C26	Mean	5.19×10^{3}	5.80×10^{3}	8.34×10^{3}	$\mathbf{4.41 \times 10^{3}}$	5.08×10^{3}
	Min	3.87×10^{3}	5.17×10^{3}	2.90×10^{3}	4.05×10^{3}	4.24×10^{3}
C27	Mean	5.11×10^{3}	5.05×10^{3}	3.76×10^{3}	$\mathbf{3.31 \times 10^{3}}$	3.81×10^{3}
	Min	3.97×10^{3}	4.66×10^{3}	3.46×10^{3}	3.27×10^{3}	3.45×10^{3}
C28	Mean	3.75×10^{3}	4.74×10^{3}	3.30×10^{3}	3.38×10^{3}	$\mathbf{3.30 \times 10^{3}}$
	Min	3.23×10^{3}	4.47×10^{3}	3.26×10^{3}	3.31×10^{3}	3.22×10^{3}
C29	Mean	4.23×10^{3}	6.51×10^{3}	4.87×10^{3}	$\mathbf{3.55 \times 10^{3}}$	4.05×10^{3}
	Min	4.13×10^{3}	6.11×10^{3}	4.29×10^{3}	3.29×10^{3}	3.84×10^{3}
C30	Mean	1.14×10^{7}	3.34×10^{8}	1.20×10^{6}	$\mathbf{1.14 \times 10^{6}}$	7.54×10^{6}
	Min	1.08×10^{7}	2.37×10^{8}	6.44×10^{5}	3.44×10^{5}	9.57×10^{5}



(a) C3 (b) C14

(c) C15 (d) C18

Figure 3. The convergence trajectories for four benchmark functions in 30-dimensional scenarios.

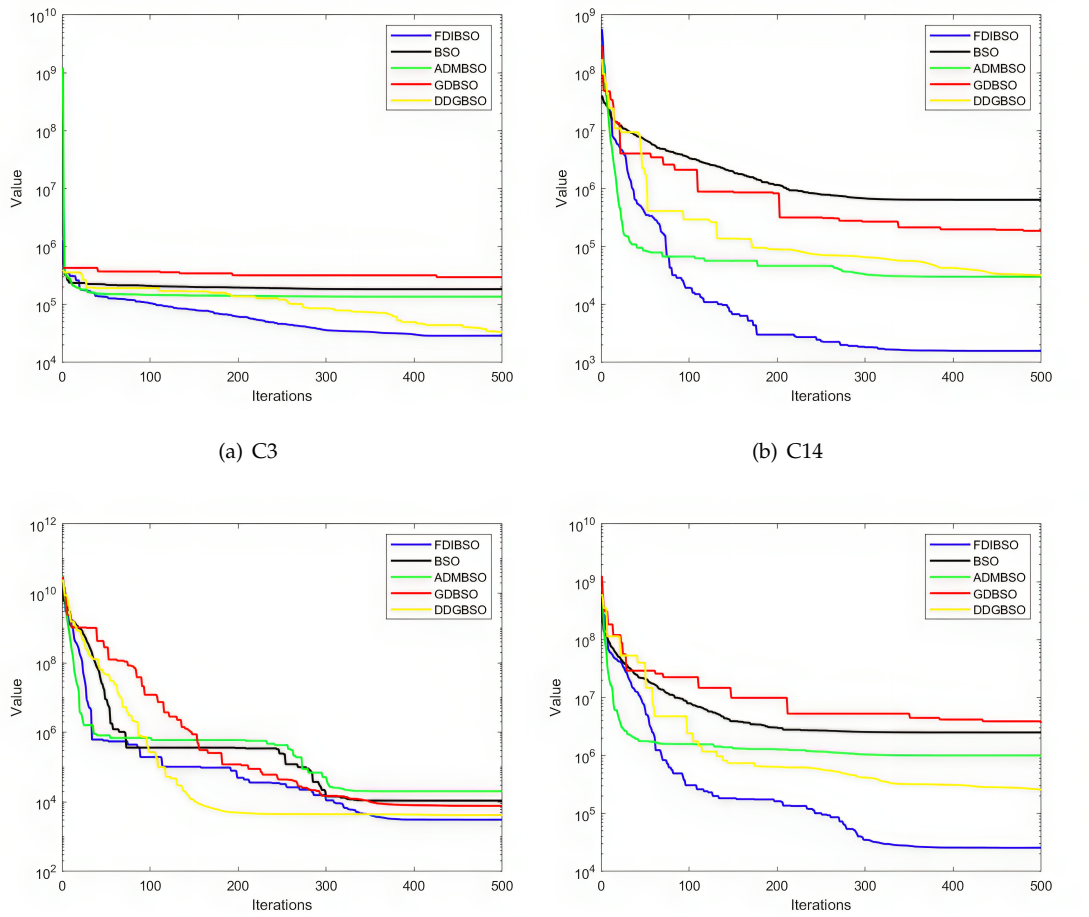
Drawing on the optimization accuracy outcomes for each algorithm, this study underscores the FDIBSO algorithm's efficacy with additional proof from the Friedman test. The outcomes of these non-parametric tests are detailed in Tables 5 and 6. Moreover, the analyses involving the Friedman test are designed explicitly to contrast the FDIBSO algorithm against other enhanced BSO algorithms, with the aim of maintaining the paper's conciseness.

Table 5. Outcomes of Friedman's test across all algorithms.

Algorithm	BSO	ADMBSO	GDBSO	DDGBSO	FDIBSO
Ranking	4.38	2.24	3.14	2.51	2.03

Table 6. Outcomes from the Wilcoxon statistical test for FDIBSO.

Algorithm	*p*-Value	R+	R−
FDIBSO vs. BSO	0.000016	417	18
FDIBSO vs. ADMBSO	0.149429	249	129
FDIBSO vs. GDBSO	0.004745	327	79
FDIBSO vs. DDGBSO	0.0750	256	119



(a) C3 (b) C14

(c) C15 (d) C18

Figure 4. The convergence trajectories for four benchmark functions in 50-dimensional scenarios.

The Friedman test facilitated the calculation of each algorithm's average rank across all test functions. Within this nonparametric statistical analysis, a lower rank indicates superior algorithm performance. The outcomes of this test are displayed in Table 5. FDIBSO ranked first (2.03), while ADMBSO ranked second (2.24). Additionally, the significance of the algorithm was assessed using the Wilcoxon statistical test, with findings presented in Table 6. Compared to FDIBSO, the *p*-values for the two algorithms were below 0.05, signifying that the FDIBSO algorithm significantly outperforms the BSO and GDBSO algorithms. Although the *p*-value of the FDIBSO algorithm is more significant than 0.05 when comparing the ADMBSO and DDGBSO algorithms, they are both much less than the value of 0.5, which proves that the FDIBSO algorithm performs better than the other two algorithms. Moreover, the resistance values R+ and R− highlight FDIBSO's outstanding performance. These outcomes further affirm the efficacy of the FDIBSO algorithm.

4.3. AUV Intelligence Evaluation Application Example

AUV intelligence evaluation can save test costs and provide guidance direction for enhancement of AUV intelligent capabilities, of which, the key technology is the solution of the combined weight model. In this paper, we drew on the multi-expert combined weight model (MMCW) proposed in [36] for evaluating AUV intelligence and simulation to verify the effectiveness and feasibility of the FDIBSO algorithm. The expression for this combined weight model is as follows.

$$\text{MinH} = \alpha \sum_{i=1}^{n} k_i \ln k_i + \beta \sum_{i=1}^{n} \left\{ \sum_{j=1}^{m} w_{cj} \ln \left(\frac{w_{cj}}{w_{ij}} \right) \right\}, \tag{10}$$

where α and β are combination coefficients and are both set to 0.5, k is an n-dimensional variable, w_{cj} is a function related to the variable k, and w_{ij} is a constant value. The settings for these parameters refer to reference [36]. This model is relatively complex, thus posing a challenge to the performance of optimization algorithms.

In this reference, for the solution of this optimization model, the comparison algorithm uses the differential evolution algorithm (DE), modified differential evolution algorithm (MDE), and GDBSO algorithm, with the same parameter settings as in [36]. These algorithms are relatively old and perform poorly, as can be seen in the experimental section of this article, where the computational performance of several algorithms is low. What is more noteworthy is that the experiments only pertain to four-dimensional variables. Therefore, as the dimensions increase, the problems with the performance of the algorithms become increasingly evident. The lower optimization performance can lead to bias in the computation of weights, which in turn, affects the performance evaluation of different AUV systems.

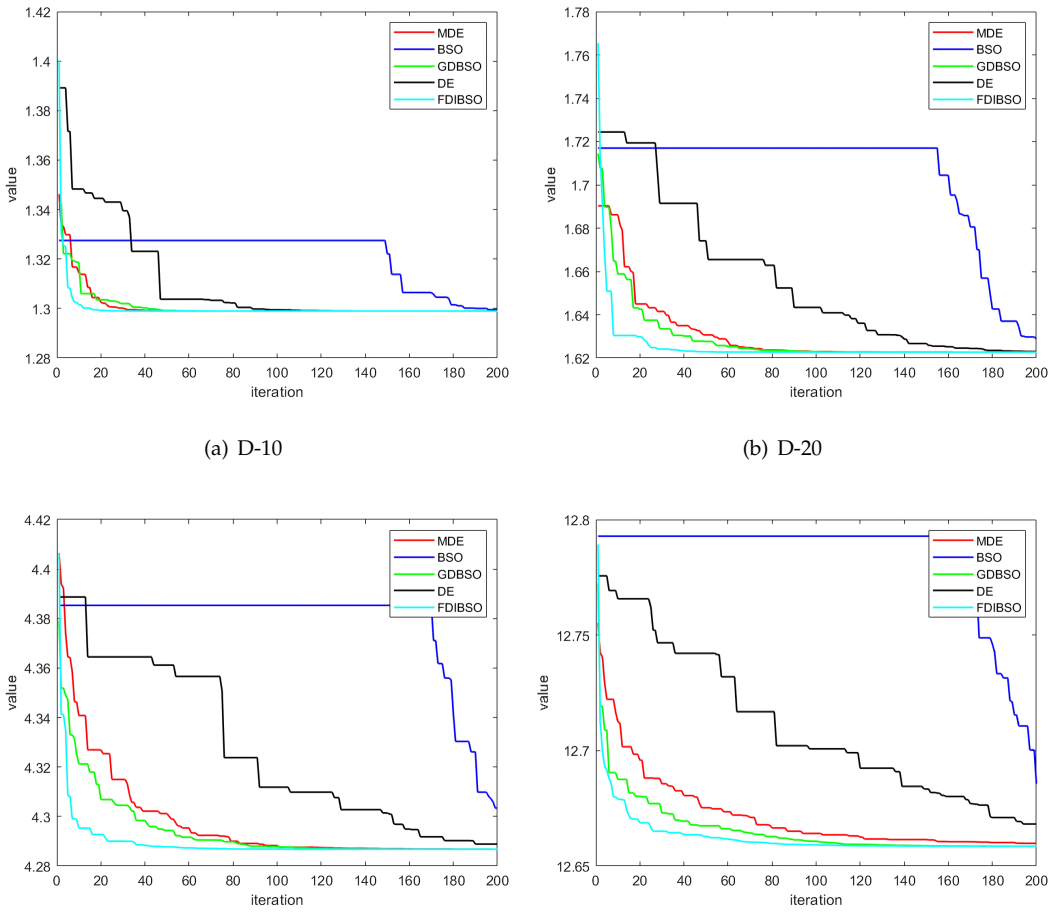
In response to the issues mentioned above, this paper introduces the FDIBSO algorithm, which has higher optimization performance, into the model's solution process to achieve a more accurate assessment of AUV intelligence. In order to examine the performance gap between the FDIBSO algorithm and the comparative algorithms at higher dimensions, this paper employs the Monte Carlo method to generate multiple sets of simulated weights. The experiments limited the number of iterations to 200. The speed at which each algorithm converged was assessed across three conditions with 10, 20, 30, and 50 dimensions, with the results depicted in Figure 5. Additionally, to ensure consistency in experimental testing of the comparative algorithms, the performance of other BSO variants used in the CEC2018 experiments, including ADMBSO and DDGBSO algorithms (the GDBSO algorithm has already been compared), was tested. The results are shown in Figure 6. The mean convergence time required to achieve the optimal solution was calculated and documented in Tables 7 and 8. A "No" in the table means the algorithm cannot find a global optimal solution within the given number of iterations.

Table 7. The mean convergence time of each algorithm (unit: seconds).

Algorithm	BSO	DE	MDE	GDBSO	FDIBSO
Mean (D-10)	1.65	0.24	0.13	0.22	0.33
Mean (D-20)	No	0.55	0.26	0.47	0.51
Mean (D-30)	No	No	0.53	0.82	0.98
Mean (D-50)	No	No	No	1.44	1.72

Table 8. The mean convergence time of different BSO variants (unit: seconds).

Algorithm	BSO	ADMBSO	DDGBSO	GDBSO	FDIBSO
Mean (D-10)	1.65	0.27	0.13	0.22	0.33
Mean (D-20)	No	0.67	0.53	0.47	0.51
Mean (D-30)	No	1.03	0.77	0.82	0.98
Mean (D-50)	No	2.23	1.49	1.44	1.72



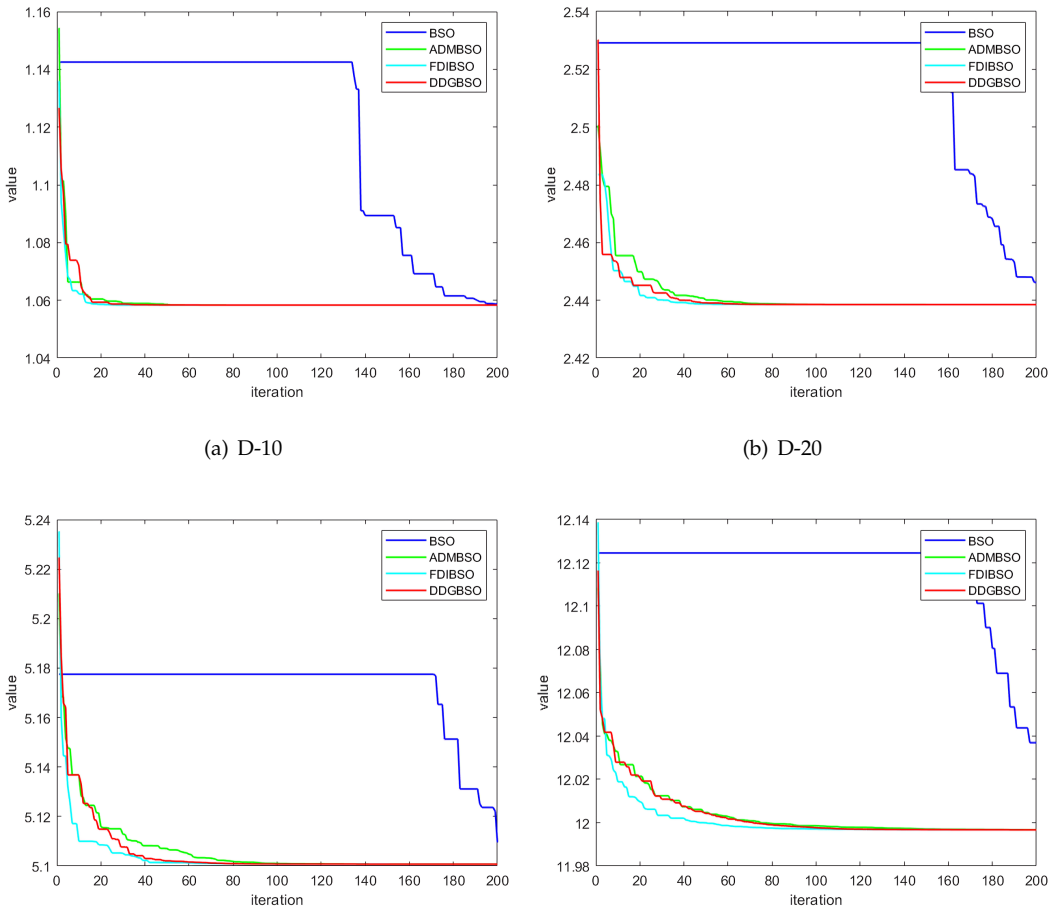
(a) D-10 (b) D-20

(c) D-30 (d) D-50

Figure 5. Convergence curves for each algorithm based on the MMCW model in four dimensions.

First, whether comparing various BSO variant algorithms or comparing the several algorithms used in reference [36], the FDIBSO algorithm proposed in this paper demonstrates the best convergence performance when solving the MMCW model. In addition, the advantage of the FDIBSO algorithm is more evident with the increase in dimensions under the four conditions. Second, under the limit of 200 iterations, some algorithms cannot find the global optimal solution under some dimensional conditions. However, the FDIBSO algorithms can all find the global optimal solution with the minimum number of iterations. When comparing the mean convergence time, the FDIBSO algorithm, due to the use of spectral clustering, leads to an increased time complexity. Although it is not advantageous compared to some other BSO algorithm variants, it is only inferior to the DDGBSO and GDBSO algorithms and is better than the ADMBSO algorithm.

The experimental findings indicate that, in terms of computing weights for AUV intelligence evaluation, the FDIBSO algorithm exhibits superior overall performance.



(a) D-10 (b) D-20

(c) D-30 (d) D-50

Figure 6. Convergence curves for BSO variants based on the MMCW model in four dimensions.

5. Discussion

Drawing upon the experimental outcomes detailed in Sections 4.2 and 4.3, it is evident that the FDIBSO algorithm possesses several advantages. First, the FDIBSO algorithm demonstrates superior optimization precision compared to other variations of the BSO algorithm, showcasing its ideal advantage. The FDIBSO algorithm can achieve optimal optimization search results on more than half of the functions featured in the CEC2018. This assertion is supported by the data in Tables 1, 2, 5 and 6. Second, the optimization precision of the FDIBSO algorithm holds a distinct edge over other contemporary competitive intelligent algorithms. The results show that although the FDIBSO algorithm achieve the same level of effectiveness as the MFO-SFR algorithm, it performs better at some functions and significantly outperforms other intelligent algorithms. This assertion is substantiated by Tables 3 and 4. Third, the rate of convergence for the FDIBSO algorithm does not match the velocity of some BSO variants in the early stage, but with advancement of iterations, the convergence speed can be realized to overtake that of the other algorithms. This assertion is supported by Figures 3 and 4. Fourth, although the FDIBSO algorithm is slightly inferior in time complexity to some BSO variant algorithms when calculating the MMCW model, its convergence performance is superior to all algorithms. It can converge to the optimal solution with the fewest iterations for problems with various dimensions, thus improving the accuracy of weight calculations. The conclusion can be confirmed by Figures 5 and 6 and Tables 7 and 8.

Numerous experiments have confirmed FDIBSO's advanced efficacy over prior enhanced BSO algorithms. As an innovative approach influenced by human behavioral

patterns, FDIBSO has shown significant promise for tackling intricate optimization issues. Additionally, incorporating the flock decision mutation strategy, good point set, and spectral clustering method within FDIBSO encourages the development of more innovative approaches for increasingly complex challenges. Moving forward, FDIBSO is set to tackle high-dimensional and large-scale applications.

Future studies may explore several avenues: further improving the optimization accuracy of the FDIBSO algorithm on all tested functions, improving the algorithm's convergence speed in the early stage, and utilizing the BSO algorithm for more real-world engineering optimization challenges.

6. Conclusions

This study introduces the improved brain storm optimization algorithm based on the flock decision mutation strategy. First, the flock decision mutation strategy is proposed to improve the BSO algorithm's optimization accuracy for various optimization issues. Second, the good point set and spectral clustering are integrated within the BSO algorithm, enhancing the algorithm's population diversity and clustering accuracy. Third, this paper compares and analyzes FDIBSO and the BSO, ADMBSO, GDBSO, DDGBSO, MSWOA, HOA, WMFO, and MFO-SFR algorithms. The results show that for the MMCW model in AUV intelligence evaluation and many benchmark functions from the CEC2018, the FDIBSO algorithm has the best performance.

Author Contributions: Conceptualization, Y.Z. and J.C. (Jianhua Cheng); methodology, Y.Z.; software, Y.Z.; validation, Y.Z., J.C. (Jianhua Cheng), and J.C. (Jing Cai); formal analysis, Y.Z.; investigation, J.C. (Jing Cai); resources, Y.Z.; data curation, Y.Z.; writing—original draft preparation, Y.Z.; writing—review and editing, Y.Z. and J.C. (Jianhua Cheng); visualization, Y.Z.; supervision, J.C. (Jianhua Cheng); project administration, J.C. (Jing Cai). All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China under grants 62073093 and 61633008, the Heilongjiang Province Science Fund for Distinguished Young Scholars under grant JC2018019, and the Basic Scientific Research Fund under grant 3072020CFT0403.

Data Availability Statement: All data generated or analyzed during this study are included in this published article.

Conflicts of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed to be a potential conflict of interest.

References

1. Liu, X.W.; Ma, Y.; Li, J. U.S. underwater unmanned vehicle development and its impact on U.S. military operational thinking. *Aerosp. Technol.* **2020**, *6*, 12–19.
2. *Autonomous Undersea Vehicle Requirement for 2025;* United States Department of Defense: Washington, DC, USA , 2016.
3. Li, L.; Huang, W.L.; Liu, Y. Intelligence testing for autonomous vehicles: A new approach. *IEEE Trans. Intell. Veh.* **2016**, *1*, 158–166. [CrossRef]
4. Shi, Y.H. Brain storm optimization algorithm. In Proceedings of the Second International Conference on Swarm Intelligence, Chongqing, China, 12–15 June 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 303–309.
5. Zhou, Q.; Gao, S. 3d uav path planning using global-best brain storm optimization algorithm and artificial potential field. In Proceedings of the Intelligent Robotics and Applications: 12th International Conference, ICIRA 2019, Shenyang, China, 8–11 August 2019; pp. 765–775.
6. Zhu, X.; Wang, Z.; Gao, G.; Chen, Y.; Wang, Y.; Li, M.; Liu, S.; Mao, H. Chaotic brain storm optimization algorithm in objective space for medical image registration. In Proceedings of the 2020 5th International Conference on Intelligent Informatics and Biomedical Sciences, Okinawa, Japan, 18–20 November 2020; pp. 81–84.
7. Zhu, H.; Shi, Y. Brain storm optimization algorithm for full area coverage of wireless sensor networks. In Proceedings of the 2016 Eighth International Conference on Advanced Computational Intelligence, Chiang Mai, Thailand, 14–16 February 2016; pp. 14–20.

8. Gupta, S.; Deep, K. A hybrid self-adaptive sine cosine algorithm with opposition based learning. *Expert Syst. Appl.* **2019**, *119*, 210–230. [CrossRef]
9. Yang, W.; Fan, S. A multi-strategy whale optimization algorithm and its application. *Eng. Appl. Artif. Intell.* **2022**, *108*, 104558. [CrossRef]
10. Nama, S.; Saha, A.K.; Chakraborty, S. Boosting particle swarm optimization by backtracking search algorithm for optimization problems. *Swarm Evol. Comput.* **2023**, *79*, 101304. [CrossRef]
11. Santana, C.J.; Macedo, M.; Siqueira, H. A novel binary artificial bee colony algorithm. *Future Gener. Comput. Syst.* **2019**, *98*, 180–196. [CrossRef]
12. Fatahi, A.; Nadimi-Shahraki, M.H.; Zamani, H. An improved binary quantum-based avian navigation optimizer algorithm to select effective feature subset from medical data: A COVID-19 case study. *J. Bionic Eng.* **2024**, *21*, 426–446. [CrossRef]
13. Chen, J.; Wang, J.; Cheng, S. Brain storm optimization with agglomerative hierarchical clustering analysis. In Proceedings of the Advances in Swarm Intelligence: 7th International Conference, Bali, Indonesia, 25–30 June 2016 ; pp. 115–122.
14. Zhao, F.; Hu, X.; Wang, L. A reinforcement learning brain storm optimization algorithm (BSO) with learning mechanism. *Knowl. Based Syst.* **2022**, *235*, 107645. [CrossRef]
15. Shen, Y.; Yang, J.; Cheng, S. BSO-AL: Brain Storm Optimization Algorithm with Adaptive Learning Strategy. In Proceedings of the 2020 IEEE Congress on Evolutionary Computation, Glasgow, UK, 19–24 July 2020; pp. 1–7.
16. Ma, W.; Gao, Y.; Li, J. Global-best difference-mutation brain storm optimization algorithm. *Syst. Eng. Electron* **2021**, *44*, 1–11.
17. Tuba, E.; Dolicanin, E.; Tuba, M. Chaotic brain storm optimization algorithm. In Proceedings of the Intelligent Data Engineering and Automated Learning–IDEAL 2017: 18th International Conference, Guilin, China, 30 October–1 November 2017; pp. 551–559.
18. Yu, Y.; Gao, S.; Cheng, S. CBSO: A memetic brain storm optimization with chaotic local search. *Memetic Comput.* **2018**, *10*, 353–367. [CrossRef]
19. Yang, Y.; Shi, Y.; Xia, S. Advanced discussion mechanism-based brain storm optimization algorithm. *Soft Comput.* **2015**, *19*, 2997–3007. [CrossRef]
20. Zhao, Y.C.; Cheng, J.H.; Cai, J. Global-best brain storm optimization algorithm based on discussion mechanism and difference step. In Proceedings of the 2023 IEEE 3rd International Conference on Computer Communication and Artificial Intelligence, Taiyuan, China 26–28 May 2023; pp. 114–119.
21. Wu, G.; Mallipeddi, R.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for the CEC 2017 Competition on Constrained Real-Parameter Optimization*; Technical Report; National University of Defense Technology: Changsha, China; Kyungpook National University: Daegu, Republic of Korea; Nanyang Technological University: Singapore, 2017.
22. Liang, X.M.; Chen, F.; Long, W. Improved particle swarm optimization based on dynamic random search technique and good point set. *J. Comput. Appl.* **2011**, *31*, 2796–2799.
23. Yuan, J.; Liu, Z.; Lian, Y. Global optimization of UAV area coverage path planning based on good point set and genetic algorithm. *Aerospace* **2022**, *9*, 86. [CrossRef]
24. Ning, G.Y.; Cao, D.Q. Improved whale optimization algorithm for solving constrained optimization problems. *Discret. Dyn. Nat. Soc.* **2021**, *2021*, 8832251 [CrossRef]
25. Von, L.U.; Ma, Y.; Li, J. A tutorial on spectral clustering. *Stat. Comput.* **2007**, *17*, 395–416.
26. Wang, Y.; Jiang, Y.; Wu, Y. Spectral clustering on multiple manifolds. *IEEE Trans. Neural Netw.* **2011**, *22*, 1149–1161. [CrossRef] [PubMed]
27. Abualigah, L.; Diabat, A.; Geem, Z.W. A comprehensive survey of the harmony search algorithm in clustering applications. *Appl. Sci.* **2020**, *10*, 3827. [CrossRef]
28. Yi, W.; Gao, L.; Li, X. A new differential evolution algorithm with a hybrid mutation operator and self-adapting control parameters for global optimization problems. *Appl. Intell.* **2015**, *42*, 642–660. [CrossRef]
29. Xue, Y.; Jiang, J.; Zhao, B. A self-adaptive artificial bee colony algorithm based on global best for global optimization. *Soft Comput.* **2018**, *22*, 2935–2952. [CrossRef]
30. Ouyang, H.; Gao, L.; Li, S. Improved global-best-guided particle swarm optimization with learning operation for global optimization problems. *Appl. Soft Comput.* **2017**, *52*, 987–1008. [CrossRef]
31. Wang, C.; Liu, K. A randomly guided firefly algorithm based on elitist strategy and its applications. *IEEE Access* **2019**, *7*, 130373–130387. [CrossRef]
32. Shen, X.; Wu, Y.; Li, L. A modified adaptive beluga whale optimization based on spiral search and elitist strategy for short-term hydrothermal scheduling. *Electr. Power Syst. Res.* **2024**, *228*, 110051. [CrossRef]
33. MiarNaeimi, F.; Azizyan, G.; Rashki, M. Horse herd optimization algorithm: A nature-inspired algorithm for high-dimensional optimization problems. *Knowl. Based Syst.* **2021**, *213*, 106711. [CrossRef]
34. Nadimi-Shahraki, M.H.; Fatahi, A.; Zamani, H. Hybridizing of whale and moth-flame optimization algorithms to solve diverse scales of optimal power flow problem. *Electronics* **2022**, *11*, 831. [CrossRef]

35. Nadimi-Shahraki, M.H.; Zamani, H. MFO-SFR: An enhanced moth-flame optimization algorithm using an effective stagnation finding and replacing strategy. *Mathematics* **2023**, *11*, 862. [CrossRef]
36. Cheng, J.H.; Zhao, Y.C.; Cai, J. An MMCW-FCE method for evaluating AUV intelligence on the algorithm level. *IEEE Access* **2022**, *10*, 132071–132082. [CrossRef]