


Multi-strategy brain storm optimization algorithm with dynamic parameters adjustment

Jianan Liu¹ · Hu Peng¹  · Zhijian Wu² · Jianqiang Chen³ · Changshou Deng¹

Abstract

As a novel swarm intelligence optimization algorithm, brain storm optimization (BSO) has its own unique capabilities in solving optimization problems. However, the performance of traditional BSO strategy in balancing exploitation and exploration is inadequate, which reduces the convergence performance of BSO. To overcome these problems, a multi-strategy BSO with dynamic parameters adjustment (MSBSO) is presented in this paper. In MSBSO, four competitive strategies based on improved individual selection rules are designed to adapt to different search scopes, thus obtaining more diverse and effective individuals. In addition, a simple adaptive parameter that can dynamically regulate search scopes is designed as the basis for selecting strategies. The proposed MSBSO algorithm and other state-of-the-art algorithms are tested on CEC 2013 benchmark functions and CEC 2015 large scale global optimization (LSGO) benchmark functions, and the experimental results prove that the MSBSO algorithm is more competitive than other related algorithms.

Keywords Brain storm optimization · Multi-strategy · Individual selection rules · Dynamic parameters adjustment

1 Introduction

In the past few decades, inspired by natural phenomena, some simple and efficient swarm intelligence algorithms or evolutionary algorithms have been proposed by researchers. Such as particle swarm optimization (PSO) [1], artificial bee colony (ABC) [2, 3], differential evolution (DE) [4], ant colony optimization (ACO) [5] and so on [6, 7]. They all have their respective advantages in solving real-world complex optimization problems.

Brain storm optimization (BSO) is an emerging swarm intelligence algorithm proposed by Shi [8], it has attracted wide attention because of its simplicity and efficiency. The core ideas of BSO are clustering and mutation, which is

conducive to searching for global optimum and maintaining population diversity. Thus, BSO is widely used in solving multimodal and high-dimensional function problems and practical optimization problems, such as predator-prey BSO for DC brushless motor [9], modified BSO algorithm for multimodal optimization [10], BSO algorithm for finding optimal location and setting of FACTS devices [11], closed-loop BSO for addressing the optimal formation reconfiguration of multiple satellites [12].

Meanwhile, owing to the imbalance between global search and local search, BSO also experiences precocity. For addressing the above problem, many excellent variants of the BSO algorithm were designed and proposed by the researchers [13, 14]. To ameliorate the performance of BSO algorithm and reduce the time spent on clustering, Zhan et al. modified the original BSO algorithm by replacing k -means clustering with simple grouping method (SGM) and proposed modified BSO (MBSO) [15]. In 2013, Yang proposed a discussion mechanism based BSO (DMBSO) which can well balance global search and local search [16]. Recently, Chu et al. presented an augmented BSO with mutation strategies (ABSO) [17]. To enhance the convergence performance of BSO, Peng et al. proposed a self-adaptive BSO with p best guided step-size (SPBSO) [18]. Although many scholars have proposed excellent

✉ Hu Peng
hu_peng@whu.edu.cn

¹ School of Information Science and Technology, Jiujiang University, Jiujiang, 332005, China

² School of Computer Science, Wuhan University, Wuhan, 430072, China

³ School of Information Technology, Jiangxi University of Finance and Economics, Nanchang, 330032, China

variants of BSO, there is still much room for improving the BSO algorithm.

By analyzing the cross-population strategy in the BSO algorithm, it can be found that this strategy uses the information exchange between different populations to generate new individual, which is essentially similar to the mutation strategy of differential evolution. In fact, many latest algorithms that combine strategies with different characteristics can improve the performance of the algorithm, such as the hybrid differential evolution adaptive algorithm (hDEBSA) in which the differential strategy is employed to update the worst individual [19]. Besides, Cheng et al. proposed a cuckoo search algorithm with multiple update rules (HCS) [20]. Guo et al. introduced a hybrid bare bones particle swarm optimizer (FHBBPSO) which combines two collaborative strategies to strength the search capability [21]. In addition, a hybrid PSO using adaptive learning strategy (ALPSO) also belongs to the above situation [22].

Inspired by the above analysis, a multi-strategy BSO algorithm with dynamic parameters adjustment is proposed in this paper. The main contributions of this paper can be summarized as follows:

- i) Combining multiple strategies based on different individual selection rules to improve the search ability of the algorithm,
- ii) A simple adaptive parameter for selecting appropriate strategies in different iteration periods to balance exploitation and exploration,
- iii) Dynamic adjustment of crossover probability (CP) to accelerate the convergence speed of the algorithm while enhancing the exploration performance of the algorithm.

The remaining sections of this paper are as follows. In Section 2, the original BSO algorithm is briefly introduced. Section 3 makes some simple summarizations and classifications of related works. Section 4 describes the new BSO variant. Experimental studies and validity tests are presented in Section 5. In Section 6, there is a conclusion for this paper, including some directions for future research.

2 Brain storm optimization algorithm

Different from other swarm intelligence algorithms, BSO algorithm is mainly abstracted from the continuous optimization process of human creative problem solving. The principle of the original BSO is to continuously generate superior ideas by communicating between populations from different backgrounds, these ideas also can be called

individuals. Generally, the structure of BSO can be divided into three parts. Firstly, BSO initializes a population contains NP individuals ($X_1, X_2, X_3 \dots X_{NP}$) each with D dimensions. Then, the clustering operation and replacement operation of cluster center will be executed in each round of iteration. Lastly, selecting one or two individuals according to P_{one} to generate new individuals, where P_{one} is the probability of selecting a random individual. The selected individuals are from ordinary individuals or cluster centers, which bases on P_{one_center} or P_{two_center} . The P_{one_center} is probability of selecting one random cluster center, and the P_{two_center} is the probability of selecting two random cluster centers. Meanwhile, the better individual selected among current individual and newly generated individual is used to renew the population. The specific clustering and generating individuals operations are as follows.

2.1 Clustering and replacement

After initialization, the fitness values of all individuals will be calculated. Then the population is divided into NC clusters by k -means method in each iteration, where the NC represents the number of cluster. At the same time, the cluster centers are generated for each cluster. Generally speaking, each cluster center is the best individual in its corresponding cluster. Each cluster center may be selected to produce new individuals, which can direct the search direction of the BSO. For the sake of keeping the search direction from being too centralized, one cluster center will be replaced with a probability $P_{replace}$ by a random individual in each iteration.

2.2 Generation and selection operators

In BSO, the quality of population can be heightened through information exchange among different clusters. The method of information exchange depends on the individual selection within mutation strategy. Further speaking, the strategy uses one random ordinary individual or one random cluster center, which is determined by P_{one_center} . Similarly, whether the newly generated individual is derived from two random ordinary individuals or two cluster centers depends on P_{two_center} . Then the algorithm iterates over and over to find a better individual until the maximum number of iteration or established accuracy is met.

In order to better distinguish the mutation strategies, the strategies are described through the following template:

$$BSO/n/s \quad (1)$$

where n denotes the number of different clusters that appear in one strategy. One strategy can adopt one or two clusters,

or even more than two clusters. The s denotes the method of selecting individuals. One strategy can select random individuals or cluster centers, and even include both the best individuals and random individuals at the same time, which can be called rand-to-best.

In accordance with above template, the strategies of BSO algorithm are expressed as follows:

1. BSO/one/rand

$$V_{i,d} = X_{r1,d} + stepsize_d \times N(0, 1)_d \quad (2)$$

2. BSO/one/center

$$V_{i,d} = X_{c1,d} + stepsize_d \times N(0, 1)_d \quad (3)$$

3. BSO/two/rand

$$V_{i,d} = (Rand) \times X_{r1,d} + (1 - Rand) \times X_{r2,d} + stepsize_d \times N(0, 1)_d \quad (4)$$

4. BSO/two/center

$$V_{i,d} = (Rand) \times X_{c1,d} + (1 - Rand) \times X_{c2,d} + stepsize_d \times N(0, 1)_d \quad (5)$$

The step-size is calculated as follows:

$$stepsize = \logsig\left(\frac{0.5 \times G - g}{k}\right) \times random(0, 1) \quad (6)$$

where the $V_{i,d}$ represents d_{th} dimension of i_{th} new individual, and the $X_{r1,d}$ represents d_{th} dimension of a random individual. Besides, the $X_{c1,d}$ denotes the d_{th} dimension of one individual, and this individual is the one of the five cluster centers. The $Rand$ and $stepsize$ represent a random number (ranging from 0 to 1) and search step respectively. The $N(0, 1)_d$ is a Gaussian random function with the mean value 0 and the standard deviation 1. In $stepsize$, the g is the number of current iteration, the G is maximum iterations, and the k is a parameter used to change the slope of transfer function ($\logsig()$).

3 Related work

On the way to continuously ameliorate the performance of BSO algorithm, new variants of BSO algorithm have been proposed by scholars one after another [23]. In order to get more acquainted with the different directions of BSO algorithm improvement, this section makes a simple classification of various BSO variants.

Some schemes for improving the step-size of BSO were proved feasible by extensive experiments. For example, Zhou et al. (2012) proposed a BSO variant with modified step-size [24], which dynamically regulates the d_{th} dimension step size according to the distribution

characteristics of all individuals so as to adjust convergence rate of algorithm. In 2015, Yang et al. proposed a novel BSO algorithm with differential step [25], which realizes dynamic adjustment of step size according to population distribution. Recently, Yu et al. proposed an improved BSO with flexible search length and memory-based selection (ASBSO) [26].

Many scholars take the modification of clustering as a breakthrough. For example, a modified BSO (MBSO) was proposed by Zhan et al. (2012), which utilizes simple random grouping (SGM) instead of k -means clustering to lessen computational burden [15]. Zhu et al. (2015) proposed a BSO with k -medians clustering, in which the median of all individuals in each cluster is the center of cluster [27]. Besides, an improved BSO with dynamic clustering strategy (BSO-DCS) was proposed by Cao et al. (2017), which employed a probabilistic parameter to determine clustering [28].

Hybrid BSO algorithm is also a feasible choice to improve performance. In 2015, Cao et al. proposed an improved version of BSO with differential evolution strategy for applications of ANNs (BSODE) [29]. They introduced the differential mutation strategy and crossover operation of DE algorithm into BSO algorithm. This novel hybrid approach can combine the advantages of differential mutation. Besides, a BSO with discrete particle swarm optimization (BSO-DPSO) is presented by Hua et al. (2016) [30]. This hybrid algorithm introduces exchange operator from DPSO [31] and utilizes inversion idea from chromosome structure.

In fact, there are many successful cases in improving the mechanism of creating new individuals. For example, Yang et al. (2013) proposed discussion mechanism based BSO algorithm (DMBSO) in which the traditional method of individual renewal is replaced by inter group discussions and intra group discussions [16]. The inter group discussions and intra group discussions are controlled by decreasing function and increasing function respectively. Additionally, Wang et al. (2017) proposed a modified BSO with learning strategy [32]. The learning strategy makes the better individuals keep away from worse individuals and enables worse individuals to learn better individuals, which may make algorithm escape local optimum.

In recent years, many scholars have combined the improved BSO algorithm with practical problems. Chen et al. (2015) proposed an enhanced BSO for wireless sensor networks deployment (EBSO) [33], which employs an improved affinity propagation (AP) clustering method and an augmented creating strategy to achieve the dynamic deployments of two different wireless sensor networks (WSN). Liang et al. (2018) combined MBSO algorithm with

powell algorithm to solve the problem of medical image registration, which improves the accuracy of the result and shortens the time of image registration [34, 35].

In addition, some improved BSO algorithms focus on the tuning of algorithm structure. For example, Cheng et al. (2014) proposed an improved BSO for maintaining population diversity [36], where the method of measuring population diversity plays an important role. A re-initialization operation is added to the algorithm after the creation operation, and two customization strategies are used to re-initialize some individuals when the measurement shows poor diversity of the population. There are many algorithms in this respect that are not explained in this paper.

What needs to be emphasized is that the work of this paper belongs to a novel variant of the BSO algorithm that focuses on the design of individual selection rules. These flexible individual selection rules improve the efficiency of exploration. In addition, this paper added dynamic adjustment parameters.

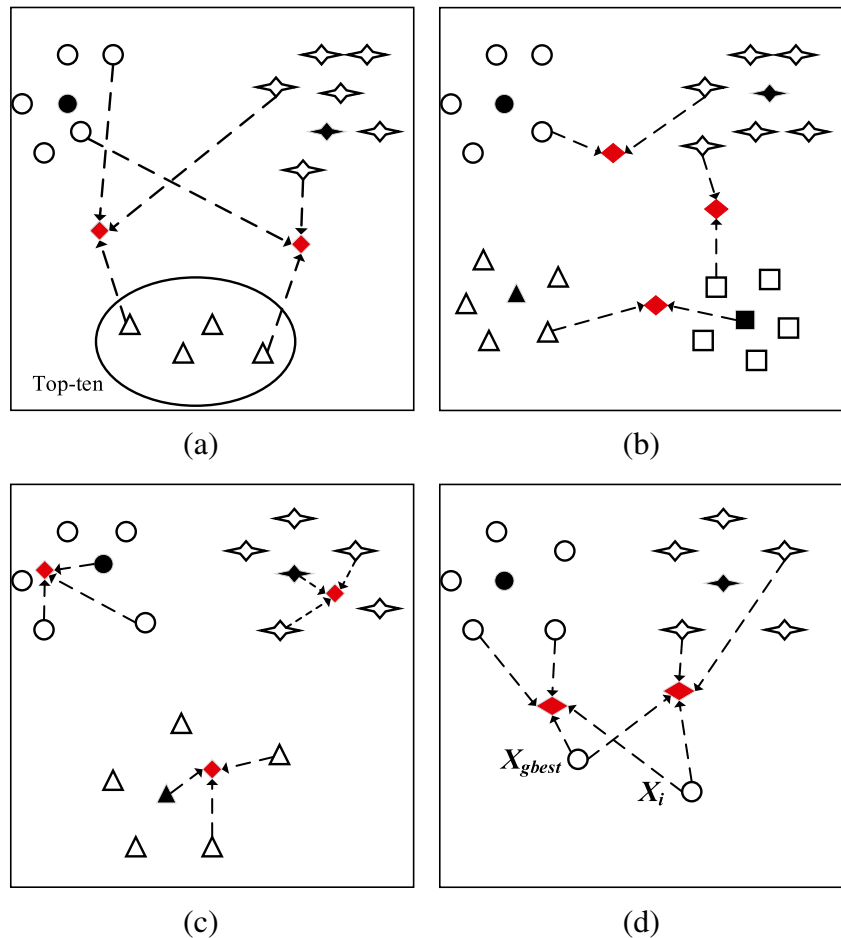
4 Multi-strategy BSO with dynamic parameters adjustment

In the view of how to better balance the exploitation and exploration of the algorithm while increasing effectiveness of new individuals, this paper improves the original BSO algorithm as follows. On the one hand, the appropriate strategies are selected in different iteration periods to enhance the adaptability of the algorithm so as to coordinate the exploitation and exploration. On the other side, the crossover probability CP from DE algorithm is dynamically adjusted to control the degree of variation of new individuals, thus strengthening the performance of exploration.

4.1 Multi-strategy in individual generation

Compared with BSO algorithm, MSBSO algorithm abandons the operation of replacing cluster center. In

Fig. 1 Schematic diagrams of four different strategies



addition, four different strategies are used to generate new individuals in this paper. These four kinds of strategies based on (1) are listed thereafter, and these strategies are selected based on P , where P is a dynamic probability parameter that will be explained in Section 4.2.

1. MSBSO/three/rand-to-best

$$V_{i,d} = X_{best,d}^{rand} + S \times (X_{i1,d} - X_{i2,d}) \quad (7)$$

2. MSBSO/two/rand

$$V_{i,d} = (Rand) \times X_{i1,d} + (1 - Rand) \times X_{i2,d} + stepsize_d \quad (8)$$

3. MSBSO/one/rand-to-center

$$V_{i,d} = X_{p_center,d} + S \times (X_{p1,d} - X_{p2,d}) \quad (9)$$

4. MSBSO/three/current-to-gbest

$$V_{i,d} = X_{i,d} + S \times (X_{gbest,d} - X_{i,d}) + S \times (X_{p1,d} - X_{p2,d}) \quad (10)$$

where the X_{best}^{rand} is one of the top ten best individuals which is inspired by literature [37]. The scaling parameter (S) is fixed as a constant, d means the d_{th} dimension of individual, $i1$ and $i2$ are two random individuals from two different clusters, i represents the current individual. p_center , $p1$ and $p2$ represent three individuals from the same one cluster in which p_center is the cluster center. The X_{gbest} represents the best individual found so far.

Figure 1 briefly illustrates the characteristics of the corresponding strategy, and the letter (a, b, c, d) of the figure corresponds to the serial number (7, 8, 9, 10) of the strategy one by one. The operation of specific selection of strategies can be referred to Algorithm 1, and all parameters of MSBSO algorithm are summarized in Table 1.

With regard to Fig. 1, the individuals included in ellipse are the top ten best individuals, and the remaining icons of different shapes represent different clusters. In addition, the solid icon in Fig. 1 represents the cluster center of a cluster. The red diamond in Fig. 1 represents the new individual generated by the corresponding strategy, and its position represents the size of search scope of the corresponding strategy. The position of the new individual

in Fig. 1a is between different clusters, and it is close to the excellent individuals. This makes the algorithm to maintain population quality while focusing on global search. Similarly, the position of new individual in Fig. 1b is between any two clusters, which is conducive to the random exploration of the algorithm among all clusters. The position of the new individual in Fig. 1c is inside a random cluster, which allows the algorithm to exploit in a smaller range. The cluster center of this random cluster controls the direction of exploitation. The position of the new individual in Fig. 1d is around the current individual, and the position of the new individual is also affected by the best individual. This mechanism facilitates algorithm to exploit around the current individual while the sum of the two sets of vector differences acts as the step size for the local search. One group of vector differences is formed by the difference of two individuals from the same cluster, which will help to narrow the search range of the algorithm.

To further demonstrate the characteristics of different search strategies in a specific search space, the Rastrigin's function [38] is used to test the process of generating new individuals in the same population using four different strategies. The formula for this function is as follows.

$$f(X) = A \times D + \sum_{i=1}^D [X_i^2 - A \cos(2\pi X_i)] \quad (11)$$

where X represents a D-dimensional individual, the value of A is 10. This function $f(X)$ has one global optimum ($X=0$, $f(X)=0$) and a large number of local optima which can be roughly regarded as points on integer coordinates (e.g. points in Fig. 2 with $X_1 = a$ and $X_2 = a$, $a=[-5, -4, \dots, 4, 5]$). So to find a local optimum near a random solution, it is only necessary to round each dimension of this random solution to the nearest integer.

These four strategies based on two-dimensional Rastrigin's function are showed in Fig. 2, where the color depth of the contour represents the individual fitness value and the search range of this function is $[-5.12, 5.12]$. Since each dimension of the local optimum approximates an integer, the number of local optima is 11^2 in Fig. 2a, b, c and d. In addition, each local optimum has a color gradation area around it, this area is actually a local optimum area that can

Table 1 The parameters of BSO and MSBSO

Algorithm	Parameter setting
BSO	$NP=100$, $NC=5$, $D=30$, $G=3000$, $P_{one} = 0.8$, $P_{replace} = 0.2$, $P_{one_center} = 0.4$, $P_{two_center} = 0.5$, $k=20$,
MSBSO	$NP=100$, $NC=5$, $D=30$, $P_{global} = 0.6$, $P_{local} = 0.8$, $S = 0.9$, $k=20$, $G=3000$

be well described by the color gradation legend ($f(X_1, X_2)$) in Fig. 2.

The headings of the four pictures in Fig. 2 correspond in turn (7), (8), (9) and (10), so the meaning of the individual marked with colored dots in each picture has been explained in (7), (8), (9) and (10). In particular, the black dot in Fig. 2 indicates the newly generated individual. Figure 1 shows four individual selection rules. The strategy (7) corresponding to Fig. 1a, which selects three individuals from two different clusters and some excellent individuals.

This description of Fig. 1a can be confirmed by Fig. 2a in which two random individuals (X_{i1}, X_{i2}) are originate from two local optimum areas that are far apart from each other. Another excellent individual (X_{best}^{rand}) is located in the central region of one local optimum area. So the situation in Fig. 2a can enlarge the search radius centered on excellent individuals. The strategy (8) corresponding to Figs. 1b and 2b which chooses two different individuals (X_{i1}, X_{i2}) from two random clusters, and these two individuals are located in two local optimum areas that are far apart, so as

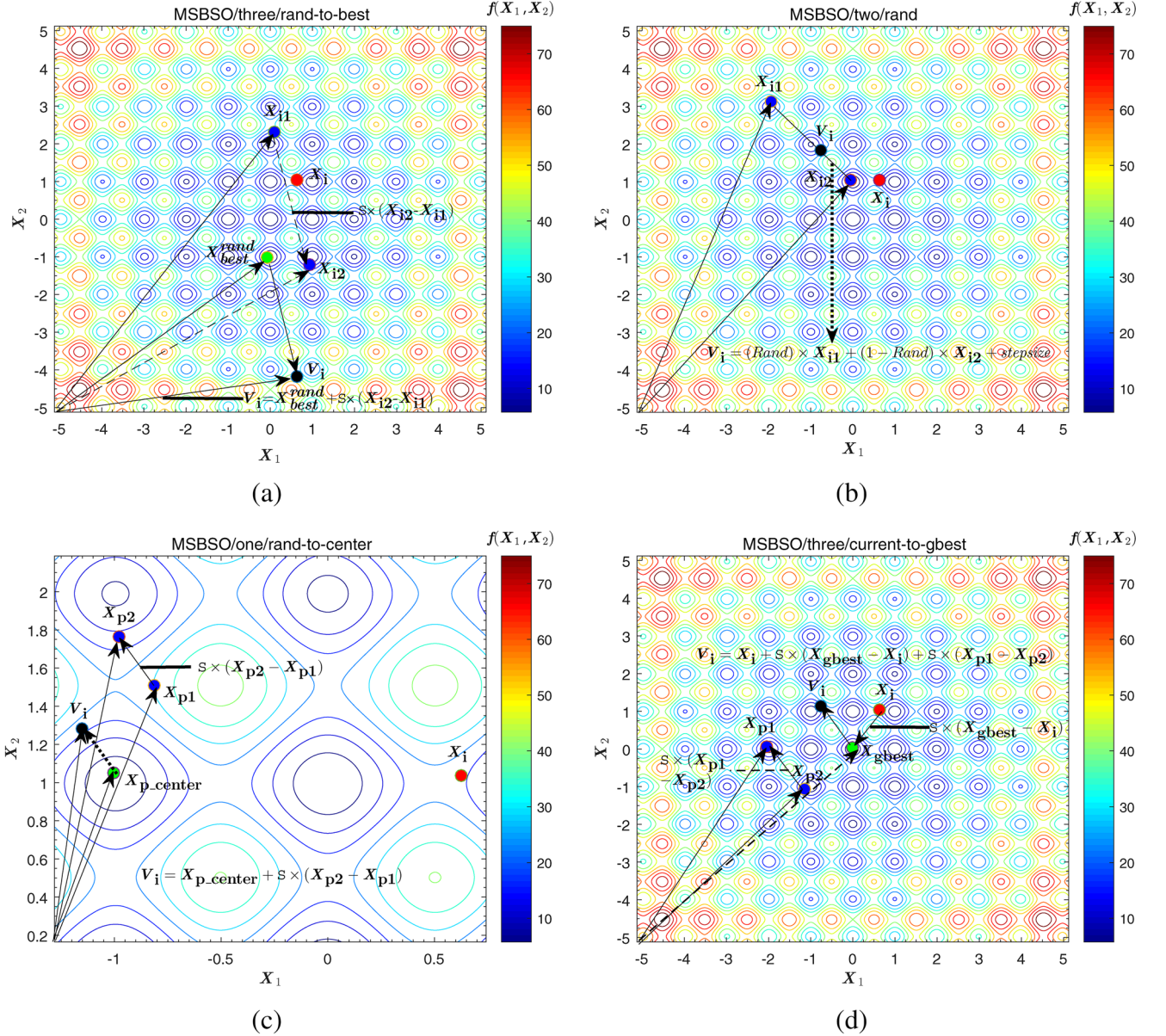


Fig. 2 Four contour line graphs of different strategies based on two-dimensional Rastrigin's function

to heighten the exploration ability of the algorithm while maintaining the diversity of the population.

The strategy (9) corresponding to Fig. 1c, which selects three different individuals from the same cluster, and one of the three individuals is the cluster center. Obviously, it can be seen from Fig. 2c that the individual information used in this strategy comes from the same cluster (three adjacent local optimum areas), which is suitable for local search around cluster center. The strategy (10) corresponding to Fig. 1d, which selects multiple individuals, including the current optimal individual and the current individual, as well as two individuals from the same cluster. As shown in Fig. 2d, the difference vectors of two individuals from the same cluster will decrease with the increasing number of iterations (the difference within the cluster becomes smaller), which will be beneficial for the current individual to search around the current optimal individual.

4.2 The selection of strategies

With the unceasing iteration of the algorithm, the space range of candidate individuals is shrinking, so that the search range should also be shrinking. Different from the original BSO algorithm, this paper adaptively adjusts P to make the algorithm focus on global search in early period and local search in the later period. The P decreases exponentially as the number of current iterations increases. In addition, the (9) is used for local search based on small probability in the early stage, which is complementary to (7). Similarly, the (8) is selected for global search based on small probability at later stage, which is complementary to (10). In general, the algorithm achieves a balance between exploitation and exploration based on the three parameters of P , P_{global} and P_{local} . Finally, the process of selecting different strategies based on P value is described in Fig. 3. In the Fig. 3, the size of the rectangle represents the probability of selecting strategy.

The formula of P

$$P = \exp(1 - \frac{G}{G-g+1}) \quad (12)$$

where \exp means finding an exponent based on e . In (12), the value of parameter P decreases exponentially from 1 to 0. It is necessary to add that each iteration in this paper contains NP function evaluations, and the value of g will increase by 1 at the end of each iteration, so the actual maximum number of function evaluations used in the following sections should be $FES \times NP$.

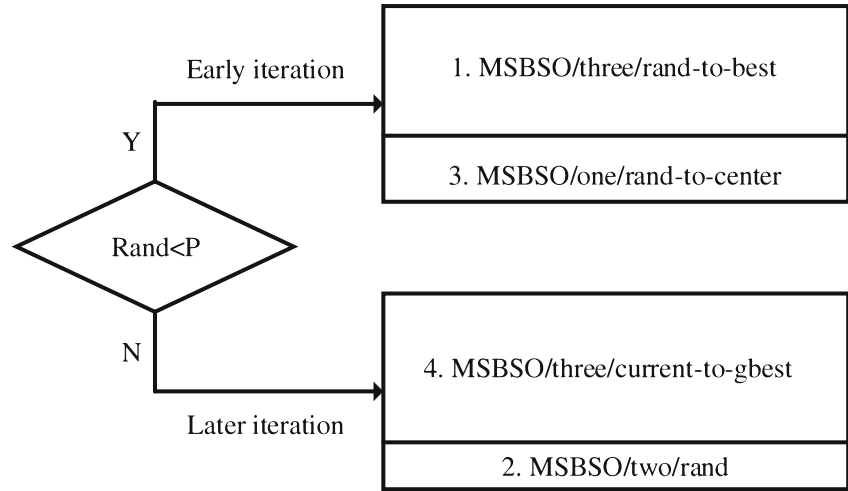
Algorithm 1 MSBSO algorithm.

Require: serial number of function, minimum, maximum
1: Initialize: $D = 30$, $NP = 100$, $NC = 5$, $G = 3000$.
2: Calculate fitness of initial population \rightarrow fitness
3: $-popu$.
while $g \leq G$ **do**
4: Dynamic update CP
5: Clustering and generating five cluster centers
6: **for** i to NP **do**
7: **if** $Rand < P$ **then**
8: **if** $Rand < P_{global}$ **then**
9: Select (7) to generate $\rightarrow V_{i,d}$.
10: **else**
11: Select (9) to generate $\rightarrow V_{i,d}$.
12: **end if**
13: **else**
14: **if** $Rand < P_{local}$ **then**
15: Select (10) to generate $\rightarrow V_{i,d}$.
16: **else**
17: Select (8) to generate $\rightarrow V_{i,d}$.
18: **end if**
19: **end if**
20: Retain better individuals, updates fitness
21: $-popu$
 end for
22: $g = g + 1$
23: **end while**
24: **return** GlobalMin, GlobalMinArray, CPUtime,

4.3 Parameter dynamic tuning

In this paper, the scaling parameter S and crossover probability CP are similar to the corresponding parameters in DE algorithm. The (13) is used for every strategies of MSBSO. More concretely, after each one-dimensional mutation of an individual, crossover operations are performed based on CP . With the purpose for making the mutation level of individual more adjustable when mutation occurs, we dynamically adjust CP according to (14), which enables the algorithm to obtain a larger crossover probability at the later iteration period. This promotes the algorithm to keep away from the local optimum by exploring the new region. Additionally, because the large step size is not conducive to the stability of the population in the later iteration stage, this paper slightly alters the step-size formula of the BSO algorithm. The (15) shows the modified step size formula which increases linearly in a very small range. So the (15) can disturb the

Fig. 3 Strategy selection based on P



population in a small range. In addition, it can accelerate the convergence of the algorithm.

$$V_{trial,d} = \begin{cases} V_{i,d}, & \text{if } Rand \leq CP \text{ or } i = d_{rand} \\ X_{i,d}, & \text{otherwise} \end{cases} \quad (13)$$

$$CP = 0.9 - \exp\left(1 - \frac{G}{G-g+1}\right) \times 0.2 \quad (14)$$

$$stepsize = \text{logsig}\left(\frac{0.5 \times g - G}{k}\right) \times \text{random}(0, 1) \quad (15)$$

where, the V_{trial} represents trial individual, and the d_{rand} represents a random integer from 1 to D . According to Algorithm 1, the algorithm will retain better individual from $V_{trial,d}$ and $X_{i,d}$ during the current cycle. k is a parameter used to change the slope of transfer function ($\text{logsig}()$), the CP is a critical parameter, it ranges from 0.7 to 0.9, and its value increases in a pseudo-exponential manner with the number of iteration rises. In particular, this adjustment method can increase the mutation degree of individual as the crossover probability increases. e.g. when CP equals 0.9, it means that the values on the 90% dimensions in the trial individual may be generated by mutation strategies.

5 Experimental studies

5.1 Benchmark functions and parameters setting

In this section, 28 benchmark functions are utilized to verify the performance of MSBSO, and the specific information of these functions can be found in Table 2. These benchmark functions were proposed in the CEC 2013 special session on real-parameter optimization [39]. Among the 28 benchmark functions, $f_1 - f_5$ belong to the unimodal functions, $f_6 - f_{20}$ are classified as basic multimodal functions, the last eight

composition functions are $f_{21} - f_{28}$. The search region of these 28 functions is $[-100, 100]^D$.

The strategies and parameter adjustment methods in this paper are inspired by DE algorithm and jDE algorithm respectively. In order to make the experiment more convincing, we selected six experimental objects, namely DE [4], jDE [40], BSO [8], MBSO [15], hDEBSA [19] and MSBSO. To avoid the effect of initial population differences on the experimental results, these six algorithms uniformly employ population with 100 individuals and 30 dimensions. The parameters of BSO algorithm and MSBSO algorithm can be referred to Table 1. The parameters of DE algorithm come from [41], and the parameters of jDE algorithm are derived from literature [40]. The parameters of hDEBSA algorithm are derived from literature [19]. Finally, we quote the parameters in literature [15] as the parameters of MBSO algorithm. In order to further verify the performance of MSBSO algorithm, the CEC 2013 test results of MSBSO algorithm and ASBSO algorithm in different dimensions are compared individually [26]. All parameters of ASBSO are derived from literature [26].

Additionally, the maximum number of function evaluations adopted in the above algorithms is $10000 \times D$ (FEs \times 100). In this paper, each experiment based on CEC 2013 test set will run 30 times separately. Meanwhile, the mean value and standard deviation of the running results will be recorded. At the end, in order to make the experimental data more objective and intuitive, this paper adopts the Wilcoxon's rank sum test with significant level of 0.05 to further analyze the experiments mentioned above [42].

Furthermore, we have carefully considered comparing the MSBSO algorithm with the winning algorithm (iCMAES-ILS) of the CEC 2013 special session on real-parameter optimization [43]. We have also done some pre-experiments of MSBSO and iCMAES-ILS on CEC 2013 benchmark functions. The experimental results show that

Table 2 Twenty-eight benchmark functions of CEC 2013

Type	Function	Specific name	Optimal solution
Unimodal Functions	f_1	Sphere Function	-1400
	f_2	Rotated High Conditioned Elliptic Function	-1300
	f_3	Rotated Bent Cigar Function	-1200
	f_4	Rotated Discus Function	-1100
	f_5	Different Powers Function	-1000
Multimodal Functions	f_6	Rotated Rosenbrock's Function	-900
	f_7	Rotated Schaffers F7 Function	-800
	f_8	Rotated Ackley's Function	-700
	f_9	Rotated Weierstrass Function	-600
	f_{10}	Rotated Griewank's Function	-500
	f_{11}	Rastrigin's Function	-400
	f_{12}	Rotated Rastrigin's Function	-300
	f_{13}	Non-Continuous Rotated Rastrigin's Function	-200
	f_{14}	Schwefel's Function	-100
	f_{15}	Rotated Schwefel's Function	100
	f_{16}	Rotated Katsuura Function	200
	f_{17}	Lunacek Bi_Rastrigin Function	300
	f_{18}	Rotated Lunacek Bi_Rastrigin Function	400
	f_{19}	Expanded Griewank's plus Rosenbrock's Function	500
	f_{20}	Expanded Scaffer's F6 Function	600
Composition Functions	f_{21}	Composition Function 1 (n=5, Rotated)	700
	f_{22}	Composition Function 2 (n=3, Unrotated)	800
	f_{23}	Composition Function 3 (n=3, Rotated)	900
	f_{24}	Composition Function 4 (n=3, Rotated)	1000
	f_{25}	Composition Function 5 (n=3, Rotated)	1100
	f_{26}	Composition Function 6 (n=5, Rotated)	1200
	f_{27}	Composition Function 7 (n=5, Rotated)	1300
	f_{28}	Composition Function 8 (n=5, Rotated)	1400
Search Region: $[-100, 100]^D$			

MSBSO does not perform as well as iCMAES-ILS. But it's not hard to find that iCMAES-ILS combines IPOP-CMA-ES and ILS into a hybrid algorithm [44, 45], and the IPOP-CMA-ES is considered nowadays as one main representative of the state-of-the-art in black-box continuous optimization. Further, original version (CMA-ES) of IPOP-CMA-ES directs the generation of new individuals by adapting the covariance matrix of normal distribution after each iteration [46], and this mechanism is missing from BSO. For the above considerations, we have not compared MSBSO with iCMAES-ILS algorithm in Section 5.

5.2 Performance of the MSBSO

This Section verifies performance of MSBSO algorithm. Table 3 shows the mean error and standard deviation of DE, jDE, BSO, MBSO, hDEBSA and MSBSO. The results of the rank sum test for each algorithms can be seen from

the bottom of Table 3, where “-”, “+”, “≈” represent separately the performance of current algorithm is worse than, better than, and similar to MSBSO [47, 48].

In accordance with Table 3, the MSBSO achieves more effective solutions than the other five algorithms. Compared with DE algorithm and BSO algorithm, more than 24 test functions of MSBSO outperform these two algorithms. Furthermore, the MSBSO is significantly better than MBSO in solution quality on most test functions except f_2 , f_{15} , f_{16} , f_{18} , f_{23} , f_{28} . Moreover, apart from several functions (f_5 , f_{14} , f_{15} , f_{17} , f_{18} , f_{23}), MSBSO performs better on most test functions than hDEBSA. Even compared with jDE algorithm recognized by academia, MSBSO algorithm has nearly half of the solutions of test functions better than it.

The performance of these six algorithms will be reflected distinctly by the result of Friedman test. Apparently, the comprehensive performance of MSBSO is better than that of the other six algorithms according to the Table 4. Based

Table 3 Experimental results of DE, BSO, jDE, MBSO, hDEBSA, MSBSO on CEC 2013 benchmark tests at $D=30$ and comparison results of these algorithms based on Wilcoxon's rank sum test

F	DE(Mean±Std)	BSO(Mean±Std)	jDE(Mean±Std)	MBSO(Mean±Std)	hDEBSA(Mean±Std)	MSBSO(Mean±Std)
f_1	1.89E+03±7.30E+02–	6.06E–14±1.01E–13–	0.00E+00±0.00E+00≈	1.59E–13±1.04E–13–	0.00E+00±0.00E+00≈	0.00E+00±0.00E+00
f_2	1.96E+08±3.52E+07–	1.60E+06±5.88E+05–	1.12E+05±7.49E+04–	6.51E+04±2.81E+04≈	2.08E+06±7.05E+05–	5.87E+04±3.71E+04
f_3	3.77E+10±6.57E+09–	1.15E+08±1.53E+08–	1.19E+06±3.61E+06–	1.96E+06±3.28E+06–	1.31E+08±1.39E+08–	4.33E+04±1.91E+00
f_4	6.24E+04±8.68E+03–	2.31E+04±4.86E+03–	2.32E+01±1.56E+01+	1.12E+03±1.53E+03–	6.64E+03±2.83E+03–	2.68E+02±1.63E+02
f_5	1.02E+02±1.87E+01–	6.99E–03±1.83E–03–	1.02E–13±3.41E–14+	1.71E–13±7.63E–14–	3.79E–15±2.04E–14+	1.21E–13±2.84E–14
f_6	3.89E+02±1.01E+02–	4.53E+01±2.52E+01–	1.32E+01±3.71E+00–	1.80E+01±2.07E+01–	4.96E+01±2.57E+01–	7.96E+00±3.47E+00
f_7	1.73E+02±2.44E+01–	1.31E+02±4.81E+01–	2.95E+00±2.84E+00–	2.44E+01±1.60E+01–	5.62E+01±1.36E+01–	1.05E+00±7.35E–01
f_8	2.10E+01±3.95E–02≈	2.09E+01±7.87E–02≈	2.09E+01±4.95E–02≈	2.10E+01±8.26E–02–	2.10E+01±4.09E–02≈	2.09E+01±5.92E–02
f_9	3.96E+01±9.92E–01–	3.08E+01±3.05E+00–	2.41E+01±6.31E+00–	2.38E+01±8.38E+00–	2.09E+01±2.52E+00–	1.84E+01±4.43E+00
f_{10}	8.70E+02±2.50E+02–	1.18E–01±1.62E–01–	4.04E–02±2.02E–02–	9.72E–02±5.71E–02–	3.41E–01±1.37E–01–	3.33E–02±1.94E–02
f_{11}	2.64E+02±2.77E+01–	4.57E+02±7.97E+01–	0.00E+00±0.00E+00+	4.66E+01±1.20E+01–	3.29E+01±1.12E+01–	8.69E+00±3.47E+00
f_{12}	3.19E+02±2.09E+01–	4.54E+02±7.67E+01–	6.14E+01±1.21E+01–	5.60E+01±1.48E+01–	9.14E+01±2.51E+01–	2.62E+01±1.14E+01
f_{13}	3.17E+02±2.49E+01–	5.57E+02±7.81E+01–	8.96E+01±1.63E+01–	1.25E+02±2.89E+01–	1.58E+02±3.40E+01–	5.71E+01±2.40E+01
f_{14}	6.86E+03±3.29E+02–	4.16E+03±4.79E+02–	2.08E–03±6.25E–03+	2.77E+03±7.80E+02–	3.30E+02±3.46E+02+	5.92E+02±3.90E+02
f_{15}	7.49E+03±2.57E+02–	4.22E+03±6.82E+02+	5.15E+03±4.30E+02+	4.05E+03±7.47E+02+	4.16E+03±8.45E+02+	6.28E+03±1.15E+03
f_{16}	2.44E+00±2.69E–01≈	1.45E–01±5.30E–02+	2.34E+00±3.22E–01≈	1.02E+00±5.35E–01+	2.41E+00±2.82E–01≈	2.29E+00±3.76E–01
f_{17}	3.65E+02±4.00E+01–	4.29E+02±8.47E+01–	3.04E+01±1.21E–06+	7.77E+01±1.19E+01–	4.29E+01±5.10E+00+	6.16E+01±8.63E+00
f_{18}	3.70E+02±2.74E+01–	3.54E+02±6.90E+01–	1.54E+02±1.65E+01≈	7.74E+01±1.22E+01+	6.58E+01±1.23E+01+	1.55E+02±3.12E+01
f_{19}	2.63E+02±2.39E+02–	9.15E+00±1.71E+00–	1.68E+00±1.38E–01+	4.76E+00±1.51E+00–	9.42E+00±3.70E+00–	2.51E+00±5.22E–01
f_{20}	1.42E+01±1.82E–01–	1.45E+01±8.72E–02–	1.18E+01±2.60E–01–	1.18E+01±1.06E+00–	1.50E+01±1.34E–01–	1.09E+01±7.33E–01
f_{21}	1.17E+03±2.16E+02–	3.22E+02±8.18E+01–	3.02E+02±8.24E+01–	3.13E+02±8.14E+01–	3.22E+02±6.56E+01≈	2.91E+02±4.41E+01
f_{22}	7.37E+03±2.70E+02–	5.10E+03±8.90E+02–	1.18E+02±3.06E+01+	3.49E+03±8.40E+02–	4.87E+02±3.29E+02–	2.85E+02±1.34E+02
f_{23}	7.87E+03±2.37E+02–	5.25E+03±6.96E+02≈	5.24E+03±4.23E+02≈	4.59E+03±6.86E+02≈	3.96E+03±8.19E+02+	5.13E+03±1.40E+03
f_{24}	3.11E+02±3.24E+00–	3.32E+02±2.71E+01–	2.14E+02±8.24E+00–	2.23E+02±8.38E+00–	2.40E+02±8.41E+00–	2.02E+02±1.35E+00
f_{25}	3.38E+02±2.97E+00–	3.66E+02±1.96E+01–	2.51E+02±1.15E+01+	2.55E+02±2.20E+01–	2.82E+02±1.02E+01–	2.55E+02±7.41E+00
f_{26}	2.16E+02±4.37E+00–	2.89E+02±8.02E+01–	2.04E+02±2.21E+01–	2.04E+02±2.32E+01–	2.00E+02±3.44E–02–	2.00E+02±1.57E–03
f_{27}	1.36E+03±3.24E+01–	1.21E+03±1.01E+02–	6.35E+02±1.99E+02–	5.51E+02±9.82E+01–	7.12E+02±6.57E+01–	3.63E+02±4.22E+01
f_{28}	1.84E+03±2.07E+02–	4.41E+03±5.92E+02–	3.00E+02±0.00E+00+	3.00E+02±2.54E–13≈	3.00E+02±1.80E–13≈	3.00E+02±2.25E–13
–	26	24	13	22	17	–
+	0	2	10	3	6	–
≈	2	2	5	3	5	–

Table 4 The result of Friedman test at $D = 30$

Algorithms	Average rankings
MSBSO	2.16
jDE	2.39
MBSO	3.16
hDEBSA	3.82
ASBSO	4.29
BSO	5.68
DE	6.50

on the Fig. 4, the convergence trend of other five algorithms is worse than MSBSO algorithm. It can be seen from Fig. 4, that the convergence speed of MSBSO is relatively stable, and has been maintained until the later stage.

Taking all the above analysis into consideration, it can be easily found that the MSBSO algorithm is more competitive than the other six algorithms mentioned in this paper. That's to say, the MSBSO algorithm has dramatically improved the performance of original BSO. This may be due to the introduction of various strategies in MSBSO, which increases the adaptability of the BSO algorithm. Moreover, MSBSO combined with the idea of differential mutation and dynamically adjusting the CP value makes the mutation operation of algorithm more effective. However, it should not be underestimated that appropriate individual selection rules play an important role in improving algorithm performance, which will be proved in Section 5.4. In the next section, the efficacy of exploration of MSBSO will be discussed in detail through a special experiment. In addition, the convergence curves of MSBSO is also discussed in the next section.

5.3 Efficacy of exploration

Many papers (including some excellent papers) have only presented experimental results and did not provide adequate discussion (neither from theoretical perspective, nor general discussions) on merits of the proposed approach [49]. The lack of this necessary discussion may lead to the inability to identify which component or approach in the algorithm is responsible for better performance. Therefore, different analytical types of experiments have been added in this section (and Sections 5.4 and 5.5) to analyze which component or approach contributes to the performance improvement of the algorithm. At the same time, these analytical types of experiments can better reflect the advantages and disadvantages of the newly proposed components or approaches. To illustrate the effectiveness of MSBSO exploration more specifically, a practical experiment of Rastrigin's function from [50] is introduced in this section. Before conducting this experiment, a concept

of basin is defined as a search area which includes some individuals tending to the same local optimum. Each basin contains only one local optimum. As described in Section 4.1, each component (each dimension) of the local optimum approximates an integer, so there are 11^2 basins in two dimensional Rastrigin's function with search range $[-5.12, 5.12]$. When the dimension of the Rastrigin's function increases to D dimensions, there are 11^D basins in search space. Therefore, the process of seeking global optimum for multimode function, especially for Rastrigin's function, is to continuously search for more suitable basins that contain better local optimum. The global optimal solution exists in the best basin in the search space. In addition, the fitness difference between random individual and local optimum is defined as the "height" of this random individual. Both this random individual and its local optimum are in the same basin. With the above analysis of the characteristics of multimode functions, this experiment is mainly to verify that when one individual is more closer (the lower the height becomes) to the its basin local optimal solution (local optimum), it is more difficult to find a better basin for this individual.

It is necessary to find a local optimal solution $X_i = -3$ ($i=1,2,\dots,30$) at the beginning of the experiment, because this solution has a medium fitness in the search space. Then, a initial random individual (X') is selected from the basin where the local optimal solution (X) is located. Besides, 10,000 individuals are obtained by uniform sampling from the search space. Subsequently, the distance between the corresponding dimensions of the X' and X are divided into 11 equal parts (e.g. $(X_5 - X'_5)/11$), and each equal part is a uniform step size. In this experiment, the process of continuous optimization is simulated by continuously accumulating the uniform step size in the corresponding dimensions of X' . The X' is bound to reach X after 11 iterations. After each iteration, we will update X' and record the height ($f(X') - f(X)$). In addition, we will resample 10,000 individuals (new individuals) and obtain the basins corresponding to the new individuals at the beginning of each iteration. These basins with a better local optimal solution than X are defined as better basins. In all better basins, sample individuals better or worse than X' are called better or worse individuals respectively. The ratios of better or worse individuals will be recorded after each iteration. The experimental results of 30 D Rastrigin's function are recorded in Fig. 5a. Additionally, the interval between each two circles in Fig. 5a represents a uniform step size.

From Fig. 5a, the effectiveness of the exploration in this simulation experiment can be observed very intuitively. In the first iteration, the "better ratio" is nearly 70%, which means that 70% individuals in all better basins are better than X' . This situation is beneficial for the algorithm to find better basins by searching better individuals. In the

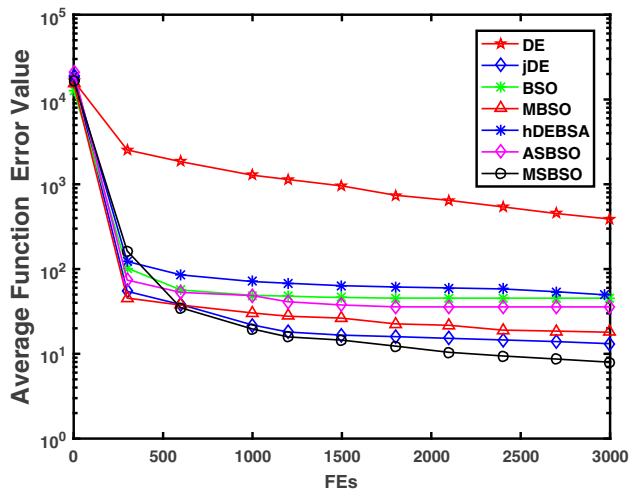
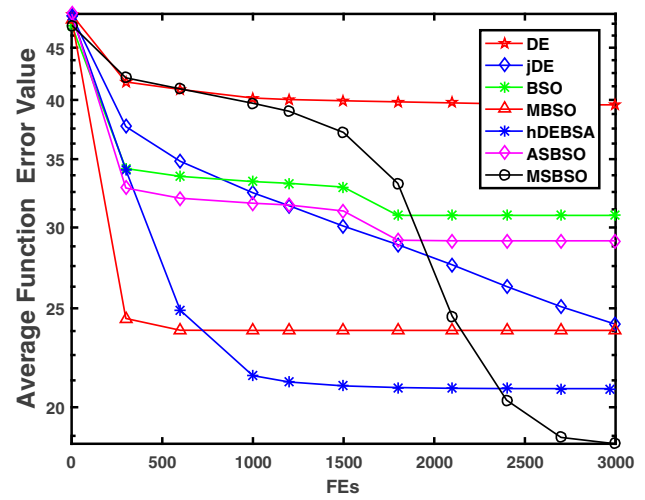
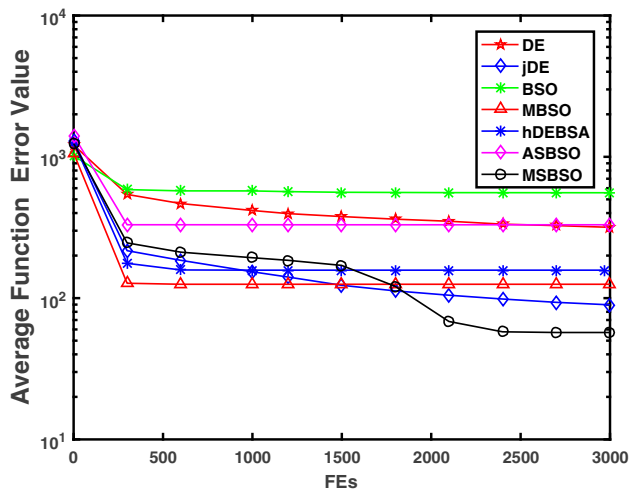
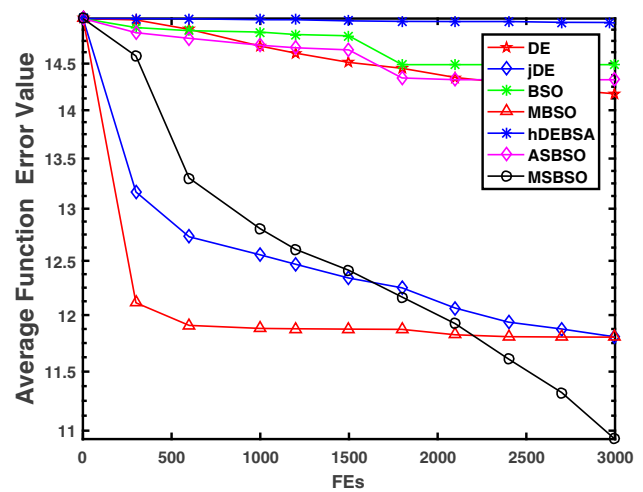
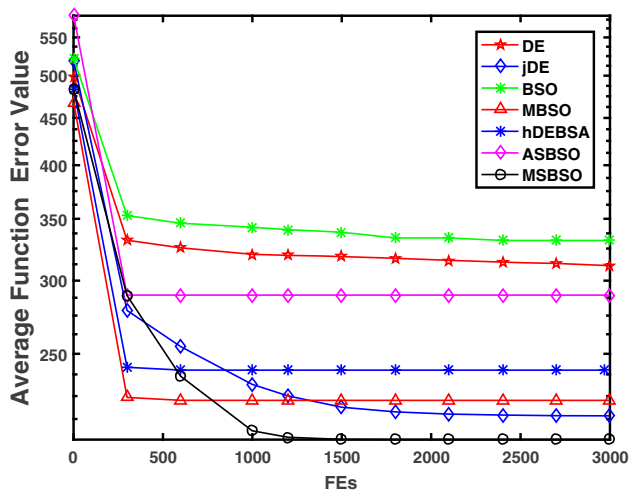
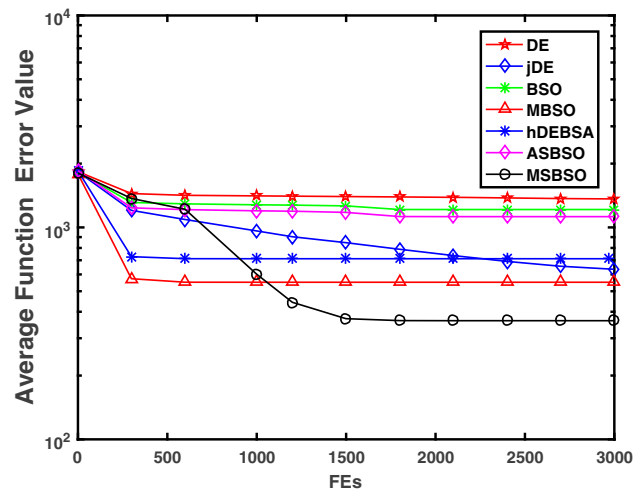
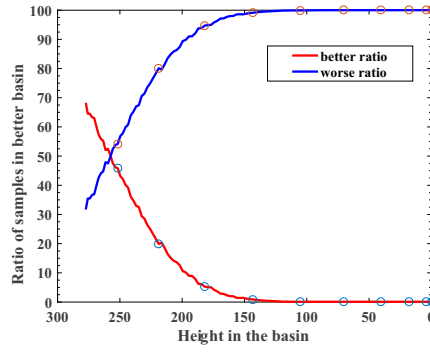
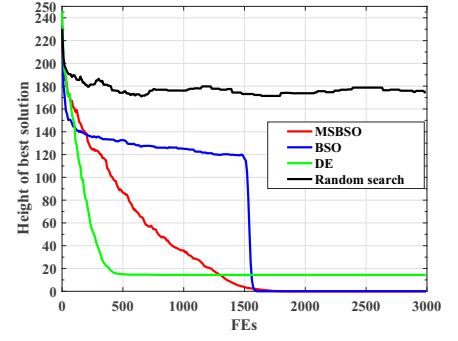
(a) f_6 (b) f_9 (c) f_{13} (d) f_{20} (e) f_{24} (f) f_{27} Fig. 4 Six representative convergent graphs of MSBSO and competitors at $D=30$

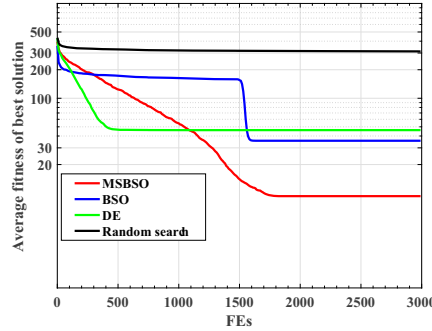
Fig. 5 Exploration performance diagrams of MSBSO and related algorithms based on 30-dimensional Rastrigin's function



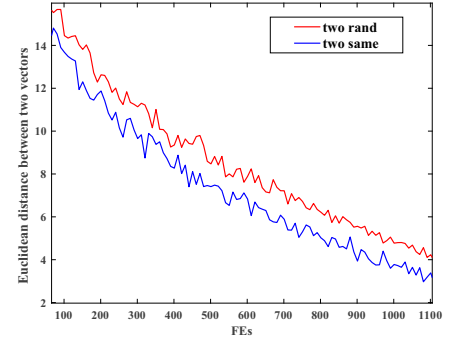
(a) When random individual gets closer to the local optimal solution in its basin, the probability that the random individual will find better individual from a better basin is smaller. The interval between circles in the graph represents one eleventh of the distance from the initial random individual to its local optimum.



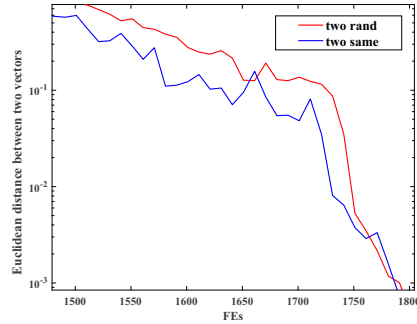
(b) The height of the best individual (solution) obtained so far in its basin based on 30 dimensional Rastrigin's function. As the search continues, the best solution is getting closer to the local optimal solution in its basin.



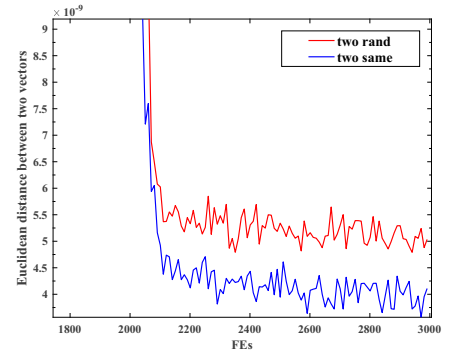
(c) The average fitness value of 30 independent tests of DE, BSO, Random search and MSBSO on a 30-dimensional Rastrigin's function.



(d) Two Euclidean distance curves of two random individuals derived from the different clusters and two individuals derived from the same cluster, these curves all based on the 30-dimensional Rastrigin's function.



(e) Two Euclidean distance curves of two random individuals derived from the different clusters and two individuals derived from the same cluster, these curves all based on the 30-dimensional Rastrigin's function.



(f) Two Euclidean distance curves of two random individuals derived from the different clusters and two individuals derived from the same cluster, these curves all based on the 30-dimensional Rastrigin's function.

second iteration, the height drops rapidly to 250 while the X' has moved only 9.09% of the distance towards X . However, the “worse ratio” exceeds the “better ratio” when height ≤ 250 , which is not beneficial for the algorithm to find a better basin. What's worse is that the “better ratio”

almost becomes zero when the height drops to 150, while X' moves only 36.36% of the distance toward X . Under these conditions, it is difficult for one random individual to move from the current basin to a better basin. The above results are mainly caused by the greedy selection (e.g.

the algorithm uses information from excellent individuals to generate new individuals or selects elite individuals to remain in new populations.). For example, if X' and 10,000 sample individuals are taken as the current individual and the newly generated individuals respectively. Then, the algorithm will have a larger probability of selecting better individual from better basins to retain and replace X' when $\text{height} > 200$. This probability is greatly reduced when $\text{height} < 150$, so the current individual is unlikely to move to a better basin. During iteration, the algorithm always retains better individuals. However, the more better individuals are retained, the more likely the current best individual is to approach its basin local optimal solution. This may make the current individuals refuse to move to other better basins.

Since random search, DE, BSO and MSBSO all adopt greedy selection, the height value is used to discuss the exploration efficacy of these algorithms. The random search algorithm adopt pure random search techniques to explore the search space, which allows it to eliminate part of the effects of greedy selection. Nevertheless, the height of random search shown in Fig. 5b drops below 180 when the number of function evaluations exceeds 50,000. It can be seen from Fig. 5a that when the height drops to 180, the value of “better ratio” will be lower than 20%. Meanwhile, the random search algorithm in Fig. 5c quickly falls into local optimization due to the limitation of strategy performance. The parameters and mutation strategy (DE/rand/1/bin) of DE algorithm used in this experiment are derived from [51]. The strategy of DE takes the difference vectors of two random individuals as the step size. Obviously, the difference between the two random individuals will become smaller as the population iterates continuously, so that the step size will also become smaller. Once the population retains one current best individual with the height below 20, it is almost impossible to find a better individual from better basins by appending a smaller step size to a random individual. As shown in Fig. 5b, when the number of function evaluations exceeds 50,000, the height of the best individual in DE population is less than 20. Figure 5c also shows that when the number of evaluations exceeds 50,000, the best individual of DE algorithm falls into local optimum and cannot move to a better basin. The above analysis shows that DE algorithm is seriously affected by greedy selection.

Additionally, an experiment has been added to better explore the efficacy of exploration of the algorithms. The fitness of the seven best individuals and their corresponding local optimal solutions (local optima of basins) are recorded in Table 5. According to Table 5, the DE algorithm obtains a individual close to the local optimal solution after 50,000 function evaluations, and the fitness of the local optimal solution does not change as the number of the function evaluations increases. This indicates that the mutation

Table 5 The fitness of current best individual and its basin local optimum in MSBSO, BSO, and DE based on Rastrigin’s function at 30 dimensions

FEs*100	DE		BSO		MSBSO	
	best	opt	best	opt	best	opt
50000	42.94	34.82	170.29	42.94	128.44	29.84
100000	42.93	34.82	161.54	41.58	63.90	18.90
150000	42.93	34.82	152.42	37.5	29.88	14.92
170000	42.93	34.82	33.66	33.66	14.00	9.94
200000	42.93	34.82	33.66	33.66	8.95	8.95
250000	42.93	34.82	33.66	33.66	8.95	8.95
300000	42.93	34.82	33.66	33.66	8.95	8.95

strategy of DE is more inclined to find a better individual in the same basin rather than in other better basins. For BSO, cluster centers play a guiding role in searching for better basins. So the (3) and (5) in Section 2.2 are mainly used to quickly find high-quality basins in the multimodal function search space, while the (2) and (4) are used to random search for other better basins. As can be seen from Fig. 5c, when the number of evaluations is less than 10,000, BSO achieve better results than DE and MSBSO. However, as can be seen from Fig. 5b, this result is not absolutely beneficial to BSO, because the height of BSO in Fig. 5b decreases faster than that of DE and MSBSO when the evaluation of function is less than 10,000 times. As a result, the convergence rate of the best solution of BSO in Fig. 5c is rather slow in the interval of more than 10,000 evaluation times and less than 150,000 evaluation times. Since the BSO finds a better basin faster and the height of the BSO solution in Fig. 5(b) is below 140 after 10,000 function evaluations, this will make it very difficult for BSO to move the current best solution to a better basin by attaching one step to one or two random solutions ((2), (4)). In this case, only (3) and (5) can be used to find a better basin. But as shown in Table 5, when the number of evaluations is from 50,000 to 150,000, the local optimal individual fitness in the current best basin of BSO is decreased by 5.44, which is far less than MSBSO decreased by 14.92.

Before explaining the efficacy of exploration of MSBSO, it is necessary to explain how MSBSO focuses on global search in the early iteration period and local search in the later iteration period. In (12), the value of parameter P decreases exponentially from 1 to 0. According to the change rule of P , it can be concluded from 7_{th} line of Algorithm 1 that (7) and (9) are given priority in the early iteration stage, while (8) and (10) are given priority in the later iteration stage. If the difference between two vectors is taken as step size, then the step size of (7) is the difference vector between two individuals which are selected from

different clusters, and the step size of (9) and (10) is the difference vector between two individuals which are select from the same cluster. Since each individual contains 30 dimensions, the Euclidean distance between two individuals is replaced by the difference vector. Figure 5d-f shows the Euclidean distance curves of two individuals from different clusters (two rand) and two individuals from the same cluster (two same) in the whole iteration process. It can be seen from Fig. 5d-f that the Euclidean distance (two rand) acting on (7) is always larger than that (two same) acting on (9) and (10). Large step size is beneficial to global search, while small step size is just the opposite. As described in Section 4.2, (9) is only a complementary strategy with a less than 50% chance of being selected. In summary, MSBSO focuses on global search in the early iteration stage and local search in the later iteration stage.

Combining with the previous analysis, MSBSO mainly adopts strategy (7) and complementary strategy (9) in the early iteration stage. The former strategy mainly carries out global search around excellent individuals, while the latter strategy carries out local and meticulous search around a cluster center. The above situations are necessary conditions for the population to find a better basin. As a parameter (CP) to control the degree of individual variation, it increases with the number of iterations, which means that its value is relatively small at the beginning of iteration. The small CP value causes the mutation strategy to act on a small fraction dimensions of the individual, so that the new individual does not approach the local optimum quickly, thus giving the individual a chance to move from a good basin to a better basin. This is a sufficient condition for the population to quickly find a satisfactory and better basin.

As shown in Table 5, MSBSO finds a basin containing a local optimal solution with very small fitness when the number of function evaluations is 50,000. This basin is obviously better than what BSO and DE found. For multimodal functions, when the best individuals in the population are fairly close to the local optimum in a basin, it is difficult to find a better basin by random search, which can be proved by the convergence graph of DE and random search algorithm in Fig. 5c. MSBSO has improved this situation. As shown in Table 5, when the number of function evaluations increases from 150,000 to 170,000, although the height of the best individual of MSBSO is as low as 14.96 (best-opt), the fitness of the local optimal solution in the basin is still reduced by 4.98. The best individual height of BSO is 114.92 when the number of evaluations is 150,000 times, but the fitness of the local optimal solution is only decreased by 3.84 when the number of evaluations rises to 170,000. It can also be seen from Table 5 that MSBSO still finds a basin containing a better local optimal solution when the number of function evaluations increases from 170,000

to 200,000, but the best individual of BSO is completely trapped in local optimum at this time.

According to the description above, MSBSO can still find better basins in the later iteration. MSBSO focuses on local search around the best individual found so far by using (10) at the later iteration period, and CP has increased to a larger value at this time, which makes the mutation strategy works on most dimensions of one individual. Based on the above situation, the best individual has opportunity to move to a better basin nearby. Unfortunately, MSBSO is still slightly affected by greedy selection. From Figs. 5c and 4e-f, it can be seen that the convergence curves of MSBSO also has a stagnant area. It can also be seen from Table 5 that the convergence rate of MSBSO will also stagnate in the later stage of iteration, which is mainly due to the fact that MSBSO algorithm utilizes the information of excellent individuals in the process of generating new individuals, but this situation is not unsolvable, and how to solve this situation will be discussed later. Combined with the above analysis, MSBSO mainly finds the basin that is as satisfactory as possible through global search in the early iteration period, and only finds a basin that is slightly better than the current basin by local search in the late iteration period. In this case, we propose a hypothesis that if the population is given enough chance to find the most potential basin in the early iteration period, the global optimal individual can be found only by local search around the most potential basin in the late iteration period. The role of CP has been discussed above, which can control the degree of individual variation. The value of CP in MSBSO rises from 0.7 to 0.9, and we expect to verify the previous hypothesis by changing the value of CP from 0.2 to 0.9. At the beginning of the iteration, the value of CP is as low as 0.2, which means that the mutation strategy only works on 20% of the dimensions of each individual. This also means

Table 6 The fitness of the current best individual and its basin local optimum in MSBSO and MSBSO-4 based on Rastrigin's function at 30 dimensions

FEs*100	MSBSO		MSBSO-4	
	best	opt	best	opt
50000	128.44	29.84	66.82	25.86
100000	63.90	18.90	12.11	4.97
150000	29.88	14.92	5E-07	0
170000	14.00	9.94	2E-12	0
200000	8.95	8.95	0	0
250000	8.95	8.95	0	0
300000	8.95	8.95	0	0

that the algorithm has enough opportunity to search for the most potential basin at the beginning of the iteration.

The new formula of CP is located at the end of this paragraph. The algorithm with a different CP value from MSBSO named MSBSO-4. Subsequently, the two algorithms are tested with Rastrigin's function, and the results of the two algorithms are recorded in Table 6. As shown in Table 6, MSBSO-4 can quickly find a basin with high potential in the early iteration period, and finally find a global optimal solution. This proves our previous hypothesis, and also shows the considerable potential of MSBSO in solving optimization problems. Based on all the analysis in Section 5.3, MSBSO is superior to DE, BSO and random search algorithm in efficacy of exploration. Of course, the experiments in this section are for multimodal functions, and "height" is only a performance reference index for algorithms using non-random sampling method.

$$CP = 0.9 - \exp(1 - \frac{G}{G-g+1}) \times 0.7 \quad (16)$$

5.4 Verification of individual selection rules

In this section, a comparative experiment is designed to demonstrate the contribution of individual selection rules, which can also prove that the superior performance of MSBSO is not only due to the combination of different strategies. The four experimental objects selected in this experiment are DE, MBSO, jDE and MSBSO-3. The parameters of MBSO algorithm and jDE algorithm have been described in Section 5.1. In this part, the strategy (17) is adopted by DE, and (17) can be extended to four different schemes (18), (19), (20) and (21) according to four different individual selection rules. These four different schemes are designed for MSBSO-3, and all these schemes are based on (17). In order to avoid the influence of dynamic

parameter adjustment on the algorithm, the CP of MSBSO-3 was set as 0.9, and the S of MSBSO-3 was set as 0.7. Similarly, the crossover probability CR and scaling factor F of DE algorithm are set to 0.9 and 0.7, respectively. The population size and dimension size of DE and MSBSO-3 algorithms can be referred to Table 1. In MSBSO-3, four different schemes are used to replace the four strategies in MSBSO, and the CP and S are set as constants. The P_{global} and P_{local} of MSBSO-3 are set to 0.8 and 0.6 respectively. In addition to the above differences, other aspects of MSBSO-3 are the same as MSBSO. The process of selecting different schemes based on P value is described in Fig. 6. In Fig. 6, the size of the rectangle represents the probability of selecting scheme. The specific strategies and schemes are as follows:

DE/* / rand-to-best

$$V_{i,d} = X_{best,d} + S \times (X_{r1,d} - X_{r2,d}) \quad (17)$$

Scheme1: MSBSO/three / rand-to-center

$$V_{i,d} = X_{center,d} + S \times (X_{i1,d} - X_{i2,d}) \quad (18)$$

Scheme2: MSBSO/two / rand-to-best

$$V_{i,d} = X_{best,d} + S \times (X_{p1,d} - X_{p2,d}) \quad (19)$$

Scheme3: MSBSO/one / rand-to-center

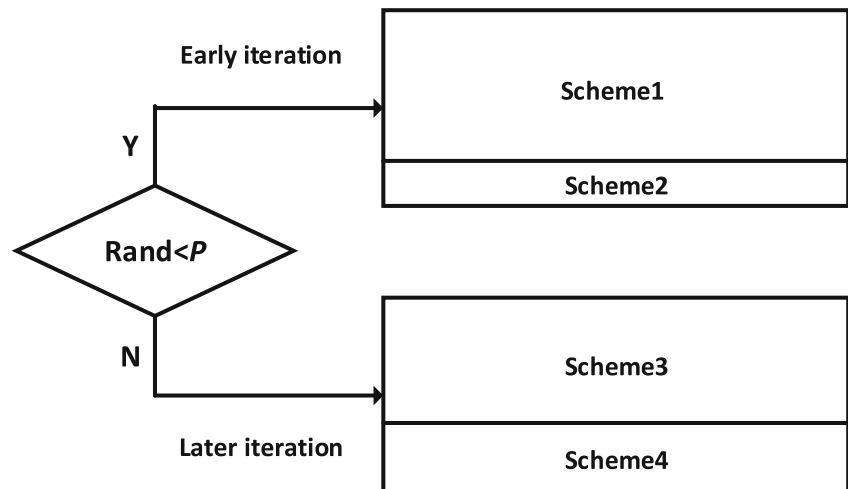
$$V_{i,d} = X_{p_center,d} + S \times (X_{p1,d} - X_{p2,d}) \quad (20)$$

Scheme4: MSBSO/three / rand-to-best

$$V_{i,d} = X_{best,d} + S \times (X_{i1,d} - X_{i2,d}) \quad (21)$$

where, "*" means that the DE algorithm does not perform clustering, so the individuals selected in the strategy come from the same population, X_{best} is the best individual in the current population, $r1$ and $r2$ refer to two random individuals in the current population. p_center , $p1$ and $p2$ represent three individuals from the same one cluster in which p_center is the cluster center, $p1$ and $p2$ represent

Fig. 6 Scheme selection based on P



two individuals from this cluster. X_{center} is a random cluster center, $i1$ and $i2$ represent two individuals from two different clusters.

The experimental results of this part are shown in Table 7. According to Table 7, the performance of DE algorithm with rand-to-best strategy is much better than that of the original DE algorithm. The schemes for MSBSO-3 also originate from rand-to-best strategy. Obviously, the test results of MSBSO-3 are much better than DE algorithm with rand-to-best strategy in most functions. Among the mean solutions of 28 benchmark functions, MSBSO-3 has more than 20 solutions which are better than MBSO. In addition, it is easy to find that the comprehensive performance of MSBSO-3 is slightly better than that of jDE according to Table 8. In summary, it can be proved that different

individual selection rules have made a great contribution to the improvement of algorithm performance, and it also verifies the effectiveness of the mechanism of generating new individuals by MSBSO.

5.5 Impact of dynamic parameter CP

Proper adjustment of parameters can achieve a satisfactory effect on the convergence accuracy. For the purpose of illustrating the practical effect of CP adjustment, we compare the experimental data of two different versions of MSBSO. One is called MSBSO-1 which has no parameter adjustment operation compared with MSBSO-2. The other is called MSBSO-2 which is equal to the MSBSO algorithm. To add that, because MSBSO-1 lacks parameter adjustment

Table 7 Experimental results of DE, MBSO, jDE, MSBSO-3 on CEC 2013 benchmark tests at $D=30$ and comparison results of these algorithms based on Wilcoxon's rank sum test

F	DE(Mean \pm Std)	MBSO(Mean \pm Std)	jDE(Mean \pm Std)	MSBSO-3(Mean \pm Std)
f_1	2.05E-13 \pm 6.82E-14-	1.59E-13 \pm 1.04E-13-	0.00E+00 \pm 0.00E+00+	4.55E-14 \pm 9.09E-14
f_2	7.51E+04 \pm 3.42E+04+	6.51E+04 \pm 2.81E+04+	1.12E+05 \pm 7.49E+04-	1.10E+05 \pm 5.62E+04
f_3	1.03E+06 \pm 2.05E+06-	1.96E+06 \pm 3.28E+06-	1.19E+06 \pm 3.61E+06-	1.16E+05 \pm 5.70E+05
f_4	5.10E+00 \pm 6.36E+00+	1.12E+03 \pm 1.53E+03-	2.32E+01 \pm 1.56E+01-	1.28E+01 \pm 1.21E+01
f_5	2.16E-13 \pm 7.96E-14-	1.71E-13 \pm 7.63E-14-	1.02E-13 \pm 3.41E-14+	1.14E-13 \pm 2.94E-14
f_6	2.92E+00 \pm 7.86E+00-	1.80E+01 \pm 2.07E+01-	1.32E+01 \pm 3.71E+00-	1.04E+00 \pm 4.76E+00
f_7	2.18E+01 \pm 1.35E+01-	2.44E+01 \pm 1.60E+01-	2.95E+00 \pm 2.84E+00 \approx	4.29E+00 \pm 4.40E+00
f_8	2.10E+01 \pm 4.31E-02 \approx	2.10E+01 \pm 8.26E-02-	2.09E+01 \pm 4.95E-02 \approx	2.09E+01 \pm 4.82E-02
f_9	1.62E+01 \pm 5.14E+00 \approx	2.38E+01 \pm 8.38E+00-	2.41E+01 \pm 6.31E+00-	1.69E+01 \pm 6.52E+00
f_{10}	4.73E-02 \pm 3.33E-02-	9.72E-02 \pm 5.71E-02-	4.04E-02 \pm 2.02E-02-	2.22E-02 \pm 1.49E-02
f_{11}	5.32E+01 \pm 1.08E+01-	4.66E+01 \pm 1.20E+01-	0.00E+00 \pm 0.00E+00+	3.08E+01 \pm 8.37E+00
f_{12}	7.29E+01 \pm 3.29E+01-	5.60E+01 \pm 1.48E+01-	6.14E+01 \pm 1.21E+01-	4.55E+01 \pm 1.13E+01
f_{13}	1.59E+02 \pm 4.03E+01-	1.25E+02 \pm 2.89E+01-	8.96E+01 \pm 1.63E+01-	8.60E+01 \pm 3.00E+01
f_{14}	1.42E+03 \pm 4.14E+02-	2.77E+03 \pm 7.80E+02-	2.08E-03 \pm 6.25E-03+	9.54E+02 \pm 3.86E+02
f_{15}	7.11E+03 \pm 4.04E+02 \approx	4.05E+03 \pm 7.47E+02+	5.15E+03 \pm 4.30E+02+	7.12E+03 \pm 4.11E+02
f_{16}	2.44E+00 \pm 2.44E-01 \approx	1.02E+00 \pm 5.35E-01+	2.34E+00 \pm 3.22E-01 \approx	2.40E+00 \pm 3.49E-01
f_{17}	1.00E+02 \pm 2.12E+01-	7.77E+01 \pm 1.19E+01-	3.04E+01 \pm 1.21E-06+	6.75E+01 \pm 8.80E+00
f_{18}	2.24E+02 \pm 2.95E+01-	7.74E+01 \pm 1.22E+01+	1.54E+02 \pm 1.65E+01+	1.86E+02 \pm 2.80E+01
f_{19}	4.84E+00 \pm 1.74E+00-	4.76E+00 \pm 1.51E+00-	1.68E+00 \pm 1.38E-01+	3.03E+00 \pm 8.51E-01
f_{20}	1.19E+01 \pm 3.88E-01-	1.18E+01 \pm 1.06E+00-	1.18E+01 \pm 2.60E-01 \approx	1.16E+01 \pm 5.74E-01
f_{21}	3.43E+02 \pm 8.21E+01-	3.13E+02 \pm 8.14E+01 \approx	3.02E+02 \pm 8.24E+01 \approx	3.11E+02 \pm 6.82E+01
f_{22}	1.42E+03 \pm 4.73E+02-	3.49E+03 \pm 8.40E+02-	1.18E+02 \pm 3.06E+01+	1.05E+03 \pm 4.99E+02
f_{23}	7.61E+03 \pm 2.32E+02-	4.59E+03 \pm 6.86E+02+	5.24E+03 \pm 4.23E+02+	7.14E+03 \pm 1.01E+03
f_{24}	2.29E+02 \pm 1.20E+01-	2.23E+02 \pm 8.38E+00-	2.14E+02 \pm 8.24E+00-	2.07E+02 \pm 6.58E+00
f_{25}	2.65E+02 \pm 1.14E+01-	2.55E+02 \pm 2.20E+01-	2.51E+02 \pm 1.15E+01 \approx	2.52E+02 \pm 1.12E+01
f_{26}	2.17E+02 \pm 4.45E+01-	2.04E+02 \pm 2.32E+01-	2.04E+02 \pm 2.21E+01 \approx	2.04E+02 \pm 1.94E+01
f_{27}	6.09E+02 \pm 8.72E+01-	5.51E+02 \pm 9.82E+01-	6.35E+02 \pm 1.99E+02-	4.02E+02 \pm 7.96E+01
f_{28}	4.40E+02 \pm 3.79E+02-	3.00E+02 \pm 2.54E-13 \approx	3.00E+02 \pm 0.00E+00+	3.00E+02 \pm 2.47E-13
-	22	21	10	-
+	4	5	11	-
\approx	2	2	7	-

Table 8 The result of Friedman test at $D = 30$

Algorithms	Average rankings
MSBSO-3	1.91
jDE	2.00
MBSO	2.75
DE	3.34

operation, we fix the CP value to 0.9. The results of this comparative experiment will be in Table 9, and this experiment is also checked by Wilcoxon's rank sum test.

According to Table 9, MSBSO-2 has achieved better results which is slightly better than MSBSO-1. More concretely, MSBSO-2 performs better on eleven test functions in 28 test functions than MSBSO-1, and the

Table 9 Experimental results of MSBSO algorithm without dynamic parameter adjustment on CEC 2013 benchmark tests at $D=30$

F	MSBSO-1(Mean±Std)	MSBSO-2(Mean±Std)
f_1	6.82E−14±1.04E−13−	0.00E+00±0.00E+00
f_2	4.50E+04±2.36E+04 ≈	5.87E+04±3.71E+04
f_3	7.60E+05±2.43E+06−	4.33E+04±1.91E+00
f_4	7.42E+00±4.72E+00+	2.68E+02±1.63E+02
f_5	1.55E−13±6.22E−14−	1.21E−13±2.84E−14
f_6	3.41E+00±2.09E+00+	7.96E+00±3.47E+00
f_7	4.75E+00±4.67E+00−	1.05E+00±7.35E−01
f_8	2.09E+01±5.80E−02 ≈	2.09E+01±5.92E−02
f_9	1.62E+01±4.00E+00 ≈	1.84E+01±4.43E+00
f_{10}	3.05E−02±1.70E−02 ≈	3.33E−02±1.94E−02
f_{11}	1.37E+01±4.26E+00−	8.69E+00±3.47E+00
f_{12}	4.18E+01±1.54E+01−	2.62E+01±1.14E+01
f_{13}	8.78E+01±2.40E+01−	5.71E+01±2.40E+01
f_{14}	3.79E+02±2.05E+02+	5.92E+02±3.90E+02
f_{15}	4.32E+03±1.09E+03+	6.28E+03±1.15E+03
f_{16}	2.36E+00±3.38E−01 ≈	2.29E+00±3.76E−01
f_{17}	4.44E+01±4.27E+00+	6.16E+01±8.63E+00
f_{18}	7.68E+01±2.32E+01+	1.55E+02±3.12E+01
f_{19}	2.31E+00±5.82E−01 ≈	2.51E+00±5.22E−01
f_{20}	1.03E+01±6.92E−01+	1.09E+01±7.33E−01
f_{21}	3.20E+02±7.57E+01 ≈	2.91E+02±4.41E+01
f_{22}	4.18E+02±1.82E+02−	2.85E+02±1.34E+02
f_{23}	4.20E+03±8.98E+02+	5.13E+03±1.40E+03
f_{24}	2.11E+02±5.87E+00−	2.02E+02±1.35E+00
f_{25}	2.56E+02±1.70E+01 ≈	2.55E+02±7.41E+00
f_{26}	2.00E+02±1.38E−03 ≈	2.00E+02±1.57E−03
f_{27}	5.21E+02±1.16E+02−	3.63E+02±4.22E+01
f_{28}	3.00E+02±2.59E−13−	3.00E+02±2.25E−13
−	11	−
+	8	−
≈	9	−

performance of MSBSO-2 is similar to MSBSO-1 on nine test functions, and worse than MSBSO-1 on eight test functions. In summary, adjusting the parameter CP value dynamically is beneficial to improve the solution quality of the test functions. That is because the parameter adjustment measures in this paper increase the variation degree of new individuals as the number of iterations increases, which greatly enhances the exploratory performance of MSBSO at the later period of iteration. In addition, the other important roles of CP have been described in detail in the last four paragraphs of Section 5.3, which will not be repeated here.

5.6 Effect of the extension of dimension

With the purpose of better illustrating the impact of increasing the dimension, this paper makes comparative experiments on MSBSO algorithm, hDEBSA algorithm and BSO algorithm on the basis of $D=50$ and $D=100$, respectively. The Table 10 shows the results of this experiment. When the dimension is 50, the result of 28 test functions on MSBSO algorithm is 25 better than that of BSO algorithm, and the remaining 3 solutions are worse than BSO. When $D=100$, the result of 28 test functions on MSBSO algorithm is 26 better than that of BSO algorithm, only 2 solutions are worse than BSO.

It can be simply said that although the dimension increases, the performance of MSBSO is significantly better than that of BSO algorithm. In addition, we can see from Table 10 that when the dimension increases, the MSBSO algorithm adds one solution among 28 test functions superior than the BSO algorithm. Equally, it is easy to find that when the dimension rises from 50 to 100, the performance of hDEBSA becomes dramatically worse on most test functions, and only four solutions of test function are better than MSBSO in the end.

Comparing the MSBSO algorithm with the most recent algorithm will help to better reflect the adaptability of the MSBSO algorithm in different dimensions. So there is a new comparative experiment on MSBSO and ASBSO. The experimental results are recorded in Table 11. Compared with the latest algorithm ASBSO, MSBSO always has more than 20 solutions of benchmark functions better than ASBSO when the dimension increases from 30 to 100 dimensions. When the dimension rises from 50 dimensions to 100 dimensions, the solutions of 16 benchmark functions of ASBSO dramatically deteriorate, while the solutions of only 12 benchmark functions of MSBSO significantly deteriorate. As can be seen directly from Fig. 7, when the dimension is 100, the convergence accuracy of MSBSO will always exceed the other four algorithms after a certain number of iterations. In summary, we conclude that the performance of MSBSO algorithm is not severely affected by the increase of dimension. Because more than half of the

Table 10 Experimental results of BSO, hDEBSA, MSBSO on CEC 2013 benchmark tests at $D=50$ and $D=100$ and the comparison results of these algorithms based on Wilcoxon's rank sum test

F	D=50	D=100				
	BSO(Mean±Std)	hDEBSA(Mean±Std)	MSBSO(Mean±Std)	BSO(Mean±Std)	hDEBSA(Mean±Std)	MSBSO(Mean±Std)
f ₁	2.53E-06±8.15E-06-	0.00E+00±0.00E+00+	2.05E-13±6.82E-14	2.35E-01±9.25E-02-	7.97E-12±3.17E-11-	4.62E-13±1.49E-13
f ₂	2.27E+06±7.81E+05-	2.97E+06±6.75E+05-	2.19E+05±6.59E+04	9.96E+06±2.16E+06-	9.14E+06±1.64E+06-	8.78E+05±1.94E+05
f ₃	2.22E+08±1.75E+08-	3.23E+08±2.13E+08-	1.91E+06±2.10E+06	1.59E+09±6.19E+08-	9.56E+09±4.00E+09-	4.94E+07±3.86E+07
f ₄	1.78E+04±4.33E+03-	6.48E+03±2.38E+03-	2.37E+03±6.93E+02	8.07E+03±2.22E+03-	1.04E+04±2.40E+03-	3.17E+03±1.27E+03
f ₅	2.46E-02±3.69E-03-	2.94E-12±4.94E-12-	2.31E-13±6.22E-14	1.64E-01±2.10E-02-	1.75E-06±5.13E-06-	5.84E-13±1.16E-13
f ₆	8.30E+01±3.79E+01-	6.68E+01±2.46E+01-	4.34E+01±1.94E-08	2.12E+02±6.48E+01-	2.19E+02±5.38E+01-	1.54E+02±5.40E+01
f ₇	1.94E+02±1.31E+02-	7.10E+01±9.72E+00-	8.66E+00±4.92E+00	1.27E+02±2.35E+01-	4.23E+02±3.38E+02-	2.80E+01±4.95E+00
f ₈	2.11E+01±5.70E-02-	2.11E+01±3.84E-02≈	2.11E+01±3.01E-02	2.13E+01±4.02E-02-	2.13E+01±1.90E-02≈	2.13E+01±2.72E-02
f ₉	5.70E+01±3.62E+00-	4.49E+01±3.60E+00-	3.90E+01±5.86E+00	1.27E+02±4.64E+00-	1.19E+02±6.07E+00≈	1.18E+02±1.80E+01
f ₁₀	1.27E+00±1.38E-01-	1.15E+00±3.93E-01-	7.59E-02±4.04E-02	4.75E+00±7.34E-01-	9.96E+00±3.70E+00-	1.07E-01±4.22E-02
f ₁₁	7.06E+02±1.07E+02-	9.75E+01±2.48E+01-	2.28E+01±4.98E+00	1.91E+03±1.92E+02-	5.41E+02±8.98E+01-	8.16E+01±1.22E+01
f ₁₂	7.95E+02±9.77E+01-	2.04E+02±2.88E+01-	3.98E+01±1.13E+01	2.20E+03±2.55E+02-	7.85E+02±1.09E+02-	1.15E+02±2.08E+01
f ₁₃	9.51E+02±9.54E+01-	3.67E+02±4.09E+01-	1.13E+02±2.46E+01	2.64E+03±1.96E+02-	1.05E+03±1.11E+02-	3.19E+02±4.76E+01
f ₁₄	7.18E+03±7.48E+02-	9.35E+02±5.80E+02≈	8.51E+02±5.17E+02	1.56E+04±1.18E+03-	5.34E+03±1.39E+03-	2.10E+03±5.76E+02
f ₁₅	8.03E+03±7.90E+02+	7.21E+03±9.35E+02+	1.39E+04±3.57E+02	1.54E+04±1.28E+03+	1.49E+04±1.36E+03+	3.06E+04±4.78E+02
f ₁₆	2.71E-01±7.30E-02+	3.35E+00±2.38E-01≈	3.27E+00±3.23E-01	5.84E-01±1.57E-01+	3.91E+00±2.22E-01≈	3.98E+00±2.28E-01
f ₁₇	8.23E+02±1.05E+02-	1.16E+02±2.41E+01≈	1.17E+02±2.16E+01	2.17E+03±2.36E+02-	5.22E+02±8.33E+01-	1.95E+02±2.38E+01
f ₁₈	6.21E+02±8.68E+01-	1.71E+02±3.79E+01+	3.37E+02±4.06E+01	1.53E+03±1.32E+02-	6.67E+02±9.96E+01+	8.72E+02±2.16E+01
f ₁₉	1.80E+01±2.33E+00-	3.66E+01±1.27E+01-	4.17E+00±7.70E-01	4.80E+01±6.70E+00-	3.29E+02±2.04E+02-	1.37E+01±2.10E+00
f ₂₀	2.43E+01±3.02E-01-	2.46E+01±4.08E-01-	2.10E+01±4.39E-01	5.00E+01±0.00E+00-	5.00E+01±0.00E+00+	5.00E+01±2.26E-11
f ₂₁	9.04E+02±3.05E+02-	8.94E+02±3.02E+02≈	8.48E+02±4.00E+02	4.35E+02±4.38E+01-	4.22E+02±1.78E+01-	3.43E+02±4.96E+01
f ₂₂	9.57E+03±1.26E+03-	1.01E+03±6.30E+02-	5.22E+02±2.72E+02	2.20E+04±2.22E+03-	5.85E+03±1.48E+03-	1.78E+03±6.03E+02
f ₂₃	1.09E+04±9.65E+02+	7.62E+03±9.74E+02+	1.38E+04±7.70E+02	2.14E+04±1.69E+03-	1.83E+04±1.53E+03+	3.08E+04±7.14E+02
f ₂₄	4.60E+02±5.54E+01-	3.03E+02±1.16E+01-	2.20E+02±6.44E+00	1.33E+03±4.86E+02-	4.99E+02±2.25E+01-	2.78E+02±1.56E+01
f ₂₅	5.13E+02±3.16E+01-	3.73E+02±1.23E+01-	3.10E+02±1.19E+01	9.46E+02±9.31E+01-	6.46E+02±2.08E+01-	4.48E+02±1.44E+01
f ₂₆	4.18E+02±9.11E+01-	2.89E+02±1.01E+02-	2.30E+02±5.46E+01	6.51E+02±1.79E+01-	5.70E+02±1.37E+01-	3.99E+02±1.38E+01
f ₂₇	2.14E+03±1.62E+02-	1.38E+03±1.06E+02-	7.54E+02±1.35E+02	4.34E+03±2.31E+02-	3.26E+03±1.69E+02-	1.28E+03±1.63E+02
f ₂₈	7.81E+03±7.68E+02-	6.66E+02±9.97E+02-	4.00E+02±3.15E-13	1.90E+04±1.81E+03-	1.08E+04±7.76E+02-	3.25E+03±9.88E+02
-	25	19	-	26	21	-
+	3	4	-	2	4	-
≈	0	5	-	0	3	-

Table 11 Experimental results of ASBSO and MSBSO on CEC 2013 benchmark tests at $D=30$, $D=50$ and $D=100$ and comparison results of these algorithms based on Wilcoxon's rank sum test

F	$D=30$		$D=50$		$D=100$	
	ASBSO(Mean±Std)	MSBSO(Mean±Std)	ASBSO(Mean±Std)	MSBSO(Mean±Std)	ASBSO(Mean±Std)	MSBSO(Mean±Std)
f_1	1.21E-13±1.52E-13-	0.00E+00±0.00E+00	6.18E-03±1.51E-02 ≈	2.05E-13±6.82E-14	2.49E-01±8.47E-01-	4.62E-13±1.49E-13
f_2	1.55E+06±4.18E+05-	5.87E+04±3.71E+04	2.33E+06±7.03E+05-	2.19E+05±6.59E+04	1.25E+07±2.21E+06-	8.78E+05±1.94E+05
f_3	8.47E+07±8.49E+07-	4.33E+04±1.91E+05	2.68E+08±1.52E+08-	1.91E+06±2.10E+06	2.73E+09±1.37E+09-	4.94E+07±3.86E+07
f_4	6.18E+03±2.21E+03-	2.68E+02±1.63E+02	9.03E+03±2.59E+03-	2.37E+03±6.93E+02	3.22E+03±7.48E+02-	3.17E+03±1.27E+03
f_5	6.34E-03±2.93E-03-	1.21E-13±2.84E-14	1.39E-02±1.29E-02-	2.31E-13±6.22E-14	1.98E-01±5.29E-01-	5.84E-13±1.16E-13
f_6	3.58E+01±2.42E+01-	7.96E+00±3.47E+00	8.27E+01±3.21E+01-	4.34E+01±1.94E-08	2.27E+02±5.19E+01-	1.54E+02±5.40E+01
f_7	9.16E+01±3.84E+01-	1.05E+00±7.35E-01	1.18E+02±3.18E+01-	8.66E+00±4.92E+00	9.49E+01±2.02E+01-	2.80E+01±4.95E+00
f_8	2.09E+01±6.59E-02-	2.09E+01±5.92E-02	2.11E+01±4.06E-02-	2.11E+01±3.01E-02	2.12E+01±4.90E-02+	2.13E+01±2.72E-02
f_9	2.91E+01±3.17E+00-	1.84E+01±4.43E+00	5.13E+01±4.30E+00-	3.90E+01±5.86E+00	1.18E+02±6.20E+00 ≈	1.18E+02±1.80E+01
f_{10}	1.07E-01±5.26E-02-	3.33E-02±1.94E-02	9.97E-01±4.44E-01-	7.59E-02±4.04E-02	6.01E+00±2.02E+00-	1.07E-01±4.22E-02
f_{11}	2.18E+02±5.31E+01-	8.69E+00±3.47E+00	6.22E+02±6.83E+01-	2.28E+01±4.98E+00	1.80E+03±1.69E+02-	8.16E+01±1.22E+01
f_{12}	2.24E+02±4.77E+01-	2.62E+01±1.14E+01	7.66E+02±1.06E+02-	3.98E+01±1.13E+01	2.13E+03±2.30E+02-	1.15E+02±2.08E+01
f_{13}	3.30E+02±6.43E+01-	5.71E+01±2.40E+01	8.98E+02±1.20E+02-	1.13E+02±2.46E+01	2.35E+03±2.11E+02-	3.19E+02±4.76E+01
f_{14}	3.78E+03±4.48E+02-	5.92E+02±3.90E+02	6.80E+03±7.80E+02-	8.51E+02±5.17E+02	1.47E+04±9.44E+02-	2.10E+03±5.76E+02
f_{15}	3.78E+03±5.64E+02+	6.28E+03±1.15E+03	7.31E+03±5.56E+02+	1.39E+04±3.57E+02	1.44E+04±1.09E+03+	3.06E+04±4.78E+02
f_{16}	1.42E-01±1.11E-01+	2.29E+00±3.76E-01	1.71E-01±1.53E-01+	3.27E+00±3.23E-01	4.46E-01±3.77E-01+	3.98E+00±2.28E-01
f_{17}	2.28E+02±5.05E+01-	6.16E+01±8.63E+00	4.69E+02±6.13E+01-	1.17E+02±2.16E+01	1.06E+03±1.55E+02-	1.95E+02±2.38E+01
f_{18}	1.98E+02±2.80E+01-	1.55E+02±3.12E+01	3.38E+02±5.11E+01 ≈	3.37E+02±4.06E+01	8.48E+02±1.59E+02+	8.72E+02±2.16E+01
f_{19}	3.78E+00±7.47E-01-	2.51E+00±5.22E-01	1.10E+01±2.76E+00-	4.17E+00±7.70E-01	3.33E+01±8.93E+00-	1.37E+01±2.10E+00
f_{20}	1.43E+01±2.89E-01-	1.09E+01±7.33E-01	2.41E+01±4.08E-01-	2.10E+01±4.39E-01	5.00E+01±2.26E-11	5.00E+01±2.26E-11
f_{21}	3.17E+02±8.28E+01-	2.91E+02±4.41E+01	7.37E+02±4.20E+02 ≈	8.48E+02±4.00E+02	4.43E+02±5.90E+01-	3.43E+02±4.96E+01
f_{22}	4.56E+03±4.62E+02-	2.85E+02±1.34E+02	9.57E+03±1.18E+03-	5.22E+02±2.72E+02	2.14E+04±2.38E+03-	1.78E+03±6.03E+02
f_{23}	5.10E+03±7.71E+02 ≈	5.13E+03±1.40E+03	9.50E+03±1.33E+03+	1.38E+04±7.70E+02	2.10E+04±1.55E+03+	3.08E+04±7.14E+02
f_{24}	2.89E+02±1.09E+01-	2.02E+02±1.35E+00	3.80E+02±1.96E+01-	2.20E+02±6.44E+00	7.15E+02±2.92E+02-	2.78E+02±1.56E+01
f_{25}	3.08E+02±1.01E+01-	2.55E+02±7.41E+00	4.85E+02±2.98E+01-	3.10E+02±1.19E+01	8.57E+02±1.10E+02-	4.48E+02±1.44E+01
f_{26}	2.59E+02±7.69E+01-	2.00E+02±1.57E-03	3.97E+02±9.09E+01-	2.30E+02±5.46E+01	6.21E+02±2.36E+01-	3.99E+02±1.38E+01
f_{27}	1.12E+03±1.05E+02-	3.63E+02±4.22E+01	1.96E+03±1.47E+02-	7.54E+02±1.35E+02	3.80E+03±2.54E+02-	1.28E+03±1.63E+02
f_{28}	6.25E+02±7.88E+02-	3.00E+02±2.25E-13	7.52E+03±5.70E+02-	4.00E+02±3.15E-13	1.79E+04±1.46E+03-	3.25E+03±9.88E+02
-	25	-	22	-	21	-
+	2	-	3	-	6	-
≈	1	-	3	-	1	-

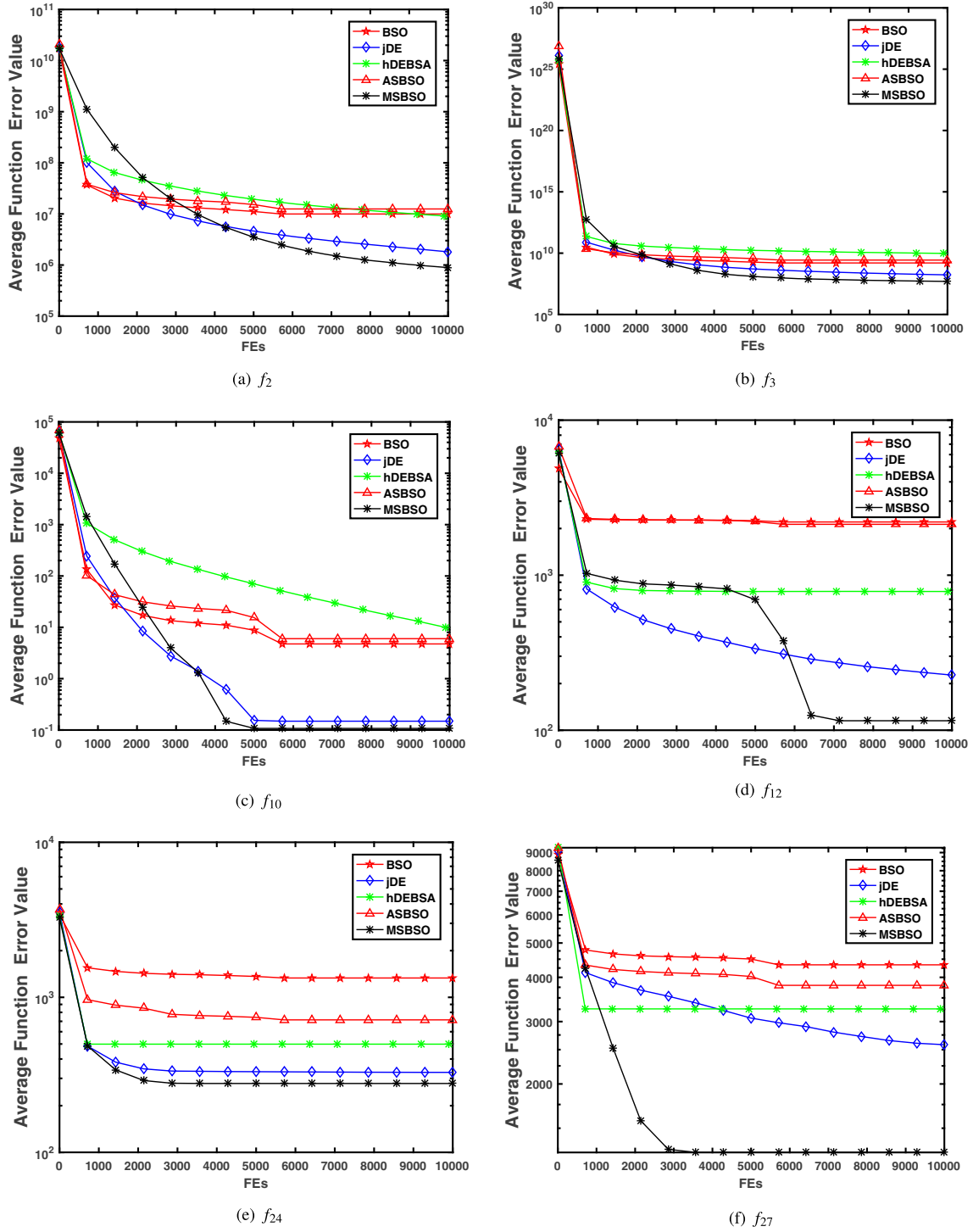
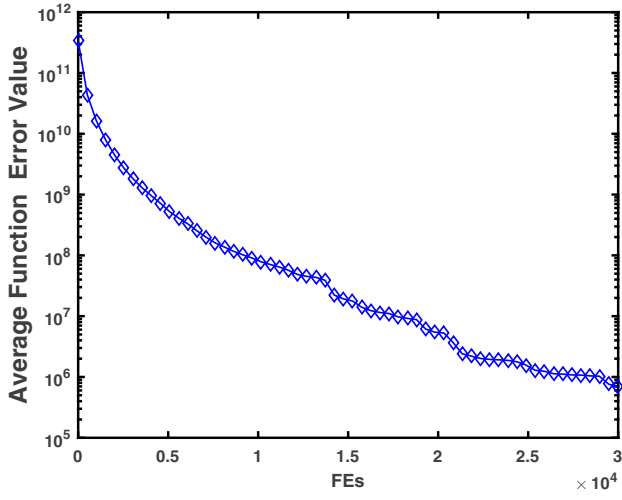
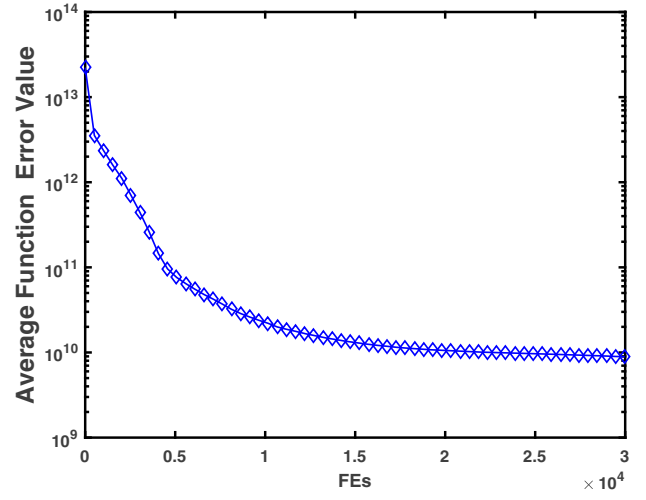
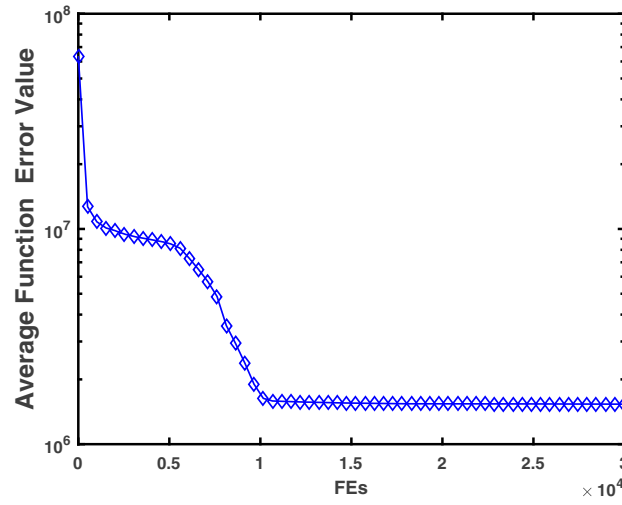
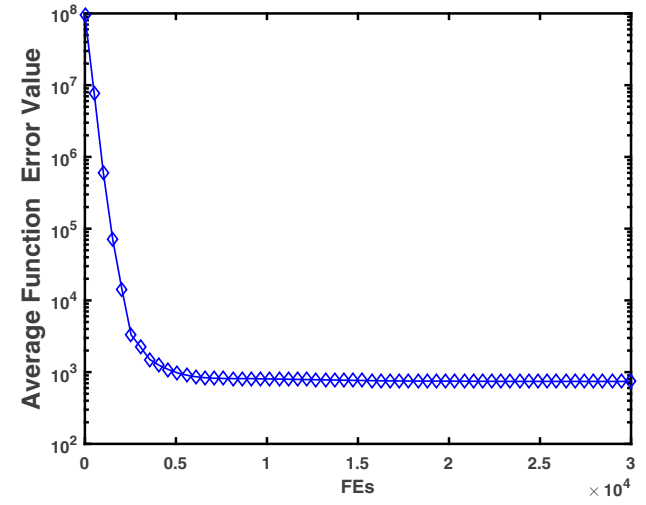
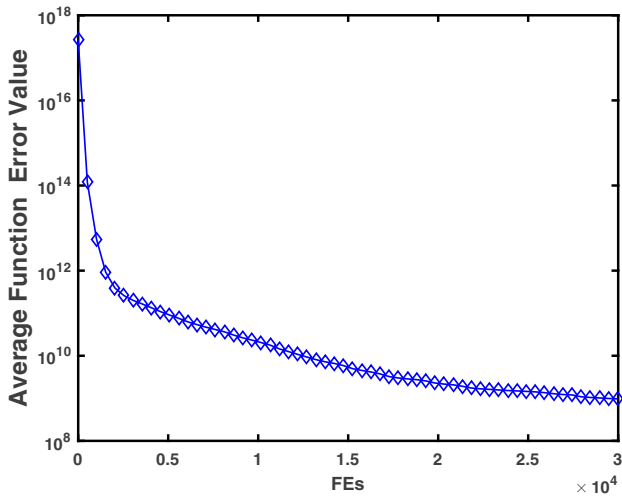
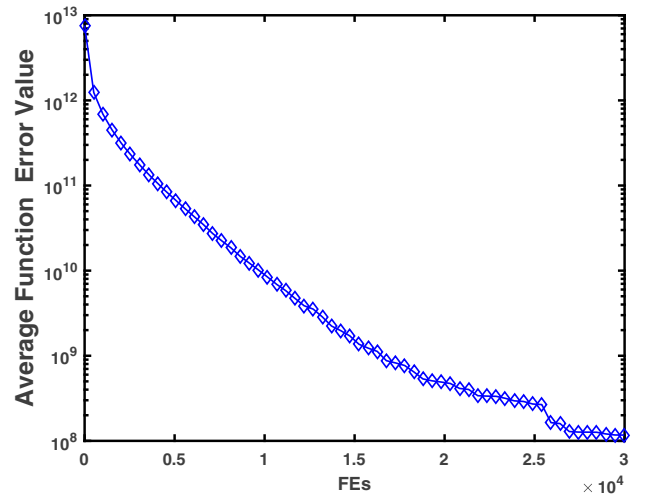


Fig. 7 Six representative convergent graphs of MSBSO and competitors at $D=100$

functions in the CEC 2013 test set are multimodal functions, we only need to explain the reason why the performance of MSBSO is still better than other related algorithms with the increase of dimension in multimodal functions. With the increase of dimension, the difficulty of finding a global optimal solution among many local optimal solutions will

increase dramatically. As shown in Fig. 5c, the performance of DE algorithm using only a single mutation strategy on multimodal functions is not satisfactory, so multiple strategies for different search scopes are designed to increase the adaptability of BSO to multimodal functions. In the early stage of iteration, MSBSO mainly adopts (7)

(a) f_1 (b) f_4 (c) f_5 (d) f_{10} (e) f_{11} (f) f_{12} Fig. 8 Six representative convergent graphs of MSBSO at $D=1000$

and (9) to generate new individuals, and (7) mainly carries out global search around some excellent individuals, while (9) as a complementary strategy of (7) mainly carries out local search around a cluster center. The cooperation of the above two strategies can make the algorithm find more excellent solutions as quickly as possible in the early iteration. In addition, keeping a small CP value in the early iteration period makes these excellent solutions not approach the local optimum quickly, so that these excellent solutions will become more potential. At the later stage of iteration, MSBSO mainly adopts (10) and (8) to generate new individuals, and (10) mainly carries out local search around the current best individual (or individuals with better potential), while (8) as a complementary strategy of (10) mainly carries out global random search. The CP value in the later iteration is relatively large, which makes the variation degree of individuals become larger, so it is conducive for algorithm to adopts (10) to find better solutions near the current best individual. The larger CP value also enables (8) to search other better solutions randomly in a larger range and dimension. The above is an analysis of the reason why MSBSO still performs better than other related algorithms as the dimension increases.

5.7 The large scale global optimization test of MSBSO

Large-scale high-dimensional optimization problems is still a big challenge for some algorithms that perform well in low-dimensional optimization functions. To further discuss the convergence performance of MSBSO in high-dimensional space, a large scale global optimization (LSGO) benchmark adopted by the CEC 2015 special session is used to test MSBSO. This LSGO benchmark set consists of 15 1000-dimensional continuous optimization functions, and the specific information of the these functions is shown in [52]. According to literature [53], most of the algorithms that are competitive on LSGO are based on hybrid approaches or cooperative co-evolution algorithms. Therefore, the MOS-based hybrid algorithm [54], iterative hybridization of DE with local search (IHDELS) and large scale evolutionary optimization using cooperative co-evolution (DECC-CG) are as three experimental subjects in this experiment [55, 56]. Among these three algorithms, the MOS-based hybrid is the winner of the CEC 2015 LSGO competition, the IHDELS and DECC-CG are in second and fourth places. It is worth noting that the experimental results of MOS-based hybrid and IHDELS are taken directly from their corresponding original literature to ensure the authenticity of the experimental data. Similarly, the experimental results of DECC-CG are derived from the literature [54]. In this

experiment, the maximum of fitness evaluation for MOS-based hybrid, IHDELS and DECC-CG are all $3.0e+06$, so does MSBSO. Each function will be run 25 times separately. The test results of the above four algorithms on 1000 dimensions are shown in Table 12. In Fig. 8, there are six convergence graphs of MSBSO on 1000D.

The mean and standard deviation of each algorithm are given in Table 12, and the number of solutions of current algorithm that are worse than the solutions of MSBSO algorithm is also shown at the bottom of Table 12, where the basis for the assessment is the mean value. Among four mean values of one function, the best one is marked in **bold**. It can be observed directly from Table 12 that the overall performance of MSBSO is not as good as that of IHDELS and MOS-based hybrids algorithm. However, MSBSO still has four solutions (f_5 , f_6 , f_9 , f_{10}) whose quality is much better than MOS-based hybrids algorithm and IHDELS. There are two reasons why the performance of MSBSO is worse than the two algorithms mentioned above. One is that IHDELS combines the global exploratory performance of SaDE with the exploitative performance of local search (LS) [57]. In addition, multiple offspring sampling (MOS) is recognized as one of the most advanced frameworks for continuous optimization problems [58], and this framework is a component that BSO does not have. Unexpectedly, the performance of MSBSO on most functions is better than that of DECC-CG algorithm specially designed for large scale global optimization (LSGO). Specifically, 11 solutions among 15 solutions of MSBSO algorithm are better than solutions of DECC-CG algorithm.

The difficulty of optimizing the problem will grow exponentially as the dimension of the problem grows. Hence, DECC-CG uses the new CC framework to split a high latitude problem into multiple low-dimensional problems [56]. So it is not surprising that DECC-CG performs better on the separable functions (f_1 - f_3) than MSBSO according to Table 12. However, the performance of MSBSO on f_6 and f_{10} is several orders of magnitude higher than the other three algorithms. This situation is mainly caused by the following several reasons. According to literature [59], the fitness values of most search spaces in f_{10} are similar, and the value of exploiting generated individuals (existing individuals) information on f_6 and f_{10} is not as good as exploring new solutions [60]. More importantly, the flexible individual selection rules give MSBSO a powerful exploration performance, so MSBSO has a greater chance of exploring new high-quality solutions. Among the six functions shown in Fig. 8, the convergence rates of the other functions (f_1 , f_4 , f_5 , f_{11} , f_{12}) are relatively stable, except that the convergence rate of f_{10} is too fast. Specifically, it can be seen from Fig. 8, the curve of these functions (f_1 , f_4 , f_5 , f_{11} , f_{12}) has been declining until the end of the iteration rather than

Table 12 Experimental results of MOS, IHDELS, DECC-CG and MSBSO on CEC2015 LSGO competition tests at $D=1000$

F	MOS(Mean±Std)	IHDELS(Mean±Std)	DECC-CG(Mean±Std)	MSBSO(Mean±Std)
f_1	0.00E+00±0.00E+00	4.34E−28±1.23E−27	2.03E−13±1.78E−14	8.57E+05±2.05E+06
f_2	8.32E+02±4.48E+01	1.32E+03±6.98E+01	1.03E+03±2.26E+01	1.16E+04±6.08E+02
f_3	9.17E−13±5.12E−14	2.01E+01±1.36E−01	2.87E−10±1.38E−11	1.74E+01±1.99E−01
f_4	1.74E+08±7.87E+07	3.04E+08±1.07E+08	2.60E+10±1.47E+10	9.17E+09±2.92E+09
f_5	6.94E+06±8.85E+05	9.59E+06±2.03E+06	7.28E+14±1.51E+05	1.49E+06±2.56E+05
f_6	1.48E+05±6.43E+04	1.03E+06±1.95E+04	4.85E+04±3.98E+04	4.30E+02±1.21E+03
f_7	1.62E+04±9.10E+03	3.46E+04±1.33E+04	6.07E+08±4.09E+08	2.31E+07±8.77E+06
f_8	8.00E+12±3.07E+12	1.36E+12±6.85E+11	4.26E+14±1.53E+14	9.39E+13±3.48E+13
f_9	3.83E+08±6.29E+07	6.74E+08±1.30E+08	4.27E+08±9.89E+07	1.22E+08±2.18E+07
f_{10}	9.02E+05±5.07E+05	9.16E+07±9.17E+05	1.10E+07±4.00E+06	7.42E+02±1.07E+02
f_{11}	5.22E+07±2.05E+07	1.07E+07±4.12E+06	2.46E+11±2.03E+11	9.71E+08±6.36E+08
f_{12}	2.47E+02±2.54E+02	3.77E+02±3.30E+02	1.04E+03±5.76E+01	1.17E+08±1.81E+08
f_{13}	3.40E+06±1.06E+06	3.80E+06±9.72E+05	3.42E+10±6.41E+09	1.28E+09±4.59E+08
f_{14}	2.56E+07±7.94E+06	1.58E+07±5.11E+06	6.08E+11±2.06E+11	9.31E+09±5.89E+09
f_{15}	2.35E+06±1.94E+05	2.81E+06±1.01E+06	6.05E+07±6.45E+06	4.64E+06±1.70E+06
—	4	5	11	—

reaching a stable stage (or local optimum). That is to say, there is a tendency for these functions to be further optimized. In summary, MSBSO has its unique advantages and potential in LSGO. If MSBSO is combined with an advanced framework or excellent adaptive mechanism [61], it may achieve better performance.

5.8 Computational cost of MSBSO

Obviously, the complexity of the framework of MSBSO is analogous to BSO, the only difference is that the MSBSO algorithm removes the operation of updating the cluster center. So we draw a conclusion that the computation time of MSBSO algorithm is shorter than BSO algorithm in the same dimension. In order to prove this conclusion, MSBSO algorithm and BSO algorithm are tested on CEC 2013 test set at $D = 30$, $D = 50$, $D = 100$, respectively. The MSBSO algorithm and BSO algorithm run 30 times on each test function, meanwhile the average running time and standard deviation of 28 test functions are recorded in Table 13. Considering that the difference of computer performance will affect the running time, all experiments in this paper

are based on a computer of Windows10 operating system with 3.60GHZ Quad-core processor and 8G RAM, which are implemented on matlab R2016b.

As can be seen from Table 13, the running time of MSBSO algorithm is less than that of BSO algorithm on CEC 2013 test functions at $D = 30$, $D = 50$ and $D = 100$. This situation is mainly caused by the following two situations. In BSO, the probability of each cluster being selected is proportional to the number of individuals included in the cluster. However, this probability variable is not used in MSBSO, so MSBSO does not need to calculate this probability. In addition, because MSBSO algorithm removes the operation of updating class centers, it is beneficial to reduce the computational burden. In summary, the MSBSO algorithm is undoubtedly better than BSO algorithm in terms of calculation cost.

6 Conclusion

BSO algorithm performs well in solving global optimization problems, but it also faces inaccuracy and premature convergence. For this reason, two operators are proposed by this paper to ameliorate this situation faced by BSO. Specifically, a multi-strategy mutation operation is introduced in this paper to balance exploitation and exploration of the algorithm and enhance the effectiveness of new individuals, while a dynamic adjustment of CP operation is applied in this paper to adjust the algorithm's exploratory ability so as to increase the possibility of algorithm escaping from local optimum. The experimental results demonstrate the performance of MSBSO algorithm

Table 13 The running time of BSO algorithm and MSBSO algorithm, which is in seconds

D	BSO(Mean±Std)	MSBSO(Mean±Std)
30	6.37E+02±8.49E+00	4.63E+02±7.33E+00
50	1.51E+03±2.00E+01	1.30E+03±1.68E+01
100	5.96E+03±1.17E+02	4.70E+03±5.22E+01

in solving global optimization problems, especially for unimodal functions.

For regard to future work, we will focus on improving the clustering method of MSBSO and adopt distributed platform of Spark computation model to strengthen the solving capability of this algorithm. Besides, the improved clustering method will be combined with Spark computation model, which will make the population evenly divided into five categories. These five categories can evolve in parallel independently. In addition, adding the improved adaptive step size to the original BSO is also the focus of future work.

Acknowledgements This work is supported by the National Natural Science Foundation of China(61763019), the Science and Technology Plan Projects of Jiangxi Provincial Education Department (GJJ161076,GJJ170953,GJJ180891).

References

- Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95 - International Conference on Neural Networks, vol 4, pp 1942–1948
- Karaboga D (2005) An idea based on honey bee swarm for numerical optimization technical report - tr06. Technical Report, Erciyes University
- Peng H, Deng C, Wu Z (2019) Best neighbor-guided artificial bee colony algorithm for continuous optimization problems. *Soft Comput* 23(18):8723–8740
- Storn R, Price K (1997) Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *J Glob Optim* 11(4):341–359
- Dorigo M, Caro GD (1999) Ant colony optimization: a new meta-heuristic. In: Congress on evolutionary computation, vol, 2, pp 1470–1477
- Cai X, Xz Gao, Xue Y (2016) Improved bat algorithm with optimal forage strategy and random disturbance strategy. *Int J Bio-Inspired Comput* 8(4):205–214
- Cui Z, Zhang J, Wang Y, Cao Y, Cai X, Zhang W, Chen J (2019) A pigeon-inspired optimization algorithm for many-objective optimization problems. *Sci China Inf Sci* 62(7):70212
- Shi Y (2011) Brain storm optimization algorithm. In: International conference in swarm intelligence. Springer, pp 303–309
- Duan H, Li S, Shi Y (2013) Predator-prey brain storm optimization for dc brushless motor. *IEEE Trans Magn* 49(10):5336–5340
- Guo X, Wu Y, Xie L (2014) Modified brain storm optimization algorithm for multimodal optimization. In: International Conference in Swarm Intelligence. Springer, pp 340–351
- Jordehi AR (2015) Brainstorm optimisation algorithm (bsoa): an efficient algorithm for finding optimal location and setting of facts devices in electric power systems. *Int J Electr Power Energy Syst* 69:48–57
- Sun C, Duan H, Shi Y (2013) Optimal satellite formation reconfiguration based on closed-loop brain storm optimization. *IEEE Comput Intell Mag* 8(4):39–51
- Cheng S, Sun Y, Chen J, Qin Q, Chu X, Lei X, Shi Y (2017) A comprehensive survey of brain storm optimization algorithms. In: 2017 IEEE Congress on Evolutionary Computation (CEC), pp 1637–1644
- Cheng S, Qin Q, Chen J, Shi Y (2016) Brain storm optimization algorithm: a review. *Artif Intell Rev* 46(4):445–458
- Zhan Zh, Zhang J, Shi Yh, Liu Hl (2012) A modified brain storm optimization. In: 2012 IEEE Congress on Evolutionary Computation (CEC). IEEE, pp 1–8
- Yang YT, Shi YH, Xia SR (2013) Discussion mechanism based brain storm optimization algorithm. *J ZheJiang Univ (Eng Sci)* 47(10):1705–1711
- Chu X, Chen J, Cai F, Chen C, Niu B (2017) Augmented brain storm optimization with mutation strategies. In: Asia-Pacific Conference on Simulated Evolution and Learning. Springer, pp 949–959
- Peng H, Deng C, Wu Z (2019) Spbsso: self-adaptive brain storm optimization algorithm with pbest guided step-size. *J Intell Fuzzy Syst* 36(6):5423–5434
- Nama S, Saha AK (2018) A new hybrid differential evolution algorithm with self-adaptation for function optimization. *Appl Intell* 48(7):1657–1671
- Cheng J, Wang L, Jiang Q, Xiong Y (2018) A novel cuckoo search algorithm with multiple update rules. *Appl Intell* 48(11):4192–4211
- Guo J, Sato Y (2019) A fission-fusion hybrid bare bones particle swarm optimization algorithm for single-objective optimization problems. *Appl Intell* 49(10):3641–3651
- Wang F, Zhang H, Li K, Lin Z, Yang J, Shen XL (2018) A hybrid particle swarm optimization algorithm using adaptive learning strategy. *Inf Sci* 436:162–177
- Cheng S, Lu H, Lei X, Shi Y (2019) Brain storm optimization algorithms: More questions than answers. In: Brain Storm Optimization Algorithms. Springer, pp 3–32
- Zhou D, Shi Y, Cheng S (2012) Brain storm optimization algorithm with modified step-size and individual generation. In: Tan Y, Shi Y, Ji Z (eds) *Advances in Swarm Intelligence*. Springer, Berlin, pp 243–252
- Yang Y, Duan D, Zhang H, Xia S (2015) Motion recognition based on hidden markov model with improved brain storm optimization. *Space Med Med Eng* 28(06):403–407
- Yu Y, Gao S, Wang Y, Cheng J, Todo Y (2018) Asbso: an improved brain storm optimization with flexible search length and memory-based selection. *IEEE Access* 6:36977–36994
- Zhu H, Shi Y (2015) Brain storm optimization algorithms with k-medians clustering algorithms. In: Proceedings of the 7th international conference on advanced computational intelligence (ICACI). IEEE, pp 107–110
- Cao Z, Rong X, Du Z (2017) An improved brain storm optimization with dynamic clustering strategy. In: MATEC Web of conferences, EDP sciences, vol, 95, pp 19002
- Cao Z, Hei X, Wang L, Shi Y, Rong X (2015) An improved brain storm optimization with differential evolution strategy for applications of anns. *Math Probl Eng* 2015(10):1–18
- Hua Z, Chen J, Xie Y (2016) Brain storm optimization with discrete particle swarm optimization for tsp. In: International conference on computational intelligence and security (CIS). IEEE, pp 190–193
- Clerc M (2004) Discrete particle swarm optimization, illustrated by the traveling salesman problem. In: New optimization techniques in engineering. Springer, pp 219–239
- Wang H, Liu J, Yi W, Niu B, Baek J (2017) An improved brain storm optimization with learning strategy. In: International Conference in Swarm Intelligence. Springer, pp 511–518
- Chen J, Shi C, Yang C, Xie Y, Shi Y (2015) Enhanced brain storm optimization algorithm for wireless sensor networks deployment. *Adv Swarm Comput Intell Lect Notes Comput Sci* 9140:373–381
- Liang Zhigang GJ (2018) Medical image registration by integrating modified brain storm optimization algorithm and powell algorithm. *J Comput Appl* 38(9):2683–2688
- Lei Y, Zhang Y (2013) An improved 2d-3d medical image registration algorithm based on modified mutual information and

- expanded powell method. In: 2013 IEEE International conference on medical imaging physics and engineering (ICMIPE). IEEE, pp 24–29
36. Cheng S, Shi Y, Qin Q, Ting TO, Bai R (2014) Maintaining population diversity in brain storm optimization algorithm. *Proc IEEE Congr Evol Comput (CEC)* 4(3):3230–3237
 37. Tang XW, Tang J, Wan S, Tang B (2013) Adaptive differential evolution algorithm with modified mutation strategy and its application. *J Astron* 34(7):1001–1007
 38. Mühlenbein H, Schomisch M, Born J (1991) The parallel genetic algorithm as function optimizer. *Parallel Comput* 17(6-7):619–632
 39. Liang J, Qu B, Suganthan P, Hernández-Díaz AG (2013) Problem definitions and evaluation criteria for the cec 2013 special session on real-parameter optimization. *Comput Intell Lab Zhengzhou Univ Zhengzhou, China Nanyang Technol Univ Singapore Techn Rep* 201212(34):281–295
 40. Brest J, Greiner S, Boskovic B, Mernik M, Zumer V (2006) Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. *IEEE Trans Evol Comput* 10(6):646–657
 41. Yong W, Zhang Q (2012) Enhancing the search ability of differential evolution through orthogonal crossover. *Inf Sci* 185(1):153–177
 42. Peng H, Wu Z (2015) Heterozygous differential evolution with taguchi local search. *Soft Comput* 19(11):3273–3291
 43. Liao T, Stuetzle T (2013) Benchmark results for a simple hybrid algorithm on the cec 2013 benchmark set for real-parameter optimization. In: 2013 IEEE Congress on Evolutionary Computation. IEEE, pp 1938–1944
 44. Auger A, Hansen N (2005) A restart cma evolution strategy with increasing population size. In: 2005 IEEE Congress on evolutionary computation. IEEE, vol, 2, pp 1769–1776
 45. Lourenço H, Martin O, Stützle T (2010) Iterated local search: Framework and applications. In: *Handbook of metaheuristics*, vol, 146, pp 363–397
 46. Hansen N, Müller SD, Koumoutsakos P (2003) Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evol Comput* 11(1):1–18
 47. Peng H, Wu Z, Deng C (2017) Enhancing differential evolution with commensal learning and uniform local search. *Chin J Electron* 26(4):725–733
 48. Guo Z, Liu G, Li D, Wang S (2017) Self-adaptive differential evolution with global neighborhood search. *Soft Comput* 21(13):3759–3768
 49. Bonyadi MR, Michalewicz Z (2017) Particle swarm optimization for single objective continuous space problems: a review. *Evol Comput* 25:1–54
 50. Gonzalez-Fernandez Y, Chen S (2015) Leaders and followers—a new metaheuristic to avoid the bias of accumulated information. In: 2015 IEEE Congress on Evolutionary Computation (CEC). IEEE, pp 776–783
 51. Mezura-Montes E, Velázquez-Reyes J, Coello Coello CA (2006) A comparative study of differential evolution variants for global optimization. In: *Proceedings of the 8th annual conference on Genetic and evolutionary computation*. ACM, pp 485–492
 52. Li X, Tang K, Omidvar MN, Yang Z, Qin K, China H (2013) Benchmark functions for the cec 2013 special session and competition on large-scale global optimization. *Gene* 7(33):8
 53. Molina D, LaTorre A, Herrera F (2018) An insight into bio-inspired and evolutionary algorithms for global optimization: review, analysis, and lessons learnt over a decade of competitions. *Cogn Comput* 10(4):517–544
 54. LaTorre A, Muelas S, Peña JM (2013) Large scale global optimization: Experimental results with mos-based hybrid algorithms. In: 2013 IEEE congress on evolutionary computation. IEEE, pp 2742–2749
 55. Molina D, Herrera F (2015) Iterative hybridization of de with local search for the cec'2015 special session on large scale global optimization. In: 2015 IEEE congress on evolutionary computation (CEC). IEEE, pp 1974–1978
 56. Yang Z, Tang K, Yao X (2008) Large scale evolutionary optimization using cooperative coevolution. *Inf Sci* 178(15):2985–2999
 57. Qin AK, Huang VL, Suganthan PN (2008) Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Trans Evol Comput* 13(2):398–417
 58. LaTorre A (2009) A framework for hybrid dynamic evolutionary algorithms: multiple offspring sampling (mos). *Universidad Politécnica de Madrid*
 59. Cano A, García-Martínez C, Ventura S (2017) Extremely high-dimensional optimization with mapreduce: scaling functions and algorithm. *Inf Sci* 415:110–127
 60. Cano A, García-Martínez C (2016) 100 million dimensions large-scale global optimization using distributed gpu computing. In: 2016 IEEE Congress on evolutionary computation (CEC). IEEE, pp 3566–3573
 61. Mohamed AW, Suganthan PN (2018) Real-parameter unconstrained optimization based on enhanced fitness-adaptive differential evolution algorithm with novel mutation. *Soft Comput* 22(10):3215–3235



Jianan Liu is currently pursuing the B.S. degree in information management and information system at Jiujiang University, Jiujiang, China. His research interests include swarm intelligence algorithm and its real-word applications.



Hu Peng received the B.S. degree in computer science and technology from Hunan Normal University, Changsha, China in 2004 and the M.S. degrees in computer technology from the Huazhong University of Science and Technology, Wuhan, China in 2008. He earned the Ph.D. degree from Wuhan University, Wuhan, China in 2016. He is an associate professor in Jiujiang University, China. His research interests include swarm intelligence algorithm and its real-word applications.



Zhijian Wu received the B.S. degree in mathematics from Jiangxi University, China, in 1983, the M.S. degree from the Department of mathematics, Wuhan University, in 1988, and the Ph.D. degree from the School of Computing, Wuhan University, Wuhan, China, in 2004. Now he is a full Professor with the School of Computer, Wuhan University, Wuhan, China. His current research interests include evolutionary computation, parallel computing, and intelligent information processing.



Changshou Deng received the B.Sc. and M.Sc degrees from Yanshan University, China in 1995 and 2001 respectively. He earned the Ph.D. degree from Tianjin University, China in 2007. He was in the postdoctoral program in Hefei University of Technology. He is a professor in Jiujiang University, China. His research interests include evolutionary algorithm and data mining.



Jianqiang Chen is a postgraduate student in Jiangxi University of Finance and Economics, Jiangxi, China. He received the B.S. degree in computer science and technology from Jiujiang University, Jiangxi, China in 2016. His research interests include swarm intelligence algorithm and its real-world applications.