Neurocomputing 98 (2012) 55–75

Contents lists available at SciVerse ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A novel modified binary differential evolution algorithm and its applications

Ling Wang *, Xiping Fu, Yunfei Mao, Muhammad Ilyas Menhas, Minrui Fei

Shanghai Key Laboratory of Power Station Automation Technology, School of Mechatronics Engineering and Automation, Shanghai University, Shanghai 200072, China

ARTICLE INFO

Available online 6 June 2012

Keywords:
Differential evolution
Binary encoding
Probability estimation operator
Numerical optimization
Multidimensional knapsack problem

ABSTRACT

Differential Evolution (DE) is a simple yet efficient global optimization algorithm. However, the standard DE and most of its variants operate in the continuous space, which cannot solve the binary-coded optimization problems directly. To tackle this problem, this paper proposes a novel modified binary differential evolution algorithm (NMBDE) inspired by the concept of Estimation of Distribution Algorithm and DE. A novel probability estimation operator is developed for NMBDE, which can efficiently maintain diversity of population and achieve a better tradeoff between the exploration and exploitation capabilities by cooperating with the selection operator. Furthermore, the parameter study of NMBDE is run and the analysis is performed to improve the global search ability and scalability of algorithm. The effectiveness and efficiency of NMBDE was verified in applications to the numerical optimization and multidimensional knapsack problems. The experimental results demonstrate that NMBDE has the better global search ability and outperforms the discrete binary DE, the modified binary DE, the discrete binary Particle Swarm Optimization and the binary Ant System in terms of accuracy and convergence speed.

1. 1 Introduction

Differential Evolution (DE), a population-based stochastic optimizer firstly proposed by Storn and Price in 1995 [1], has become a new research hotspot in evolutionary computation. The standard DE algorithm, which is simple yet efficient in global optimization, has been successfully applied in scientific and engineering fields. As a versatile evolutionary algorithm, DE does not require any gradient information so that it is capable of solving non-convex, nonlinear, non-differentiable and multimodal problems. Moreover, there are only two control parameters in the updating formulas of DE, thus it is easy for implementation and parameter tuning. Literatures have reported that DE is superior to Particle Swarm Optimization (PSO) and real-coded Genetic Algorithm (GA) in some real-world applications [2–4].

Due to its simplicity, robustness and effectiveness, DE has attracted more and more attention in recent years, and a number of improved variants have been proposed. First of all, adaptive parameter tuning strategies have been studied and adopted to enhance the performance of DE. Liu and Lampinen [5] proposed a fuzzy adaptive DE which used fuzzy logic controllers to tune control parameters according to the relative fitness values and individuals of the successive generations. Qin et al. [6] proposed a self-adaptive DE algorithm, which enabled DE to switch between four different mutant schemes, and consequently it enhanced the performance of the algorithm. Zhang and Sanderson [7] introduced a new DE by implementing a novel mutation strategy with the optimal external archive and adaptively tuning control parameters. In addition, the hybrids of DE with some deterministic optimization techniques or meta-heuristic algorithms were also well addressed, such as the fusion of DE with the interior point algorithm [8], PSO [9], Harmony Search [10], Clonal Selection algorithm [11] and Estimation of Distribution Algorithm [12]. Hybrid DEs usually outperform the original DE as the differential information provided by DE and the optimization information extracted by the cooperating algorithms are combined and used to guide the search. Besides, some other improved techniques were also presented. For instance, Noman and Iba [13] used an adaptive cross-based local search strategy to accelerate the convergence speed of DE where the length of the local search was adjusted based on a hill-climbing heuristic strategy. This algorithm was reported to outperform the standard DE and improved DEs with other cross-based local search strategies. Rahnamayan et al. [14] proposed an opposition-based DE algorithm using the opposition-based learning for population initialization and generation jumping, in which a new parameter, namely the jumping rate, was introduced and a comprehensive study was performed to verify its effectiveness. Das et al. [15] introduced a modified mutation operator to balance the exploration and exploitation ability of DE by linearly combining two different mutation operators, which is inspired by the neighborhood concept in PSO. So far, DEs have been successfully applied in

* Corresponding author.
E-mail addresses: wangling@shu.edu.cn, shwl_1212@163.com (L. Wang).

0925-2312/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2011.11.033

filter designing [16], power system planning [17], neural network training [11,18], graph theory problems [19], etc. More details can be found in [20,21].

However, the standard DE and most of its improved variants operate in the continuous space, which are not suitable for solving binary-coded combinational optimization problems. Therefore, several binary DE algorithms were proposed to extend the applications of DE. He and Han [22] presented a binary DE based on Artificial Immune System (AIS-DE), in which the scaling factor was treated as a random bit-string and the trial individuals were generated by Boolean operators. An extra threshold parameter was introduced to improve the performance but weaken the flexibility of AIS-DE as it was significantly problem dependent. Later, Wu and Tseng [23] improved the Boolean mutation operator based on the binary bit-string framework and developed a modified binary DE (MBDE) algorithm. Gong and Tuson [24] proposed a binary-adapted DE (BADE) where the scaling factor was regarded as the probability of the scaled difference bit to take on "1". However, AIS-DE, MBDE and BADE discarded the updating formulas of the standard DE and generated new individuals based on different Boolean operators. Inspired by angle modulated PSO algorithm, Pampará et al. [25] proposed a new binary DE named angle modulated DE (AMDE). In AMDE, standard DE was adopted to update the four real-coded parameters of angle modulated function which was used to generate the binary-coded solutions by sampling till the global best solution was found. Thus AMDE actually worked in the continuous space. Chen et al. [26] developed a discrete binary differential evolution (DBDE) where the sigmoid function used in discrete binary Particle Swarm Optimization algorithm (DBPSO) [27] was directly taken to convert the real individuals to bit strings. DBDE searches in the binary space directly so that it is easy to implement, but it is very sensitive to the setting of control parameters. Moreover, the value transformed by the sigmoid function is not symmetrical in DBDE, which results in the deterioration of global searching ability.

In this work, a novel modified binary DE (NMBDE) is proposed which develops a novel probability estimation operator to generate the offspring individuals. On the one hand, NMBDE reserves the updating strategy of the standard DE so that the excellent characteristics of DE such as ease of implementing and tuning parameters are inherited. On the other hand, the proposed probability estimation operator can keep the diversity of population better and is robust to the setting of parameters. Therefore, NMBDE offers a new efficient methodology for discrete binary optimization problems.

This paper is organized as follows. Section 2 gives a brief review of the standard DE. The proposed NMBDE is introduced in detail in Section 3. Then the parameter setting of algorithm is discussed in Section 4. Section 5 presents the applications of NMBDE to the numerical optimization and MKP, where the comparisons of NMBDE with four other binary-coded optimization algorithms, i.e., MBDE, DBDE, DBPSO and the binary Ant System (BAS) [28] on a suite of benchmark functions and MKP instances are conducted. Finally, conclusions are remarked in Section 6.

2. The standard DE

The population of the standard DE consists of a group of floating-point encoded vectors randomly initialized in the continuous space. Three evolutionary operators, i.e., the mutation operator, the crossover operator and the selection operator are commonly used for DE to update the population. In the evolutionary process, the mutation operator and crossover operator are used to generate the new trail individual, and the selection operator chooses the better one between the target individual and its trial alternative for the next generation by comparing their fitness values.

Mutation: There are several mutation schemes in DE, and "*DE/rand/1*" as Eq. (1) is the most popular one.

$$u_{ij}^{t+1} = x_{r1,j}^{t} + F*(x_{r2,j}^{t} - x_{r3,j}^{t}) \tag{1}$$

In Eq. (1), $u_{ij}$is the element of the mutant individual u_i; F is the scaling factor which is a positive constant; t is the index of generation; $x_{r1,j}$,$x_{r2,j}$ and $x_{r3,j}$ are three bits of the randomly chosen individuals with index $r_1 \neq r_2 \neq r_3 \neq i$.

Crossover: The trial individual v_i is generated by crossing the target individual x_i with its mutant counterpart u_i. The widely used binomial crossover is defined as the following equation:

$$v_{ij}^{t+1} = \begin{cases} u_{ij}^{t+1}, & \text{if } (rand\, j \le CR) \text{ or } (j = rand\,(i)) \\ x_{ij}^{t}, & \text{otherwise} \end{cases} \tag{2}$$

where v_{ij} is the element of the trial individual v_i and CR is the crossover probability ranged in (0, 1). The *randj* are a stochastic number uniformly distributed within [0,1); *rand* (*i*) are a random integer within $1, 2, \ldots, N$ where N is the length of individual; j is the index of the dimensionality with $j = 1, 2, \ldots, N$.

Selection: The selection operator is defined as the following equation:

$$x_i^{t+1} = \begin{cases} v_i^{t+1}, & \text{if } f(v_i^{t+1},) < f(v_i^{t},) \\ x_i^{t}, & \text{otherwise} \end{cases} \tag{3}$$

As shown in Eq. (3), the trial individual $v_i$replaces the target individual x_i if its fitness value is better. Otherwise, the target individual is reserved in the next generation. Therefore, the population is updated according to these three operators.

3. Novel modified binary differential evolution

NMBDE adopts the binary coding scheme and each individual is represented by a bit string denoted as $px_i = px_{ij}, |px_{ij} \in 0,1;$ $i = 1,2,\ldots,NP;$ $j = 1,2,\ldots,N$, where NP is the population size and N is the dimensionality of solution. NMBDE reserves the updating formulas of the standard DE, including the mutation operator, the crossover operator and the selection operator. Since the standard mutant operator generates real-coded vectors not bit strings, a new probability estimation operator is proposed to tackle this problem in NMBDE, into which the mutant operator is integrated.

3.1. Probability estimation operator

Inspired by the idea of population based incremental learning (PBIL) algorithm [29], a novel probability estimation operator is proposed and utilized to build the probability model $P(px_i) = (px_{i,1}, px_{i,2}, \ldots, px_{i,N})$, which is used to generate the binary-coded mutated individual in NMBDE. However, different from PBIL, NMBDE constructs the multiple probability models at each iteration according to the information extracted from the parent individuals by using the mutant operator. The probability estimation operator $P(px_i^t)$ is defined as the following equation:

$$\begin{cases} P(px_{ij}^{t+1}) = 1/[1 + e^{-2b*(MO-0.5)/(1+2F)}] \\ MO = px_{r1,j}^{t} + F*(px_{r2,j}^{t} - px_{r3,j}^{t}) \end{cases} \tag{4}$$

where F is the scaling factor; $px_{r1,j}^{t}$,$px_{r2,j}^{t}$ and $px_{r3,j}^{t}$ are the *j-th* bits of three randomly chosen individuals; b, called bandwidth factor, is a positive real constant. As seen, the mutant operator of the standard DE represented as MO is reserved and embedded into

the probability estimation operator in NMBDE. The developed probability estimation operator uses the standard mutation operator to derive the differential information of three parent individuals to construct the probability distribution model of the mutant individual to be bit "1". The bandwidth factor b tunes the range and shape of the probability distribution model, and an appropriate b value can improve the search efficiency and maintain population diversity simultaneously.

Then, the corresponding mutant individual pu_i^{t+1} of the current target individual px_i^t is generated as Eq. (5) according to the probability estimation vector $P(px_i^t)$,

$$pu_{ij}^{t+1}=\begin{cases}1, & \text{if } rand(\,)\le P(px_{ij}^{t+1})\\ 0, & \text{otherwise}\end{cases} \tag{5}$$

where $rand(\,)$ is a random number; $P(px_{ij}^{t+1})$ is the j-th component of the probability vector of the i-th target individual.

The mechanism of NMBDE to produce the mutant individual can be depicted in Fig. 1.

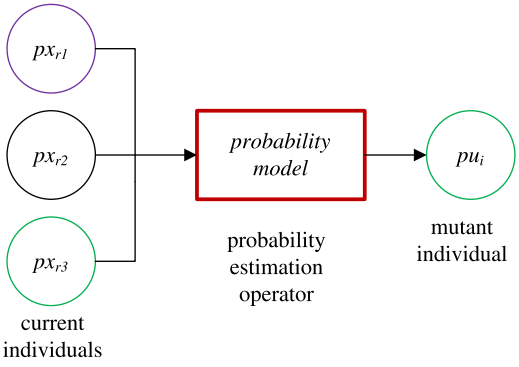


Fig. 1. The operating of probability estimation operator in NMBDE.

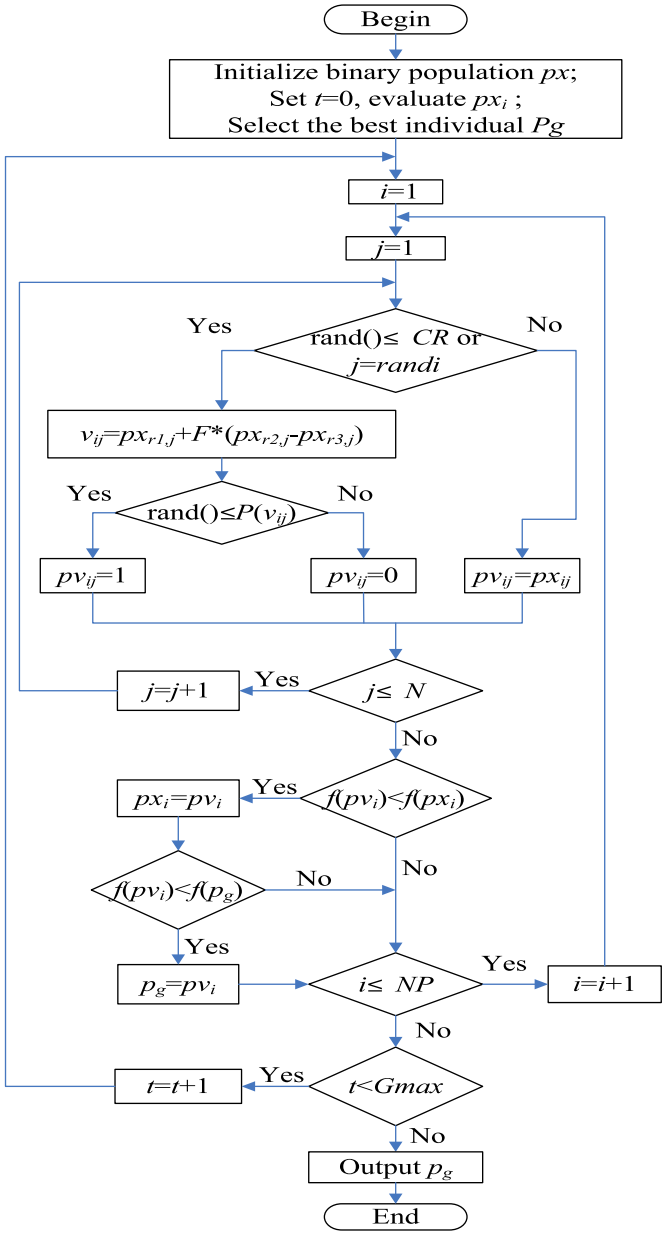


Fig. 2. Flowchart of the NMBDE algorithm.

3.2. Crossover operator

The crossover operator is used to produce the trail individual pv_i=($pv_{i,1}$, $pv_{i,2}$, … , $pv_{i,N}$) by mixing the target individual and its mutant individual. The trial vector pv_i can be obtained according to the following equation:

$$pv_{ij}^{t+1}=\begin{cases}pu_{ij}^{t+1}, & \text{if } (r \text{ and } j\le CR) \text{ or } (j=r \text{ and } i)\\ px_{ij}^{t}, & \text{otherwise}\end{cases} \tag{6}$$

Table 1
Probability of being "1" for the mutant individual of NMBDE with different values of F and b=6.

$px_{r1,j}$	$px_{r2,j}$	$px_{r3,j}$	u_{ij}	$P(u_{ij})$		
				F=0.5	F=1.0	F=2.0
0	0	0	0	0.0474	0.1192	0.2315
0	0	1	$-F$	0.0025	0.0025	0.0025
0	1	0	F	0.5000	0.8808	0.9734
0	1	1	0	0.0474	0.1192	0.2315
1	0	0	1	0.9526	0.8808	0.7685
1	0	1	$1-F$	0.5000	0.1192	0.0266
1	1	0	$1+F$	0.9975	0.9975	0.9975
1	1	1	1	0.9526	0.8808	0.7685

Table 2
Numerical benchmark functions.

No.	Name	Dimension	Minimum	Type
f_1	Rosenbrock	2(30)	0	unimodal
f_2	Goldstein & Price	2	3	unimodal
f_3	Freudenstein–Roth	2	0	unimodal
f_4	Glankwahmdee	2	0	unimodal
f_5	Griewangk	2(30)	0	unimodal
f_6	Schaffer F6	2	−1	multimodal
f_7	Shubert	2	−186.730908	multimodal
f_8	Dixon	2(30)	0	multimodal
f_9	Pathological	2(30)	0	multimodal
f_{10}	Ackley	2	0	multimodal
f_{11}	Levy F5	2	−176.1375	multimodal
f_{12}	Camel Back-6	2	−1.03162845	multimodal
f_{13}	Alpine	2	−7.8856	multimodal
f_{14}	Levy F3	2	−174.5417	multimodal
f_{15}	Beale	2	0	multimodal
f_{16}	Hartman 3	3	−3.86278	multimodal
f_{17}	Sphere function	2(30)	0	unimodal
f_{18}	Sum squares function	2(30)	0	unimodal
f_{19}	Sum of different power	2(30)	0	unimodal
f_{20}	Schwefel's problem 2.22	2(30)	0	unimodal

where r *and* j, CR and r *and* $i \in \{1,2,\ldots,N\}$ are identical to those defined in standard DE. When a random number is less than the predefined CR or the index j is equal to the random index r *and* i, pv_{ij}^{t+1} takes the value of pu_{ij}^{t+1}; otherwise, it is set equal to px_{ij}^{t}. From Eq. (6), it is obvious that at least one bit of the trial individual is inherited from the mutant individual so that NMBDE is able to avoid duplicating individuals and effectively search within the neighborhood.

3.3. Selection operator

The selection operator is adopted to determine whether a trial individual can survive in the next generation in NMBDE. pv_i^{t+1} replaces px_i^t and passes to the next population if its fitness value is better than that of the target individual px_i^t; otherwise the original individual px_i^t is reserved as shown in Eq. (7). Therefore, the selection operator is also called one to one

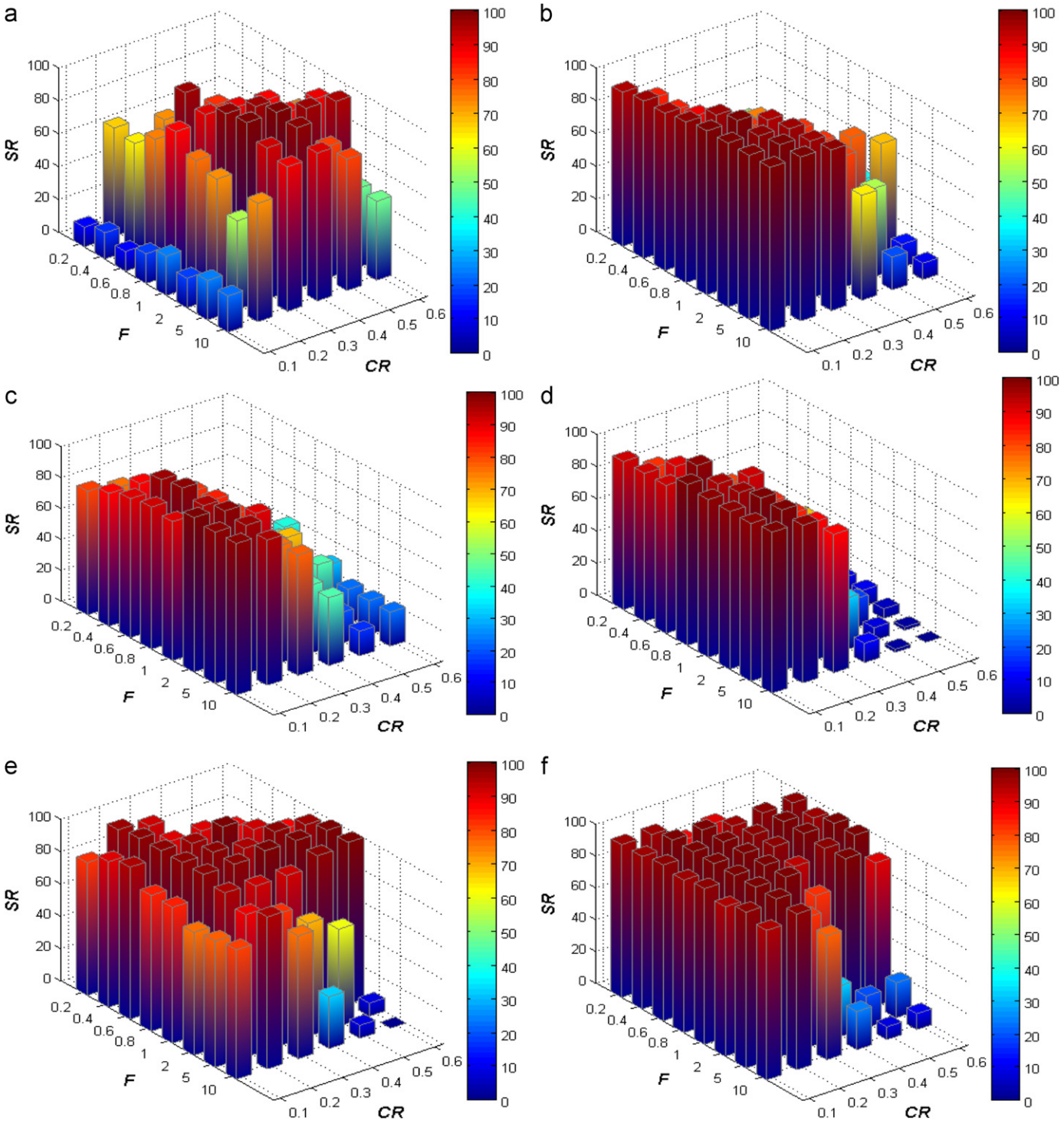


Fig. 3. The success rates of NMBDE with different settings of F and CR. (a) f_1, (b) f_2, (c) f_3, (d) f_4, (e) f_6 and (f) f_7.

elitism selection.

$$px_i^{t+1} = \begin{cases} pv_i^{t+1}, & \text{if } f(pv_i^{t+1}) < f(px_i^t) \\ px_i^t, & \text{otherwise} \end{cases} \quad (7)$$

In summary, the procedure of NMBDE can be stated as follows:

Step 1: Set control parameters and initialize the population randomly;
Step 2: Generate the binary mutant individuals according to the probability estimation operating as Eq. (4) and Eq. (5);
Step 3: Produce the trial individuals by using the binary crossover operator following Eq. (6);
Step 4: Evaluate the target individual and the corresponding trial individual, and choose the better one to survive into the next generation;
Step 5: If the terminal conditions are met, terminate the iteration; otherwise go to step 2.

Generally, the evolution process terminates if the global optimal solution is found or some exit condition is satisfied, for instance, the maximum generation is reached or the minimum fitness error is satisfied. The detailed process of NMBDE is depicted in Fig. 2.

4. Analysis and discussion

4.1. Analysis of the probability estimation operator

Unlike PBIL using the whole population information to construct one probability model, the probability model of NMBDE is built on the three individuals with the mutation operator for each individual. Table 1 shows the corresponding probability models of NMBDE with $F=0.5$, 1.0 and 2.0. According to Table 1, we can find that the probability of (0,1,1), (1,0,1) and (1,1,0) with two "1" bits and one "0" bit are 0.2315, 0.0266 and 0.9975 respectively due to the different sampling order when $F=0.5$. Especially, NMBDE is still able to generate "1" bit with a certain probability even when the bits of population are all "0"s, and vice versa. These two characteristics of probability estimation operator enable NMBDE to maintain the

Table 3
Experimental results with different bandwidth factor *b*.

b	f_1			f_2		
	SR%	*Mean* ± *StdVar*	*MGN*	*SR%*	*Mean* ± *StdVar*	*MGN*
0	8	$8.17E-9 \pm 8.43E-9$	1966	100	3 ± 0	1310
2	30	$1.14E-9 \pm 2.24E-9$	2357	**100**	$\mathbf{3 \pm 0}$	**789**
4	64	$2.09E-11 \pm 2.82E-11$	2091	98	$3.0000000005 \pm 3.55E-9$	504
6	78	$1.28E-11 \pm 2.43E-11$	1870	90	$3.0000000025 \pm 7.62E-9$	540
8	90	$5.82E-12 \pm 1.76E-11$	1802	90	$3.0000000025 \pm 7.62E-9$	531
10	92	$4.65E-12 \pm 1.59E-11$	1460	90	$3.0000000025 \pm 7.62E-9$	466
20	**96**	$\mathbf{2.32E-12 \pm 1.15E-11}$	**1658**	74	$3.0000000065 \pm 1.11E-8$	314
50	88	$6.98E-12 \pm 1.91E-11$	1544	70	$3.0000000075 \pm 1.16E-8$	319
100	88	$6.98E-12 \pm 1.91E-11$	1341	68	$3.0000000080 \pm 1.18E-8$	335
200	88	$6.98E-12 \pm 1.91E-11$	1621	68	$3.0000000080 \pm 1.18E-8$	294
500	82	$1.25E-10 \pm 8.22E-10$	1674	70	$3.0000000050 \pm 1.01E-8$	384
1000	86	$1.23E-10 \pm 8.22E-10$	1498	66	$3.0000000060 \pm 1.08E-8$	309
b	f_3			f_4		
	SR%	*Mean* ± *StdVar*	*MGN*	*SR%*	*Mean* ± *StdVar*	*MGN*
0	88	$2.24E-10 \pm 6.11E-10$	1400	92	$1.21E-9 \pm 4.16E-9$	1369
2	90	$1.86E-10 \pm 5.64E-10$	974	94	$5.86E-10 \pm 2.91E-9$	1037
4	92	$1.49E-10 \pm 5.10E-10$	793	96	$5.77E-10 \pm 2.87E-9$	695
6	**100**	$\mathbf{0 \pm 0}$	**813**	98	$2.48E-11 \pm 1.76E-10$	500
8	96	$7.45E-11 \pm 3.69E-10$	526	98	$2.69E-10 \pm 3.95E-9$	553
10	96	$7.45E-11 \pm 3.69E-10$	581	**100**	$\mathbf{0 \pm 0}$	**520**
20	96	$7.45E-11 \pm 3.69E-10$	591	**100**	$\mathbf{0 \pm 0}$	**423**
50	96	$7.45E-11 \pm 4.47E-10$	473	**100**	$\mathbf{0 \pm 0}$	**413**
100	96	$3.73E-11 \pm 2.63E-10$	519	98	$3.17E-10 \pm 3.97E-9$	440
200	96	$1.86E-10 \pm 6.11E-10$	469	94	$3.66E-10 \pm 3.36E-9$	471
500	92	$1.49E-10 \pm 5.10E-10$	462	94	$5.59E-10 \pm 3.41E-9$	464
1000	92	$1.49E-10 \pm 5.10E-10$	588	94	$3.66E-10 \pm 1.92E-9$	420
b	f_6			f_7		
	SR%	*Mean* ± *StdVar*	*MGN*	*SR%*	*Mean* ± *StdVar*	*MGN*
0	74	-0.999806 ± 0.001374	1573	52	$-186.730906 \pm 4.91E-6$	2101
2	94	-0.999806 ± 0.001374	1160	82	$-186.730908 \pm 1.73E-6$	2010
4	94	-0.999611 ± 0.001923	1008	92	$-186.730908 \pm 7.31E-7$	1704
6	**98**	$\mathbf{-0.999999 \pm 0.001923}$	**900**	92	$-186.730908 \pm 1.89E-6$	1550
8	**98**	$\mathbf{-0.999806 \pm 0.001374}$	**949**	94	$-186.730908 \pm 8.73E-7$	1442
10	96	-0.999998 ± 0.001374	781	**98**	$\mathbf{-186.730908 \pm 1.60E-6}$	**1503**
20	94	-0.999417 ± 0.002331	782	**98**	$\mathbf{-186.730908 \pm 1.03E-6}$	**1452**
50	94	-0.999611 ± 0.001923	910	**98**	$\mathbf{-186.730908 \pm 1.30E-6}$	**1315**
100	94	-0.999417 ± 0.002331	715	**98**	$\mathbf{-186.730908 \pm 1.07E-6}$	**1432**
200	94	-0.999417 ± 0.002331	847	96	$-186.730908 \pm 9.24E-7$	1574
500	94	$-0.999805 \pm 5.27E-10$	847	96	$-186.730908 \pm 6.62E-7$	1473
1000	94	-0.999611 ± 0.001923	784	94	$-186.730908 \pm 1.15E-6$	1441

population diversity better and escape from the local optima more effectively. Furthermore, the probability model varies with different F. For instance, the probability of being "1" is 0.0474, 0.1192 and 0.2315 for (0,0,0) when F is 0.5, 1.0, and 2.0, respectively. Obviously, $F=2$ is big for NMBDE as the mutation rate achieves 0.2315 which will spoil the performance while a too small F cannot effectively improve the diversity of algorithm. Thus, a proper F as well as b is vital for the search ability of NMBDE, which determines the probability model. However, each probability model is used to create one mutant individual in NMBDE, which is similar to DE, but different from PBIL. Therefore, the risk from wrong probability prediction can be reduced and the optimal information can be maintained effectively with the selection operator.

Table 4
Parameter settings of NMBDE, DBDE, DBPSO and BAS.

Algorithm	Control parameters
NMBDE	$CR=0.2$, $F=0.8$, $b=20$
MBDE [23]	$CR=0.8$, $F1=0.5$, $F2=0.005$
DBDE [26]	$CR=0.1$, $F=0.9$
DBPSO [27]	$w=0.2$, $c_1=c_2=2.0$, $Vmax=6.0$
BAS [28]	$m=N$, $\tau_0=0.5$, $\Delta\tau=1.0$, $\rho=0.1$, $cf=0.9$

*m is the number of ants and N is the length of individuals.

4.2. Parameter analysis

Like standard DE, NMBDE is sensitive to the control parameters, so the value of parameter need be chosen carefully.

Table 5
Results of NMBDE, DBDE, DBPSO, BAS and MBDE on low-dimensional benchmark functions.

	Algorithm	*SR*	*Mean ± StdVar*	*MGN*	*t-test*		Algorithm	*SR*	*Mean ± StdVar*	*MGN*	*t-test*
f_1	NMBDE	**90**	**5.82E−12 ± 1.75E−11**	**1484**		f_{11}	NMBDE	**100**	**−176.137537 ± 0.0**	**345**	
	DBDE	0	2.93E−9 ± 5.69E−9	/	+		DBDE	90	−176.135921 ± 0.0056	1147	+
	DBPSO	14	4.98E−9 ± 1.27E−8	2048	+		DBPSO	100	−176.137537 ± 0.0	720	≈
	BAS	24	6.03E−5 ± 4.83E−4	920	≈		BAS	77	−175.683130 ± 3.7883	664	≈
	MBDE	21	1.58E−4 ± 3.91E−3	1389	+		MBDE	55	−173.927526 ± 7.29645	186	+
f_2	NMBDE	76	3.00000000603 ± 1.07E−8	351		f_{12}	NMBDE	**98**	**−1.03162844 ± 3.18E−8**	**240**	
	DBDE	**99**	**3.00000000025 ± 2.51E−9**	**1089**	−		DBDE	86	−1.03162841 ± 8.72E−8	1384	+
	DBPSO	37	3.00000001584 ± 1.22E−8	1147	+		DBPSO	37	−1.03162634 ± 1.03E−5	1634	+
	BAS	35	3.00262863376 ± 0.0191	240	≈		BAS	26	−1.02481310 ± 0.0617	541	≈
	MBDE	41	3.00000001483 ± 0.00	432	+		MBDE	42	−1.03148589 ± 0.00119	169	≈
f_3	NMBDE	**95**	**9.31E−11 ± 4.07E−10**	**558**		f_{13}	NMBDE	**100**	**−7.885600 ± 0.0**	**543**	
	DBDE	89	2.04E−10 ± 5.85E−10	1047	≈		DBDE	100	−7.885600 ± 0.0	963	≈
	DBPSO	12	8.91E−7 ± 7.12E−7	1295	+		DBPSO	40	−7.866341 ± 0.018537	690	+
	BAS	13	1.51E−4 ± 9.06E−4	103	≈		BAS	67	−7.875115 ± 0.0168419	570	+
	MBDE	31	2.91E−2 ± 1.45	498	+		MBDE	58	−7.879108 ± 0.01299405	533	+
f_4	NMBDE	**89**	**1.19E−9 ± 3.96E−9**	**427**		f_{14}	NMBDE	**100**	**−176.541767 ± 0.0**	**542**	
	DBDE	80	3.36E−9 ± 7.17E−9	1136	+		DBDE	100	−176.541767 ± 0.0	800	≈
	DBPSO	10	3.62E−5 ± 2.28E−4	1654	≈		DBPSO	99	−176.541753 ± 1.33E−4	760	≈
	BAS	13	9.78E−4 ± 0.0052	235	≈		BAS	75	−176.537196 ± 0.0329	573	≈
	MBDE	29	8.37E−6 ± 2.79E−4	409	+		MBDE	89	−176.541373 ± 0.00182	496	+
f_5	NMBDE	**100**	**0.0 ± 0.0**	**291**		f_{15}	NMBDE	**98**	**7.34E−12 ± 5.17E−11**	**587**	
	DBDE	91	6.6678E−4 ± 0.002130	1129	+		DBDE	38	2.53E−10 ± 2.10E−10	2029	+
	DBPSO	82	5.9170E−4 ± 0.002016	1104	+		DBPSO	33	8.12E−10 ± 1.33E−9	1687	+
	BAS	86	3.02E−4 ± 0.001470	653	+		BAS	52	3.01E−5 ± 1.32E−4	408	+
	MBDE	85	1.14E−4 ± 9.8 E−3	205	+		MBDE	20	6.41E−6 ± 0.0064024	558	≈
f_6	NMBDE	**97**	**−0.999708 ± 0.001665**	**797**		f_{16}	NMBDE	**100**	**−3.862781 ± 0.0**	**364**	
	DBDE	35	−0.994364 ± 0.004819	1495	+		DBDE	72	−3.862773 ± 1.91E−5	1418	+
	DBPSO	75	−0.998542 ± 0.003486	1312	+		DBPSO	66	−3.862768 ± 2.79E−5	1698	+
	BAS	77	−0.998815 ± 0.0031	793	+		BAS	55	−3.862330 ± 0.0017	991	+
	MBDE	68	−0.996988 ± 0.0045161	529	+		MBDE	52	−3.859679 ± 0.01572	269	+
f_7	NMBDE	**97**	**−186.73090843 ± 6.24E−7**	**1502**		f_{17}	NMBDE	**100**	**0.0 ± 0.0**	**36**	
	DBDE	88	−186.73090806 ± 1.34E−6	1506	+		DBDE	100	0.0 ± 0.0	239	≈
	DBPSO	51	−186.73085096 ± 5.05E−4	1895	≈		DBPSO	100	0.0 ± 0.0	884	≈
	BAS	51	−186.72336875 ± 0.064715	989	≈		BAS	94	4.81E−8 ± 3.11E−7	150	≈
	MBDE	50	−186.73041944 ± 0.0026432	1049	≈		MBDE	100	0.0 ± 0.0	47	≈
f_8	NMBDE	**0**	**9.51E−9 ± 2.46E−8**	**/**		f_{18}	NMBDE	**100**	**0.0 ± 0.0**	**36**	
	DBDE	0	1.21E−8 ± 2.71E−8	/	≈		DBDE	100	0.0 ± 0.0	236	≈
	DBPSO	0	3.42E−8 ± 4.60E−8	/	+		DBPSO	100	0.0 ± 0.0	840	≈
	BAS	0	2.14E−5 ± 1.85E−4	/	+		BAS	98	1.46E−9 ± 1.03E−8	132	≈
	MBDE	0	0.00226319 ± 0.0169754	/	+		MBDE	100	0.0 ± 0.0	47	≈
f_9	NMBDE	**0**	**3.13E−8 ± 2.12E−7**	**/**		f_{19}	NMBDE	**100**	**0.0 ± 0.0**	**38**	
	DBDE	0	9.55E−8 ± 2.25E−7	/	+		DBDE	100	0.0 ± 0.0	245	≈
	DBPSO	0	6.88E−7 ± 1.46E−6	/	+		DBPSO	100	0.0 ± 0.0	929	≈
	BAS	0	5.69E−6 ± 1.50E−5	/	+		BAS	91	3.40E−9 ± 2.43E−8	37	≈
	MBDE	0	9.38E−7 ± 3.15E−6	/	+		MBDE	100	0.0 ± 0.0	48	≈
f_{10}	NMBDE	100	0.0 ± 0.0	88		f_{20}	NMBDE	**100**	**0.0 ± 0.0**	**36**	
	DBDE	100	0.0 ± 0.0	454	≈		DBDE	100	0.0 ± 0.0	241	≈
	DBPSO	100	0.0 ± 0.0	852	≈		DBPSO	100	0.0 ± 0.0	962	≈
	BAS	93	0.034132 ± 0.2208	141	≈		BAS	85	0.0011 ± 0.0055	88	+
	MBDE	**100**	**0.00 ± 0.00**	**52**	≈		MBDE	100	0.0 ± 0.0	46	≈

Although the parameter setting of DE was discussed and the recommended values were given [30], these values are not appropriate for NMBDE due to the changes in the definition and meaning of parameters. Therefore, it is necessary to conduct the parameter analysis to understand the mechanism of NMBDE better. However, as the parameter setting is problem dependent, the aim of parameter study here is to find the proper parameter values rather than the optimal values.

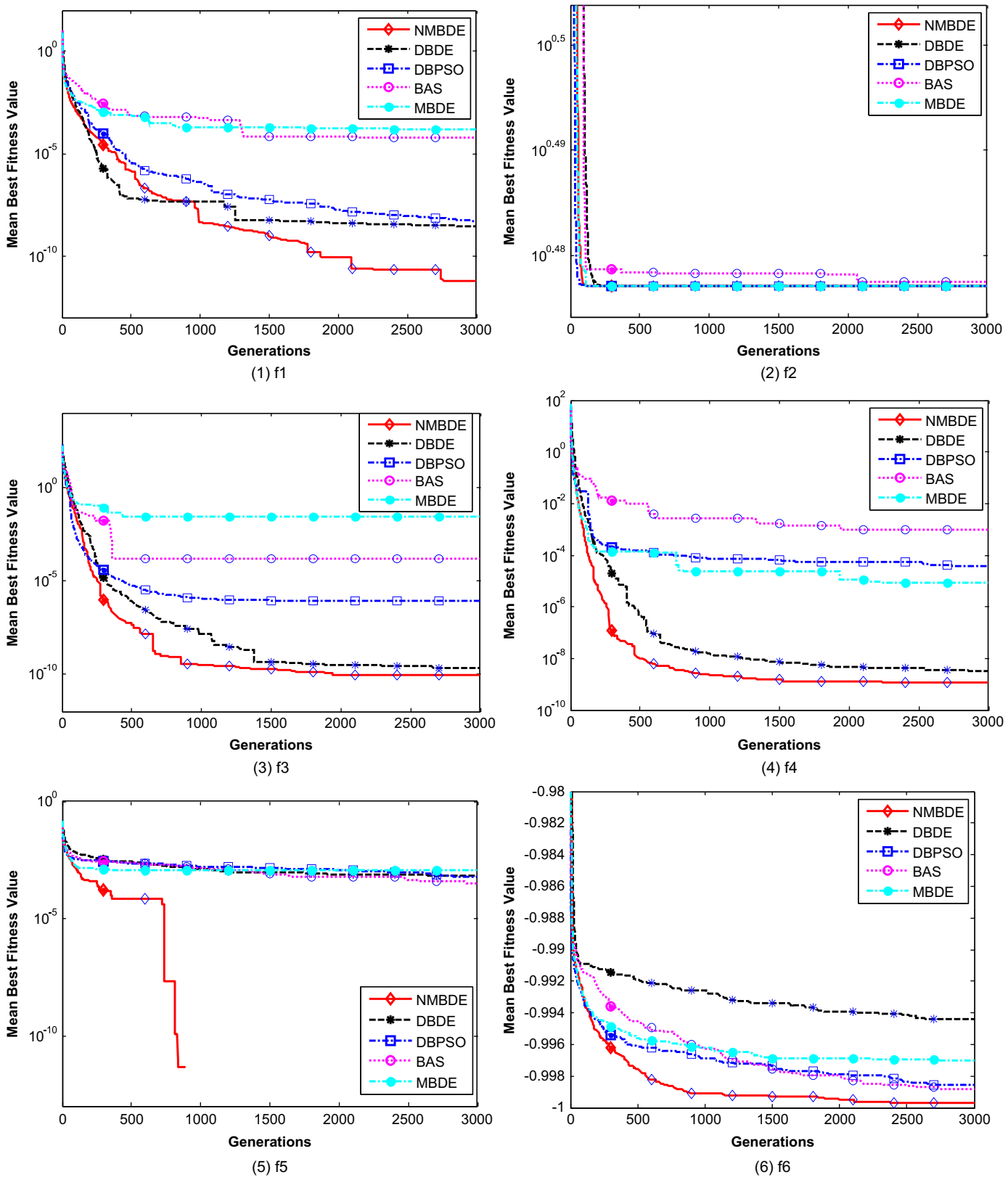


Fig. 4. Convergence curves of NMBDE, DBDE, DBPSO, BAS and MBDE on the low-dimensional functions.

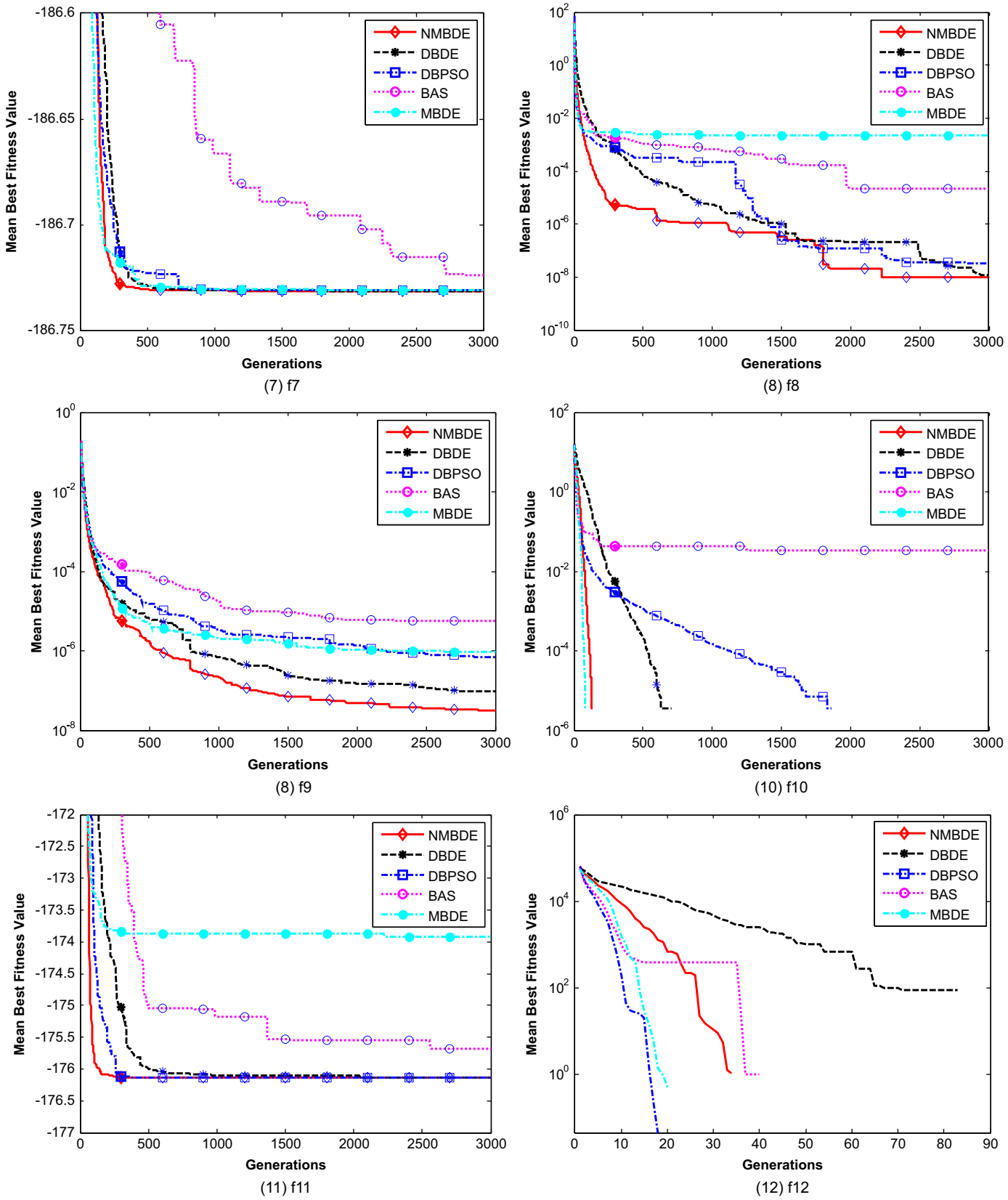


Fig. 4. (*continued*)

In NMBDE, the crossover rate *CR* decides the contribution of the mutant individual to the generation of the trial solution; the scaling factor *F* determines the scale of the differential information in the mutation operator *MO*; and the bandwidth factor *b* tunes the range and shape of the probability estimation operator of NMBDE. Therefore, all three control parameters, i.e.,

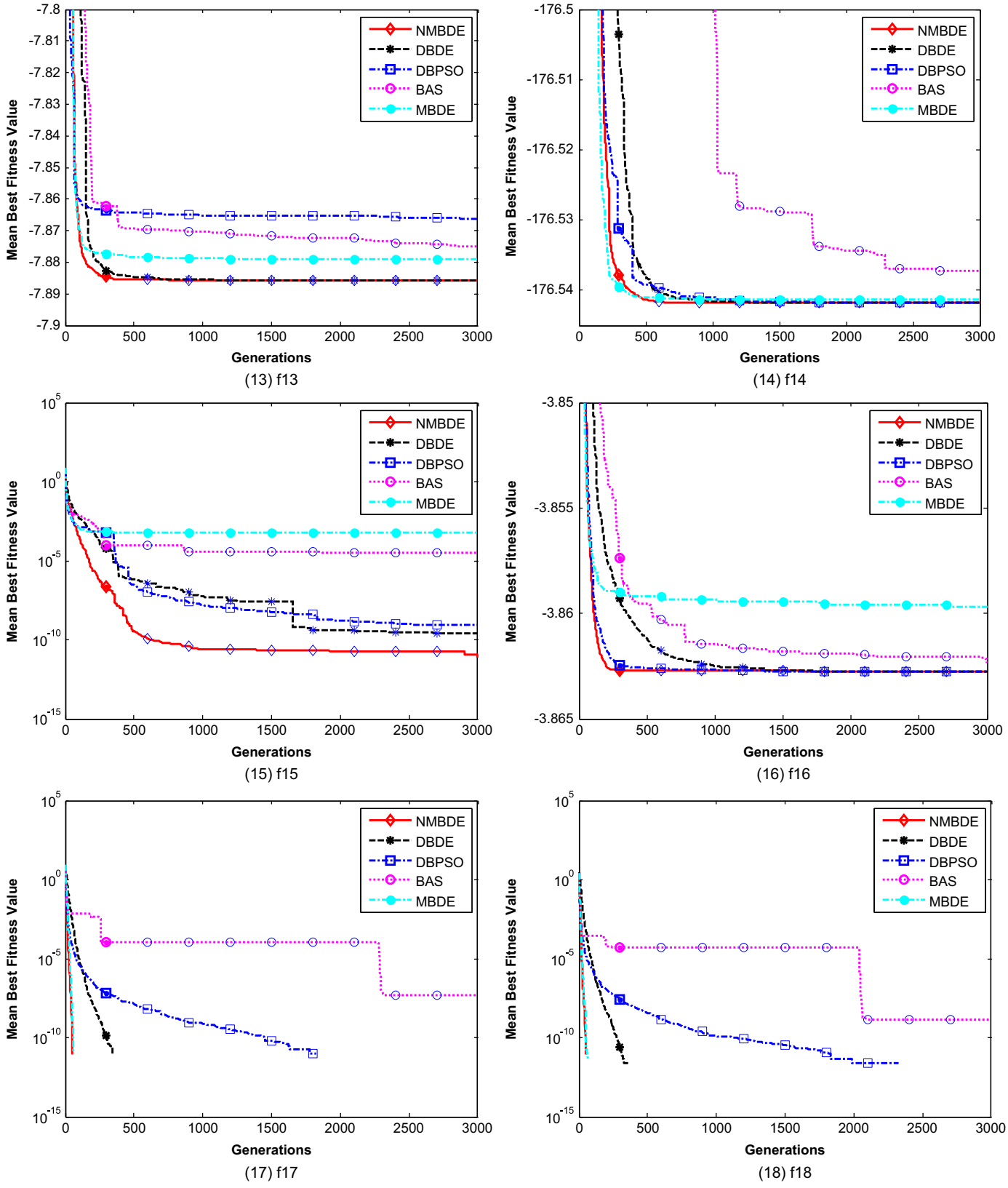


Fig. 4. (*continued*)

the crossover rate CR, the scaling factor F and the bandwidth factor b, have influence on the optimization performance of NMBDE. A suite of 20 well-known functions [14] listed in the Appendix were adopted as the numerical optimization benchmarks in this work, and six of them (i.e., f_1, f_2, f_3, f_4, f_6, and f_7), comprising four unimodal and two multimodal functions, were used for parameter analysis. The characteristics and minima of all functions are presented in Table 2. For a general analysis, the trial-and-error strategy in [31] was adopted for parameter study.

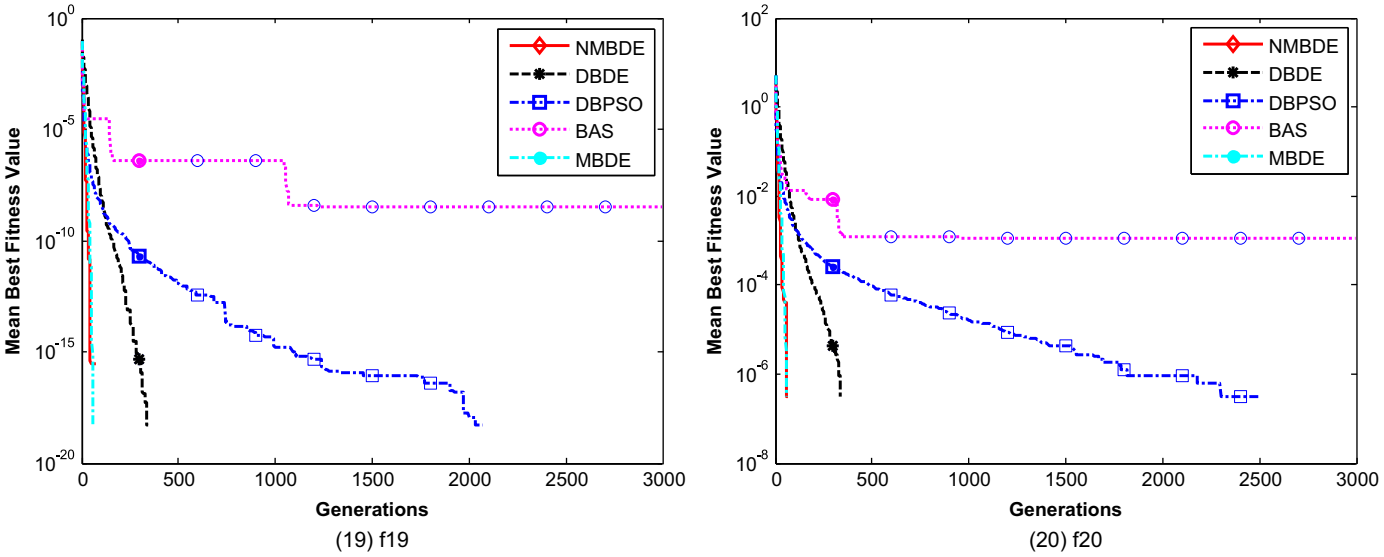


Fig. 4. (*continued*)

Table 6
Results of NMBDE, DBDE, DBPSO, BAS and MBDE on 30-dimensional benchmark functions.

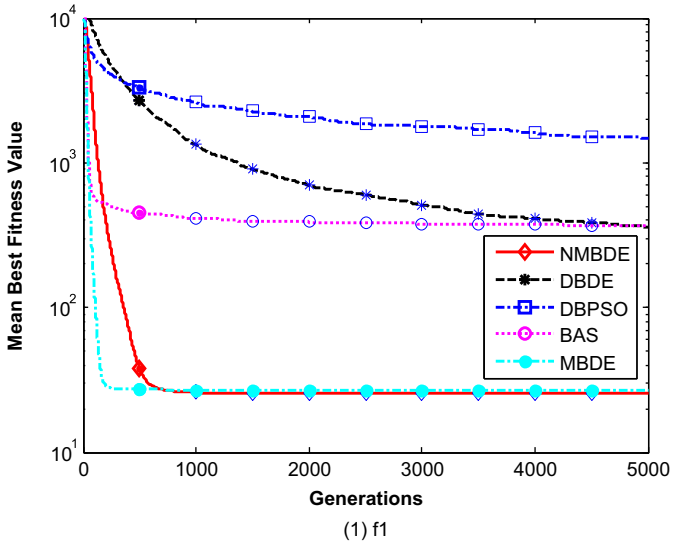
	Algorithm	*SR*	*Mean* ± *StdVar*	*MGN*	*t-test*		Algorithm	*SR*	*Mean* ± *StdVar*	*MGN*	*t-test*
f_1	NMBDE	0	**25.4575 ± 0.5370**	–		f_{17}	NMBDE	**100**	**0.0 ± 0.0**	**743**	
	DBDE	0	360.9936 ± 36.2519	–	+		DBDE	0	21.0394 ± 1.8366	–	+
	DBPSO	0	1489.0703 ± 269.2322	–	+		DBPSO	0	129.3599 ± 16.9387	–	+
	BAS	0	369.8605 ± 776.0672	–	+		BAS	0	7.3822 ± 22.4639	–	+
	MBDE	0	27.0071 ± 0.46428	–	+		MBDE	100	0.0 ± 0.0	955	≈
f_5	NMBDE	**100**	**0.0 ± 0.0**	**983**		f_{18}	NMBDE	**100**	**0.0 ± 0.0**	**745**	
	DBDE	0	0.5715 ± 0.0408	–	+		DBDE	0	61.7924 ± 6.2845	–	+
	DBPSO	0	1.0314 ± 0.0061	–	+		DBPSO	0	411.3443 ± 411.3443	–	+
	BAS	0	0.1349 ± 0.1684	–	+		BAS	0	34.8005 ± 184.8619	–	≈
	MBDE	100	0.0 ± 0.0	1087	≈		MBDE	100	0.0 ± 0.0	966	≈
f_8	NMBDE	0	**0.6667 ± 0.0002**	–		f_{19}	NMBDE	**100**	**0.0 ± 0.0**	**1489**	
	DBDE	0	2422.7499 ± 453.2116	–	+		DBDE	0	4.1973*E*−6 ± 1.7779*E*−6	–	+
	DBPSO	0	84034.0850 ± 22479.2289		+		DBPSO	0	0.0208 ± 0.0100	–	+
	BAS	0	11484.5819 ± 74269.9085	–	+		BAS	0	0.0017 ± 0.0096	–	≈
	MBDE	0	0.6677 ± 0.0074	–	≈		MBDE	100	0.0 ± 0.0	1755	≈
f_9	NMBDE	0	**1.5437 ± 0.4626**	–		f_{20}	NMBDE	**100**	**0.0 ± 0.0**	**723**	
	DBDE	0	2.6227 ± 0.04642	–	+		DBDE	0	19.6213 ± 0.9937982	–	+
	DBPSO	0	7.4392 ± 0.3293	–	+		DBPSO	0	48.4762 ± 4.4078	–	+
	BAS	0	4.7013 ± 0.8866	–	+		BAS	0	4.5244 ± 3.3154	–	+
	MBDE	0	2.6199 ± 0.0437	–	+		MBDE	100	0.0 ± 0.0	741	≈

Firstly, the sensitivity of NMBDE on *CR* and *F* was test. The bandwidth factor *b* was set as $b=6.0$. Then *CR* was tuned from 0.1 to 0.6 with step 0.1 and *F* was set as 0.2, 0.4, 0.6, 0.8, 1.0, 2.0, 5.0 and 10.0 respectively. Each variable was encoded by 20 bits. The population size was 40 and the maximal generation number was set as 3000. The experiments on each function were repeated 50 times independently.

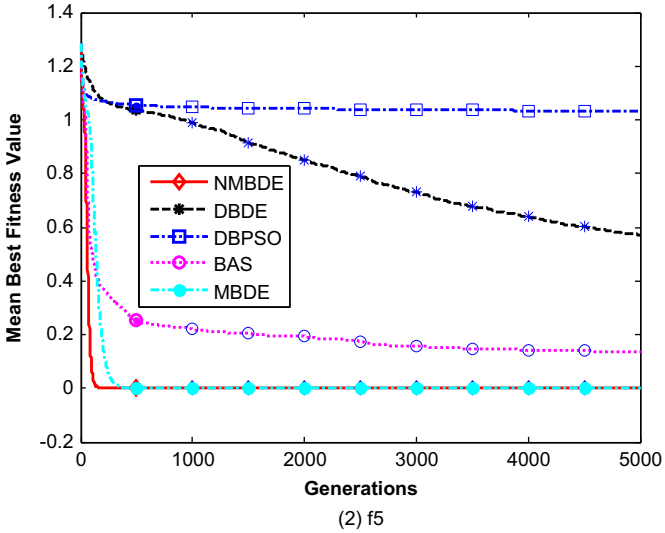
The success rates of finding the global optima (*SR*) with different parametric settings are shown in Fig. 3. We can find that both *CR* and *F* have influence on the performance of NMBDE. High success rates are obtained when $0.1 < CR \leq 0.3$ while the performance of NMBDE greatly deteriorates when $CR \geq 0.4$ for unimodal functions. Compared with *CR*, NMBDE is less sensitive to *F* and the success rates hardly fluctuate in a wide range of *F* with a proper *CR*. However, it should avoid setting a too big or too small *F*. Obviously, *CR* and *F* have not to be set a big value simultaneously, otherwise the local search ability will be spoiled, which results in poor performances.

Based on the results of Fig. 3, $CR \in (0.1,0.3)$ and $F \in (0.2,5)$ are recommended. In this work, $CR=0.2$ and $F=0.8$ were adopted as the default values. Then the bandwidth factor *b* was set within {0,2,4,6,8,10,20,50,100,200,500,1000} to test its influence on the optimization ability. Four performance indicators were taken into consideration, i.e., *SR*, the mean best fitness value, the standard variance of the best fitness value and the mean generation number (*MGN*). The experimental results of 50 times independent runs are given in Table 3.

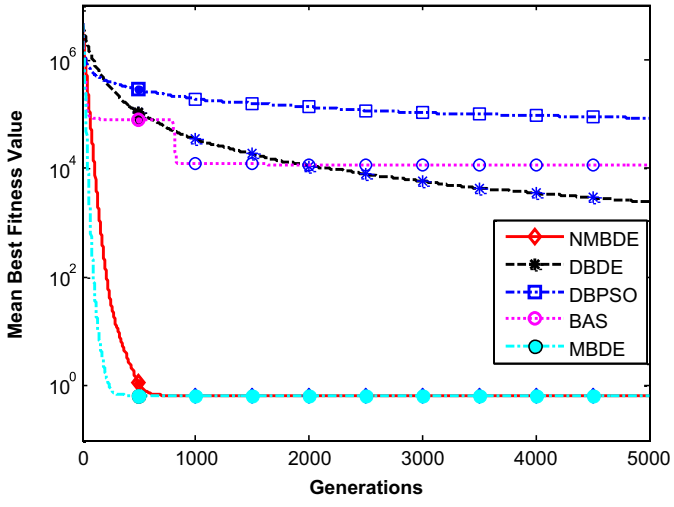
The results in Table 3 show that NMBDE is not very sensitive to the bandwidth factor *b*, but the optimal value of *b* is obviously problem dependent. For f_2, NMBDE achieves the best performance when *b* is less than 4, while *b* should be more than 6 for NMBDE on other 5 benchmark functions. Based on an overall analysis of experimental results, we suggest that *b* is set between 6 and 100, and 20 was selected as the default value for *b* in our work to achieve a balance performance on various problems.



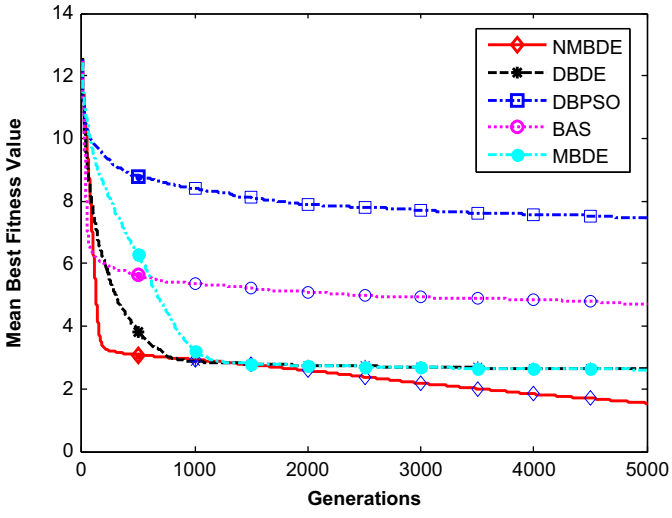
(1) f1



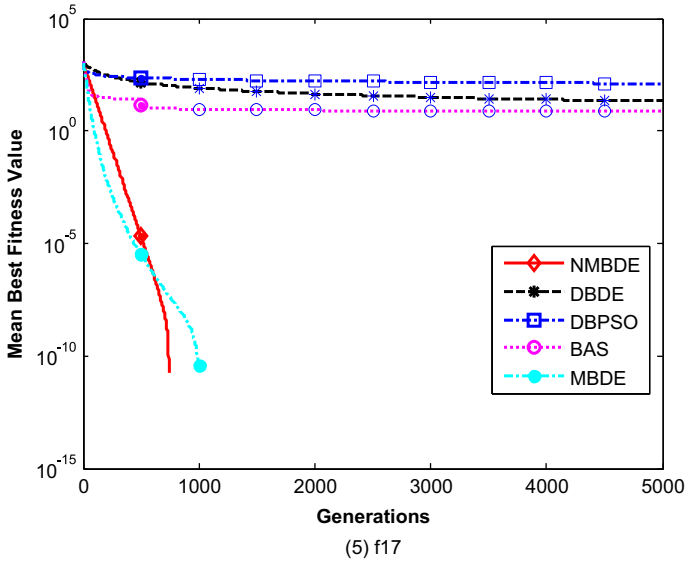
(2) f5



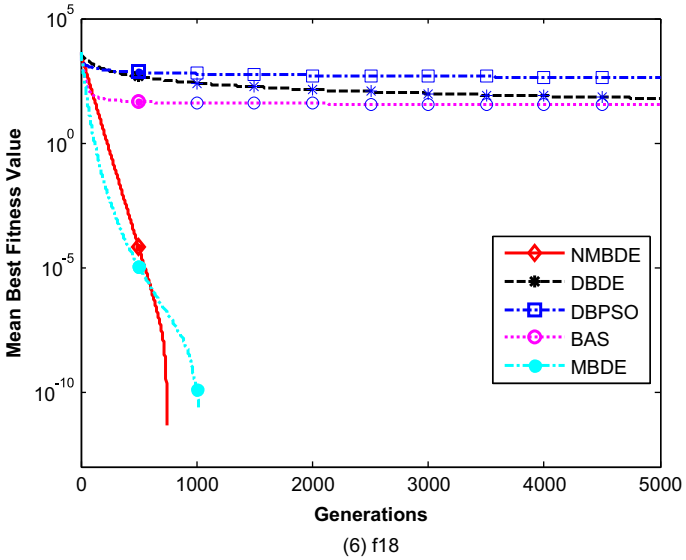
(3) f8



(4) f9



(5) f17



(6) f18

Fig. 5. Convergence curves of NMBDE, DBDE, DBPSO, BAS and MBDE on the 30-dimensional functions.

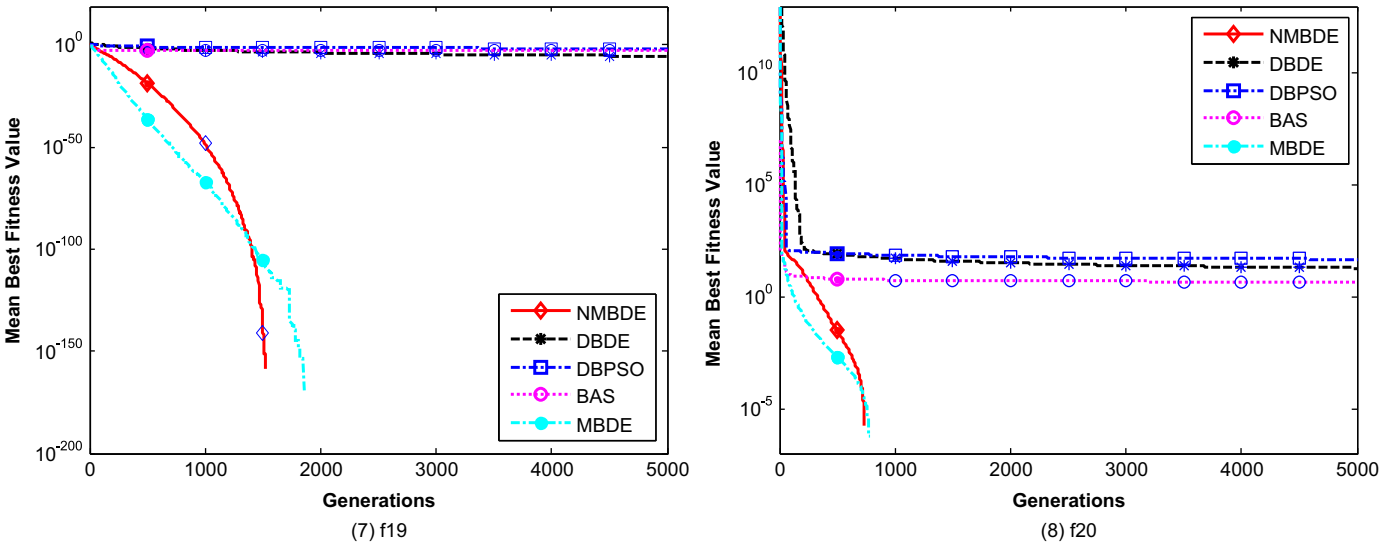


Fig. 5. (*continued*)

Table 7
Experimental results of NMBDE, DBDE, DBPSO, BAS and MBDE on 10.100 MKP instances.

MKP cases	Best Known	Algorithm	*SR%*	*Mean* ± *StdVar*	*MGN*	*t-test*	MKP cases	Best Known	Algorithm	*SR%*	*Mean* ± *StdVar*	*MGN*	*t-test*
10.100.00	23064	NMBDE	**34**	**23058.12 ± 4.95**	**1225**		*10.100.15*	42995	NMBDE	100	42995.00 ± 0.00	211	
$n=100$		DBDE	17	23055.55 ± 4.97	2594	+	$n=100$		DBDE	83	42982.39 ± 28.79	2599	+
$m=10$		DBPSO	3	23046.12 ± 13.49	2866	+	$m=10$		DBPSO	0	42669.82 ± 93.11	–	+
		BAS	13	23036.24 ± 26.20	2126	+			BAS	14	42879.11 ± 62.07	1667	+
		MBDE	0	23050.61 ± 2.77	–	+			MBDE	**100**	**42995.00 ± 0.00**	**105**	≈
10.100.01	22801	NMBDE	**84**	**22792.87 ± 18.72**	**1429**		*10.100.16*	43559	NMBDE	**1**	**43552.91 ± 7.33**	**2042**	
$n=100$		DBDE	19	22751.44 ± 30.09	2820	+	$n=100$		DBDE	0	43381.50 ± 54.97	–	+
$m=10$		DBPSO	0	22652.60 ± 34.68	–	+	$m=10$		DBPSO	0	43232.88 ± 91.02	–	+
		BAS	5	22684.94 ± 58.89	1333	+			BAS	0	43421.77 ± 68.25	–	+
		MBDE	19	22738.75 ± 36.12	308	+			MBDE	0	43497.54 ± 45.83	–	+
10.100.02	22131	NMBDE	**94**	**22127.54 ± 13.89**	**1537**		*10.100.17*	42970	NMBDE	**100**	**42970.00 ± 0.00**	**344**	
$n=100$		DBDE	31	22089.39 ± 28.78	2352	+	$n=100$		DBDE	69	42956.64 ± 20.74	2720	+
$m=10$		DBPSO	0	22003.05 ± 41.19	–	+	$m=10$		DBPSO	5	42839.44 ± 59.68	2311	+
		BAS	1	22026.10 ± 43.22	856	+			BAS	26	42908.58 ± 52.11	1041	+
		MBDE	10	22069.90 ± 27.27	257	+			MBDE	38	42936.48 ± 26.79	177	+
10.100.03	22772	NMBDE	**47**	**22764.13 ± 14.12**	**1077**		*10.100.18*	42212	NMBDE	**100**	**42212.0 ± 0.00**	**177**	
$n=100$		DBDE	20	22760.42 ± 15.96	1724	≈	$n=100$		DBDE	100	42212.00 ± 0.00	233	≈
$m=10$		DBPSO	9	22719.86 ± 43.63	1820	+	$m=10$		DBPSO	33	42160.52 ± 44.19	2258	+
		BAS	5	22694.23 ± 58.17	2192	+			BAS	40	42174.53 ± 38.50	604	+
		MBDE	8	22722.94 ± 30.69	277	+			MBDE	91	42207.46 ± 15.06	277	+
10.100.04	22751	NMBDE	2	22633.51 ± 26.92	1020		*10.100.19*	41207	NMBDE	**4**	**41165.74 ± 10.74**	**217**	
$n=100$		DBDE	**3**	**22685.78 ± 24.36**	**2136**	–	$n=100$		DBDE	2	41161.03 ± 16.17	2914	+
$m=10$		DBPSO	0	22628.92 ± 13.65	–	≈	$m=10$		DBPSO	0	40913.44 ± 76.79	–	+
		BAS	2	22623.31 ± 27.75	2248	+			BAS	0	41066.14 ± 62.86	–	+
		MBDE	0	22602.18 ± 35.78	–	+			MBDE	0	41129.91 ± 40.66	–	+
10.100.05	22777	NMBDE	**3**	**22717.40 ± 13.79**	**1414**		*10.100.20*	57375	NMBDE	**100**	**57375.00 ± 0.00**	**107**	
$n=100$		DBDE	0	22708.98 ± 26.62	–	+	$n=100$		DBDE	100	57375.00 ± 0.00	541	≈
$m=10$		DBPSO	3	22681.99 ± 33.24	2162	+	$m=10$		DBPSO	97	57371.31 ± 25.30	887	≈
		BAS	1	22661.39 ± 40.07	352	+			BAS	65	57316.68 ± 85.05	793	+
		MBDE	0	22673.18 ± 32.47	–	+			MBDE	95	57374.10 ± 3.94	109	+
10.100.06	21875	NMBDE	2	21823.03 ± 8.90	3856		*10.100.21*	58978	NMBDE	**100**	**58978.00 ± 0.00**	**226**	
$n=100$		DBDE	**11**	**21831.58 ± 18.78**	**2869**	–	$n=100$		DBDE	91	58973.29 ± 15.16	1693	+
$m=10$		DBPSO	3	21804.57 ± 30.99	3645	+	$m=10$		DBPSO	0	58912.18 ± 17.27	–	+
		BAS	4	21794.06 ± 41.55	2760	+			BAS	4	58905.46 ± 33.53	3161	+
		MBDE	0	21819.78 ± 9.19	–	+			MBDE	18	58932.32 ± 21.58	105	+
10.100.07	22635	NMBDE	**23**	**22570.32 ± 35.53**	**1199**		*10.100.22*	58391	NMBDE	**1**	**58354.378 ± 3.70**	**188**	
$n=100$		DBDE	5	22449.29 ± 67.18	3653	+	$n=100$		DBDE	6	58353.39 ± 13.49	3216	≈
$m=10$		DBPSO	0	22387.92 ± 72.04	–	+	$m=5$		DBPSO	0	58354.01 ± 8.89	–	≈
		BAS	7	22481.47 ± 82.23	1871	+			BAS	1	58342.31 ± 20.98	3745	+
		MBDE	11	22556.88 ± 28.81	169	+			MBDE	0	58350.43 ± 10.85	–	+
10.100.08	22511	NMBDE	**3**	**22427.40 ± 11.17**	**162**		*10.100.23*	61966	NMBDE	**1**	**61895.03 ± 6.29**	**714**	
$n=100$		DBDE	5	22411.59 ± 26.98	2818	+	$n=100$		DBDE	0	61873.53 ± 22.74	–	+
$m=10$		DBPSO	1	22373.94 ± 40.56	4793	+	$m=10$		DBPSO	0	61887.84 ± 17.38	–	+

Table 7 (*continued*)

MKP cases	Best Known	Algorithm	*SR%*	*Mean* ± *StdVar*	*MGN*	*t-test*	MKP cases	Best Known	Algorithm	*SR%*	*Mean* ± *StdVar*	*MGN*	*t-test*
		BAS	1	22371.78 ± 56.52	4658	+			BAS	0	61887.92 ± 25.86	–	+
		MBDE	0	22399.32 ± 15.29	–	+			MBDE	0	61891.92 ± 5.80	–	+
10.100.09	22702	NMBDE	**27**	**22699.08 ± 1.78**	**693**		*10.100.24*	60803	NMBDE	**100**	**60803.00 ± 0.00**	**113**	
$n=100$		DBDE	0	22543.66 ± 40.07	–	+	$n=100$		DBDE	100	60803.00 ± 0.00	586	≈
$m=10$		DBPSO	0	22665.96 ± 59.55	–	+	$m=10$		DBPSO	61	60799.76 ± 7.44	1973	+
		BAS	13	22658.83 ± 74.74	752	+			BAS	39	60794.53 ± 22.82	1404	+
		MBDE	41	22683.61 ± 42.63	54	+			MBDE	95	60802.70 ± 1.31	65	+
10.100.10	41395	NMBDE	**19**	**41372.53 ± 11.14**	**3569**		*10.100.25*	61437	NMBDE	**1**	**61368.49 ± 7.20**	**284**	
$n=100$		DBDE	0	41338.10 ± 35.22	–	+	$n=100$		DBDE	0	61304.68 ± 33.43	–	+
$m=10$		DBPSO	0	40946.81 ± 91.38	–	+	$m=10$		DBPSO	0	61316.20 ± 32.85	–	+
		BAS	0	41263.81 ± 81.72	–	+			BAS	1	61304.70 ± 39.86	827	+
		MBDE	0	41351.67 ± 39.56	–	+			MBDE	0	61305.76 ± 45.18	–	+
10.100.11	42344	NMBDE	**77**	**42325.67 ± 33.72**	**1539**		*10.100.26*	56377	NMBDE	**36**	**56360.78 ± 12.25**	**1466**	
$n=100$		DBDE	5	42245.81 ± 30.45	1764	+	$n=100$		DBDE	8	56344.78 ± 12.75	3109	+
$m=10$		DBPSO	0	41977.73 ± 69.60	–	+	$m=10$		DBPSO	0	56264.55 ± 33.99	–	+
		BAS	2	42168.88 ± 90.92	4496	+			BAS	0	56281.67 ± 46.04	–	+
		MBDE	13	42249.78 ± 45.89	876	+			MBDE	1	56335.04 ± 9.82	1186	+
10.100.12	42401	NMBDE	**1**	**42350.51 ± 5.10**	**209**		*10.100.27*	59391	NMBDE	**100**	**59391.00 ± 0.00**	**343**	
$n=100$		DBDE	1	42291.85 ± 28.14	4990	+	$n=100$		DBDE	100	59391.00 ± 0.00	1313	≈
$m=10$		DBPSO	0	42199.82 ± 60.01	–	+	$m=10$		DBPSO	1	59282.1 ± 42.16	3775	+
		BAS	0	42283.37 ± 106.99	–	+			BAS	12	59299.12 ± 53.23	1328	+
		MBDE	0	42345.84 ± 10.82	–	+			MBDE	23	59345.57 ± 24.95	820	+
10.100.13	45624	NMBDE	**14**	**45591.55 ± 28.18**	**1798**		*10.100.28*	60205	NMBDE	100	60205.00 ± 0.00	134	
$n=100$		DBDE	1	45537.17 ± 43.10	4129	+	$n=100$		DBDE	100	60205.00 ± 0.00	919	≈
$m=10$		DBPSO	0	45223.55 ± 85.05	–	+	$m=10$		DBPSO	26	60099.72 ± 66.86	2644	+
		BAS	0	45424.93 ± 108.57	–	+			BAS	25	60105.97 ± 64.59	1160	+
		MBDE	1	45535.24 ± 33.84	124	+			**MBDE**	**100**	**60205.00 ± 0.00**	**55**	≈
10.100.14	41884	NMBDE	1	41804.02 ± 29.21	671		*10.100.29*	60633	NMBDE	**41**	**60628.87 ± 3.46**	**342**	
$n=100$		DBDE	**0**	**41830.22 ± 20.09**	–	−	$n=100$		DBDE	19	60580.55 ± 44.10	3440	+
$m=10$		DBPSO	0	41631.18 ± 71.45	–	+	$m=10$		DBPSO	24	60588.00 ± 43.00	1151	+
		BAS	0	41740.41 ± 51.12	–	+			BAS	26	60577.61 ± 48.79	669	+
		MBDE	0	41783.10 ± 30.05	–	+			MBDE	67	60626.78 ± 22.37	97	≈

5. Applications of NMBDE

5.1. Parameter settings

In the following section, NMBDE are used to solve the numerical optimization problems and multidimensional knapsack problems to test its optimization ability. In order to compare the solution quality and performance of NMBDE, four binary-coded optimization algorithms, i.e., DBDE [26], DBPSO [27], BAS [28], and MBDE [23] with the recommended parameter values were applied to the both applications as well. Table 4 lists the parameter settings of all the algorithms.

5.2. Numerical optimization

Firstly, NMBDE, DBDE, DBPSO, BAS and MBDE were applied to optimize 20 benchmark functions in low-dimension. Then eight functions of them, which can be extended to high-dimension, were increased to 30-dimensions to verify the optimization ability and scalability of the algorithms. Each variable in the solution vector was encoded by 20 bits.

5.2.1. Low-dimensional numerical experiment

For low-dimension tests, the maximum generation and the population size of all algorithms were set to 3000 and 40 respectively, and the experiments were repeated for 100 times independently. The numerical results and the *t*-test results are both given in Table 5 where "+" indicates that NMBDE is significantly better than the compared algorithm at the 95% confidence; "−" means that NMBDE is significantly worse than the compared algorithm; and "≈" represents that the difference is not significant. The convergence curves of four algorithms on parts of the benchmarks are drawn in Fig. 4.

Table 5 shows that NMBDE outperforms DBDE, DBPSO, BAS and MBDE on almost all the test functions in terms of search accuracy and convergence speed. Specifically, NMBDE is only inferior to DBDE and MBDE on f_2 on f_{10}, respectively.

From Fig. 4, it is obvious that NMBDE searches out the better solutions and converges faster than the other algorithms on the majority of functions as the proposed probability estimation operator can provide better global searching ability and prevent NMBDE from trapping in the local optima.

5.2.2. High-dimensional numerical experiment

On high-dimensional numerical experiment, the population size of all algorithms was set as $NP=200$ and the maximal generation number was 5000. The experimental results are listed in Table 6 and the convergence curves of the average best fitness values are displayed in Fig. 5, which show that NMBDE is superior to DBDE, DBPSO, BAS and MBDE on all eight high-dimensional benchmark functions. For f_5, f_{17}, f_{18}, f_{19} and f_{20}, NMBDE searched out the global optima with 100% success rate with a faster convergence speed as well as MBDE while DBDE, DBPSO and BAS failed to achieve the global best values. However, the high-dimension numerical optimization is a very difficult problem as the search space is expanded to 2^{600}. NMBDE did not find out the optimal values of f_1, f_8 and f_9, but its solutions are still better than those of DBDE, DBPSO, BAS and MBDE. Therefore, it is fair to claim that NMBDE has the better optimization ability especially for the complicated problems.

5.3. Multidimensional knapsack problem

5.3.1. Problem definition

The multidimensional knapsack problem (MKP), also called the multi-constraint knapsack problem, is a strongly NP-hard combinational optimization problem [32]. The MKP has good practical application background because many real-world applications can be formulated as MKP, the goal of which is to find the optimal subset of items that provides the maximum profit without exceeding any allocated resource. Generally, the MKP can be

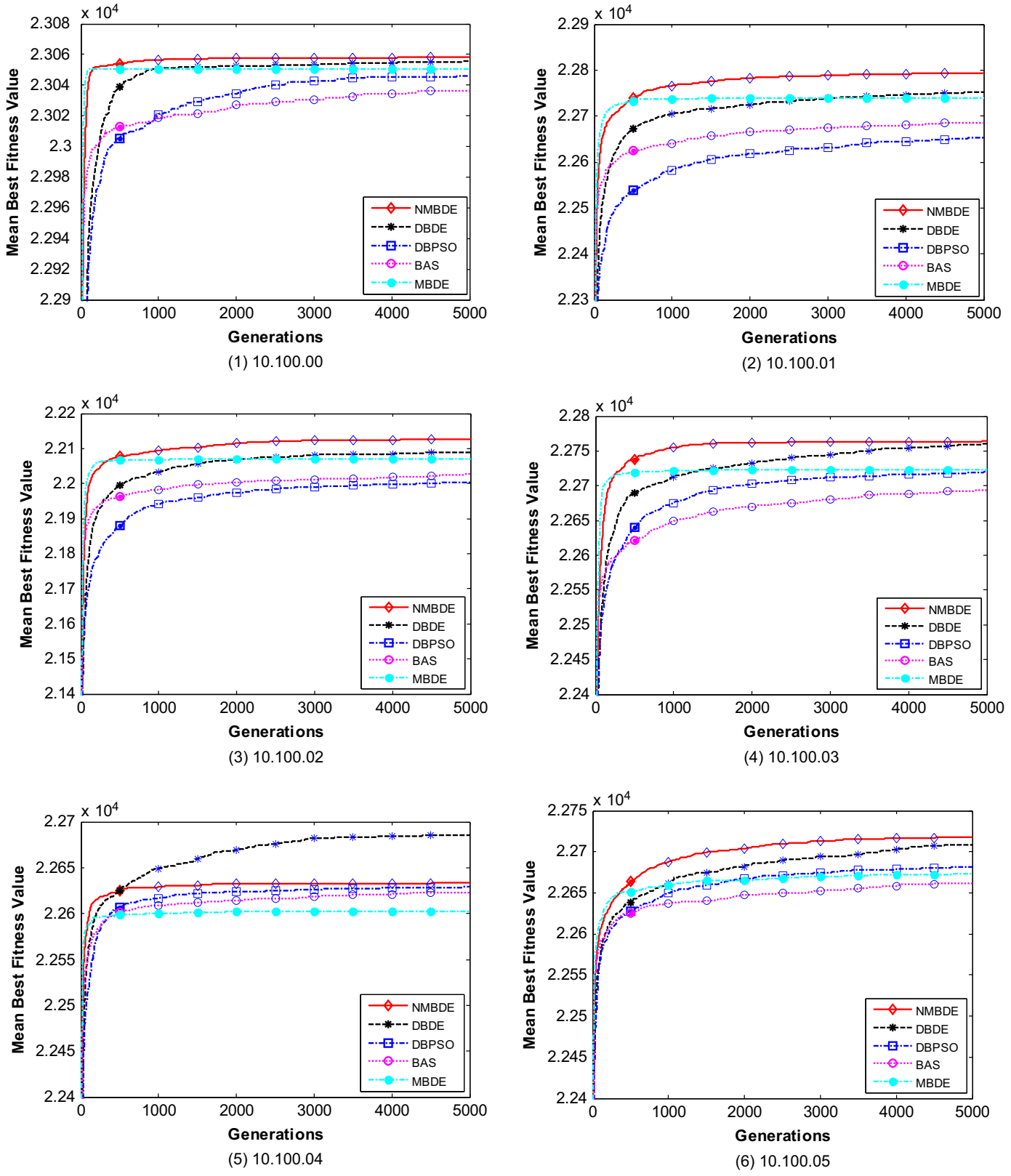
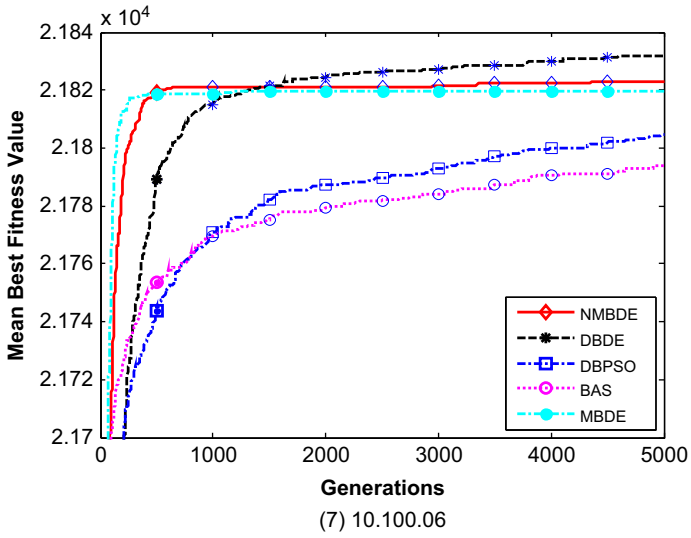
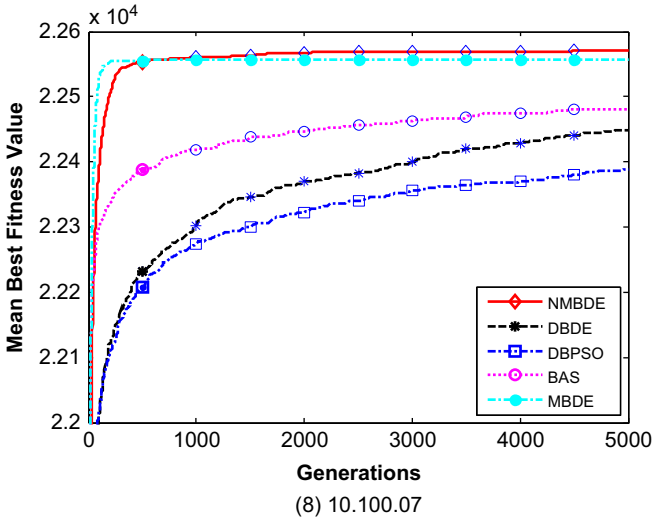


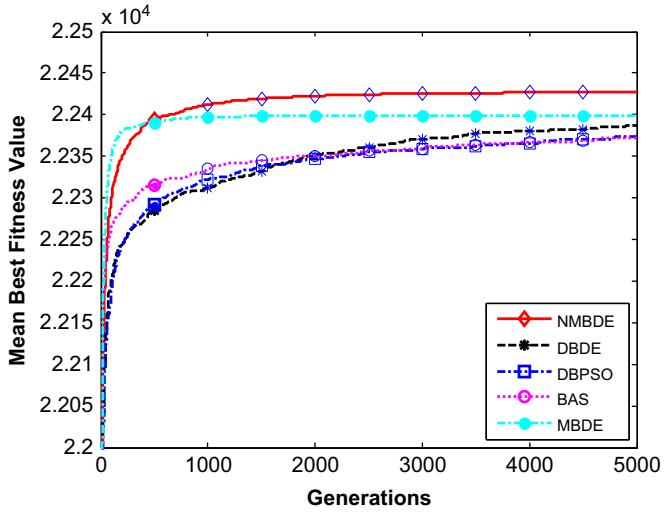
Fig. 6. Convergence curves of NMBDE, DBDE, DBPSO,BAS and MBDE on the MKP instances.



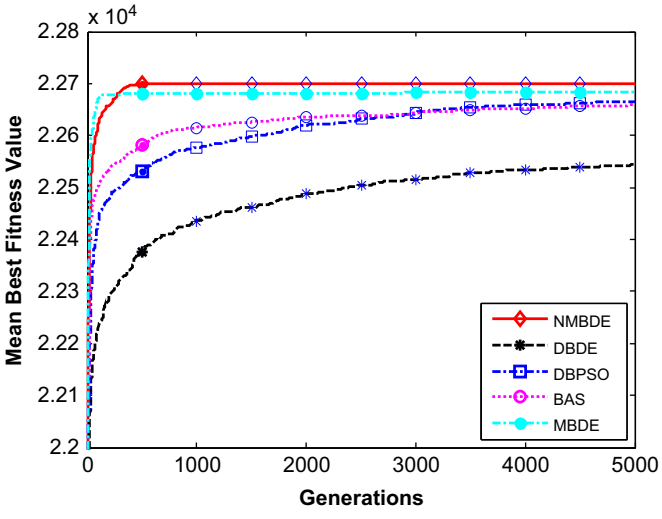
(7) 10.100.06



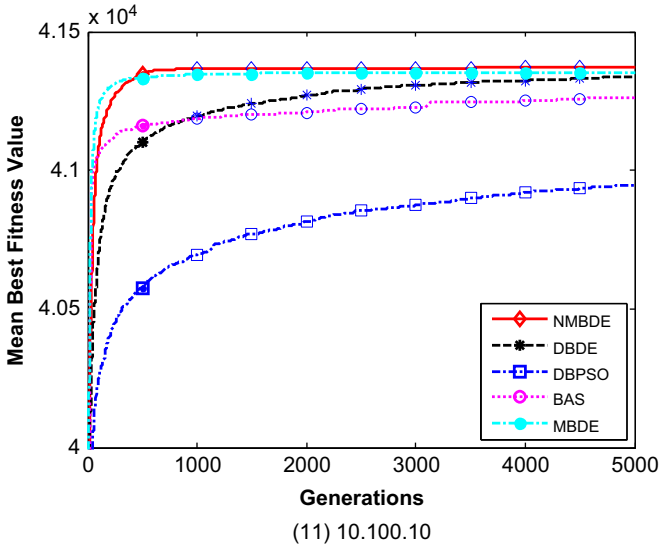
(8) 10.100.07



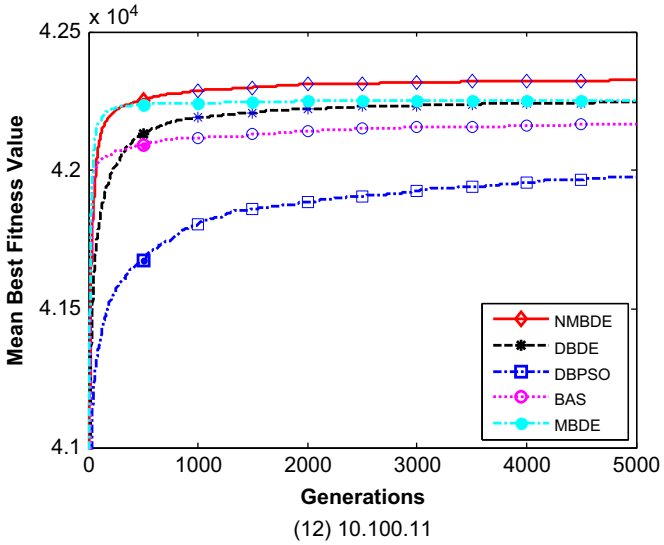
(9) 10.100.08



(10) 10.100.09

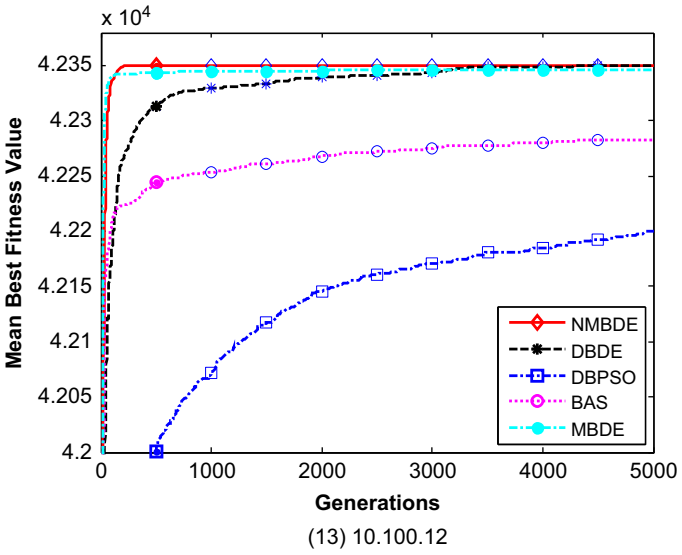


(11) 10.100.10

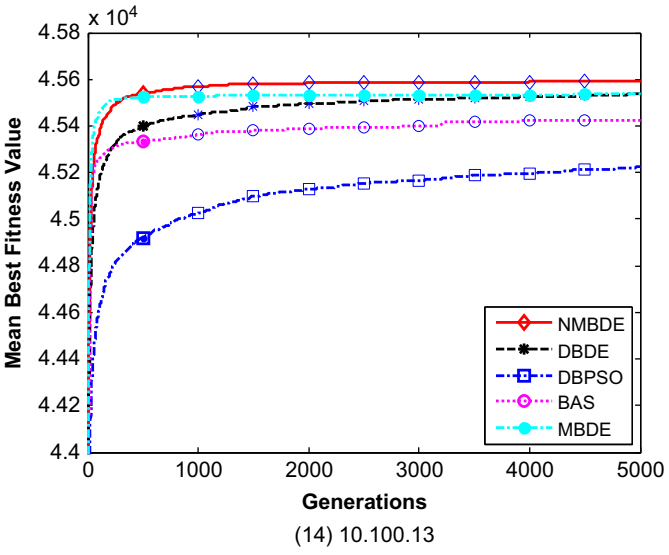


(12) 10.100.11

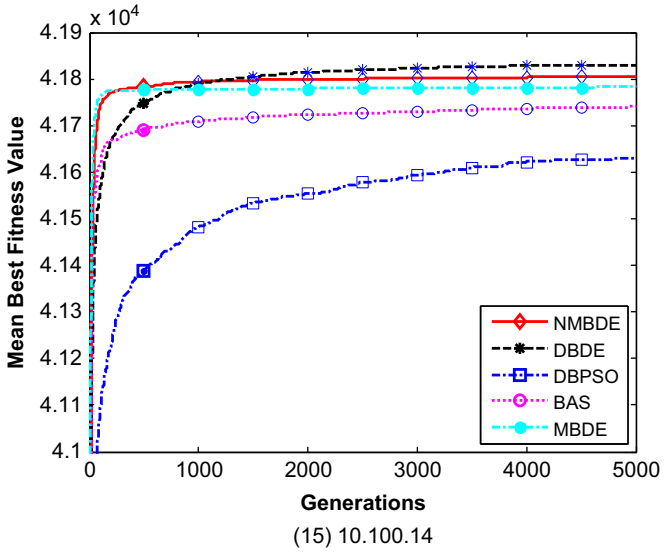
Fig. 6. (*continued*)



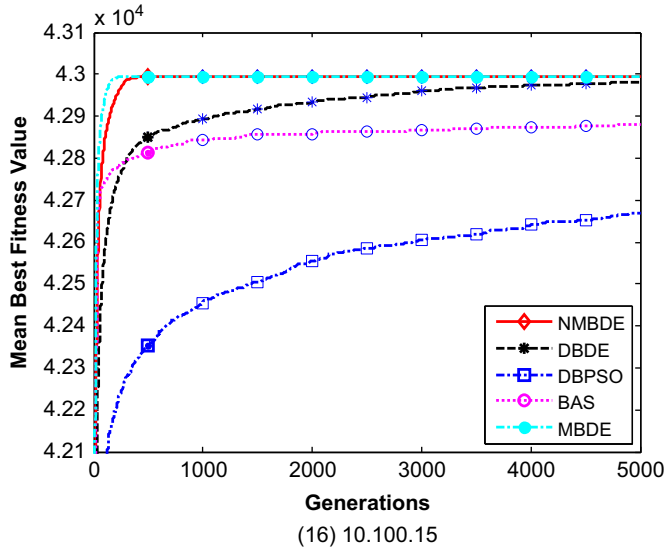
(13) 10.100.12



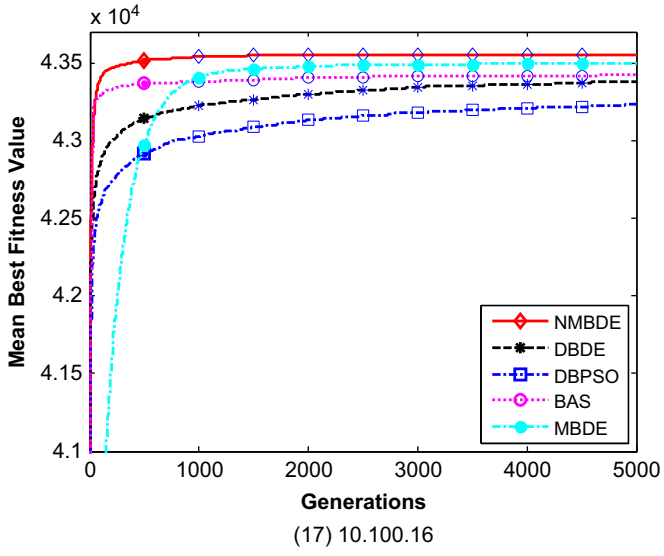
(14) 10.100.13



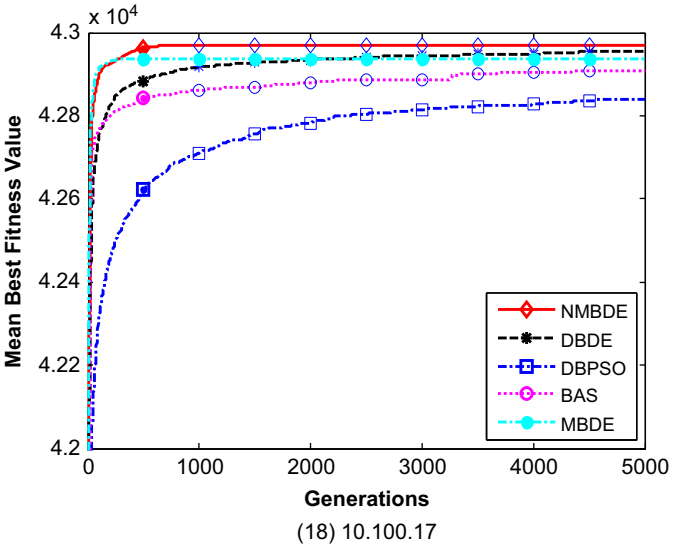
(15) 10.100.14



(16) 10.100.15

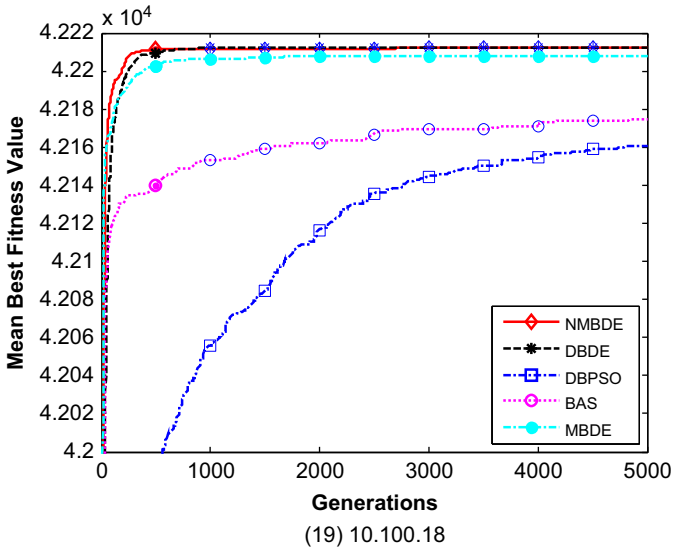


(17) 10.100.16

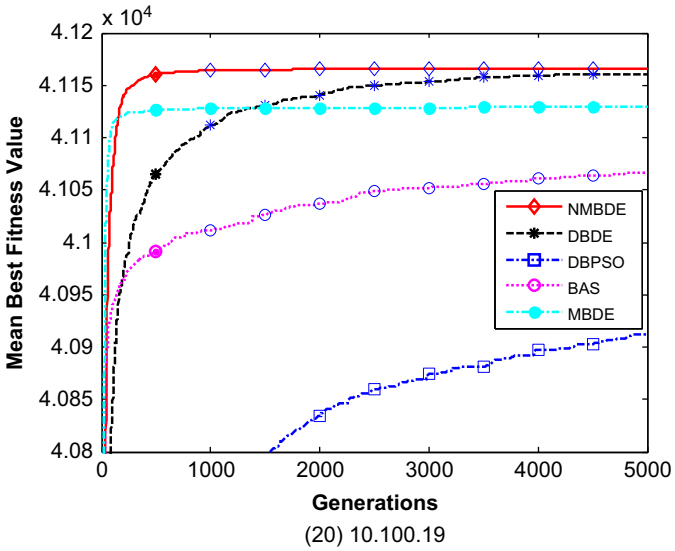


(18) 10.100.17

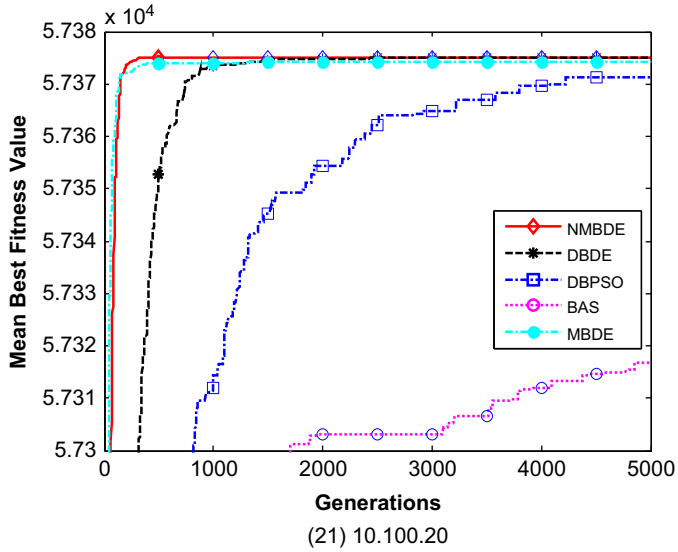
Fig. 6. (*continued*)



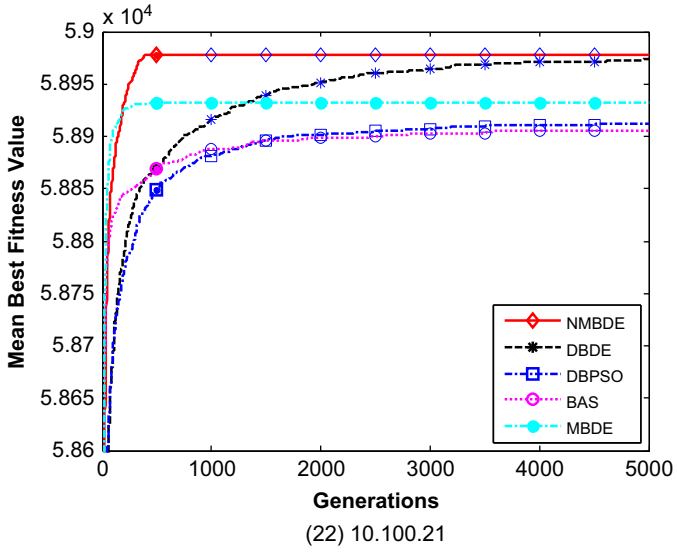
(19) 10.100.18



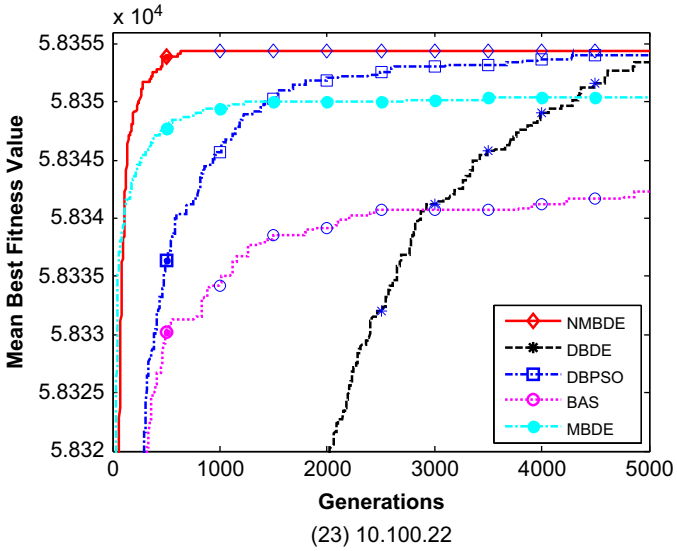
(20) 10.100.19



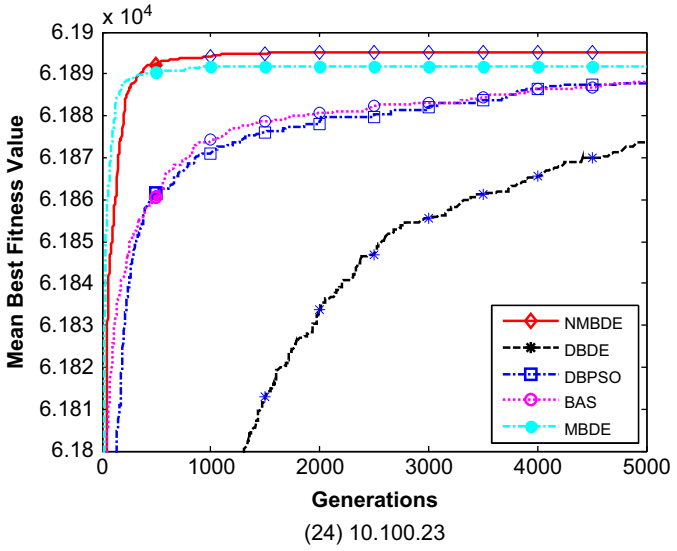
(21) 10.100.20



(22) 10.100.21

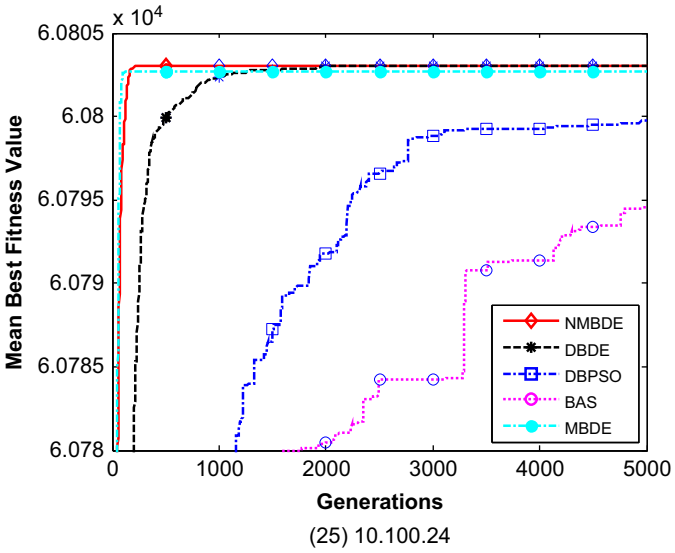


(23) 10.100.22

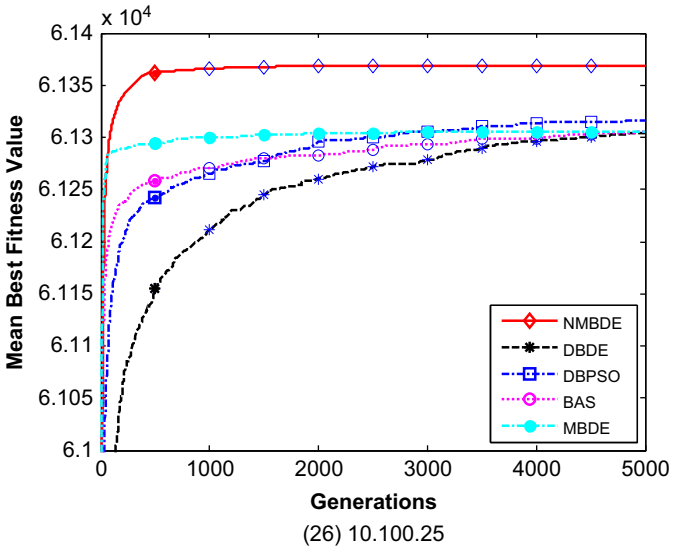


(24) 10.100.23

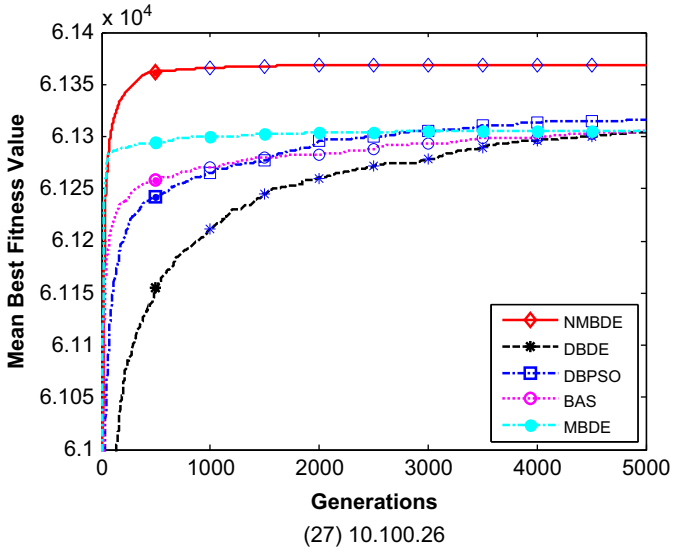
Fig. 6. (*continued*)



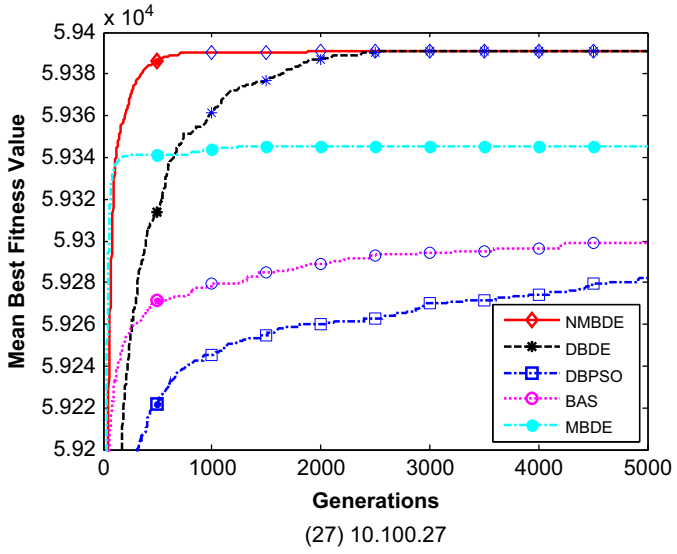
(25) 10.100.24



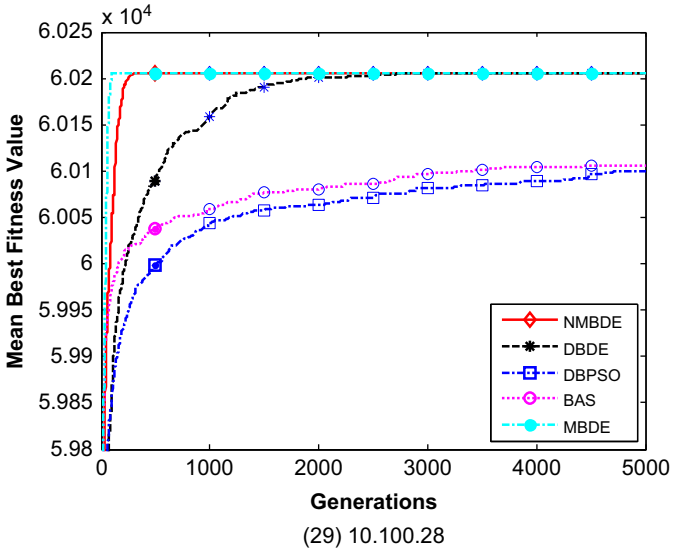
(26) 10.100.25



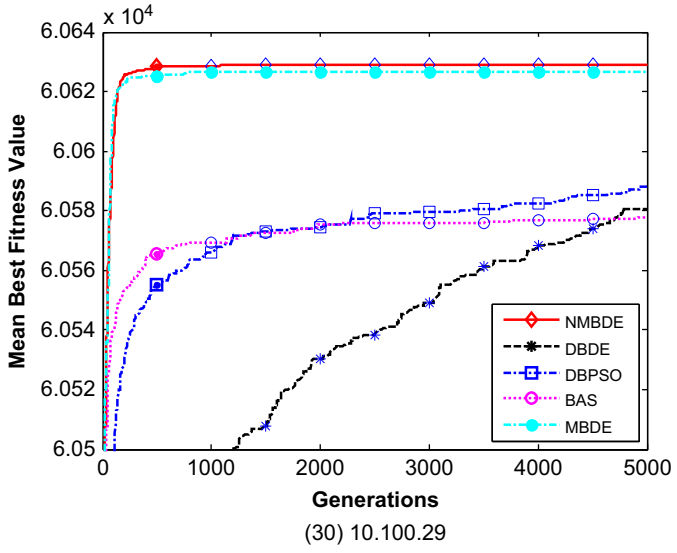
(27) 10.100.26



(27) 10.100.27



(29) 10.100.28



(30) 10.100.29

Fig. 6. (*continued*)

defined as the following equations:

$$\text{maximize } f(x)=\sum_{j=1}^{n} p_j x_j \tag{8}$$

$$\text{Subject to } \sum_{j=1}^{n} w_{ij}x_j \le M_i$$
$$\text{with } p_j>0,\ w_{ij}\ge 0,\ M_i\ge 0,\ x_j\in 0,1,\ i=1,\ldots,m;\ j=1,\ldots,n \tag{9}$$

where n is the number of items; m is the number of knapsack constraints with maximum capacities; M_i is the *i-th* maximal resource budget; w_{ij} is the consumption of the i-th resource by the j-th item; p_j is the profit yielded by the j-th item. The binary decision variable x_j decides the items to be selected. Eq. (8) is the objective function representing the total profit of the selected items and Eq. (9) is the multiple resource constraints.

To deal with the constraints, several methods have been studied and used in evolutionary algorithms to solve MKP [33], such as the repair operator [34,35] and penalty function [36]. The repair operator involving the surrogate relaxation method can transform infeasible solutions into feasible solutions so that it extend the feasible domain and improve the quality of feasible solution which is used in this work.

5.3.2. Results and discussions

To investigate the performance of NMBDE, 10.100 MKP instances in the *OR-library* are used as the benchmarks. The control parameters of NMBDE, DBDE, DBPSO, BAS and MBDE are the values given in Section 5.1, but the population size was set to the twice of item number, i.e., $NP=2n$. The maximum number of iterations was $Gmax=5000$ and 100 independent runs were executed for each MKP instance. The experiment results are shown in Table 7, and the average convergence curves are drawn in Fig. 6.

The results in Table 7 indicate that NMBDE has the best performance on solving the MKP and only NMBDE can search out the global optima on all the 30 instances. Moreover, NMBDE outperforms DBPSO and BAS on all instances while it is inferior to DBDE and MBDE on 3 and 2 cases, respectively. In Fig. 6, we can clearly observe that NMBDE offers the better solution quality and converges to the optima faster than DBDE, DBPSO, BAS and MBDE on most instances.

6. Conclusion

This paper presents a novel modified binary DE algorithm for solving binary-coded optimization problems, which is inspired by the concept of Estimation of Distribution Algorithm and standard DE. The updating strategy of the standard DE is reserved in the proposed NMBDE so that the advantages of DE, such as easy implementation and parameter tuning, are inherited. Furthermore, the proposed probability estimation operator can effectively preserve the diversity of population and enhance the global search ability. The efficiency and effectiveness of NMBDE were verified on low-dimensional numerical optimization problems, high-dimensional numerical optimization problems and MKP. The comparisons with discrete binary PSO, binary Ant System, the discrete binary DE and the modified binary DE on the benchmarks demonstrate that NMBDE has the best performances in terms of the search accuracy and convergence speed, especially on the complicated optimization problems.

Acknowledgment

This work is supported by Research Fund for the Doctoral Program of Higher Education of China (20103108120008), the Projects of Shanghai Science and Technology Community (10ZR1411800, 10JC1405000 & 08160512100), National Natural Science Foundation of China (Grant no. 60804052), ChenGuang Plan (2008CG48), Shanghai University "11th Five-Year Plan" 211 Construction Project, Mechatronics Engineering Innovation Group project from Shanghai Education Commission and the Graduate Innovation Fund of Shanghai University (SHUCX102218 and SHUCX112171).

Appendix. Benchmark functions

(1) Rosenbrock's function:

$$\min f_1=\sum_{i=1}^{n-1}[100\cdot(x_i^2-x_{i+1})^2+(1-x_i)^2],$$
$$\text{subject to } -2.048\le x_i\le 2.048.$$

The global minimum is located at $(1,1,\ldots,1)$. $f_1^*(1,1,\ldots,1)=0$.

(2) Goldstein & price problem:

$$\min f_2=[1+(x_1+x_2+1)^2\cdot(19-14\cdot x_1+3\cdot x_1^2-14\cdot x_2+6\cdot x_1\cdot x_2+3\cdot x_2^2)]\cdot[30+(2\cdot x_1-3\cdot x_2)^2\cdot(18-32\cdot x_1+12\cdot x_1^2+48\cdot x_2-36\cdot x_1\cdot x_2+27\cdot x_2^2],$$

subject to $-2\le x_1,x_2\le 2$. The global minimum is located at $(0,-1)$. $f_2^*(0,-1)=3$.

(3) Freudenstein–Roth function:

$$\min f_3=[-13+x_1+((5-x_2)\cdot x_2-2)\cdot x_2]^2+[-29+x_1+((x_2+1)\cdot x_2-14)\cdot x_2]^2,$$
$$\text{subject to } -10\le x_1,x_2\le 10.$$

The global minimum is located at $(5,4)$. $f_3^*(5,4)=0$.

(4) Glankwahmdee's function:
$\min f_4=(x_1^2+x_2-11)^2+(x_1+x_2^2-7)^2$, subject to $-10\le x_1, x_2\le 10$. The global minimum is located at $(3, 2)$. $f_4^*(3,2)=0$.

(5) Griewangk's function:

$$\min f_5=\sum_{i=1}^{n}(x_i^2/4000)-\prod_{i=1}^{n}\cos(x_i/\sqrt{i})+1,$$
$$\text{subject to } -10\le x_i\le 10.$$

The global minimum is located at the origin $(0, 0, \ldots, 0)$. $f_5^*(0,0,\cdots,0)=0$.

(6) Schaffer F6:

$$\min f_6=-0.5-\left[\left(\sin\sqrt{x^2+y^2}\right)^2-0.5\right]/[1+0.001\cdot(x^2+y^2)]^2,$$
$$\text{subject to } -10\le x_1,x_2\le 10.$$

The global maximum is located at the origin. $f_6^*(0,0,\cdots,0)=-1$.

(7) Shubert problem:

$$\min f_7=\left\{\sum_{i=1}^{5} i\cdot\cos[(i+1)\cdot x+i]\right\}\cdot\left\{\sum_{i=1}^{5} i\cdot\cos[(i+1)\cdot y+i]\right\},$$
$$\text{subject to } -10\le x_1,x_2\le 10.$$

There are 18 global minimal solutions. $f_7^*\approx -186.730908$

(8) Dixon function:

$$\min f_8=(x_1-1)^2+\sum_{i=2}^{n} i(2x_i^2-x_{i-1})^2,$$
$$\text{subject to } -10\le x_i\le 10.$$

(9) The global minimum is located at the origin. $f_8^*(1,0,\cdots,0)=0$
Pathological function:

$$\min f_9=\sum_{i=1}^{n-1}\left(0.5+\frac{\sin^2\sqrt{100x_i^2+x_{i+1}^2}-0.5}{1+0.001(x_i^2-2x_ix_{i+1}+x_{i+1}^2)^2}\right),$$
$$-100\le x_1,x_2\le 100.$$

The global minimum is located at (0, 0, …, 0). $f_9^*(0,0,\cdots,0)=0$

(10) Ackley's function:

$$\min f_{10}=-20\cdot e^{-0.2\cdot\sqrt{(1/n)\cdot\sum_{j=1}^{n}x_j^2}}-e^{(1/n)\cdot\sum_{j=1}^{n}\cos(2\cdot\pi\cdot x_j)}+20+e,$$
subject to $-32\le x_i\le 32$.

The global minimum is located at (0, 0, …, 0). $f_{10}^*(0,0,\cdots,0)=0$

(11) Levy function 5:

$$\min f_{11}=\left\{\sum_{i=1}^{5}i\cdot\cos[(i-1)\cdot x+i]\right\}\cdot\left\{\sum_{i=1}^{5}i\cdot\cos[(i+1)\cdot y+i]\right\}$$
$$+[(x+1.42513)^2+(y+0.80032)^2],$$
subject to $-10\le x_1,x_2\le 10$.

The global minimum is located at $(-1.3068,\ -1.4248)$. $f_{11}^*(-1.3068,-1.4248)\approx -176.1375$

(12) Camel back-6 six hump problem:

$$\min f_{12}=\left(4-2.1\cdot x_1^2+\frac{x_1^4}{3}\right)\cdot {x_1}^2+x_1\cdot x_2+(-4+4\cdot x_2^2)\cdot x_2^2,$$
subject to $-10\le x_1,x_2\le 10$.

There are four global minimal solutions $(\pm 0.0898,\ \pm 0.7126)$. $f_{12}^*\approx -1.03162845$.

(13) Alpine function:

$$\min f_{13}=-\left[\prod_{i=1}^{n}\sin(x_i)\right]\cdot\sqrt{\prod_{i=1}^{n}(x_i)},\quad \text{subject to } 0\le x_i\le 10.$$

The global maximum is located at (7.917, 7.917). $f_{13}^*(7.917,7.917)\approx -7.8856$.

(14) Levy function 3:

$$\min f_{14}=\left\{\sum_{i=1}^{5}i\cdot\cos[(i-1)\cdot x+i]\right\}\cdot\left\{\sum_{i=1}^{5}i\cdot\cos[(i+1)\cdot y+i]\right\},$$
subject to $-10\le x_1,x_2\le 10$.

There are 18 global minimal solutions. $f_{14}^*\approx -174.5417$.

(15) Beale function:

$$\min f_{15}=(1.5-x_1+x_1x_2)^2+(2.25-x_1+x_1x_2^2)^2$$
$$+(2.625-x_1+x_1x_2^3)^2,$$
subject to $-4.5\le x_1,x_2\le 4.5$.

The global minimum is located at (3,0.5). $f_{15}^*(3,0.5)=0$.

(16) Hartman 3 problem:

$$\min f_{16}=-\sum_{i=1}^{4}\left(c_ie^{-\sum_{j=1}^{3}a_{ij}(x_j-p_{ij})}\right),\quad \text{subject to } 0\le x_{ij}\le 1$$

$$c=\begin{bmatrix}1\\1.2\\3\\3.2\end{bmatrix},\quad a=\begin{bmatrix}3&10&30\\0.1&10&35\\3&10&30\\0.1&10&35\end{bmatrix},$$

$$p=\begin{bmatrix}0.3689&0.117&0.2673\\0.4699&0.4387&0.747\\0.1091&0.8732&0.5547\\0.03815&0.5743&0.8828\end{bmatrix},$$

The global minimum is located at (0.114614, 0.555649, 0.852547).

$f_{16}^*(0.114614,0.555649,0.852547)=-3.86278$.

(17) Sphere function:
$\min f_{17}=\sum_{i=1}^{n}x_i^2$, subject to $-10\le x_i\le 10$. The global minimum is located at the origin. $f_{17}^*(0,0,\cdots,0)=0$

(18) Sum squares function:
$\min f_{18}(x)=\sum_{i=1}^{n}ix_i^2$, subject to -5.12 $\le x_i\le$ 5.12. The global minimum is located at the origin.$f_{18}^*(0,0,\cdots,0)=0$

(19) Sum of different power:
$\min f_{19}=\sum_{i=1}^{n}|x_i|^{(i+1)}$, subject to $-1\le x_i\le 1$. The global minimum is located at the origin.$f_{19}^*(0,0,\cdots,0)=0$

(20) Schwefel's poblem 2.22:

$$\min f_{20}=\sum_{i=1}^{n}|x_i|+\prod_{i=1}^{n}|x_i|,\quad \text{subject to } -10\le x_i\le 10.$$

The global minimum is located at the origin. $f_{20}^*(0,0,\cdots,0)=0$

References

[1] R. Storn, K.V. Price, Differential Evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces, in: Technology Report. Berkeley, CA, TR-95-012, 1995.
[2] J. Vesterstrom, R. Thomsen, A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems, IEEE Congress Evol. Comput. (2004) 1980–1987.
[3] I.T. Rekanos, Shape reconstruction of a perfectly conducting scatterer using differential evolution and particle swarm optimization, IEEE, Trans. Geosci. Remote Sensing 46 (2008) 1967–1974.
[4] A. Ponsich, C.A.C. Coello, Differential evolution performances for the solution of mixed-integer constrained process engineering problems, Appl. Soft Comput. 11 (2011) 399–409.
[5] J. Liu, J. Lampinen, A. Fuzzy, Adaptive differential evolution algorithm, Soft Comput. 9 (2004) 448–462.
[6] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2009) 398–417.
[7] J.Q. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (2009) 945–958.
[8] X.B. Zhu, Y. Qian, X.J. Wang, A hybrid differential evolution algorithm for solving nonlinear bilevel programming with linear constraints, in: 5th IEEE International Congress on Cognitive Informatics, ICCI'06, 2006, pp. 126–131.
[9] C. Zhang, J. Ning, S. Lu, D. Ouyang, T. Ding, A novel hybrid differential evolution and particle swarm optimization algorithm for unconstrained optimization, Oper. Res. Lett. 37 (2009) 117–122.
[10] L. Wang, L.B. Li, Hybrid algorithms based on harmony search and differential evolution for global optimization, in: the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation, GEC'09, Shanghai, 2009, pp. 271–278.
[11] X.Z. Gao, X. Wang, S.J. Ovaska, Fusion of clonal selection algorithm and differential evolution method in training cascade–correlation neural network, Neurocomputing 72 (2009) 2483–2490.
[12] J. Sun, Q. Zhang, E. Tsang, DE/EDA: a new evolutionary algorithm for global optimization, Inform. Sci. 169 (2005) 249–262.
[13] N. Noman, H. Iba, Accelerating differential evolution using an adaptive local search, IEEE Trans. Evol. Comput. 12 (2008) 107–125.
[14] S. Rahnamayan, H.R. Tizhoosh, M.M.A. Salama, Opposition-based differential evolution, IEEE Trans. Evol. Comput. 12 (2008) 64–79.
[15] S. Das, A. Abraham, U. Chakraborty, A. Konar, Differential evolution using a neighborhood-based mutation operator, IEEE Trans. Evol. Comput. 13 (2009) 526–553.
[16] R. Storn, Designing nonstandard filters with differential evolution, IEEE Signal Process. Mag. 22 (2005) 103–106.
[17] Y. Guang Ya, D. Zhao Yang, W. Kit Po, A modified differential evolution algorithm with fitness sharing for power system planning, IEEE, Trans. Power Syst. 23 (2008) 514–522.

[18] J. Du, D. Huang, X. Wang, X. Gu, Shape recognition based on neural networks trained by differential evolution algorithm, Neurocomputing 70 (2006) 896–903.
[19] G.W. Greenwood, Using differential evolution for a subclass of graph theory problems, IEEE, Trans. Evol. Comput. 13 (2009) 1190–1192.
[20] U.K. Chakraborty, Advances in Differential Evolution, Springer, Berlin, 2008.
[21] S. Das, P.N. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (2011) 4–31.
[22] X. He, L. Han, A novel binary differential evolution algorithm based on artificial immune system, in: IEEE Congress on Evolutionary Computation, CEC 2007, 2007, pp. 2267–2272.
[23] C.Y. Wu, K.Y. Tseng, Topology optimization of structures using modified binary differential evolution, Struct. Multidisciplinary Optim. 42 (2010) 939–953.
[24] T. Gong, A.L. Tuson, Differential evolution for binary encoding, in: Soft Computing in Industrial Applications, ASC 39, 2007, pp. 251–262.
[25] G. Pampara, A.P. Engelbrecht, N. Franken, Binary differential evolution, in: IEEE Congress on Evolutionary Computation, CEC 2006, 2006, pp. 1873–1879.
[26] P. Chen, J. Li, Z. Liu, Solving 0-1 knapsack problems by a discrete binary version of differential evolution, in: Second International Symposium on Intelligent Information Technology Application, IITA'08, 2008, pp. 513–516.
[27] J. Kennedy, R.C. Eberhart, A discrete binary version of the particle swarm algorithm, in: Proceedings of IEEE International Conference on Computation, Cybernetics and Simulation, System, Man and Cybernetics, 1997, pp. 4104–4108.
[28] M. Kong, P. Tian, A binary ant colony optimization for the unconstrained function optimization problem, in: Computational Intelligence and Security (CIS 2005), Part I, LNAI 3801, Springer, 2005, pp. 682–687.
[29] S. Baluja, Population-based incremental learning: a method for integrating genetic search based function optimization and competitive learning, Report, 1994.
[30] R. Gämperle, S.D. Müller, P. Koumoutsakos, A parameter study for differential evolution, in: WSEAS International Conference on Advances in Intelligent System, Fuzzy System, Evolutionary Computation, 2002, pp. 293–298.
[31] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer., Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, IEEE Trans. Evol. Comput. 10 (2006) 646–657.
[32] A. Fréville, The multidimensional 0-1 knapsack problem: an overview, Eur. J. Oper. Res. 155 (2004) 1–21.
[33] C.A.Coello Coello, Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state-of-the art, Comput. Methods Appl. Mech. Eng. 191 (2002) 1245–1287.
[34] P.C. Chu, J.E. Beasley, A genetic algorithm for the multidimensional knapsack problem, J. Heuristics 4 (1998) 63–86.
[35] V. Boyer, M. Elkihel, D. El Baz, Heuristics for the 0-1 multidimensional knapsack problem, Eur. J. Oper. Res. 199 (2009) 658–664.
[36] A.L. Oken, Penalty functions and the knapsack problem, in: the first IEEE Conference on Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence, Orlando, FL, USA, 1994, pp. 554–558.

Ling Wang is an associate professor in Control Theory and Control Engineering at Shanghai University. He received B.Sc. degree in Automation from East China University of Science and Technology in 2002 and Ph.D. degree in control theory and control engineering from East China University of Science and Technology in 2007. His research interests include intelligent optimization algorithm design and automatic control.

Xiping Fu received his B.Sc. degree in Automation from Shanghai University in 2008. Currently he is a postgraduate student major in Control Science and Engineering at Shanghai University. His research interests include evolutionary computation and industrial wireless sensor networks.

Yunfei Mao is currently a postgraduate student major in Control Science and Engineering at Shanghai University. He received B.Sc. degree in Measurement and Control Technology and Instrumentation Program from Yangzhou University in 2009. His research interests include industrial wireless sensor networks and evolutionary computation.

Muhammad Ilyas Menhas received his B.Sc in electrical engineering from University of Azad Jammu and Kashmir in 2002. He has been an assistant engineer with Azad Jammu and Kashmir Electricity Department. Currently he is doctoral student at Shanghai University. His research interests include intelligent optimization algorithms, automation and control.

Minrui Fei received his Ph.D. degree in Control Theory and Control Engineering from Shanghai University in 1997. Since 1998, he has been a professor and doctorial supervisor in Shanghai University. His current research interests are in the areas of intelligent control, networked learning control systems, wireless sensor networks, etc. He is a vice-chairman of Chinese Association for System Simulation, a standing director of China Instrument & Control Society, and a director of Chinese Artificial Intelligence Association.