Applied Soft Computing Journal 74 (2019) 679–692

Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

Artificial bee colony algorithm based on Parzen window method

Weifeng Gao [a,*], Zhifang Wei [a], Yuting Luo [a], Jin Cao [b]

[a] School of Mathematics and Statistics, Xidian University, Xi'an 710126, China
[b] School of Cyber and Engineering, Xidian University, Xi'an 710126, China

HIGHLIGHTS

- Three diverse popular search strategies are adopted to build the strategy candidate pool.
- The Parzen window method is employed to evaluate the quality of candidate individuals.
- Two different neighborhood mechanisms are adopted.
- Numerical simulation demonstrates the effectiveness of the proposed algorithm.

ARTICLE INFO

Article history:
Received 17 June 2018
Received in revised form 1 October 2018
Accepted 10 October 2018
Available online 29 October 2018

Keywords:
Parzen window method
Artificial bee colony algorithm
Strategy candidate pool
Neighborhood mechanism

ABSTRACT

Artificial Bee Colony (ABC) algorithm, based on the metaphor in foraging behavior of honey fee swarm, has been repeatedly criticized for its poor convergence, due to its known exploration bias. In order to enhance the performance of ABC, the paper develops a novel approach (named ABCPW). First, three popular search strategies with different characteristics are employed to construct a strategy candidate pool for obtaining high quality candidate individuals. Next, to cut down on computational cost, the Parzen window method is applied to estimate these candidate individuals and then select one as the offspring. In addition, two different neighborhood mechanisms are adopted to balance the convergence and the population diversity. Finally, the performance of ABCPW is tested on a series of benchmark functions. The experimental results not only demonstrate the stability and convergence of ABCPW, but also show ABCPW outperforms several popular algorithms.

1. Introduction

Since optimization problems arise in industrial production and daily life, optimization techniques have been attracting more and more attention of researchers. As a matter of fact, a number of real-world optimization problems are very difficult to handle due to their complex characteristics, such as discontinuity, nonlinearity, multi-extremum, and non-differentiability. The traditional gradient-based methods tend to be powerless to deal with these problem. Under such circumstances, motivated by the natural law of survival of the fittest, evolutionary algorithms (EAs) were proposed and widely applied to tackle these problems. Until now, many EAs are developed, such as genetic algorithm (GA) [1], ant colony optimization (ACO) [2], differential evolution (DE) [3], biogeography-based optimization (BBO) [4], particle swarm optimization (PSO) [5], artificial bee colony algorithm (ABC) [6], covariance matrix adaptation evolution strategy (CMA-ES) [7] and so on.

In the paper, ABC is studied, which was proposed by Karaboga [6] under the inspiration of the foraging behavior of bees. The experimental results indicated that the performance of ABC is better than or at the least equivalent to the other EAs [8–11]. Owing to its concise structure, superior performance and easy to implement, ABC has been used to deal with numerous engineering optimization problems, such as timetabling [12], chaotic systems [13], optimal filter design [14], and so on [15].

However, ABC also suffers from some drawbacks like other EAs, the biggest one of which is poor convergence. This is because both the search direction and the step size have strong randomicity. This may result in that the search strategy of ABC prefers the exploration to the exploration. However, the two aspects should be balanced perfectly for constructing an efficient algorithm. Therefore, it is necessary to seek a right tradeoff between the exploitation ability and the exploration ability.

Recently, various methods which combine different search strategies are presented to improve the search ability of the algorithm. The most widely used method is the probability model-based method. In this method, one search strategy is selected based on the probability which is calculated according to the previous successful experience. For example, the reference [16] developed an adaptive DE (called SaDE), in which one mutation strategy is selected from the strategy candidate pool based on the previous

* Corresponding author.
E-mail address: gaoweifeng2004@126.com (W. Gao).

https://doi.org/10.1016/j.asoc.2018.10.024

Step 1) Initialization:

Step 1.1) Randomly generate *SN* points in the search space to form an initial population.

Step 1.2) Evaluate the objective function values of the population.

Step 1.3) $FES= SN$.

Step 2) The employed bee phase:

For $i=1,\ldots SN$, do

Step 2.1)

Step 2.1.1) Generate a candidate solution V_i by Eq. (2).

Step 2.1.2) Evaluate $f(V_i)$ and set $FES=FES+1$.

Step 2.2) If $f(V_i)<f(X_i)$, set $X_i=V_i$, $trial_i=1$, otherwise, set $trial_i= trial_i+1$.

Step 3) Calculate the probability values p_i by Eq. (3), set $t = 0$, $i = 1$.

Step 4) The onlookers phase:

While $t<= SN$, do

Step 4.1)

If $rand(0,1)< p_i$

Step 4.1.1) Generate a candidate solution V_i by Eq. (2).

Step 4.1.2) Evaluate $f(V_i)$ and set $FES=FES+1$.

Step 4.1.3) If $f(V_i)<f(X_i)$, set $X_i=V_i$, $trial_i=1$, otherwise, set $trial_i= trial_i+1$.

Step 4.1.4) Set $t=t+1$.

End If

Step 4.2) Set $i=i+1$, if $i= SN$, set $i=1$.

Step 5) The scouts phase:

If $max(trial_i)>limit$, replace X_i with a new randomly produced solution by Eq. (1).

Step 6) If $FES>= Max.FES$, stop and output the best solution achieved so far, otherwise, go to Step 2.

Fig. 1. Framework of ABC.

successful experience. Similar to SaDE, Wang et al. [17] presented a multi-strategy ensemble ABC. In this method, the distinct search strategies construct a strategy candidate pool to generate offspring and the search strategy is dynamically selected based the quality of new generated candidate solutions. However, how to construct an appropriate probability model to select one strategy remains a challenge [18].

Instead of choosing one strategy based on the probability model for each individual to generate one candidate individual, the multistrategy technique employs multiple search strategies for each individual to generate multiple candidate individuals at the same time, and then the best one is selected as the offspring. However, this technique increases computational cost since the algorithm needs to evaluate multiple candidate individuals for each individual in each generation. Concerning this issue, this paper employs a density estimation method to estimate these candidate individuals and then select one as the offspring. What is more, to generate high quality candidate individuals, three popular search strategies with different characteristics which are applied to build a search strategy candidate pool and two neighborhood mechanisms are introduced.

Based on this consideration, a novel approach (named ABCPW) is developed in this paper which is based on multistrategy technique, density estimation method and neighborhood mechanism. To test the performance of ABCPW, the experiments are conducted on a set of benchmark functions. The experimental results indicate that the approach proposed in the paper has obvious advantages over several popular algorithms in terms of solution quality, convergence rate, and numerical stability.

The rest of this paper is structured as follows. The framework of classic ABC and some studies on modified ABC are introduced in Section 2. Section 3 describes in detail the proposed framework. Next, the experimental studies on a set of test instances are carried out in Section 4. Finally, Section 5 concludes this paper.

2. ABC and related work

2.1. ABC framework

Inspiration from the foraging behavior of honey bees, Karaboga proposed a relatively new heuristic optimization method — ABC [6]. ABC involves three phases: the scout phase, the onlooker phase and the employed bee phase. The framework of classical ABC is presented in Fig. 1.

ABC initializes a population of P with SN individuals in a D-dimensional search space. Each initial individual $X_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$ is generated as follow.

$$x_{i,j} = x_{min,j} + rand(0, 1)(x_{max,j} - x_{min,j}), \tag{1}$$

where $j = 1, 2, \ldots, D$, $i = 1, 2, \ldots, SN$, $x_{max,j}$ and $x_{min,j}$ are the upper and lower bounds for the jth dimension, respectively.

After the initialization, a candidate individual V_i is generated by the following solution search strategy.

$$v_{i,j} = x_{i,j} + \phi_{i,j}(x_{i,j} - x_{k,j}), \tag{2}$$

where $j \in \{1, 2, \ldots, D\}$ and $k \in \{1, 2, \ldots, SN\}$ are randomly selected indexes; $\phi_{i,j}$ is a uniform random number in $[-1, 1]$; k is different from i.

The probability value p_i of an individual is calculated as follows:

$$p_i = fit_i / \sum_{j=1}^{Np} fit_j, \tag{3}$$

where fit_i means the fitness value of the ith individual.

In the scout phase, when an individual cannot be further improved after the *limit* cycles, it is discarded. Then, a new random individual is produced by Eq. (1).

2.2. Improved ABCs

Owing to the attractive properties of ABC, it in recent years has witnessed a boom in development and a mass of ABC variants have been proposed. Next, a brief survey of the studies on enhanced ABC variants is presented, which can be roughly classified into two categories.

The first category covers different approaches to develop a new search strategy. For example, Zhu and Kwong [19] exploited a novel variant of ABC, named as GABC, by introducing the potential information hidden in the current best solution to accelerate the convergence. Moreover, Li et al. [20] presented a novel method which introduces the inertia weight and the acceleration coefficients into ABC. Banharnsakun et al. [21] suggested a new search strategy for onlookers. In the method, the useful message contained in the current best individual is exploited to enhance the convergence rate. Wang et al. [17] proposed a new ABC based on multi-strategy ensemble to achieve a balance between the exploitation and the exploration. Diwold et al. [22] designed two novel variants of ABC in which two novel search mechanisms are adopted. Das et al. [23] introduced an improved algorithm where the saccadic flight strategy is introduced into ABC. Gao et al. [24] developed the novel search strategies to improve the exchange of the information among different subpopulations. Banitalebi et al. [25] developed an improved compact ABC which benefits from the search logic of ABC. Cui et al. [26] presented a novel variant of ABC, in which the rankings of the individuals decide whether the individuals take part the search. Karaboga and Gorkemli [27] exploited a new algorithm where the behavior of onlooker bees are modeled more accurately to improve local search ability. Cui et al. [28] presented a depth-first search mechanism in which two modified search strategies are introduced to the onlooker phase and the employed bee phase, respectively. Inspired by the gravity model, Xiang et al. [29] proposed a novel algorithm in which one force model is used to choose a suitable neighborhood of the current individual to enhance the search capacity.

The second category is often known as the hybrid approaches. For example, Gao et al. [30] combined ABC variants with the orthogonal learning mechanism and proposed a new general algorithmic framework to improve the search ability. Kang et al. [31] exploited a novel hybrid ABC method where the Nelder–Mead simplex method is introduced. Das et al. [32] proposed a novel hybrid algorithm which employs the weighted selection scheme and the fitness learning mechanism. Kang et al. [33] combined ABC with the Rosenbrock method and proposed a novel algorithm. Alatas [34] suggested a chaotic ABC. In the approach, the chaotic map is embedded into the scouts phase and the initialization. Xiang and An [35] introduced three additional operations and presented a novel ABC. Kiran and Gunduz [36] suggested a hybrid method, which combines ABC with PSO based on recombination procedure. Jadon et al. [37] provided a hybridization of DE and ABC which is a platform for exploiting a heuristic method with better convergence rate. Cui et al. [38] presented a novel variant of ABC where the population size can be dynamically updated. Kiran and Findik [39] combined ABC with directional information for accelerating convergence of the method. The experimental results indicate that the proposed method is very effective method. Li and Yang [40] developed a novel ABC variant where a memory mechanism is constructed to retain the previous successful search experiences of the artificial bees. Li and Pan [41] presented a novel hybrid algorithm which combines tabu search with ABC to deal with the flow shop scheduling problem.

However, the studies are not restricted to the above-mentioned two aspects and more related work can refer to [42].

3. Proposed approach

3.1. Basic idea

It is all known that different search equations have different influences on the performance of ABC. However, it is very difficult to select a suitable search equation in different evolutionary stages or for a specific problem. To address this issue, the multistrategy technique is often introduced in the EAs community. The multistrategy technique simultaneously utilizes multiple search strategies for each individual. Then, multiple candidate individuals are generated, and the best one is survived according to the fitness value. A major disadvantage of the multistrategy technique is its expensive computational cost. For the purpose of decreasing the number of fitness evaluations (FES), the Parzen window method is employed to evaluate the quality of candidate individuals and thus the highest quality one is selected as the offspring. Inspired by the work in [18] and based on the above considerations, a new approach is developed in the paper. In this method, multiple candidate individuals are first produced by multiple search strategies for each individual, and then the Parzen window method is employed to estimate these candidate individuals and selects one as the offspring. Besides, two different neighborhood mechanisms are adopted to balance the population diversity and the convergence. The framework of the proposed method is demonstrated in Fig. 2, and more details are illustrated in the following subsections.

3.2. Multiple search strategies

The search strategy has a great influence on the performance of ABC. In general, the different evolutionary stages or the diverse types of the problems need the diverse search strategies according to the evolution process or the properties of problems. The search strategies belonging to the strategy candidate pool ought to contain diverse properties. Thus, these strategies can exhibit predominant performance in different evolutionary stages or for a specific problem. In the paper, three diverse popular search strategies are adopted to build this strategy candidate pool: (1) classic ABC; (2) CABC [30]; and (3) improved ABCbest [43].

The search strategy of classic ABC can be shown in Eq. (2). The detail on the search strategy of improved ABCbest and CABC is shown as follows, respectively.

$$v_{i,j} = x_{best,j} + \phi_{i,j}(x_{best,j} - x_{r_1,j}), \tag{4}$$

$$v_{i,j} = x_{r_1,j} + \phi_{i,j}(x_{r_1,j} - x_{r_2,j}), \tag{5}$$

where $x_{best,j}$ is the jth dimension of the best individual in the whole population; j is a randomly chosen index from $\{1, 2, \ldots, D\}$; r_1 and r_2 are two randomly selected indexes from $\{1, 2, \ldots, SN\}$ and $r_1 \neq r_2 \neq i$; $\phi_{i,j}$ is a randomly selected number from $[-1, 1]$.

Based on the analysis in [30] and [43], the search strategy of classic ABC Eq. (2) has the global exploration bias. While, the search strategy of improved ABCbest Eq. (4) places emphasis on the local exploitation. The search strategy of CABC Eq. (5) has a balance between global search and local search. It is obvious that these search strategies present different characteristics. Therefore, they can show distinct capacities in different evolutionary stages or for a specific problem.

3.3. Parzen window method

There are many density estimation methods in pattern classification field. The Parzen window method is among the most

Input: population size N, the predetermined number of cycles *limit*, the maximum number of function evaluations *Max.FES*, set the counter of the number of function evaluations *FES*=0.

Output: the fitness value of the best solution.

Step 1) Initialization:

Step 1.1) Randomly generate *SN* points in the search space to form an initial population.

Step 1.2) Evaluate the objective function values of the population.

Step 1.3) *FES=SN*.

Step 2) The employed bee phase:

For $i=1,\dots SN$, do

Step 2.1) Use three search strategies, each with a neighborhood form the neighborhood candidate pool, to generate three candidate individuals V_{i_1}, V_{i_2}, and V_{i_3} for X_i.

Step 2.2) Estimate the density of V_{i_1}, V_{i_2}, and V_{i_3}.

Step 2.3) Choose the best one (denote as V_i) as the offspring.

Step 2.4) Evaluate $f(V_i)$, and set $FES=FES+1$.

Step 2.5) If $f(V_i)<f(X_i)$, set $X_i=V_i$, $trial_i=1$, otherwise, set $trial_i= trial_i+3$.

Step 3) The onlookers phase:

Calculate the probability values p_i by Eq. (3), set $t = 0$, $i = 1$.

While $t<= SN$, do

Step 3.1)

If $rand(0,1)< p_i$

Step 3.1.1) Use three search strategies, each with a neighborhood form the neighborhood candidate pool, to generate three candidate individuals V_{i_1}, V_{i_2}, and V_{i_3} for X_i.

Step 3.1.3) Evaluate $f(V_{i\_1})$, $f(V_{i\_2})$ and $f(V_{i\_3})$, and set $FES=FES+3$.

Step 3.1.4) Choose the best one (denote as V_i).

Step 3.1.5) If $f(V_i)<f(X_i)$, set $X_i=V_i$, $trial_i=1$, otherwise, $trial_i= trial_i+3$.

Step 3.1.6) Set $t=t+1$.

End If

Step 3.2) Set $i=i+1$, if $i=SN$, set $i=1$.

Step 4) The scouts phase:

If $max(trial_i)>limit$, replace X_i with a new randomly produced solution by Eq. (1).

Step 5) If $FES>= Max.FES$, stop and output the best solution achieved so far, otherwise, go to Step 2.

Fig. 2. Framework of ABCPW.

representative density estimation methods, where the density estimation is computed by the following equation.

$$\tilde{F}(Y) = \frac{1}{\mu}\sum_{i=1}^{\mu}(\frac{r_i}{\mu}\frac{1}{w}\varphi(\frac{\|Y - X_i\|}{w})) \tag{6}$$

where

(1) r_i denotes the rank of the ith individual which is sorted in descending order based on the fitness.

(2) w means the window width which can be computed by the following equation.

$$w = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(\bar{a}_j - \underline{b}_j)^2} \tag{7}$$

where $\bar{a}_j = arg\ max_{i=1,\dots,\mu}\ x_{i,j}$ and $\underline{b}_j = arg\ min_{i=1,\dots,\mu}\ x_{i,j}$.

(3) $\varphi(u)$ is the *Epanechnikov Kernel* which is computed as follows:

$$\varphi(u) = \frac{3}{4}(1 - u^2) \tag{8}$$

It should be noted that if the jth dimension of a candidate individual Y violates the lower or upper bound i.e., $y_j < \underline{b}_j$ or $y_j > \bar{a}_j$, $\underline{b}_j$ or $\bar{a}_j$ should be updated timely. It can be seen that: (1) we can directly calculate the parameters w and r_i in the density estimation modal Eq. (6) from the given data; (2) the better individuals make the more contribution to the density by using the rank order of each individual; (3) the computational cost needed by the Parzen window method is considerably low contrasted to that required by the fitness evaluations of multiple candidate individuals, especially expensive optimization problems.

The density estimation of each candidate individual can be computed by Eq. (6) which is used to evaluate the quality of the candidate individuals. Then, the greedy selection is applied to choose the one candidate individual with the largest density estimation as the offspring.

3.4. Neighborhood mechanism

The neighborhood mechanism has been widely exploited to improve the performance of the algorithm in EAs. Furthermore, it is well known that the performance of EAs depends on the neighborhood mechanism. Many methods are used to study the neighborhood mechanism of population in PSO and DE. However, few studies consider different neighborhood mechanisms during

Table 1
Benchmark functions used in experiments.

Name	Function	Search range	Accept
Sphere	$f_1(X) = \sum_{i=1}^{D} x_i^2$	$[-100,100]^D$	1×10^{-8}
Elliptic	$f_2(X) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} x_i^2$	$[-100,100]^D$	1×10^{-8}
SumSquare	$f_3(X) = \sum_{i=1}^{D} ix_i^2$	$[-10,10]^D$	1×10^{-8}
SumPower	$f_4(X) = \sum_{i=1}^{D} \lvert x_i \rvert^{(i+1)}$	$[-1,1]^D$	1×10^{-8}
Schwefel 2.22	$f_5(X) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	$[-10,10]^D$	1×10^{-8}
Schwefel 2.21	$f_6(X) = max_i\{\lvert x_i \rvert, 1 \leq i \leq D\}$	$[-100,100]^D$	4×10^{1}
Step	$f_7(X) = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	$[-100,100]^D$	1×10^{-8}
Exponential	$f_8(X) = exp(0.5 * \sum_{i=1}^{D} x_i) - 1$	$[-1.28,1.28]^D$	1×10^{-8}
Quartic	$f_9(X) = \sum_{i=1}^{D} ix_i^4 + random[0, 1)$	$[-1.28,1.28]^D$	1×10^{-1}
Rosebrock	$f_{10}(X) = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-5,10]^D$	5×10^{0}
Rastrigin	$f_{11}(X) = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$	$[-5.12,5.12]^D$	1×10^{-8}
NCRastrigin	$f_{12}(X) = \sum_{i=1}^{D} [y_i^2 - 10\cos(2\pi y_i) + 10]$ $y_i = \begin{cases} x_i & \lvert x_i \rvert < \frac{1}{2} \\ \frac{round(2x_i)}{2} & \lvert x_i \rvert \geq \frac{1}{2} \end{cases}$	$[-5.12,5.12]^D$	1×10^{-8}
Griewank	$f_{13}(X) = \frac{1}{4000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-600,600]^D$	1×10^{-8}
Schwefel 2.26	$f_{14}(X) = 418.98288727243369 * D - \sum_{i=1}^{D} x_i \sin(\sqrt{\lvert x_i \rvert})$	$[-500,500]^D$	1×10^{-8}
Ackley	$f_{15}(X) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i))$ $+20 + e$	$[-32,32]^D$	1×10^{-8}
Penalized1	$f_{16}(X) = \frac{\pi}{D}\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})]$ $+(y_D - 1)^2\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{1}{4}(x_i + 1)$ $u_{x_i,a,k,m} = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a \leq x_i \leq a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	$[-50,50]^D$	1×10^{-8}
Penalized2	$f_{17}(X) = \frac{1}{10}\{\sin^2(\pi x_1) + \sum_{i=1}^{D-1}(x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})] +$ $(x_D - 1)^2[1 + \sin^2(2\pi x_{i+1})]\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	$[-50,50]^D$	1×10^{-8}
Alpine	$f_{18}(X) = \sum_{i=1}^{D} \lvert x_i \cdot \sin(x_i) + 0.1 \cdot x_i \rvert$	$[-10,10]^D$	1×10^{-8}
Levy	$f_{19}(X) = \sum_{i=1}^{D-1}(x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})] + \sin^2(3\pi x_1)$ $+\lvert x_D - 1 \rvert[1 + \sin^2(3\pi x_D)]$	$[-10,10]^D$	1×10^{-8}
Weierstrass	$f_{20}(X) = \sum_{i=1}^{D}(\sum_{k=0}^{k_{max}}[a^k cos(2\pi b^k(x_i + 0.5))]) - D\sum_{k=0}^{k_{max}}[a^k$ $\cos(2\pi b^k 0.5)]$, $a = 0.5$, $b = 3$, $k_{max} = 20$	$[-0.5,0.5]^D$	1×10^{-8}
Himmelblau	$f_{21}(X) = \frac{1}{D}\sum_{i=1}^{D}(x_i^4 - 16x_i^2 + 5x_i)$	$[-5,5]^D$	-78
Michalewicz	$f_{22}(X) = -\sum_{i=1}^{D}\sin(x_i)\sin^{20}(\frac{i \times x_i^2}{\pi})$	$[0, \pi]^D$	$-49, -95, -190$

the evolution process in ABC. To fill this gap, two neighborhood mechanisms are employed to build the neighborhood candidate pool in this paper: (1) fitness-based neighborhood; (2) distance-based neighborhood. The one puts emphasis on the convergence, while the other one focuses on the population diversity. They can work together to improve the performance of the algorithm.

(1) *Fitness-based neighborhood*: The selection probability of the ith individual taking part in the search is computed by the following equation.

$$P_i = \frac{r_i}{SN}, i = 1, \ldots, SN \tag{9}$$

where r_i represents the rank of the ith individual which is sorted in descending order based on the fitness. It is obvious from Eq. (9) that the better individuals have the more selection probability. Based on this point, the individuals with better fitness have more opportunity to be chosen to participate in the search. Therefore, this neighborhood mechanism is beneficial to the convergence.

(1) *Distance-based neighborhood*: The selection probability of the ith individual participating in the search for the jth parent individual is assigned as

$$P_i = 1 - \frac{\|X_i - X_j\|}{\sum_{i=1}^{SN}\|X_i - X_j\|}, i = 1, \ldots, SN \tag{10}$$

It can be seen from Eq. (10) that the individuals which are nearer to the parent individual have more probability to be selected. It is obvious that the distance-based neighborhood contributes to keeping the population diversity.

4. Numerical experiments

4.1. Experimental setup

The performance of the proposed approach is tested on a set of benchmark functions. This test set contains twenty 15-dimensional, 30-dimensional, or 60-dimensional benchmark functions in [10,19,50,51] and two 50-dimensional, 100-dimensional, or 200-dimensional benchmark functions in [50,51], which are concluded in Table 1. These benchmark functions are classified into three groups according to the number of the dimension, i.e., the high-, middle-, and low-dimensional functions, so that it

Table 2
Comparison among ABCs on the low-dimensional cases.

Fun		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
ABCPW	Mean	1.18e−63	3.71e−57	4.97e−65	3.47e−109	7.84e−35	2.08e−03	0	0	3.26e−03	1.23e−02	0
	SD	1.21e−63	3.63e−57	8.21e−64	7.33e−109	6.51e−35	1.82e−03	0	0	2.30e−03	4.85e−02	0
CABC	Mean	3.89e−36^{+}	3.10e−32^{+}	3.20e−37^{+}	7.02e−50^{+}	3.42e−19^{+}	3.81e−01^{+}	0$^{=}$	3.52e−17^{+}	1.60e−02^{+}	1.59e−01^{+}	0$^{=}$
	SD	4.02e−36	2.96e−32	6.70e−37	3.10e−49	3.81e−19	2.00e−01	0	8.34e−17	6.31e−03	1.39e−01	0
ABCbest	Mean	2.37e−34^{+}	3.10e−31^{+}	3.01e−36^{+}	1.72e−56^{+}	4.10e−19^{+}	2.31e−01^{+}	0$^{=}$	3.52e−17^{+}	5.20e−03^{+}	2.49e−00^{+}	0$^{=}$
	SD	1.21e−34	3.01e−31	2.30e−36	4.40e−56	1.10e−19	5.61e−02	0	9.89e−17	1.78e−03	4.63e−00	0
GABC	Mean	4.58e−24^{+}	2.89e−20^{+}	4.30e−25^{+}	7.01e−34^{+}	5.21e−13^{+}	5.21e−01^{+}	0$^{=}$	1.89e−16^{+}	1.98e−02^{+}	8.78e−01^{+}	2.87e−17^{+}
	SD	3.96e−24	3.42e−20	2.89e−25	2.01e−33	2.16e−13	1.87e−01	0	4.78e−17	7.89e−03	2.12e−00	1.21e−16
ABC	Mean	7.10e−14^{+}	9.02e−10^{+}	2.98e−15^{+}	8.11e−22^{+}	4.86e−08^{+}	3.41e−00^{+}	0$^{=}$	3.55e−16^{+}	5.12e−02^{+}	2.31e−01^{+}	3.01e−07^{+}
	SD	9.20e−14	1.78e−09	2.11e−15	1.89e−21	2.01e−08	8.22e−01	0	1.20e−16	2.37e−02	1.88e−01	8.23e−07
Fun		f_{12}	f_{13}	f_{14}	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}	f_{21}	f_{22}
ABCPW	Mean	0	4.43e−16	0	1.14e−15	3.14e−32	1.34e−32	3.86e−35	1.34e−31	0	−78.3323	−49.2638
	SD	0	9.02e−16	0	1.28e−15	0	0	8.46e−35	0	0	1.58e−14	6.71e−02
CABC	Mean	0$^{=}$	1.10e−11^{+}	1.78e−13^{+}	3.02e−14^{+}	3.14e−32$^{=}$	1.34e−32$^{=}$	3.78e−20^{+}	1.34e−31$^{=}$	0$^{=}$	−78.3323$^{=}$	−48.7012^{+}
	SD	0	5.23e−09	4.12e−13	2.78e−15	1.01e−47	5.20e−48	4.12e−20	1.78e−47	0	1.25e−11	1.56e−01
ABCbest	Mean	0$^{=}$	8.12e−03^{+}	2.34e−12^{+}	2.69e−14^{+}	3.14e−32$^{=}$	1.34e−32$^{=}$	4.20e−19^{+}	1.34e−31$^{=}$	0$^{=}$	−78.3323$^{=}$	−48.1002^{+}
	SD	0	1.41e−02	4.06e−12	2.51e−15	0	0	4.34e−19	0	0	1.66e−08	3.59e−01
GABC	Mean	6.35e−17^{+}	6.01e−04^{+}	4.14e−13^{+}	5.27e−12^{+}	1.11e−25^{+}	3.28e−24^{+}	2.23e−06^{+}	3.78e−16^{+}	3.27e−13^{+}	−78.3322^{+}	−45.5665^{+}
	SD	3.55e−16	2.04e−03	4.45e−13	2.18e−12	2.23e−25	1.47e−24	5.41e−06	3.59e−16	2.12e−13	1.32e−05	5.21e−01
ABC	Mean	9.36e−06^{+}	2.01e−05^{+}	1.65e−00^{+}	1.24e−06^{+}	6.97e−15^{+}	4.23e−14^{+}	8.16e−06^{+}	5.05e−10^{+}	1.23e−04^{+}	−78.2752^{+}	−44.6741^{+}
	SD	1.78e−05	5.02e−05	4.23e−00	6.12e−07	8.45e−15	3.25e−14	4.36e−06	7.32e−10	1.75e−05	5.86e−02	3.58e−01

Table 3
Comparison among ABCs on the middle-dimensional cases.

Fun		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
ABCPW	Mean	6.79e−60	5.87e−55	1.65e−60	7.41e−111	6.90e−33	1.47e−00	0	1.42e−16	2.31e−02	8.21e−02	0
	SD	1.34e−59	7.97e−55	1.77e−60	1.61e−110	3.03e−33	5.21e−01	0	1.32e−17	4.36e−03	2.34e−02	0
CABC	Mean	4.39e−35^{+}	5.12e−31^{+}	6.89e−36^{+}	2.84e−40^{+}	2.30e−18^{+}	5.27e−00^{+}	0$^{=}$	2.22e−16$^{=}$	5.32e−02^{+}	2.23e−01^{+}	0$^{=}$
	SD	6.76e−35	7.21e−31	8.03e−36	8.45e−40	6.10e−19	9.02e−01	0	1.89e−16	2.07e−02	2.02e−01	0
ABCbest	Mean	4.23e−32^{+}	3.75e−28^{+}	6.03e−33^{+}	2.85e−57^{+}	3.20e−17^{+}	5.41e−00^{+}	0$^{=}$	2.22e−16$^{=}$	4.01e−02^{+}	2.60+01^{+}	0$^{=}$
	SD	4.30e−32	7.05e−28	2.84e−33	6.78e−57	8.69e−18	2.01e−00	0	6.39e−17	1.83e−03	3.30+01	0
GABC	Mean	2.01e−22^{+}	7.68e−19^{+}	1.85e−23^{+}	3.12e−34^{+}	2.86e−12^{+}	6.74e−00^{+}	0$^{=}$	6.56e−09^{+}	9.02e−02^{+}	1.09e−00^{+}	1.09e−15^{+}
	SD	8.89e−23	6.03e−19	2.20e−23	4.98e−34	8.45e−13	9.32e−01	0	2.89e−08	3.25e−02	3.02e−00	2.89e−15
ABC	Mean	1.89e−13^{+}	2.78e−09^{+}	3.87e−14^{+}	2.03e−20^{+}	2.36e−07^{+}	2.10e+01^{+}	0$^{=}$	7.58e−16^{+}	2.56e−01^{+}	2.40e−01^{+}	2.02e−06^{+}
	SD	1.89e−13	1.87e−09	2.65e−14	1.93e−20	3.78e−08	3.26e−00	0	6.78e−16	4.58e−02	2.46e−01	3.31e−06
Fun		f_{12}	f_{13}	f_{14}	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}	f_{21}	f_{22}
ABCPW	Mean	0	6.55e−15	0	3.68e−15	1.57e−32	1.34e−32	6.10e−34	1.34e−31	0	−78.3323	−98.6268
	SD	0	2.43e−15	0	2.70e−14	0	0	2.41e−34	0	0	8.20e−14	3.23e−01
CABC	Mean	0$^{=}$	5.32e−04^{+}	2.02e−12^{+}	3.69e−14^{+}	2.64e−32$^{=}$	2.13e−32$^{=}$	4.12e−19^{+}	1.34e−31$^{=}$	0$^{=}$	−78.3323$^{=}$	−95.5750^{+}
	SD	0	1.78e−03	4.26e−13	4.23e−15	3.12e−48	3.20e−48	3.21e−19	0	0	2.14e−09	3.87e−01
ABCbest	Mean	0$^{=}$	5.65e−11^{+}	1.57e−12^{+}	5.45e−14^{+}	2.01e−32$^{=}$	3.25e−32$^{=}$	6.87e−17^{+}	6.20e−31$^{=}$	1.38e−15^{+}	−78.3323$^{=}$	−93.4171^{+}
	SD	0	3.21e−11	7.58e−13	2.56e−15	6.23e−34	2.30e−32	3.86e−17	5.23e−31	2.85e−15	5.32e−08	7.20e−01
GABC	Mean	6.23e−14^{+}	1.89e−03^{+}	1.56e−11^{+}	1.89e−11^{+}	2.78e−24^{+}	5.23e−23^{+}	6.21e−06^{+}	1.32e−16^{+}	8.21e−12^{+}	−78.3322^{+}	−89.3012^{+}
	SD	1.02e−13	7.12e−03	7.41e−12	1.27e−11	3.56e−24	4.01e−23	1.20e−05	2.76e−16	6.43e−12	4.57e−05	6.32e−01
ABC	Mean	2.01e−01^{+}	4.25e−09^{+}	4.05e+01^{+}	1.78e−06^{+}	2.30e−14^{+}	1.86e−13^{+}	6.27e−05^{+}	2.45e−10^{+}	3.76e−04^{+}	−78.1651^{+}	−87.6101^{+}
	SD	4.25e−02	1.12e−08	2.32e−00	6.25e−07	2.30e−14	2.14e−13	7.23e−05	3.27e−10	7.53e−05	2.14e−01	5.32e−01

can be easily described in the later part. For example, the high-dimensional functions involve the 60-dimensional functions f_1−f_{20} and the 200-dimensional functions f_{21} and f_{22}.

Table 1 reports the test set of 22 benchmark functions. There are 9 unimodal functions f_1-f_9, a Rosenbrock function f_{10} which is unimodal in low dimensional situations while has multiple optima in high dimensional situations [52], 12 multimodal functions $f_{11}-f_{22}$. Furthermore, Table 1 also shows "Accept" (column 4) which is used to decide if the search is considered successful or not. If optimal value got by the algorithm is not more than "Accept", the run is considered successful.

In ABCPW, *limit* is set at 200 [20,24,53] and the population size *SN* is set at 50 [10,12,26,33]. When the functions belong to the low-, middle-, and high-dimensional benchmark functions, the stopping criteria is set as 50 000, 100 000, and 200 000 function evaluations (FES), respectively. All results are reported over 25 independent runs.

4.2. Comparison among ABC variants

The performance of ABCPW is compared to that of CABC [30], ABCbest [43], GABC [19], and ABC [6]. The parameter settings of these four competing algorithms remain the same as the original references. The results acquired by each method on the set of 22 benchmark functions are provided in Tables 2–4 in terms of the mean and the standard deviation. Furthermore, the results of the Wilcoxon statistical test at 5% significance difference are reported. The statistical result is indicated as "− / = / +", that denotes ABCPW performs worse than, equal to, or better than the compared algorithm, respectively. For convenience, the best results of all the methods are bolded.

From Tables 2–4, it can be observed that ABCPW beats CABC, ABCbest, GABC, and ABC on the most cases, respectively. Specially, From Table 2, ABCPW has better performance than CABC, ABCbest,

Table 4
Comparison among ABCs on the high-dimensional cases.

Fun		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
ABCPW	Mean	5.00e−58	2.86e−53	3.29e−58	4.38e−98	1.63e−31	3.23e+01	0	1.34e−17	1.03e−01	1.32e−02	0
	SD	5.79e−58	3.49e−53	2.36e−58	9.77e−98	6.66e−32	4.21e−00	0	1.01e−17	4.22e−02	1.35e−02	0
CABC	Mean	3.02e−34^{+}	2.89e−30^{+}	2.14e−34^{+}	3.23e−47^{+}	4.25e−18^{+}	4.12e+01^{+}	0$^{=}$	8.23e−16^{+}	2.01e−01^{+}	3.31e−01^{+}	0$^{=}$
	SD	3.02e−34	4.20e−30	2.32e−34	6.20e−47	1.98e−18	8.03e−00	0	1.23e−16	5.69e−02	7.20e−01	0
ABCbest	Mean	2.32e−29^{+}	6.85e−26^{+}	3.89e−30^{+}	2.53e−58^{+}	4.51e−16^{+}	2.85e+01^{+}	0$^{=}$	5.21e−16^{+}	1.68e−01^{+}	2.40+01^{+}	0$^{=}$
	SD	8.12e−30	4.36e−26	3.20e−30	2.87e−58	2.20e−16	4.23e−00	0	2.89e−16	4.36e−02	3.16+01	0
GABC	Mean	3.68e−21^{+}	2.56e−17^{+}	6.32e−22^{+}	6.25e−29^{+}	2.64e−11^{+}	5.23e+01^{+}	0$^{=}$	2.56e−08^{+}	4.24e−01^{+}	1.11e+01^{+}	2.12e−13^{+}
	SD	1.17e−21	8.21e−18	3.15e−22	1.52e−28	1.79e−12	5.32e−00	0	7.55e−08	4.33e−02	1.93e+01	1.73e−13
ABC	Mean	3.23e−12^{+}	2.12e−08^{+}	4.02e−13^{+}	2.38e−17^{+}	4.02e−07^{+}	3.25e+01^{+}	0$^{=}$	2.01e−15^{+}	6.91e−01^{+}	4.39e−01^{+}	7.92e−01^{+}
	SD	5.20e−13	9.91e−09	1.84e−13	3.55e−17	5.59e−08	6.48e−00	0	2.71e−16	1.68e−01	2.70e−01	6.19e−01
Fun		f_{12}	f_{13}	f_{14}	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}	f_{21}	f_{22}
ABCPW	Mean	0	0	3.63e−11	2.31e−14	7.85e−33	1.34e−32	3.62e−31	1.34e−31	0	−78.3323	−196.6476
	SD	0	0	0	1.78e−15	0	0	9.26e−32	0	0	1.12e−14	3.30e−01
CABC	Mean	0$^{=}$	2.12e−08^{+}	2.31e−10^{+}	9.23e−14^{+}	8.23e−33$^{=}$	1.34e−32$^{=}$	5.36e−18^{+}	1.34e−31$^{=}$	9.32e−15^{+}	−78.3323$^{=}$	−187.3762^{+}
	SD	0	2.58e−08	2.56e−11	7.25e−15	3.25e−48	6.02e−48	9.01e−18	1.58e−47	8.23e−15	2.03e−09	7.98e−01
ABCbest	Mean	0$^{=}$	1.58e−07^{+}	1.89e−10^{+}	8.69e−14^{+}	7.36e−32^{+}	2.01e−30^{+}	4.10e−14^{+}	8.36e−30^{+}	3.25e−14^{+}	−78.3323$^{=}$	−181.1510^{+}
	SD	0	2.25e−07	4.25e−11	6.10e−15	3.89e−32	5.78e−31	4.39e−15	2.02e−29	5.69e−15	7.25e−08	8.23e−01
GABC	Mean	1.98e−11^{+}	1.86e−02^{+}	5.35e−10^{+}	5.86e−11^{+}	2.03e−23^{+}	4.36e−22^{+}	1.58e−05^{+}	7.03e−16^{+}	2.03e−10^{+}	−78.3322^{+}	176.5461^{+}
	SD	3.26e−11	2.69e−02	2.87e−10	1.98e−11	5.98e−24	2.03e−22	2.00e−05	5.36e−16	8.02e−11	7.36e−05	2.20e−01
ABC	Mean	3.26e−00^{+}	7.36e−11^{+}	5.30e−00^{+}	2.98e−06^{+}	2.37e−14^{+}	6.31e−13^{+}	5.75e−04^{+}	8.36e−10^{+}	6.12e−04^{+}	−77.9601^{+}	−174.3301^{+}
	SD	7.65e−01	7.26e−11	3.32e+01	2.14e−06	7.52e−15	3.16e−13	8.13e−04	5.85e−10	6.13e−05	2.01e−01	2.03e−00

Table 5
Comparison in successful rate and convergence speed among ABCs on the low-dimensional cases.

Fun		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
ABCPW	SR	100	100	100	100	100	100	100	100	100	100	100
	AVEN	9.46e+03	1.18e+04	8.35e+03	2.96e+03	1.34e+04	1.10e+03	3.46e+03	6.25e+03	2.24e+03	1.01e+04	1.30e+04
CABC	SR	100	100	100	100	100	100	100	100	100	100	100
	AVEN	1.57e+04	2.10e+04	1.39e+04	5.65e+03	2.39e+04	1.17e+04	6.01e+03	1.06e+04	3.55e+04	1.41e+04	1.79e+04
ABCbest	SR	100	100	100	100	100	100	100	100	100	70	100
	AVEN	1.59e+04	2.06e+04	1.43e+04	4.90e+03	2.43e+04	3.10e+03	6.20e+03	1.10e+04	4.89e+03	1.88e+04	1.88e+04
GABC	SR	100	92	100	100	100	100	100	100	4	96	100
	AVEN	1.93e+04	2.91e+04	1.89e+04	6.69e+03	3.37e+04	1.20e+04	7.89e+03	1.40e+04	3.90e+04	1.98e+04	3.11e+04
ABC	SR	100	100	100	100	0	100	100	100	0	100	50
	AVEN	3.29e+04	4.60e+04	2.98e+04	9.99e+03	–	1.99e+04	1.21e+04	2.13e+04	–	2.17e+04	4.80e+04
Fun		f_{12}	f_{13}	f_{14}	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}	f_{21}	f_{22}
ABCPW	SR	100	100	100	100	100	100	100	100	100	100	100
	AVEN	1.22e+04	1.54e+04	1.32e+04	1.47e+04	7.85e+03	8.76e+03	1.43e+04	9.06e+03	1.53e+04	1.06e+04	3.09e+04
CABC	SR	100	96	100	100	100	100	100	100	100	100	82
	AVEN	1.89e+04	2.66e+04	1.79e+04	2.57e+04	1.34e+04	1.48e+04	2.30e+04	1.69e+04	2.79e+04	1.42e+04	3.60e+04
ABCbest	SR	100	68	100	100	100	100	100	100	100	100	0
	AVEN	1.92e+04	3.13e+04	2.19e+04	2.59e+04	1.39e+04	1.49e+04	2.52e+04	1.66e+04	2.86e+04	1.60e+04	–
GABC	SR	100	92	100	100	100	100	56	100	100	100	0
	AVEN	3.39e+04	3.50e+04	2.99e+04	3.69e+04	1.85e+04	2.02e+04	4.69e+04	2.39e+04	4.11e+04	2.41e+04	–
ABC	SR	0	52	36	0	100	100	0	100	0	100	0
	AVEN	–	4.60e+04	4.82e+04	–	2.99e+04	3.28e+04	–	3.97e+04	–	4.19e+04	–

GABC, and ABC on 14, 14, 21, and 21 out of 22 low-dimensional benchmark functions, respectively, while CABC, ABCbest, GABC, and ABC cannot outperform ABCPW on any test case. They have the similar performance on the remaining cases, while ABCPW has more stability than CABC, ABCbest, GABC, and ABC on the most cases, such as f_{21}.

From Table 3 on the results of the middle-dimensional benchmark functions, it can be seen that ABCPW is superior to CABC, ABCbest, GABC, and ABC on the most of 22 test functions. Specially, ABCPW is always better than CABC, ABCbest, GABC, and ABC on 13, 14, 21, and 21 cases, respectively. ABCPW is equal to CABC, ABCbest, GABC, and ABC on the remaining cases, while ABCPW has better robustness on the most cases.

From Table 4, it can be see that ABCPW also overcomes CABC, ABCbest, GABC, and ABC on the most cases. Specially, ABCPW is better than CABC, ABCbest, GABC, and ABC on 15, 18, 21, and 21 cases, respectively, while CABC, ABCbest, GABC, and ABC cannot exceed ABCPW on any case.

Further, the stability and the convergence rate of each algorithm is measured by the successful rate and the average evaluation number, respectively. The successful rate (SR%) denotes the number of success runs over 25 independent runs. The average evaluation number (AVEN) denotes the average FES required to reach "Accept", which has been specified in Table 1. SR% and AVEN are shown in Tables 5–8. What is more, the convergence curves of several benchmark functions are shown in Figs. 3–4.

From Tables 5–7 and Figs. 3–4, it can be observed that ABCPW has faster convergence and more stability than CABC, ABCbest, GABC, and ABC on all the test cases. To be specific, compared with ABC, AVEN of ABCPW can be decreased no less than one order of magnitude on several benchmark functions. Moreover, from Tables 5–7 it can be seen that the successful rate of ABCPW is 100%

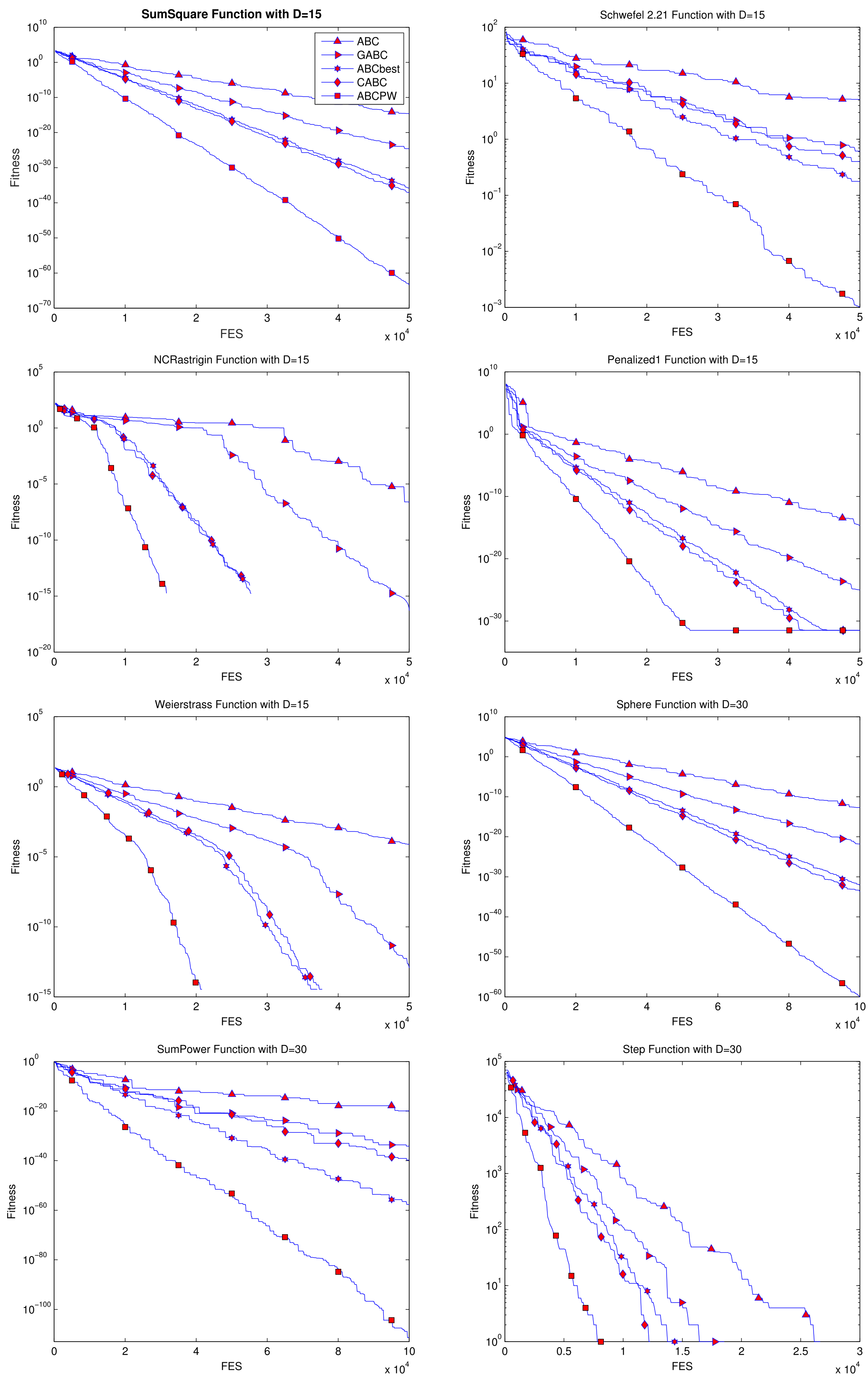

Fig. 3. Convergence curves of different ABCs on eight 15-dimensional or 30-dimensional functions.

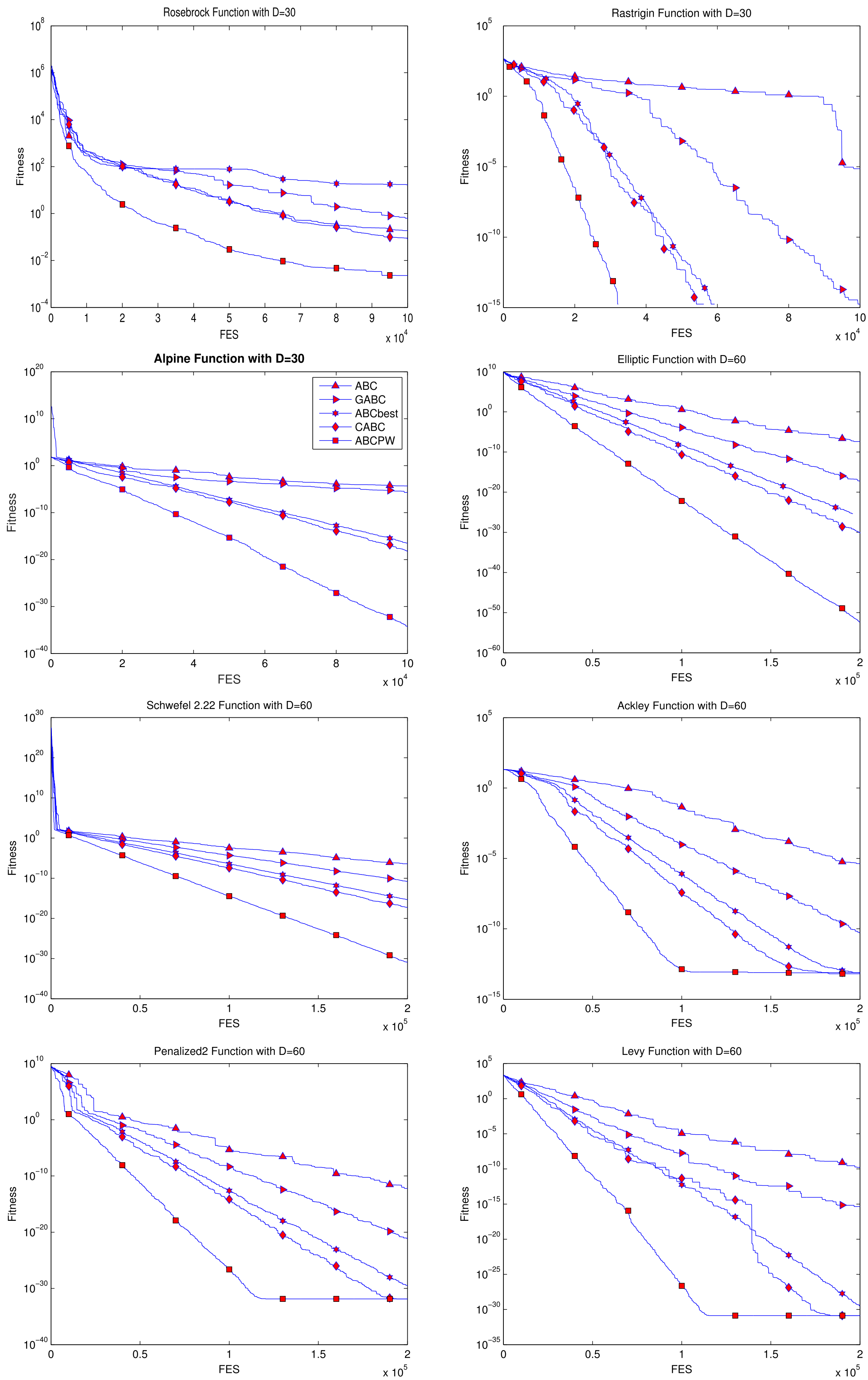

Fig. 4. Convergence curves of different ABCs on eight 30-dimensional or 60-dimensional functions.

Table 6
Comparison in successful rate and convergence speed among ABCs on the middle-dimensional cases.

Fun		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
ABCPW	SR	100	100	100	100	100	100	100	100	100	100	100
	AVEN	1.99e+04	2.57e+04	1.90e+04	5.21e+03	2.95e+04	8.80e+03	7.74e+03	1.38e+04	1.66e+04	2.12e+04	3.08e+04
CABC	SR	100	100	100	100	100	100	100	100	100	100	100
	AVEN	3.29e+04	4.40e+04	3.10e+04	1.21e+04	5.10e+04	5.29e+04	1.35e+04	2.26e+04	3.64e+04	3.49e+04	4.20e+04
ABCbest	SR	100	100	100	100	100	100	100	100	100	60	100
	AVEN	3.49e+04	4.59e+04	3.38e+04	9.79e+03	5.45e+04	1.66e+04	1.44e+04	2.50e+04	3.29e+04	5.49e+04	4.35e+04
GABC	SR	100	100	100	100	100	100	100	94	0	94	100
	AVEN	4.69e+04	6.10e+04	4.30e+04	1.26e+04	7.29e+04	6.16e+04	1.79e+04	3.20e+04	–	5.40e+04	7.10e+04
ABC	SR	100	100	100	100	0	32	100	100	0	100	0
	AVEN	8.53e+04	9.53e+04	6.62e+04	2.10e+04	–	9.30e+04	2.84e+04	4.70e+04	–	5.41e+04	–
Fun		f_{12}	f_{13}	f_{14}	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}	f_{21}	f_{22}
ABCPW	SR	100	100	100	100	100	100	100	100	100	100	100
	AVEN	2.69e+04	2.89e+04	2.79e+04	3.12e+04	1.65e+04	2.13e+04	2.79e+04	1.88e+04	3.34e+04	2.31e+04	4.66+04
CABC	SR	100	96	100	100	100	100	100	100	100	100	82
	AVEN	4.25e+04	4.46e+04	3.90e+04	5.29e+04	2.80e+04	3.08e+04	5.02e+04	3.49e+04	5.85e+04	3.11e+04	9.46e+04
ABCbest	SR	100	100	100	100	100	100	100	100	100	100	0
	AVEN	4.30e+04	4.25e+04	4.10e+04	5.54e+04	3.11e+04	3.38e+04	5.59e+04	2.45e+04	6.18e+04	4.48e+04	–
GABC	SR	100	94	100	100	100	100	6	100	100	100	0
	AVEN	7.74e+04	5.69e+04	7.20e+04	7.90e+04	3.40e+04	4.39e+04	9.91e+04	4.69e+04	8.81e+04	6.00e+04	–
ABC	SR	0	92	6	0	100	100	0	100	0	100	0
	AVEN	–	8.42e+04	9.76e+04	–	6.15e+04	7.08e+04	–	7.91e+04	–	8.62e+04	–

Table 7
Comparison in successful rate and convergence speed among ABCs on the high-dimensional cases.

Fun		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}
ABCPW	SR	100	100	100	100	100	100	100	100	0	100	100
	AVEN	4.21e+04	5.46e+04	4.11e+04	1.13e+04	6.20e+05	1.11e+05	1.71e+04	2.92e+04	–	5.45e+04	7.52e+04
CABC	SR	100	100	100	100	100	62	100	100	0	100	100
	AVEN	6.95e+04	9.08e+04	6.65e+04	2.45e+04	1.05e+05	1.53e+05	2.85e+04	4.82e+04	–	8.68e+04	8.19e+04
ABCbest	SR	100	100	100	100	100	100	100	100	0	22	100
	AVEN	7.90e+04	9.99e+04	7.59e+04	2.09e+04	1.20e+05	1.25e+05	3.23e+04	5.50e+04	–	1.69e+05	9.12e+05
GABC	SR	100	100	100	100	100	0	100	86	0	62	100
	AVEN	9.98e+04	1.29e+05	9.68e+04	2.63e+05	1.58e+05	–	4.19e+04	6.96e+04	–	1.30e+05	1.60e+05
ABC	SR	100	48	100	100	0	0	100	100	0	100	0
	AVEN	1.49e+05	1.96e+05	1.45e+05	4.30e+04	–	–	8.24e+04	1.01e+05	–	1.31e+05	–
Fun		f_{12}	f_{13}	f_{14}	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}	f_{21}	f_{22}
ABCPW	SR	100	100	100	100	100	100	100	100	100	100	100
	AVEN	5.82e+04	4.51e+04	6.12e+04	6.62e+04	3.38e+04	3.98e+04	6.18e+04	3.89e+04	7.01e+04	4.35e+04	1.08e+05
CABC	SR	100	86	100	100	100	100	100	100	100	100	0
	AVEN	8.55e+04	8.62e+04	8.23e+04	1.09e+05	5.59e+04	6.48e+04	1.02e+05	7.39e+04	1.19e+05	6.40e+04	–
ABCbest	SR	100	100	100	100	100	100	100	100	100	100	0
	AVEN	9.50e+04	9.01e+04	1.22e+05	1.19e+05	6.45e+04	7.45e+04	1.25e+05	7.50e+04	1.10e+05	7.25e+04	–
GABC	SR	100	84	100	100	100	100	0	100	100	100	0
	AVEN	1.72e+05	1.11e+05	1.58e+05	1.63e+05	8.30e+04	9.51e+04	–	9.78e+04	1.83e+05	1.19e+05	–
ABC	SR	0	100	0	0	100	100	0	100	0	48	0
	AVEN	–	1.65e+05	–	–	1.25e+05	1.48e+05	–	1.59e+05	–	1.96e+05	–

on all the benchmark functions except f_9. This may benefit from the fact that ABCPW can make use of multiple search strategies, density estimation method and neighborhood mechanism to achieve a right tradeoff between the exploitation and the exploration.

4.3. Comparison ABCPW with other state-of-the-art algorithms

4.3.1. Comparison several variant EAs with ABCPW

Table 8 shows the comparison results between ABCPW and several state-of-the-art EAs, which are reported as follows:

- LEA [51].
- OGA/Q [50].
- CEP [54].
- FEP [55].
- ALEP [56].

The experimental results of LEA, OGA/Q, CEP, FEP, and ALEP are based on their original references. Particularly, NA donates that the relative results cannot be provided in the original reference. It can be shown from Table 8 that ABCPW overcomes LEA, OGA/Q, CEP, FEP, and ALEP on nearly all the cases, except that ABCPW is worse than OGA/Q on Sphere and Schwefel 2.22 .

4.3.2. Comparison several variant DEs with ABCPW

Further, ABCPW is compared to several variant DEs in Table 9. These variant DEs are shown as follows:

- JADE [45].
- Classic DE [3].
- SaDE [16].
- jDE [44].

Table 8
Comparison between several variant evolutionary algorithms and ABCPW.

Method	Sphere			Schwefel 2.22			Schwefel 2.26		
	Mean.FE	Mean	SD	Mean.FE	Mean	SD	Mean.FE	Mean	SD
ABCPW	50,000	1.32e−27	1.36e−27	100,000	6.90e−33	3.03e−33	100,000	**0**	**0**
LEA	110,654	4.7e−16	6.2e−17	110,031	4.2e−19	4.2e−19	302,116	3.0e−02	6.4e−04
OGA/Q	112,559	**0**	**0**	112,612	**0**	**0**	302,116	3.0e−02	6.4e−04
CEP/best	250,000	3.9e−07	NA	250,000	1.9e−03	NA	NA	NA	NA
FEP	150,000	5.7e−04	1.3e−04	200,000	8.1e−03	7.7e−04	900,000	1.4e+01	5.2e+01
ALEP	150,000	6.3e−04	7.6e−05	NA	NA	NA	150,000	1.1e+03	5.8e+01
	Rastrigin			Griewank			Penalized 1		
	Mean.FE	Mean	SD	Mean.FE	Mean	SD	Mean.FE	Mean	SD
ABCPW	100,000	**0**	**0**	100,000	**0**	**0**	50,000	**7.19e−30**	**5.75e−30**
LEA	223,803	2.1e−18	3.3e−18	140,498	6.1e−16	2.5e−17	132,642	2.4e−06	2.2e−06
OGA/Q	224,710	**0**	**0**	134,000	**0**	**0**	134,556	6.0e−06	1.1e−06
CEP/best	250,000	4.7e−00	NA	250,000	2.7e−07	NA	NA	NA	NA
FEP	500,000	4.6e−02	1.2e−02	200,000	1.6e−02	2.2e−02	150,000	9.2e−06	3.6e−06
ALEP	150,000	5.8e−00	2.1e−00	150,000	2.4e−02	2.8e−02	150,000	6.0e−06	1.0e−06
	Penalized 2			Himmelblau			Michalewicz		
	Mean.FE	Mean	SD	Mean.FE	Mean	SD	Mean.FE	Mean	SD
ABCPW	50,000	**1.29e−28**	**2.40e−28**	150,000	**−78.3323**	**1.85e−14**	150,000	**−99.2086**	**3.21e−02**
LEA	130,213	1.7e−04	1.2e−04	243,895	−78.3100	6.1e−03	289,863	−93.01	2.3e−02
OGA/Q	134,143	1.8e−04	2.6e−05	245,930	−78.3000	6.2e−03	302,773	−92.83	2.6e−02
EDA/L	114,570	3.4e−21	NA	153,116	−78.3107	NA	168,885	−94.3757	NA
FEP	150,000	1.6e−04	7.3e−05	NA	NA	NA	NA	NA	NA
ALEP	150,000	9.8e−05	1.2e−05	NA	NA	NA	NA	NA	NA

Table 9
Comparison between variant DEs and ABCPW.

Fun	Max.FEs	DE [3]	jDE [44]	SaDE [16]	JADE [45]	ABCPW
Sphere	150,000	9.8e−14 (8.4e−14)	1.46e−28 (1.78e−28)	3.28e−20 (3.63e−20)	2.69e−56 (1.41e−55)	**1.08e−92 (2.21e−93)**
Schwefel 2.22	200,000	1.6e−09 (1.1e−09)	9.02e−24 (6.01e−24)	3.51e−25 (2.74e−25)	3.18e−25 (2.05e−24)	**5.59e−67 (3.86e−67)**
Step	10,000	4.7e+03 (1.1e+03)	6.13e+02 (1.72e+02)	5.07e+01 (1.34e+01)	5.62e+00 (1.87e+00)	**0 (0)**
Rastrigin	100,000	1.8e+02 (1.3e+01)	3.32e−04 (6.39e−04)	2.43e+00 (1.60e+00)	1.33e−01 (9.74e−02)	**0 (0)**
Griewank	50,000	2.0e−01 (1.1e−01)	7.29e−06 (1.05e−05)	2.52e−09 (1.24e−08)	1.57e−08 (1.09e−07)	**6.02e−12 (2.39e−12)**
Schwefel 2.26	100,000	5.9e+03 (1.1e+03)	1.70e−10 (1.71e−10)	1.13e−08 (1.08e−08)	2.62e−04 (3.59e−04)	**0 (0)**
Ackley	50,000	1.1e−01 (3.9e−02)	2.37e−04 (7.10e−05)	3.81e−06 8.26e−07	3.35e−09 (2.84e−09)	**5.82e−14 (7.12e−14)**
Penalized 2	50,000	7.5e−02 (3.8e−02)	1.80e−05 (1.42e−05)	1.93e−09 (1.53e−09)	1.87e−10 (1.09e−09)	**1.29e−28 (2.40e−28)**
Penalized 1	50,000	1.2e−02 (1.0e−02)	7.03e−08 (5.74e−08)	8.25e−12 (5.12e−12)	1.67e−15 (1.02e−14)	**7.19e−30 (5.75e−30)**
Alpine	300,000	2.3e−04 (1.7e−04)	6.08e−10 (8.36e−10)	2.94e−06 (3.47e−06)	2.78e−05 (8.43e−06)	**1.36e−98 (6.03e−98)**

Table 10
Comparison between variant PSOs and ABCPW.

Fun	PSO [5]	FIPS [46]	CLPSO [47]	HPSO-TVAC [48]	OLPSO-G [49]	ABCPW
Sphere	3.34e−14 (5.39e−14)	2.42e−13 (1.73e−13)	1.58e−12 (7.70e−13)	2.83e−33 (3.19e−33)	4.12e−54 (6.34e−54)	**6.2e−120 (1.5e−120)**
Schwefel 2.22	1.70e−10 (1.39e−10)	2.76e−08 (9.04e−09)	2.51e−08 (5.84e−09)	9.03e−20 (9.58e−20)	9.85e−30 (1.01e−29)	**5.59e−67 (3.86e−67)**
Step	**0 (0)**	**0 (0)**	**0 (0)**	**0 (0)**	**0 (0)**	**0 (0)**
Rosenbrock	2.80e+01 (2.17e+01)	2.51e+01 (5.10e−01)	1.13e+01 (9.85e−00)	2.39e+01 (2.65e+01)	2.15e+01 (2.99e+01)	**2.30e−03 (3.71e−03)**
NCRastrigin	4.36e+01 (1.12e+01)	7.01e+01 (1.47e+01)	1.54e−00 (2.75e−00)	1.03e+01 (8.24e−00)	2.18e−00 (6.31e−01)	**0 (0)**
Rastrigin	3.57e+01 (6.89e−00)	6.51e+01 (1.33e+01)	9.09e−05 (1.25e−04)	9.43e−00 (3.48e−00)	1.07e−00 (9.92e−01)	**0 (0)**
Griewank	1.53e−03 (4.32e−03)	9.01e−12 (1.84e−11)	9.02e−09 (8.57e−09)	9.75e−03 (8.33e−03)	4.83e−03 (8.63e−03)	**0 (0)**
Schwefel 2.26	3.16e+03 (4.06e+02)	9.93e+02 (5.09e+02)	3.82e−04 (1.28e−05)	1.59e+03 (3.26e+02)	3.84e+02 (2.17e+02)	**0 (0)**
Ackley	8.20e−08 (6.73e−08)	2.33e−07 (7.19e−08)	3.66e−07 (7.57e−08)	7.29e−14 (3.00e−14)	7.98e−15 (2.03e−15)	**1.01e−15 (2.20e−15)**
Penalized 2	3.26e−13 (3.70e−13)	2.70e−14 (1.57e−14)	1.25e−12 (9.45e−12)	2.79e−28 (2.18e−28)	4.39e−04 (2.20e−03)	**1.35e−32 (0)**
Penalized 1	8.10e−16 (1.07e−15)	1.96e−15 (1.11e−15)	6.45e−14 (3.70e−14)	2.71e−29 (1.88e−28)	1.59e−32 (1.03e−33)	**1.57e−32 (0)**

The results of DE, jDE, SaDE, and JADE are based on the references [45,57]. From Table 9, it is clear that ABCPW overcomes DE, jDE, SaDE, and JADE on all the 11 cases.

4.3.3. Comparison several variant PSOs with ABCPW

Additionally, Table 10 shows ABCPW is compared to several variant PSOs. These PSOs are summarized as follows:

- Classic PSO [5].
- HPSO-TVAC [48].
- FIPS [46].
- OLPSO-G [49].
- CLPSO [47].

The results of these variant PSOs are directly from the reference [49]. It is obvious from Table 10 that ABCPW performs best on all the cases.

4.3.4. Comparison CMA-ES with ABCPW

Next, the comparison of ABCPW and CMA-ES [7] is discussed. The parameter settings of ABCPW and CMA-ES are the same as [10]. The benchmark functions can also be found in [10]. The comparison results of ABCPW and CMA-ES are reported in Table 11. The experimental results of CMA-ES are from [10]. It can be seen from

Table 11
Comparison between CMA-ES and ABCPW.

Method	Sphere		Rosenbrock		Step		Quartic	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
ABCPW	**2.0e−23**	**8.3e−24**	**3.4e−02**	**4.1e−02**	**0**	**0**	**1.3e−02**	**6.5e−03**
CMA-ES	9.7e−23	3.8e−23	4.0e−01	1.2e−00	1.4e−00	1.7e−00	2.3e−01	8.7e−02
Method	Schwefel		Rastrigin		Griewank		Penalized	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
ABCPW	**2.3e−04**	**5.3e−04**	**1.6e−04**	**3.8e−04**	8.5e−04	2.9e−04	**1.4e−04**	**2.7e−03**
CMA-ES	4.9e+03	8.9e+02	5.1e+01	1.3e+01	**7.4e−04**	**2.7e−03**	1.2e−04	3.2e−02
Method	Penalized 2		Foxholes		Goldstein-Price		Shekel 10	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
ABCPW	**3.8e−04**	**5.1e−04**	**0.998**	**2.1e−04**	**3.0e+00**	**2.5e−04**	**−10.536**	**2.2e−04**
CMA-ES	1.7e−03	4.5e−03	10.44	6.87	14.34	25.05	−7.03	3.74

Table 12
Comparison between WWO, LOA and ABCPW.

Method	Hybrid function 6				Composition function 1				Composition function 2			
	Maximum	Minimum	Median	SD	Maximum	Minimum	Median	SD	Maximum	Minimum	Median	SD
ABCPW	2.21e+03	2.21e+03	2.21e+03	1.56e+00	2.72e−33	2.45e+03	2.55e+03	3.87e+01	2.66e+03	2.61e+03	2.61e+03	2.73e+02
WWO	2.85e+03	2.22e+03	2.48e+03	1.43e+02	2.62e+03	2.62e+03	2.62e+03	1.45e−01	2.63e+03	2.62e+03	2.63e+03	6.89e+00
LOA	2.21e+03	2.21e+03	2.21e+03	4.86e−01	2.74e+03	2.47e+03	2.55e+03	8.93e+01	2.67e+03	2.60e+03	2.62e+03	2.33e+01
	Composition function 3				Composition function 4				Composition function 5			
	Maximum	Minimum	Median	SD	Maximum	Minimum	Median	SD	Maximum	Minimum	Median	SD
ABCPW	2.71e+03	2.52e+03	2.54e+03	7.56e+01	2.61e+03	2.60e+03	2.60e+03	2.54e+00	2.72e+03	2.70e+03	2.70e+03	6.23e+00
WWO	2.72e+03	2.70e+03	2.71e+03	2.00e+00	2.70e+03	2.70e+03	2.70e+03	6.50e−02	3.50e+03	3.10e+03	3.10e+03	5.90e+01
LOA	2.71e+03	2.52e+03	2.56e+03	6.93e+01	2.61e+03	2.60e+03	2.61e+03	3.06e+00	2.72e+03	2.70e+03	2.71e+03	5.79e+00
	Composition function 6				Composition function 7				Composition function 8			
	Maximum	Minimum	Median	SD	Maximum	Minimum	Median	SD	Maximum	Minimum	Median	SD
ABCPW	5.21e+03	3.09e+03	3.65e+03	4.20e+02	4.95e+03	3.42e+03	3.63e+03	4.72e+02	3.15e+03	3.02e+03	3.05e+03	6.23e+01
WWO	5.39e+03	3.10e+03	3.78e+03	3.61e+02	5.06e+03	3.56e+03	4.02e+03	3.60e+02	7.66e+03	4.25e+03	5.63e+03	7.38e+02
LOA	7.09e+03	3.15e+03	4.25e+03	1.27e+03	6.50e+04	3.31e+03	1.80e+04	2.14e+04	3.18e+03	3.02e+03	3.06e+03	5.31e+01

Table 13
The CEC 2005 benchmark functions.

Pro.	Name	D
F1	Shifted Sphere function	30
F2	Shifted Schwefel problem 1.2	30
F3	Shifted Rosenbrock function	30
F4	Shifted Rastrign function	30
F5	Shifted Griewank function	30
F6	Shifted Ackley function	30
F7	Shifted Alpine function	30
F8	Shifted Rotated Ackley function	30
F9	Shifted Rotated Rastrign function	30
F10	Shifted Rotated Weierstrass function	30
F11	Expanded Extended Griewank plus Rosenbrock function	30

Table 11 that ABCPW is better than CMA-ES on 11 out of 12 test cases except Griewank function. In particular, CMA-ES easily falls into a local optimum on the complex multimodal functions such as Schwefel, Rastrigin, Foxholes, Goldstein-Price, and Shekel 10 functions. This is because CMA-ES is a local method which prefers the exploitation to the exploration.

4.3.5. Comparison WWO and LOA with ABCPW

Further, ABCPW is compared with two popular methods, i.e., water wave optimization algorithm (WWO) [58] and lion optimization algorithm (LOA) [59] on a comprehensive set of CEC 2014 benchmark test functions which can be seen in [58,59]. The experimental results of WWO and LOA are from original literatures [58] and [59], respectively. The comparison results of ABCPW, WWO and LOA are shown in Table 12. It can be seen that ABCPW is better than WWO and LOA on the most test cases. However, it should be indicated that the advantage of ABCPW is not obvious on Hybrid function 6, Composition function 4, and Composition function 5 when compared to LOA.

4.4. Test on CEC 2005 benchmark functions

We further test the performance of the proposed approach on a set of 11 CEC 2005 optimization problems. Table 13 shows these test problems, the detail of which can be seen in [60]. Table 14 reports the comparison results of ABCPW, CABC, ABCbest, GABC, and ABC under the maximum number of 100000 FES. It can be seen form Table 14 that ABCPW surpasses CABC, ABCbest, GABC, and ABC on 7, 11, 11, and 11 out of 11 cases, respectively. They perform the same on the remaining cases.

4.5. Test on real-world problem

In this subsection, one real-world problem, i.e., Frequency-modulated (FM) sound synthesis [61] is used to test the performance of the proposed approach. In FM, The equation of the target sound wave and the estimated sound wave are shown as follows, respectively.

$$y_0(t) = 1.0 sin(5.0 \cdot t \cdot \theta - 1.5 \cdot sin(4.8 \cdot t \cdot \theta + 2.0 \cdot sin(4.9 \cdot t \cdot \theta))), \quad (11)$$

Table 14
Comparison among ABCs on CEC 2005 30-dimensional functions.

Fun		F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11
ABC	Mean	$7.25e{-13}^{+}$	$6.30e{+04}^{+}$	$1.25e{+01}^{+}$	$7.25e{-02}^{+}$	$4.02e{-10}^{+}$	$1.67e{-05}^{+}$	$8.66e{-04}^{+}$	$2.11e{+01}^{+}$	$3.20e{+02}^{+}$	$3.10e{+01}^{+}$	$1.59e{+00}^{+}$
	SD	8.25e−13	2.45e+04	2.56e+01	3.47e−01	5.27e−10	6.67e−06	1.39e−03	8.20e−02	6.11e+01	1.65e+00	2.03e−01
GABC	Mean	$2.35e{-22}^{+}$	$4.23e{+04}^{+}$	$1.32e{+01}^{+}$	$1.03e{-14}^{+}$	$2.65e{-11}^{+}$	$4.12e{-11}^{+}$	$6.16e{-06}^{+}$	$2.11e{+01}^{+}$	$1.70e{+02}^{+}$	$3.08e{+01}^{+}$	$1.51e{+00}^{+}$
	SD	2.45e−22	6.24e+03	1.55e+01	6.58e−14	2.37e−10	1.65e−11	8.20e−06	5.41e−02	2.52e+01	2.29e+00	3.02e−01
ABCbest	Mean	$7.36e{-26}^{+}$	$5.25e{+04}^{+}$	$3.20e{+01}^{+}$	$4.26e{-15}^{+}$	$2.57e{-11}^{+}$	$7.28e{-13}^{+}$	$4.58e{-05}^{+}$	$2.11e{+01}^{+}$	$1.64e{+02}^{+}$	$3.02e{+01}^{+}$	$1.45e{+00}^{+}$
	SD	1.58e−26	2.34e+03	2.01e+01	2.46e−15	5.39e−11	5.43e−13	7.29e−05	3.60e−02	2.40e+01	3.09e+00	2.98e−01
CABC	Mean	$\mathbf{0}^{=}$	$2.34e{+04}^{+}$	$3.07e{+01}^{+}$	$\mathbf{0}^{=}$	$2.31e{-12}^{+}$	$8.81e{-15}^{+}$	$1.23e{-16}^{=}$	$2.09e{+01}^{=}$	$1.62e{+02}^{+}$	$2.86e{+01}^{+}$	$1.32e{+00}^{+}$
	SD	**0**	7.29e+03	5.23e+01	**0**	1.13e−11	2.32e−15	1.19e−16	4.89e−02	2.26e+01	1.53e+00	1.40e−01
ABCPW	Mean	**0**	**2.92e+04**	**1.18e+01**	**0**	**6.98e−15**	**3.64e−15**	**1.10e−16**	**2.09e+01**	**1.28e+02**	**2.77e+01**	**1.12e+00**
	SD	**0**	**9.89e+03**	**1.07e+01**	**0**	**4.82e−15**	**2.89e−15**	**1.73e−17**	**1.56e−02**	**5.47e+01**	**1.98e+00**	**2.30e−01**

Table 15
Comparison among ABCs on frequency modulator synthesis problem.

	Best	Worst	Median	Mean	SD
ABCPW	**3.35e−07**	**9.82e−06**	**8.46e−07**	**1.03e−06**	**5.75e−07**
CABC	4.26e−04	9.85e−03	3.27e−03	$5.16e{-03}^{+}$	4.18e−03
ABCbest	6.49e−03	2.75e−02	1.23e−02	$2.04e{-02}^{+}$	1.80e−02
GABC	9.05e−03	7.32e−01	2.03e−02	$6.26e{-02}^{+}$	6.03e−02
ABC	2.98e+01	3.14e+01	3.10e+01	$3.08e{+01}^{+}$	5.94e−01

$$y(t) = a_1 sin(\omega_1 \cdot t \cdot \theta - a_2 \cdot sin(\omega_2 \cdot t \cdot \theta + a_3 \cdot sin(\omega_3 \cdot t \cdot \theta))), \quad (12)$$

where $\theta = 2\pi/100$.

The goal is to minimize the sum of squared errors shown in (13). This problem is an actually complex multimodal optimization problem whose optimum value is 0.

$$f(X) = \sum_{t=0}^{100} (y(t) - y_0(t))^2. \quad (13)$$

The upper and lower bounds of the optimal parameters $X = (a_1, \omega_1, a_2, \omega_2, a_3, \omega_3)$ are −6.4 and 6.35 for each dimension, respectively. The maximum number of function evaluations is to be 500,000. The experimental results in terms of the best, worst, median, mean and standard deviation values obtained by ABCPW, CABC, ABCbest, GABC, and ABC over 30 independent runs are reported in Table 15. It can be seen that ABCPW has the best performance than CABC, ABCbest, GABC, and ABC.

5. Conclusion

With regard to the drawback of slow convergence of ABC, this paper considers to employ the multistrategy technique to generate the high quality solutions. While the major disadvantage of the multistrategy technique is its expensive computational cost. To address the problem, a density estimation method is employed to estimate the quality of the candidate individuals and choose one as the offspring. In addition, fitness-based neighborhood and distance-based neighborhood are introduced to not only keep the population diversity, but also improve the convergence. Based on the above consideration, a novel approach, named ABCPW, is proposed.

When compared to several popular methods, ABCPW exhibits superior performance on a series of benchmark functions. How to develop ABCPW to handle complicated big data optimization problems is our further work. It may also be meaningful to extend this approach to treat dynamic optimization, constrained optimization and multimodal optimization.

Acknowledgments

This work was supported in part by the National Nature Science Foundation of China under Grant 61772391 and Grant 61402534, in part by the Natural Science Basic Research Plan in Shaanxi Province of China under Grant 2018JQ6051, in part by the Young Talent fund of University Association for Science and Technology in Shaanxi, China, and in part by Hong Kong Scholars Program.

References

[1] K.S. Tang, K.F. Man, S. Kwong, Q. He, Genetic algorithms and their applications, IEEE Signal Process. Mag. 13 (6) (1996) 22–37.
[2] M. Dorigo, T. Stutzle, Ant Colony Optimization, MIT Press, Cambridge, MA, USA, 2004.
[3] R. Storn, K. Price, Differential evolution–A simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.
[4] D. Simon, Biogeography-based optimization, IEEE Trans. Evol. Comput. 12 (6) (2008) 702–713.
[5] J. Kennedy, R.C. Eberhart, Particle swarm optimization, in: Proc. IEEE Int. Conf. Neural Netw. vol. 4, Perth, WA, Australia, 1995, pp. 1942–1948.
[6] D. Karaboga, An idea based on honey bee swarm for numerical optimization, Dept. Comput. Sci. Erciyes Univ. Kayseri, Turkey, Tech. Rep. TR06, Oct(2005).
[7] N. Hansen, The CMA Evolution Strategy: A Comparing Review, Towards a New Evolutionary Computation. Advances on Estimation of Distribution Algorithms, Springer, 2006, pp. 75–102.
[8] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm, J. Global Optim. 39 (3) (2007) 459–471.
[9] D. Karaboga, B. Basturk, On the performance of artificial bee colony (ABC) algorithm, Appl. Soft Comput. 8 (1) (2008) 687–697.
[10] D. Karaboga, B. Basturk, A comparative study of artificial bee colony algorithm, Appl. Math. Comput. 214 (1) (2009) 108–132.
[11] D. Karaboga, Artificial bee colony algorithm, Scholarpedia 5 (3) (2010) 6915.
[12] C. Fong, H. Asmuni, B. McCollum, A hybrid swarm-based approach to university timetabling, IEEE Trans. Evol. Comput. 19 (6) (2015) 870–884.
[13] W. Gao, S. Liu, F. Jiang, An improved artificial bee colony algorithm for directing orbits of chaotic systems, Appl. Math. Comput. 218 (7) (2011) 3868–3879.
[14] R. Vural, T. Yildirim, T. Kadioglu, A. Basargan, Performance evaluation of evolutionary algorithms for optimal filter design, IEEE Trans. Evol. Comput. 16 (1) (2012) 135–147.
[15] D. Karaboga, B. Gorkemli, C. Ozturk, N. Karaboga, A comprehensive survey: Artificial bee colony (ABC) algorithm and, Artif. Intell. Rev. 42 (1) (2014) 21–57.
[16] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2009) 398–417.
[17] H. Wang, et al., Multi-strategy ensemble artificial bee colony algorithm, Inf. Sci. 279 (2014) 587–603.
[18] W. Gong, A. Zhou, Z. Cai, A multioperator search strategy based on cheap surrogate models for evolutionary optimization, IEEE Trans. Evol. Comput. 19 (5) (2015) 746–758.
[19] G. Zhu, S. Kwong, Gbest-guided artificial bee colony algorithm for numerical function optimization, Appl. Math. Comput. 217 (7) (2010) 3166–3173.

[20] G.Q. Li, P.F. Niu, X.J. Xiao, Development and investigation of efficient artificial bee colony algorithm for numerical function optimization, Appl. Soft Comput. 12 (1) (2012) 320–332.
[21] A. Banharnsakun, T. Achalakul, B. Sirinaovakul, The best-so-far selection in artificial bee colony algorithm, Appl. Soft Comput. 11 (2) (2011) 2888–2901.
[22] K. Diwold, A. Aderhold, A. Scheidler, M. Middendorf, Performance evaluation of artificial bee colony optimization and new selection schemes, Memetic Comput. 3 (3) (2011) 149–162.
[23] S. Das, S. Biswas, B.K. Panigrahi, S. Kundu, D. Basu, A spatially informative optic flow model of bee colony with saccadic flight strategy for global optimization, IEEE Trans. Cybern. 44 (10) (2014) 1884–1897.
[24] W. Gao, L.L. Huang, S.Y. Liu, C. Dai, Artificial bee colony algorithm based on information learning, IEEE Trans. Cybern. 45 (12) (2015) 2827–2839.
[25] A. Banitalebi, et al., Enhanced compact artificial bee colony, Inf. Sci. 298 (2015) 491–511.
[26] L. Cui, et al., A ranking-based adaptive artificial bee colony algorithm for global numerical optimization, Inf. Sci. 417 (2017) 169–185.
[27] D. Karaboga, B. Gorkemli, A quick artificial bee colony (qABC) algorithm and its performance on optimization problems, Appl. Soft Comput. 23 (2014) 227–238.
[28] L. Cui, et al., A novel artificial bee colony algorithm with depth-first search framework and elite-guided search equation, Inf. Sci. 367–368 (2016) 1012–1044.
[29] W. Xiang, et al., An improved artificial bee colony algorithm based on the gravity model, Inf. Sci. 429 (2013) 49–71.
[30] W. Gao, S.Y. Liu, L.L. Huang, A novel artificial bee colony algorithm based on modified search equation and orthogonal learning, IEEE Trans. Cybern. 43 (3) (2013) 1011–1024.
[31] F. Kang, J.J. Li, Q. Xu, Structural inverse analysis by hybrid simplex artificial bee colony algorithms, Comput. Struct. 87 (13–14) (2009) 861–870.
[32] S. Das, S. Biswas, S. Kundu, Synergizing fitness learning with proximity-based food source selection in artificial bee colony algorithm for numerical optimization, Appl. Soft Comput. 13 (12) (2013) 4676–4694.
[33] F. Kang, J.J. Li, Z.Y. Ma, Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions, Inf. Sci. 181 (16) (2011) 3508–3531.
[34] B. Alatas, Chaotic bee colony algorithms for global numerical optimization, Expert Syst. Appl. 37 (8) (2010) 5682–5687.
[35] W.L. Xiang, M.Q. An, An efficient and robust artificial bee colony algorithm for numerical optimization, Comput. Oper. Res. 40 (5) (2013) 1256–1265.
[36] M.S. Kiran, M. Gunduz, A recombination-based hybridization of particle swarm optimization and artificial bee colony algorithm for continuous optimization problems, Appl. Soft Comput. 13 (4) (2013) 2188–2203.
[37] S. Jadon, et al., Hybrid artificial bee colony algorithm with differential evolution, Appl. Soft Comput. 58 (2017) 11–24.
[38] L. Cui, et al., A novel artificial bee colony algorithm with an adaptive population size for numerical function optimization, Inf. Sci. 414 (2017) 53–67.
[39] M. Kiran, O. Findik, A directed artificial bee colony algorithm, Appl. Soft Comput. 216 (2015) 454–462.
[40] X. Li, G. Yang, Artificial bee colony algorithm with memory, Appl. Soft Comput. 41 (2016) 362–372.
[41] J. Li, Q. Pan, Solving the large-scale hybrid flow shop scheduling problem with limited buffers by a hybrid artificial bee colony algorithm, Inf. Sci. 316 (2015) 487–502.
[42] D. Karaboga, B. Gorkemli, C. Ozturk, N. Karaboga, A comprehensives survey: artificial bee colony (ABC) algorithm and applications, Artif. Intell. Rev. 42 (1) (2014) 21–57.
[43] W.F. Gao, S.Y. Liu, L.L. Huang, A global best artificial bee colony algorithm for global optimization, J. Comput. Appl. Math. 236 (11) (2012) 2741–2753.
[44] J. Brest, S. Greiner, B. Boskovic, M. Mernik, V. Zumer, Selfadapting control parameters in differential evolution: A comparative study on numerical benchmark problems, IEEE Trans. Evol. Comput. 10 (6) (2006) 646–657.
[45] J. Zhang, A.C. Sanderson, JADE: Adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958.
[46] R. Mendes, J. Kennedy, J. Neves, The fully informed particle swarm: Simpler, maybe better, IEEE Trans. Evol. Comput. 8 (3) (2004) 204–210.
[47] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evol. Comput. 10 (3) (2006) 281–295.
[48] A. Ratnaweera, S. Halgamuge, H. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, IEEE Trans. Evol. Comput. 8 (3) (2004) 240–255.
[49] Z.H. Zhan, J. Zhang, Y. Li, Y.H. Shi, Orthogonal learning particle swarm optimization, IEEE Trans. Evol. Comput. 15 (6) (2011) 832–847.
[50] Y.W. Leung, Y. Wang, An orthogonal genetic algorithm with quantization for global numerical optimization, IEEE Trans. Evol. Comput. 5 (1) (2001) 41–53.
[51] Y.P. Wang, C.Y. Dang, An evolutionary algorithm for global optimization based on level-set evolution and Latin squares, IEEE Trans. Evol. Comput. 11 (5) (2007) 579–595.
[52] Y.W. Shang, Y.H. Qiu, A note on the extended Rosenbrock function, Evol. Comput. 14 (1) (2006) 119–126.
[53] B. Basturk, D. Karaboga, A modified artificial bee colony algorithm for real-parameter optimization, Inf. Sci. 192 (1) (2012) 120–142.
[54] K. Chellapilla, Combining mutation operators in evolutionary programming, IEEE Trans. Evol. Comput. 2 (3) (1998) 91–96.
[55] X. Yao, Y. Liu, G.M. Lin, Evolutionary programming made faster, IEEE Trans. Evol. Comput. 3 (2) (1999) 82–102.
[56] C.Y. Lee, X. Yao, Evolutionary programming using mutations based on the Levy probability distribution, IEEE Trans. Evol. Comput. 8 (1) (2004) 1–13.
[57] W.Y. Gong, Z.H. Cai, C.X. Ling, H. Li, Enhanced differential evolution with adaptive strategies for numerical optimization, IEEE Trans. Syst. Man, Cybern. B, Cybern. 41 (2) (2011) 397–413.
[58] Y.J. Zheng, Water wave optimization: A new nature-inspired metaheuristic, Comput. Oper. Res 55 (2015) 1–11.
[59] M. Yazdani, F. Jolai, Lion optimization algorithm (LOA): a nature-inspired metaheuristic algorithm, J. Comput. Des. Eng. 3 (1) (2016) 24–36.
[60] P.N. Suganthan, et al., Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization, Dept. Electr. Electron. Eng. Nanyang Technol. Univ. Singapore, Tech. Rep. 2005, 2005.
[61] S. Das, A. Abraham, U.K. Chakraborty, A. Konar, Differential evolution using a neighborhood based mutation operator, IEEE Trans. Evol. Comput. 13 (3) (2009) 526–553.