Hindawi
Journal of Control Science and Engineering
Volume 2017, Article ID 4851493, 13 pages
https://doi.org/10.1155/2017/4851493

Research Article

Self-Adaptive Artificial Bee Colony for Function Optimization

Mingzhu Tang,[1,2] Wen Long,[3] Huawei Wu,[2] Kang Zhang,[1] and Yuri A. W. Shardt[4]

[1] *School of Energy and Power Engineering, Changsha University of Science & Engineering, Changsha 410114, China*
[2] *Hubei Key Laboratory of Power System Design and Test for Electrical Vehicle, Xiangyang 441053, China*
[3] *Guizhou Key Laboratory of Economics System Simulation, Guizhou University of Finance & Economics, Guiyang 550004, China*
[4] *Department of Chemical Engineering, University of Waterloo, ON, Canada N2L 3G1*

Correspondence should be addressed to Wen Long; lw227@mail.gufe.edu.cn

Received 16 February 2017; Revised 19 April 2017; Accepted 28 May 2017; Published 14 August 2017

Academic Editor: Shen Yin

Artificial bee colony (ABC) is a novel population-based optimization method, having the advantage of less control parameters, being easy to implement, and having strong global optimization ability. However, ABC algorithm has some shortcomings concerning its position-updated equation, which is skilled in global search and bad at local search. In order to coordinate the ability of global and local search, we first propose a self-adaptive ABC algorithm (denoted as SABC) in which an improved position-updated equation is used to guide the search of new candidate individuals. In addition, good-point-set approach is introduced to produce the initial population and scout bees. The proposed SABC is tested on 12 well-known problems. The simulation results demonstrate that the proposed SABC algorithm has better search ability with other several ABC variants.

1. Introduction

Population-based optimization algorithms, such as whale optimization algorithm (WOA) [1], flower pollination algorithm (FPA) [2], bacterial foraging optimizer (BFO) [3], cuckoo search algorithm (CSA) [4], fruit fly optimization (FFO) [5], gravitational search optimizer (GSO) [6], and chemical reaction optimization (CRO) [7], have many advantages over classical optimization methods and have been successfully and broadly applied to solve global continuous optimization problems in the last few decades [6].

In this paper, we used the advantage of ABC, presented by Karaboga [8], which mimics the hunting behaviors of honey bee swarm. Simulation results show that ABC is superior to many other population-based optimization methods, namely, genetic algorithm (GA), evolution strategies (ES), and particle swarm optimization (PSO) [9, 10]. ABC has caused wide attention and applications since its invention in 2005, owing to its simplicity and fewer control parameters. However, ABC faces some challenging problems, namely, low precision and slow convergence. As a result, many improved versions of ABC have been proposed to overcome these shortcomings. Zhu and Kwong [11] presented an improved Gbest-guided ABC (denoted as GABC) through combining the global best (Gbest) individual with the position-updated equation to enhance the ability of local search. Luo et al. [12] presented an improved position-updated equation by using global best individual information to generate offspring individuals. Inspired by DE, Gao et al. [13] developed an improved ABC variant applying the bees search only near the global best one to enhance the exploitation. Xiang and An [14] developed a modified position-updated equation to accelerate the convergence speed. Moreover, chaotic optimization mechanism is introduced to avoid being trapped in local minima. Li et al. [15] presented an improved ABC variant based on inertia weight and accelerating factors and to coordinate the ability of global and local search. Gao and Liu [16] developed two modified position-updated equations, namely, "ABC/best/1" and "ABC/rand/1." Kang et al. [17] developed a hybrid method which combines ABC algorithm and pattern search method to speed up convergence. Gao et al. [18] presented an improved version of ABC (denoted as CABC) based on a modified position-updated equation. In addition, the orthogonal experimental design is introduced.

As we know, for an optimization algorithm, both global and local search are necessary and they should be well coordinated to obtain better global performance [11]. Different from the previous work, the position-updated equation is modified by self-adaptive adopting of the previous and global best solution to generate new candidate offspring to coordinate the ability of global and local search. Moreover, the good-point-set approach is used to generate initial population. The proposed SABC is tested on 12 benchmark global optimization problems. The experimental results show that our SABC algorithm is superior to basic ABC and other ABC variants.

The remainder of this study is organized as follows. The standard ABC is described in Section 2. The improved ABC called SABC algorithm is proposed and analyzed in Section 3. In Section 4, 12 benchmark global optimization problems are used to test the proposed SABC algorithm. Finally, Section 5 summarized the conclusions.

2. Artificial Bee Colony

ABC is metaheuristic optimization method inspired by hunting behavior of honey bee swarm. At the initialization step, generate randomly N solutions to construct an initial population:

$$x_{i,j} = l_{i,j} + \text{rand}(0,1) \cdot \left(u_{i,j} - l_{i,j}\right), \tag{1}$$

where $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, n$; rand(0, 1) is a uniformly distributed random number; $l_{i,j}$ and $u_{i,j}$ are the lower and upper bounds for the dimension j, respectively.

In onlooker bee stage, food source is chosen by probability p_i

$$p_i = \frac{\text{fit}_i}{\sum_{i=1}^{N} \text{fit}_i}, \tag{2}$$

where fit_i denotes the fitness value of $\overrightarrow{X}_i$ and is defined as

$$\text{fit}_i = \begin{cases} \dfrac{1}{1 + f\left(\overrightarrow{X}_i\right)}, & f\left(\overrightarrow{X}_i\right) \geq 0 \\ 1 + \left|f\left(\overrightarrow{X}_i\right)\right|, & f\left(\overrightarrow{X}_i\right) < 0, \end{cases} \tag{3}$$

where $f(\overrightarrow{X}_i)$ denotes the objective function values of the decision vector $\overrightarrow{X}_i$.

A candidate food source position $\overrightarrow{V}_i = (v_{i1}, v_{i2}, \ldots, v_{in})$ can be generated from the old one $\overrightarrow{X}_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ as

$$v_{ij} = x_{ij} + \phi_{ij}\left(x_{ij} - x_{kj}\right), \tag{4}$$

where $j = \{1, 2, \ldots, n\}$ and $k = \{1, 2, \ldots, N\}$ are randomly selected individuals; k is different from i; and ϕ_{ij} is a random uniformly distributed number in $[-1, 1]$.

Pseudocode of basic ABC is given in Algorithm 1.

3. Self-Adaptive Artificial Bee Colony (SABC)

3.1. Population Initialization. Note that, for ABC algorithm, in order to find the existing area of optimal solutions faster, initial population should cover the whole search space. Good-point-set method is widely applied to generate many uniformly distributed candidate individuals [19]. Therefore, we apply good-point-set technique for producing initial population of ABC to maintain diversity of population. The pseudocode of good-point-set technique is given in Algorithm 2.

The uniformity properties of random method and good-point-set method are compared as given in Figure 1 and it displays the 80 points on a unit square generated using random number generator and good-point-set approach.

From Figure 1, the candidate individuals produced through good-point-set technique are more uniform than the candidate individuals produced by random method. Thus, good-point-set method is preferred technique for generating initial population.

3.2. Modified Search Equation. On the basis of the position-updated equation depicted by (4), the offspring individual is produced through moving the previous individual towards (or far away from) another individual chosen randomly in population. Nevertheless, the randomly chosen individual is a good individual or a bad one; the probability is the same. Therefore, the offspring individual could not be guaranteed to be better than the previous one. In addition, the coefficient ϕ_{ij} in (4) is a random number over the range $[-1, 1]$ and x_{kj} is a randomly selected solution from population. Thus, the position-updated equation depicted by (4) is skilled in global search and bad at local search [11].

For the sake of enhancing the performance of ABC, one active research trend is to investigate its position-updated equation. As mentioned above, the characteristics of the search equations of ABC have been extensively investigated. ABC researchers have suggested many empirical guidelines for modifying search equation during the last decade. It has been clear that some search equations can speed up the convergence [11–13], and some others are suitable for the global search [20]. Indubitably, these experiences are extremely helpful for improving the performance of ABC. "ABC/rand/1" and "ABC/best/1" are invariably employed in lots of ABC variants and their characteristics have been commendably investigated. The equations of "ABC/rand/1" and "ABC/best/1" are stated as [16]

$$\text{"ABC/rand/1"}: v_{i,j} = x_{i,j} + \phi_{ij}\left(x_{r_1,j} - x_{r_2,j}\right) \tag{5}$$

$$\text{"ABC/best/1"}: v_{i,j} = x_{\text{best},j} + \phi_{ij}\left(x_{r_1,j} - x_{r_2,j}\right), \tag{6}$$

where r_1 and r_2 are randomly selected from $\{1, 2, \ldots, N\}$, and $r_1 \neq r_2 \neq i$. $x_{\text{best},j}$ is global best solution position vector and $j \in \{1, 2, \ldots, n\}$ denotes j dimension. ϕ_{ij} is a uniformly distributed random number in $[-1, 1]$. The "ABC/rand/1" search equation is one of many often used in the paper. In "ABC/rand/1" equation, all positions are randomly chosen in population and, consequently, it does not have any bias

```
(01) Initialize the parameters.
(02) Initialize $N$ solutions to construct an initial population.
(03) Evaluate the fitness values of each solution.
(04) cycle = 1.
(05) **Repeat**
(06)     Generate a offspring individual by Eq. (4) and evaluate its quality.
(07)     Compare and select the better one.
(08)     Calculate probabilities through Eq. (2).
(09)     Produce randomly a number in $[0, 1]$.
(10)     Generate a offspring individual by Eq. (4) and evaluate its quality.
(11)     Compare and select the better one.
(12)     Memorize the best solution achieved so far.
(13)     cycle = cycle + 1.
(14) **Until** cycle=Maximum cycle number
```

Algorithm 1: Pseudo-code of basic ABC algorithm.

```
(01) Set the population size $N$; the decision variables dimension $n$, $i = 1$, $j = 1$.
(02) **For** $i = 1$ to $n$ **do**
(03)     **For** $j = 1$ to $N$ **do**
(04)         $a = 2 * n + 3$;
(05)         **While** $p \sim= a$ **do**
(06)             Set individual counter $k = 2$.
(07)             **For** $k = 2$ to $a - 1$ **do**
(08)                 **If** $\mathrm{mod}(a, i) == 0$ **then**
(09)                     $a = a + 1$;
(10)                 **Else**
(11)                     $p = a$;
(12)                 **End if**
(13)             **End for**
(14)         **End while**
(15)         $r(i) = 2 * \cos(2 * pi * i/p)$;
(16)         $x(i) = \mathrm{mod}(r(i) * N, 1)$;
(17)     **End for**
(18) **End for**
```

Algorithm 2: Good point set method.

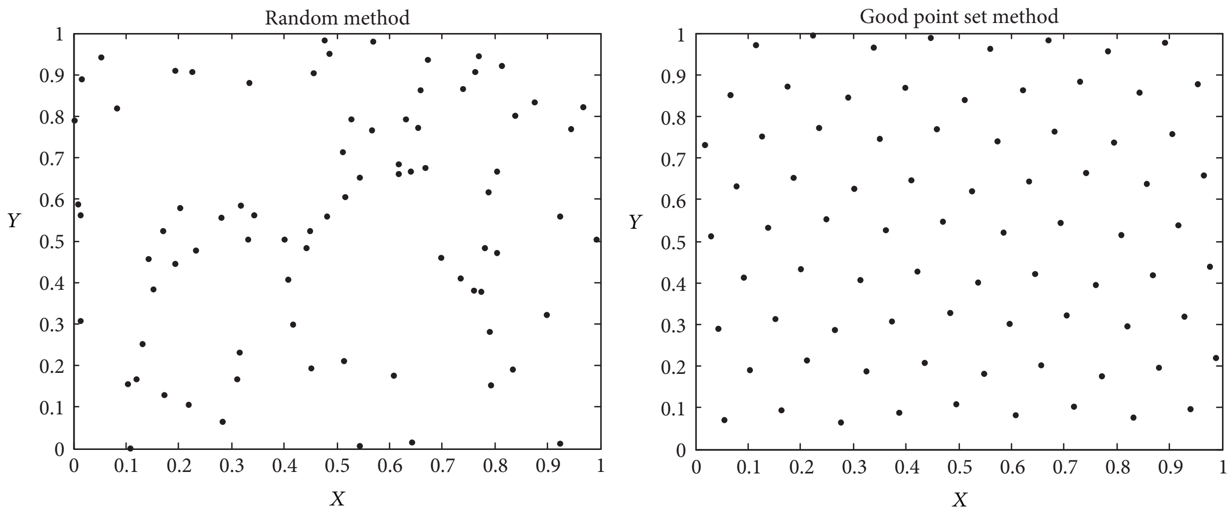

Figure 1: 80 points in the unit squares generated through random and good-point-set approaches.

in particular positions and randomly selects new positions. Therefore, they often show better global search ability with slow convergence. "ABC/best/1" has faster convergence based on global optima position. However, they are more likely to fall into local optima.

In the ABC algorithm, appropriate coordinate of global search and local search is very important for obtaining the optima effectively. In order to bring about a balance between global and local search abilities of ABC, we first introduces a parameter λ, which coordinates the effect of previous position on the current one, by modifying (4) to

$$v_{i,j} = \lambda x_{i,j} + (1-\lambda) x_{\text{best},j} + \phi_{i,j}\left(x_{r_1,j} - x_{r_2,j}\right), \quad (7)$$

where $\lambda \in [0,1]$ is utilized to influence the balance between global and local search abilities of individuals, the indices r_1 and r_2 are randomly selected from $\{1,2,\ldots,N\}$, and $r_1 \neq r_2 \neq i$. $x_{\text{best},j}$ is global best solution position vector and $j \in \{1,2,\ldots,n\}$ denotes j dimension. ϕ_{ij} is a uniformly distributed random number.

Note that the parameter λ is crucial in balancing the abilities of global and local search. When λ is equal to 1, (7) becomes (5). When λ decreases from 1 to 0, the global search ability of (7) will also decrease correspondingly. When λ takes 0, (7) is (6). The search performance of algorithm will adaptively adjust through changing the parameter λ. From (7), with a large value of λ in the early stage, individuals are admitted to move in all directions of solution space, instead of moving towards the best individual. A small value of λ allows the population to converge to the global optimal solution in later stage. Therefore, a well-tuned λ is very important. The parameter λ in this paper is calculated according to the following function:

$$\lambda = \frac{T-t}{T}, \quad (8)$$

where t denotes number of iterations and T denotes a predefined maximum number of iterations. Thus, the improved position-updated equation developed by (7) is able to balance the abilities of global and local search of SABC.

3.3. Rank Selection. As we know, the roulette wheel selection mechanism is employed in classical ABC. This selection strategy makes more chance to select individuals with higher fitness [14]. To keep population diversity better, a rank-based selection mechanism is introduced in this paper. Selection probability p_i is defined [21]:

$$p_i = \frac{1}{N} + a(t)\frac{N+1-2i}{N(N+1)}, \quad i = 1,2,\ldots,N. \quad (9)$$

$$a(t) = 0.2 + \frac{3t}{4t_{\max}}, \quad (10)$$

where N is the population size, $a(t)$ is parameters, t is current iteration, $t_{\max}$ is maximum iteration.

3.4. The Proposed SABC Algorithm. The flow chart of proposed SABC algorithm is presented in Figure 2.

4. Simulation Results and Comparisons

4.1. Benchmark Test Functions. To evaluate the performance of SABC, 12 benchmark test problems from [14, 15, 22] are used and are shown in Table 1.

4.2. Comparison between SABC and Basic ABC Algorithm. In SABC, population size N is 40, limit is 100, and the maximum iteration is 1000. Each experiment is repeated 30 runs independently. The experimental results of SABC and basic ABC are given in Table 2 regarding the best, the mean, the worst, the standard deviation (St.dev), and the convergence iteration (CI).

As seen from Table 2, SABC is able to obtain global optima for three functions (f_6, f_8, and f_{10}). Moreover, in seven functions (f_1–f_4, f_9, f_{11}, and f_{12}), the results obtained by SABC are pretty near to global optimal solution. Compared with ABC, SABC can obtain much better results than ABC on 12 functions, that is, f_1–f_{12}. The convergence performance of SABC and basic ABC on 12 functions are drawn in Figure 3 so as to show the performance of SABC more clearly.

4.3. Comparison between SABC and ABC Variants. SABC is compared against four high-performance ABC algorithms under three performance evaluation criteria: Mean, St.dev, and CI. These selected ABC variants such as Gbest-guided ABC (denoted as GABC) algorithm [11], an efficient and robust ABC (abbreviated as ERABC) algorithm [14], improved ABC (abbreviated as IABC) algorithm [15] and prediction and selection ABC (denoted as PSABC) [15]. The parameters settings are the same as those of ERABC [14] and PSABC [15] together with SABC. The comparative results have been shown in Tables 3 and 4. The results provided by other algorithms were directly taken from the original references for each approach.

From Tables 3 and 4, compared to GABC, SABC could achieve much better "mean" and "standard deviation" results than GABC on 11 problems except for function f_7. For the function f_7, SABC algorithm obtained similar results. In addition, the convergence iteration obtained by SABC is smaller than those of ABC, except for function f_7.

As can be seen from Tables 3 and 4, with respect to ERABC, SABC obtained better "mean" and "standard deviation" values on two functions (f_3 and f_4) and similar results on four test functions (f_6, f_8, f_9, and f_{10}). However, ERABC algorithm provided better results on six problems (f_1-f_2, f_5, f_7, and f_{11}-f_{12}) than SABC.

From Tables 3 and 4, compared to IABC, SABC obtained better results on five functions (f_3, f_5, f_6, f_{11}, and f_{12}) and similar solutions on four functions (f_7, f_8, f_9, and f_{10}). However, IABC provided better solutions on three functions (f_1, f_2, and f_4) than SABC algorithm.

In comparison with PSABC, in Tables 3 and 4, SABC provided better solutions on four test functions (f_3, f_5, f_6, and f_{11}) and similar results on five functions (f_7, f_8, f_9, f_{10}, and f_{12}). However, the PSABC algorithm found better solutions on three functions (f_1, f_2, and f_4).

Table 1: Benchmark test functions.

Test functions	Dimension	Search range	Min
$f_1(X)=\sum_{i=1}^{n}x_i^2$	30 and 50	$[-100,100]^n$	0
$f_2(X)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	30 and 50	$[-10,10]^n$	0
$f_3(X)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}x_j\right)^2$	30 and 50	$[-100,100]^n$	0
$f_4(X)=\max_i\{\lvert x_i\rvert,\ 1\le i\le n\}$	30 and 50	$[-100,100]^n$	0
$f_5(X)=\sum_{i=1}^{n-1}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(x_i-1\right)^2\right]$	30 and 50	$[-30,30]^n$	0
$f_6(X)=\sum_{i=1}^{n}\left(\lfloor x_i+0.5\rfloor\right)^2$	30 and 50	$[-100,100]^n$	0
$f_7(X)=\sum_{i=1}^{n}ix_i^4+\text{random}[0,1)$	30 and 50	$[-1.28,1.28]^n$	0
$f_8(X)=\sum_{i=1}^{n}\left(x_i^2-10\cos\left(2\pi x_i\right)+10\right)$	30 and 50	$[-5.12,5.12]^n$	0
$f_9(X)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos\left(2\pi x_i\right)\right)+20+e$	30 and 50	$[-32,32]^n$	0
$f_{10}(X)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	30 and 50	$[-600,600]^n$	0
$f_{11}(X)=\frac{\pi}{n}\left\{10\sin\left(\pi y_1\right)+\sum_{i=1}^{n-1}\left(y_i-1\right)^2\left[1+10\sin\left(\pi y_{i+1}\right)^2\right]+\left(y_n-1\right)^2\right\}+\sum_{i=1}^{n}u\left(x_i,10,100,4\right)$ $y_i=1+\frac{x_i+1}{4};\ u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m, & x_i>a\\ 0, & -a\le x_i\le a\\ k(-x_i-a)^m, & x_i<-a\end{cases}$	30 and 50	$[-50,50]^n$	0
$f_{12}(X)=0.1\left\{\sin\left(3\pi x_1\right)^2+\sum_{i=1}^{n}\left(x_i-1\right)^2\left[1+\sin\left(3\pi x_i+1\right)^2\right]+\left(x_n-1\right)^2\left[1+\sin\left(2\pi x_n\right)^2\right]\right\}+\sum_{i=1}^{n}u\left(x_i,5,100,4\right)$	30 and 50	$[-50,50]^n$	0

Table 2: Experimental results of SABC and ABC in f_1–f_{12}.

Number	Dimension	Methods	Best	Mean	Worst	SD	CI
f_1	30	ABC	$4.6231E-10$	$7.1171E-10$	$1.1766E-09$	$4.0299E-10$	982
		SABC	$4.1163E-25$	$1.1061E-24$	$1.5769E-23$	$1.8645E-24$	467
	50	ABC	$1.1506E-05$	$6.1887E-05$	$1.3418E-04$	$6.4204E-05$	1000
		SABC	$1.4239E-25$	$6.0779E-24$	$1.5665E-23$	$4.3555E-24$	567
f_2	30	ABC	$7.1038E-07$	$2.1352E-06$	$1.8472E-05$	$1.6161E-06$	1000
		SABC	$2.2631E-17$	$4.3501E-17$	$6.8724E-17$	$5.9788E-18$	988
	50	ABC	$1.4589E-03$	$2.5687E-03$	$4.7673E-03$	$7.2431E-04$	1000
		SABC	$8.3862E-13$	$3.2128E-12$	$7.0340E-12$	$2.6239E-12$	951
f_3	30	ABC	$2.3747E+02$	$4.0476E+02$	$7.3452E+02$	$1.8433E+02$	1000
		SABC	$1.1928E-24$	$4.6715E-24$	$8.3532E-24$	$2.9293E-24$	393
	50	ABC	$1.0263E+03$	$1.3498E+03$	$1.8414E+03$	$3.1333E+02$	1000
		SABC	$3.7284E-24$	$8.7797E-24$	$1.8362E-23$	$5.5293E-24$	565
f_4	30	ABC	19.2342	21.7466	24.3791	1.6750	1000
		SABC	$3.7668E-13$	$1.5260E-12$	$4.4526E-12$	$8.9061E-13$	952
	50	ABC	48.3573	56.7818	70.2795	9.5092	1000
		SABC	$1.0362E-12$	$2.5018E-12$	$6.5874E-12$	$1.8606E-12$	983
f_5	30	ABC	1.0598	2.4438	4.6864	1.3676	1000
		SABC	$1.0148E-04$	$1.9505E-04$	$3.7937E-04$	$2.7217E-04$	999
	50	ABC	18.4882	23.7003	28.5976	4.8506	1000
		SABC	$1.2218E-01$	$2.3657E-01$	$4.1856E-01$	$1.1026E-01$	1000
f_6	30	ABC	$8.5740E-20$	$2.2257E-19$	$3.2408E-19$	$8.3640E-20$	588
		SABC	0	0	0	0	97
	50	ABC	$1.7016E-02$	$3.3668E-01$	$9.6818E-01$	$3.7865E-01$	985
		SABC	0	0	0	0	108
f_7	30	ABC	$8.9359E-02$	$1.8702E-01$	$2.6427E-01$	$5.6650E-02$	1000
		SABC	$7.8137E-03$	$1.4728E-02$	$3.6362E-02$	$6.9294E-03$	1000
	50	ABC	$1.9388E-01$	$4.3249E-01$	$8.9758E-01$	$3.1615E-01$	1000
		SABC	$5.9068E-02$	$8.4158E-02$	$1.0092E-01$	$1.7730E-02$	998
f_8	30	ABC	$3.7327E-08$	$1.4133E-05$	$4.2206E-05$	$2.4312E-05$	956
		SABC	0	0	0	0	109
	50	ABC	5.59127	6.47346	7.19213	0.81286	1000
		SABC	0	0	0	0	253
f_9	30	ABC	$1.4010E-05$	$3.2429E-05$	$5.0576E-05$	$1.8280E-05$	1000
		SABC	$4.4409E-15$	$7.9906E-15$	$1.1546E-14$	$9.3294E-16$	802
	50	ABC	$2.6146E-02$	$5.1473E-02$	$8.5138E-02$	$3.0367E-02$	1000
		SABC	$1.5099E-14$	$1.5099E-14$	$1.5099E-14$	0	817
f_{10}	30	ABC	$8.3116E-08$	$4.9243E-03$	$1.4773E-02$	$8.5290E-03$	1000
		SABC	0	0	0	0	243
	50	ABC	$2.1268E-05$	$1.8178E-02$	$5.4485E-02$	$3.1443E-02$	1000
		SABC	0	0	0	0	624
f_{11}	30	ABC	$2.3733E-12$	$7.2076E-12$	$1.4228E-11$	$6.2224E-12$	1000
		SABC	$1.0582E-24$	$3.7213E-24$	$7.4688E-24$	$2.2940E-24$	546
	50	ABC	$7.9534E-08$	$1.8292E-07$	$3.2155E-07$	$1.2480E-07$	1000
		SABC	$1.5732E-19$	$2.2735E-19$	$3.9719E-19$	$8.7756E-20$	681
f_{12}	30	ABC	$2.3336E-10$	$6.4815E-10$	$1.3293E-09$	$5.9458E-10$	1000
		SABC	$2.3648E-18$	$4.6324E-18$	$7.2155E-18$	$1.3751E-18$	696
	50	ABC	$6.3245E-07$	$2.2850E-06$	$5.5056E-06$	$2.7894E-06$	1000
		SABC	$4.4409E-15$	$6.8094E-15$	$7.9936E-15$	$2.0512E-15$	834

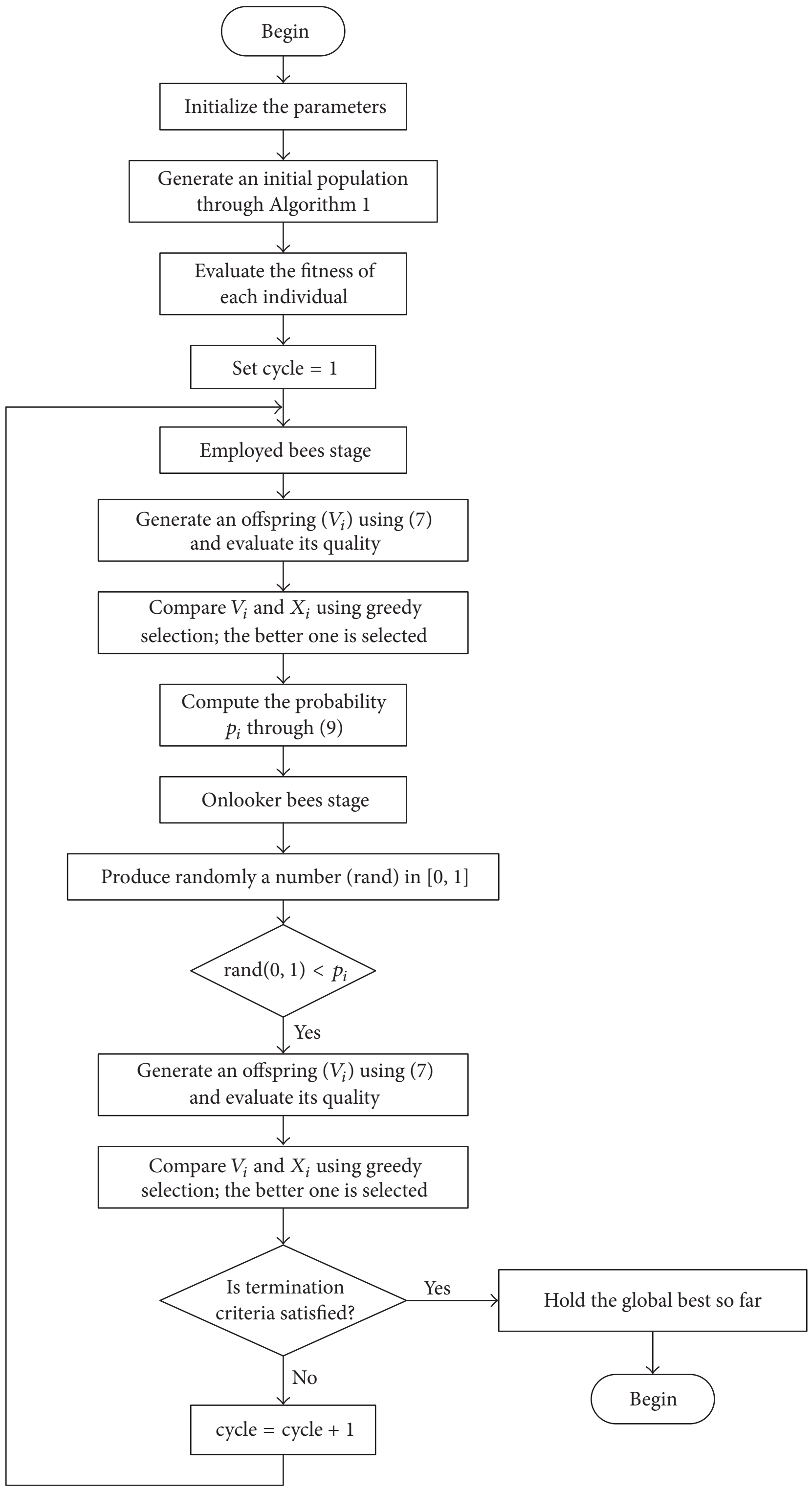
Begin
Initialize the parameters
Generate an initial population through Algorithm 1
Evaluate the fitness of each individual
Set cycle = 1
Employed bees stage
Generate an offspring (V_i) using (7) and evaluate its quality
Compare V_i and X_i using greedy selection; the better one is selected
Compute the probability p_i through (9)
Onlooker bees stage
Produce randomly a number (rand) in [0, 1]
rand(0, 1) < p_i
Yes
Generate an offspring (V_i) using (7) and evaluate its quality
Compare V_i and X_i using greedy selection; the better one is selected
Is termination criteria satisfied?
Yes
Hold the global best so far
Begin
No
cycle = cycle + 1

FIGURE 2: Flow chart of proposed SABC algorithm.

4.4. Effects of Limit on the Performance of SABC. To investigate the impact of limit, four different test functions with $D = 30$ are used. They are Sphere (f_1), Rastrigin (f_8), Ackley (f_9), and Griewank (f_{10}). Five different values of limit (i.e., 50, 100, 150, 200, and 250) are used to optimize the four high dimensional test functions. The mean values (mean) and the standard deviation values (St.dev) are shown in Table 5. The box plots of different limit values for four functions are presented in Figure 4.

From Table 5, parameter limit is able to influence the performance of SABC. When limit is equal to 100, SABC achieves better performance for three functions (f_1, f_8, and

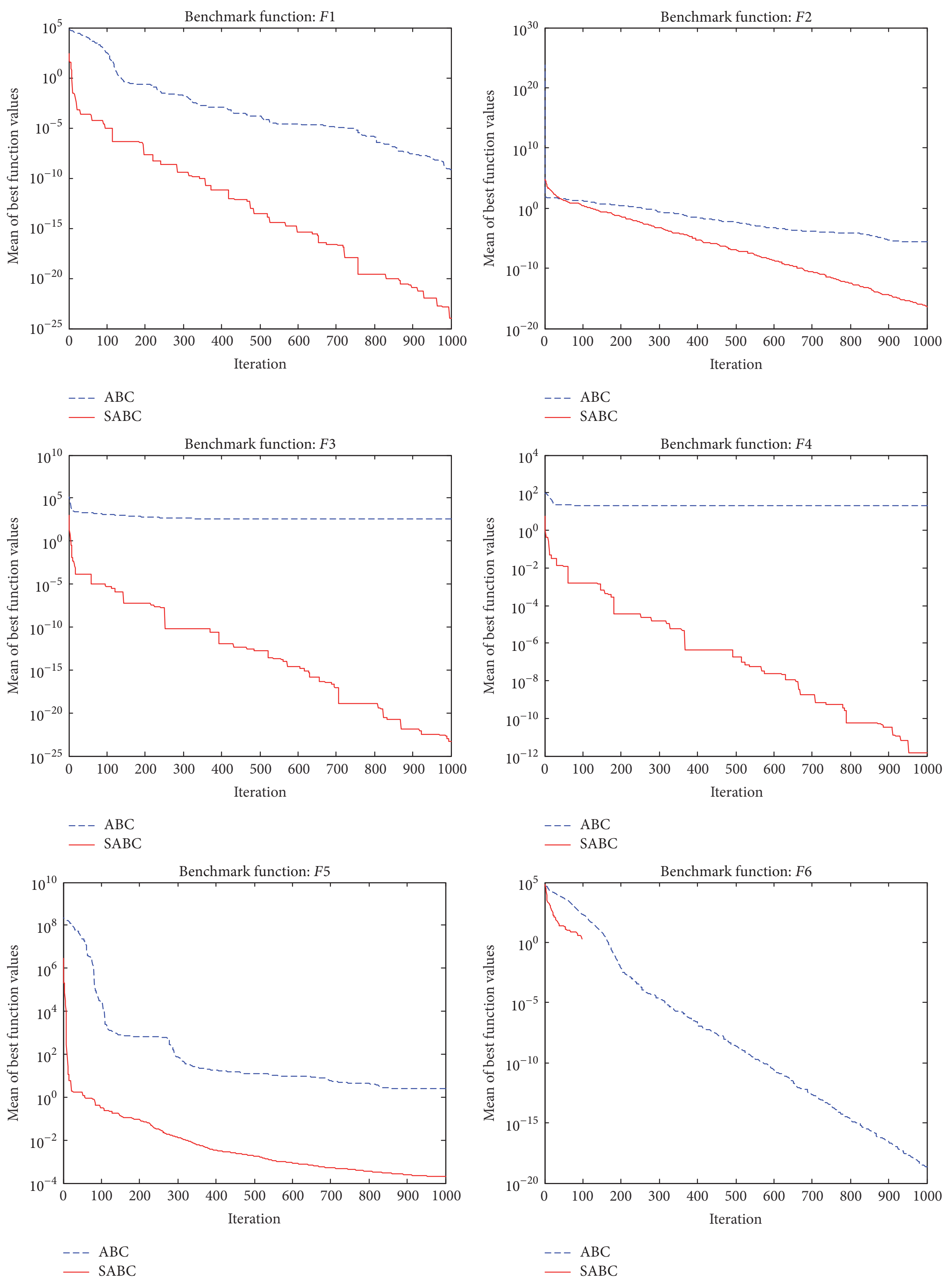

FIGURE 3: Continued.

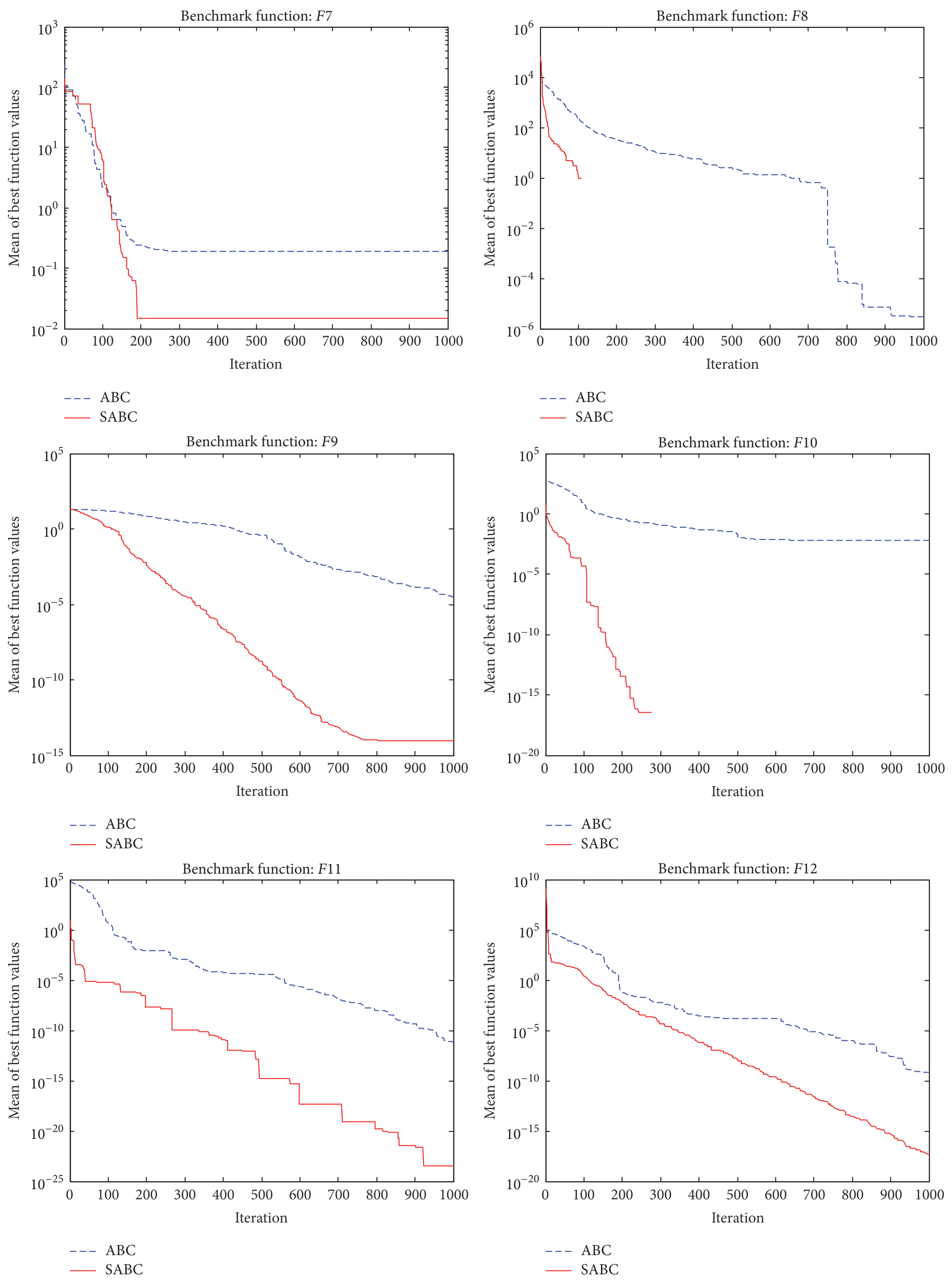

FIGURE 3: Convergence curve of SABC and basic ABC for twelve functions ($D = 30$).

Table 3: Performance comparison of GABC, ERABC, IABC, PSABC, and SABC on test functions f_1–f_{12} (Dim = 30).

Number		GABC	ERABC	IABC	PSABC	SABC
f_1	Mean	$6.2643E-16$	**0**	**0**	**0**	$1.1061E-24$
	St.dev	$1.0859E-16$	**0**	**0**	**0**	$1.8645E-24$
	CI	986	842	**322**	336	467
f_2	Mean	$1.3019E-10$	$2.1049E-233$	**0**	**0**	$4.3501E-17$
	St.dev	$4.6859E-11$	0	**0**	**0**	$5.9788E-18$
	CI	1000	1000	30	**22**	988
f_3	Mean	$1.0939E+04$	$1.2202E-20$	$1.4344E+04$	$6.1069E+03$	**4.6715E − 24**
	St.dev	$2.5670E+03$	0	$2.7291E+03$	$1.6947E+03$	**2.9293E − 24**
	CI	1000	998	999	1000	**393**
f_4	Mean	12.6211	$2.7073E-08$	**1.207E − 197**	$8.591E-115$	$1.5260E-12$
	St.dev	2.6556	$1.2604E-07$	**0**	$4.705E-114$	$8.9061E-13$
	CI	1000	985	1000	1000	**952**
f_5	Mean	7.47961	**1.5450E − 06**	26.4282	1.5922	$1.9505E-04$
	St.dev	19.0926	**7.5654E − 06**	1.3956	4.4066	$2.7217E-04$
	CI	1000	**999**	1000	1000	999
f_6	Mean	$6.4499E-16$	**0**	$3.8463E-10$	$5.7169E-16$	**0**
	St.dev	$1.1126E-16$	**0**	$2.3239E-10$	$8.2549E-17$	**0**
	CI	997	**20**	999	671	97
f_7	Mean	$8.4786E-02$	**9.7185E − 05**	$1.9609E-02$	$2.1514E-02$	$1.4728E-02$
	St.dev	$2.7907E-02$	**6.7013E − 05**	$9.3459E-03$	$6.8816E-02$	$6.9294E-03$
	CI	983	**797**	1000	997	1000
f_8	Mean	$3.3165E-01$	**0**	**0**	**0**	**0**
	St.dev	$1.8165E-01$	**0**	**0**	**0**	**0**
	CI	1000	143	84	**80**	109
f_9	Mean	$7.7828E-10$	$1.0066E-15$	**8.8817E − 16**	**8.8817E − 16**	$7.9906E-15$
	St.dev	$2.9817E-10$	$6.4863E-16$	**0**	**0**	$9.3294E-16$
	CI	1000	**93**	156	188	802
f_{10}	Mean	$6.9655E-04$	**0**	**0**	**0**	**0**
	St.dev	$2.2609E-03$	**0**	**0**	**0**	**0**
	CI	1000	**58**	684	834	243
f_{11}	Mean	$5.8570E-16$	**1.5705E − 32**	$7.1096E-12$	$5.5312E-16$	$3.7213E-24$
	St.dev	$1.1349E-16$	**5.5674E − 48**	$5.2513E-12$	$8.6858E-17$	$2.2940E-24$
	CI	938	**105**	1000	625	546
f_{12}	Mean	$2.1724E-07$	**1.3498E − 32**	$4.7831E-08$	$6.0601E-18$	$4.6324E-18$
	St.dev	$5.6676E-07$	**5.5674E − 48**	$2.0359E-07$	$5.6064E-18$	$1.3751E-18$
	CI	996	**105**	995	567	696

f_{10}). For the Ackley function (f_9), the effect of limit on the performance of SABC is very little. From Figure 3 and Table 5, parameter limit = 100 is a suitable choice in SABC algorithm.

5. Conclusion

An improved version of ABC, called SABC, is developed by using good-point-set initialization employed to enhance the population distribution, rank-based selection strategy used to enhance the global search ability, self-adaptive position-updated equation applying for balancing the exploration and exploitation. The proposed SABC is tested on 12 well-known global optimization problems. The simulation results show that our algorithm is superior to the conventional ABC and other ABC variants. The further work includes the studies on how to develop SABC to deal with those constrained and engineering design problems.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (nos. 61403046, 61463009), the Natural Science Foundation of Hunan Province, China (no. 2015JJ3005), Science and Technology Foundation of Guizhou

Table 4: Performance comparison of GABC, ERABC, IABC, PSABC, and SABC on test functions f_1–f_{12} (Dim = 50).

Number		GABC	ERABC	IABC	PSABC	SABC
f_1	Mean	$1.2546E-05$	**0**	**0**	**0**	$6.0779E-24$
	St.dev	$6.0511E-09$	**0**	**0**	**0**	$4.3555E-24$
	CI	1000	934	726	628	**567**
f_2	Mean	$2.3671E-05$	$1.2627E-226$	**0**	**0**	$3.2128E-12$
	St.dev	$6.1989E-06$	0	**0**	**0**	$2.6239E-12$
	CI	1000	1000	**61**	74	951
f_3	Mean	$4.1236E+04$	**1.0937E − 141**	$4.6927E+04$	$3.0118E+04$	$8.7797E-24$
	St.dev	$5.8269E+03$	**5.9904E − 141**	$7.3584E+03$	$4.1073E+03$	$5.5293E-24$
	CI	1000	1000	1000	1000	**565**
f_4	Mean	45.3075	$1.3695E-08$	25.5055	19.6683	**2.5018E − 12**
	St.dev	4.3151	$5.0911E-08$	5.6662	6.3094	**1.8606E − 12**
	CI	1000	992	999	1000	**983**
f_5	Mean	25.7164	**1.1000E − 03**	47.0287	34.4913	$2.3657E-01$
	St.dev	31.75811	**3.7000E − 03**	0.8601	30.3412	$1.1026E-01$
	CI	1000	**983**	1000	1000	1000
f_6	Mean	$5.6529E-09$	**0**	$1.8434E-05$	$1.1674E-15$	**0**
	St.dev	$3.6854E-09$	**0**	$1.7466E-05$	$1.4114E-16$	**0**
	CI	1000	**44**	1000	1000	108
f_7	Mean	$2.4609E-01$	**9.4755E − 05**	$8.8306E-02$	$6.5309E-02$	$8.4158E-02$
	St.dev	$4.7278E-02$	**6.5181E − 05**	$2.5540E-02$	$1.7705E-02$	$1.7730E-02$
	CI	1000	**979**	998	966	998
f_8	Mean	2.1733	**0**	**0**	**0**	**0**
	St.dev	1.0728	**0**	**0**	**0**	**0**
	CI	1000	173	172	**140**	253
f_9	Mean	$1.1137E-04$	$1.2434E-15$	**8.8817E − 16**	**8.8817E − 16**	$1.5099E-14$
	St.dev	$3.8873E-05$	$1.0840E-15$	**0**	**0**	0
	CI	1000	109	**306**	365	817
f_{10}	Mean	$1.0470E-03$	**0**	**0**	**0**	**0**
	St.dev	$2.7482E-03$	**0**	**0**	**0**	**0**
	CI	1000	**71**	696	836	624
f_{11}	Mean	$9.3017E-11$	**9.4233E − 33**	$5.4223E-07$	$1.0252E-15$	$2.2735E-19$
	St.dev	$7.9664E-11$	**2.7837E − 48**	$2.9821E-07$	$1.5815E-16$	$8.7756E-20$
	CI	1000	**113**	999	979	681
f_{12}	Mean	$8.8776E-07$	**1.3498E − 32**	$2.4156E-05$	$5.0541E-18$	$6.8094E-15$
	St.dev	$1.5324E-06$	**5.5674E − 48**	$4.3568E-05$	$1.5350E-16$	$2.0512E-15$
	CI	999	**123**	998	959	834

Table 5: The experimental results of SABC with different limit values.

Function		Limit = 50	Limit = 100	Limit = 150	Limit = 200	Limit = 250
Sphere (f_1)	Mean	$1.6723E-22$	**1.1061E − 24**	$2.5730E-23$	$1.7838E-20$	$3.1027E-19$
	St.dev	$9.1091E-23$	**1.8645E − 24**	$1.4855E-23$	$1.7195E-20$	$7.3276E-19$
Rastrigin (f_8)	Mean	$1.8629E-14$	**0**	**0**	$2.0400E-14$	$6.6501E-14$
	St.dev	$3.2113E-14$	**0**	**0**	$1.9637E-14$	$7.7353E-14$
Ackley (f_9)	Mean	$1.5328E-14$	**7.9906E − 15**	$8.4720E-15$	$9.4136E-15$	$1.8826E-14$
	St.dev	$4.4680E-15$	**9.3294E − 16**	$1.8026E-15$	$3.3117E-15$	$8.7152E-15$
Griewank (f_{10})	Mean	$4.8762E-12$	**0**	$5.9952E-16$	$8.6898E-12$	$5.9682E-07$
	St.dev	$1.1261E-11$	**0**	$1.5413E-15$	$4.5680E-11$	$2.2794E-06$

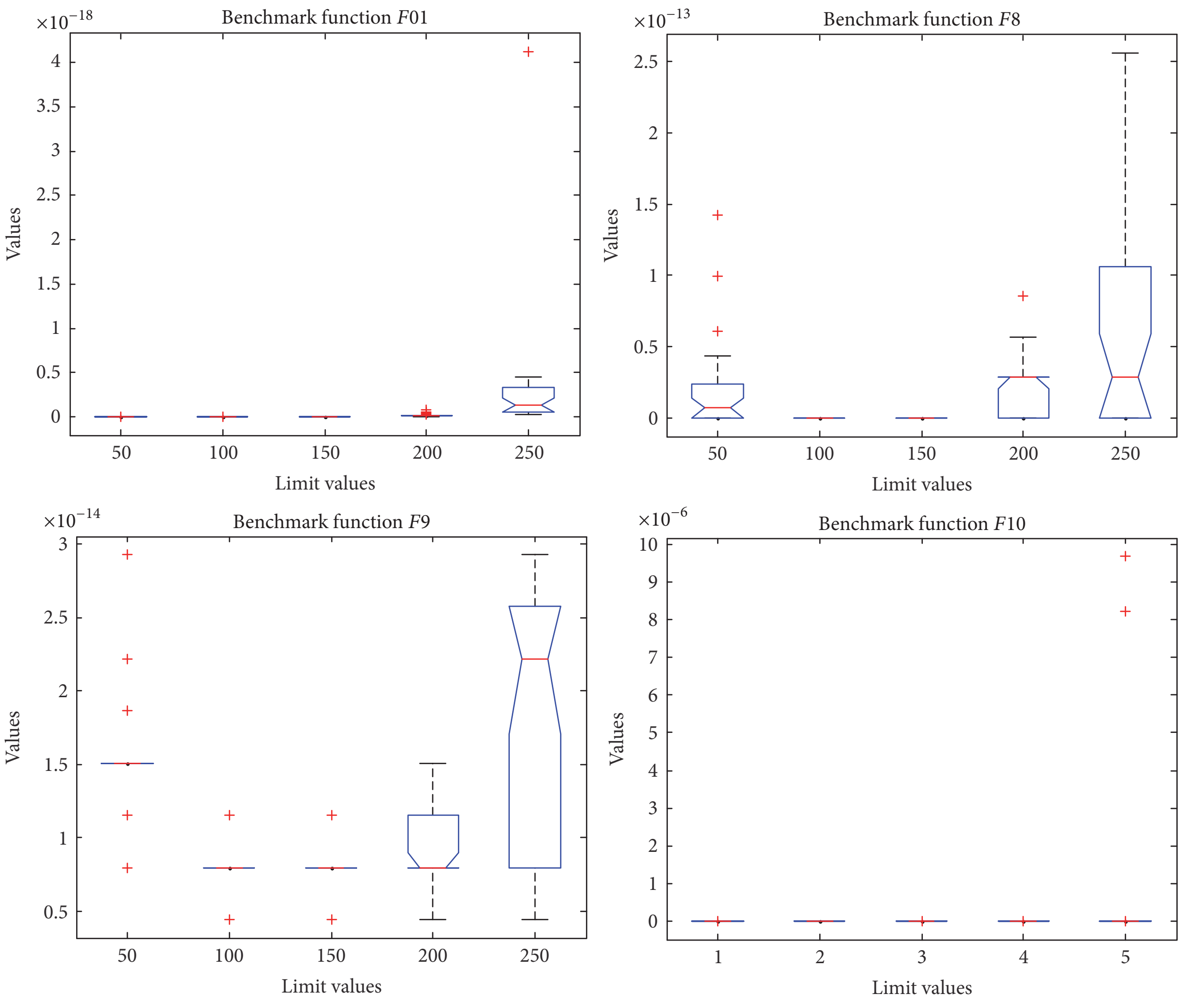

FIGURE 4: Box plots of different limit values for function f_1, f_8, f_9, and f_{10} over 30 independent runs.

Province (Grant no. [2016]1022), China Scholarship Council, Key Laboratory of Renewable Energy Electric-Technology of Hunan Province, Key Laboratory of Efficient & Clean Energy Utilization of Hunan Province, Hubei Key Laboratory of Power System Design and Test for Electrical Vehicle, and Hunan Province 2011 Collaborative Innovation Center of Clean Energy and Smart Grid.

References

[1] S. Mirjalili and A. Lewis, "The Whale Optimization Algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

[2] E. Nabil, "A modified flower pollination algorithm for global optimization," *Expert Systems with Applications*, vol. 57, pp. 192–203, 2016.

[3] W. Zhao and L. Wang, "An effective bacterial foraging optimizer for global optimization," *Information Sciences*, vol. 329, pp. 719–735, 2016.

[4] L. Huang, S. Ding, S. Yu, J. Wang, and K. Lu, "Chaos-enhanced cuckoo search optimization algorithms for global optimization," *Applied Mathematical Modelling. Simulation and Computation for Engineering and Environmental Systems*, vol. 40, no. 5-6, pp. 3860–3875, 2016.

[5] Y. Zhang, G. Cui, J. Wu, W.-T. Pan, and Q. He, "A novel multi-scale cooperative mutation Fruit Fly Optimization Algorithm," *Knowledge-Based Systems*, vol. 114, pp. 24–35, 2016.

[6] A. Yadav, K. Deep, J. H. Kim, and A. K. Nagar, "Gravitational swarm optimizer for global optimization," *Swarm and Evolutionary Computation*, vol. 31, pp. 64–89, 2016.

[7] Z. Y. Li, Z. Li, T. T. Nguyen, and S. M. Chen, "Orthogonal chemical reaction optimization algorithm for global numerical optimization problems," *Expert Systems with Applications*, vol. 42, no. 6, pp. 3242–3252, 2015.

[8] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Tech. Rep. Technical Report-TR06, Erciyes University, Kayseri, Turkey, 2005.

[9] A. n. Yurtkuran and E. Emel, "An adaptive artificial bee colony algorithm for global optimization," *Applied Mathematics and Computation*, vol. 271, pp. 1004–1023, 2015.

[10] J. Liu, H. Zhu, Q. Ma, L. Zhang, and H. Xu, "An Artificial Bee Colony algorithm with guide of global & local optima and asynchronous scaling factors for numerical optimization," *Applied Soft Computing Journal*, vol. 37, pp. 608–618, 2015.

[11] G. Zhu and S. Kwong, “Gbest-guided artificial bee colony algorithm for numerical function optimization,” *Applied Mathematics and Computation*, vol. 217, no. 7, pp. 3166–3173, 2010.

[12] J. Luo, Q. Wang, and X. Xiao, “A modified artificial bee colony algorithm based on converge-onlookers approach for global optimization,” *Applied Mathematics and Computation*, vol. 219, no. 20, pp. 10253–10262, 2013.

[13] W. Gao, S. Liu, and L. Huang, “A global best artificial bee colony algorithm for global optimization,” *Journal of Computational and Applied Mathematics*, vol. 236, no. 11, pp. 2741–2753, 2012.

[14] W.-L. Xiang and M.-Q. An, “An efficient and robust artificial bee colony algorithm for numerical optimization,” *Computers & Operations Research*, vol. 40, no. 5, pp. 1256–1265, 2013.

[15] G. Li, P. Niu, and X. Xiao, “Development and investigation of efficient artificial bee colony algorithm for numerical function optimization,” *Applied Soft Computing Journal*, vol. 12, no. 1, pp. 320–332, 2012.

[16] W. Gao and S. Liu, “Improved artificial bee colony algorithm for global optimization,” *Information Processing Letters*, vol. 111, no. 17, pp. 871–882, 2011.

[17] F. Kang, J. Li, and H. Li, “Artificial bee colony algorithm and pattern search hybridized for global optimization,” *Applied Soft Computing*, vol. 13, no. 4, pp. 1781–1791, 2013.

[18] W.-F. Gao, S.-Y. Liu, and L.-L. Huang, “A novel artificial bee colony algorithm based on modified search equation and orthogonal learning,” *IEEE Transactions on Cybernetics*, vol. 43, no. 3, pp. 1011–1024, 2013.

[19] R. Wang, Y. Ru, and Q. Long, “Improved adaptive and multigroup parallel genetic algorithm based on good-point set,” *Journal of Software*, vol. 4, pp. 348–356, 2009.

[20] W.-f. Gao, S.-y. Liu, and L.-l. Huang, “Enhancing artificial bee colony algorithm using more information-based search equations,” *Information Sciences. An International Journal*, vol. 270, pp. 112–133, 2014.

[21] L. Bao and J. Zeng, “Comparison and analysis of the selection mechanism in the artificial bee colony algorithm,” in *Proceedings of the 2009 Ninth International Conference on Hybrid Intelligent Systems*, pp. 411–416, Shenyang, China, 2009.

[22] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, “GSA: a gravitational search algorithm,” *Information Sciences*, vol. 213, pp. 267–289, 2010.