Journal of Computations & Modelling, vol.6, no.4, 2016, 1-30
ISSN: 1792-7625 (print), 1792-8850 (online)
Scienpress Ltd, 2016

Artificial Bee Colony Algorithm with Mixed Search Equation

Zhigang Wang[1]

Abstract

Artificial bee colony (ABC) algorithm is a relatively new optimization algorithm which has been used to solve many kinds of numerical function optimization problems. It performs well in most cases, however, the solution search equation of artificial bee colony algorithm exists some disadvantages when solving complex functions with high dimensions, such as the convergence speed is not fast enough, easy to fall into local optimum. In order to solve these issues, artificial bee colony algorithm with mixed search equation is proposed. In this algorithm, two different search equations are dynamically used when the employed bees and the onlooker bees search around the neighborhood of the food source. It can keep the diversity of the population and improve the global search ability at the initial generations, and improve the local search ability at a later time. In addition, we use a more robust calculation to determine and compare the quality of alternative solutions. Experiments are conducted on a set of 28 benchmark functions, and the results demonstrate that the new algorithm has fast convergence and high accuracy than several other ABC-based algorithms.

[1] Nanjing Normal University, Taizhou College Taizhou, P.R. China.

Article Info: *Received:* October 30, 2016. *Revised:* November 26, 2016.
Published online: December 30, 2016.

Keywords: Artificial bee colony algorithm; Search equation; Optimization

1 Introduction

In recent years, people have developed several kinds of biological-inspired optimization algorithms, such as ant colony optimization (ACO) [1] inspired by the foraging behavior of ant colonies, particle swarm optimization (PSO) [2] inspired by the social behavior of bird flocking or fish schooling, artificial fish swarm algorithm (AFSA) [3] inspired by the collective movement of the fish and their various social behaviors, and cuckoo search (CS) [4] inspired by cuckoo birds that lay eggs in the nests of other birds. By simulating the foraging behavior of honey bees, Karaboga [5] invented a new evolutionary algorithm called artificial bee colony (ABC) algorithm, a set of experimental results on function optimization [6-9] show that ABC algorithm is competitive with other optimization algorithms such as genetic algorithm (GA), differential evolution (DE), particle swarm optimization (PSO), evolution strategies (ES). Due to its simplicity and ease of implementation, ABC algorithm has been applied to solve many practical optimization problems [10-18] since its invention in 2005.

However, similar to other evolutionary algorithms, the convergence speed of ABC algorithm is not fast enough when handing the unimodal problems. What is more, the ABC algorithm can also easily get trapped in the local optima when solving complex multimodal problems. These weaknesses have restricted the wider applications of the ABC algorithm. In order to accelerate convergence speed and avoid the local optima, a number of variant ABC algorithms have been proposed. To mention a few, Kang et al. [19] used the rotational direction method to complete the exploitation phase and proposed the Rosenbrock ABC algorithm, Alatas [20] introduced the chaotic maps into the initialization and chaotic search into the scout bee phase and proposed the chaotic ABC algorithm, Zhu and Kwong [21] added the global best information in the population into the solution

search equation to improve the exploitation, Gao et al. [22] proposed two ABC-based algorithms that use two update rules of differential evolution called ABC/Best/1 and ABC/Best/2, Gao and Liu [23] used the update rule of the ABC/Best/1 algorithm for employed bees and the update rule of basic ABC algorithm for onlooker bees to reinforce the exploration ability of the method, Imanian et al. [24] changed the update rule of the basic ABC algorithm to increase the convergence speed for solving high dimensional continuous optimization problems inspired by PSO, Wang et al. [25] proposed the MEABC algorithm to improve the local and global search capability of the basic ABC algorithm, Kiran et al. [26] proposed the integration of multiple solution update rules with ABC algorithm to efficiently solve optimization problems with different characteristics, Banharnsakun et al. [27] used the best-so-far selection for onlooker bees to improve the capability of convergence of the ABC algorithm to a global optimum.

In this paper, we propose a novel artificial bee colony algorithm with mixed search equation (ABCMSE for short). In our method, two different solution search equations are mixed. Then, we use a parameter ω to take advantage of them and avoid the shortages of them. In addition, the comparison and the selection of the new solution are changed from a fitness-based comparison to an objective-value based.

The rest of this paper is organized as follows. In Section 2, the ABC algorithm is described. The improved ABC algorithm is proposed in Section 3. In Section 4, experimental results from several basic benchmark functions are shown and compare with the performance of basic ABC and five classical improved ABC algorithms. Finally, Section 5 gives conclusions.

2 Artificial bee colony algorithm

Artificial Bee Colony (ABC) algorithm is a biological-inspired optimization algorithm which simulates the foraging behavior of honey bee colony. The

artificial bee colony consists of three groups of bees: employed bees, onlooker bees and scout bees. In the ABC algorithm, the position of a food source represents a possible solution of the optimization problem, and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. Furthermore, the number of employed bees is equal to the number of onlooker bees, and equal to the number of the solutions in the population.

In the beginning, an initial population containing SN solutions is produced randomly. $x_i = (x_{i1}, x_{i2}, \cdots, x_{iD})$ is the i th solution in the population, and x_i is generated as follows:

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

where $i = 1, 2, \cdots, SN$, $j \in \{1, 2, \cdots, D\}$. $x_{\min}^j$ and $x_{\max}^j$ are the lower and upper bounds for the dimension j, $rand(0,1)$ is a random number in the range $[0,1]$. These solutions are randomly assigned to SN number of employed bees.

After initialization, each employed bee x_i generates a new food source v_i by using search equation as follows:

$$v_{ij} = x_{ij} + \varphi_{ij}(x_{ij} - x_{kj}) \tag{2}$$

where j is a random integer in the range $[1, D]$, k is a random integer in the range $[1, SN]$ and $k \neq i$, φ_{ij} is a random number in the range $[-1,1]$.

Once v_i is obtained, the fitness value is calculated for every v_i by

$$fitness_i = \begin{cases} \dfrac{1}{1+f_i}, & f_i \geq 0 \\ 1+|f_i|, & f_i < 0 \end{cases} \tag{3}$$

where f_i is the objective function value of solution v_i. If the fitness value of v_i is equal to or better than that of x_i, v_i will replace x_i and become a new solution of the population; otherwise x_i is retained

After all employed bees complete their searches, the onlooker bees tend to select attractive food sources from the information shared by employed bees, and

search around the selected food sources. The probability of a food source being selected is proportional to the profitability of the food source. The probability of a food source being selected by an onlooker bee is calculated by

$$P_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \tag{4}$$

where fit_i is the fitness value of the solution x_i. After the onlooker selects her food source x_i, she produces a modification on x_i by using Eq. (2). If the modified food source has a better or equal nectar amount than x_i, the modified food source will replace x_i and become a new solution in the population.

In ABC algorithm, if a food source x_i cannot improve the fitness value through a predetermined number of $\lim it$, the food source will be abandoned, and the corresponding employed bee becomes a scout bee. The scout bee produces a food source by using Eq. (1).

3 Artificial bee colony algorithm with mixed search equation

According to the solution search equation of ABC algorithm described by Eq. (2), the new candidate solution is generated by moving the old solution towards (or away from) another solution selected randomly from the population. Therefore, the solution search equation described by Eq. (2) is good at exploration but poor at exploitation and diversity.

In order to improve exploitation and increase diversity in the population, two solution search equations are given as follows:

$$v_{ij} = x_{rj} + \varphi_{ij}(x_{r_1 j} - x_{r_2 j}) \tag{5}$$

$$v_{ij} = x_{best,j} + \varphi_{ij}(x_{r_1 j} - x_{r_2 j}) \tag{6}$$

In Eq. (5) [28], r, r_1 and r_2 are mutually exclusive integers randomly

generated within the range $[1, SN]$, which are also different from the index i. In Eq. (6) [22], x_{best} is the global best solution in the population.

We know that both exploration and exploitation are important for the population-based optimization algorithms. In the optimization algorithms, the exploration refers to the ability to investigate the various unknown regions in the solution space to discover the global optimum, and the exploitation refers to the ability to apply the knowledge of the previous good solutions to find better solutions. However, the exploration and exploitation contradict with each other. To achieve good optimization performance, the two abilities should be well balanced. According to the Eq. (5), the new candidate solution is generated around the solution which is selected randomly from the population. Therefore, the solution search dominated by Eq. (5) is random enough for exploration, and it can also effectively maintain population diversity. According to the Eq. (6), the new candidate solution is generated around the global best solution. The global best solution are used to direct the movement of the current population, therefore Eq. (6) can increase the exploitation of ABC algorithm.

From the above explanation, it is clear that Eq. (6) has a good capacity of the exploitation. Unfortunately, Eq. (6) can reduce exploration of ABC algorithm. If all bees produce new solution using Eq. (6), the algorithm can easily get trapped in the local optima when solving complex multimodal problems. However, the solution search equation described by Eq. (5) is good at exploration but poor at exploitation results in a slow convergence. In order to balance the exploration of the solution search equation (5) and the exploitation of the solution search equation (6), we combine the two solution search equation and get a novel solution search equation as follows:

$$v_{ij} = \omega x_{rj} + (1-\omega) x_{best,j} + \varphi_{ij}(x_{r_1 j} - x_{r_2 j}) \tag{7}$$

where ω is a real number in the range $[0,1]$.

Note that the parameter ω plays an important role in balancing the exploration and exploitation. When ω takes 0, Eq. (7) is identical to Eq. (6);

when ω takes 1, Eq. (7) is identical to Eq. (5). When ω decreases from 1 to 0, the exploration of Eq. (7) will also decrease correspondingly, but the exploitation of Eq. (7) will increase. Therefore, in the early stage, ω should be larger to maintain population diversity and increase exploration, it can prevent bees from falling into the local minimum. Moreover, the speed reduction of ω should be faster to accelerate the convergence speed. At a later stage, the speed reduction of ω should be slow down to search for local optimum effectively. The parameter ω is given as follows [29]:

$$\omega = \exp(-30 \cdot (FE / Max.FE)^S) \tag{8}$$

where FE is the number of function evaluation, $Max.FE$ is the maximum number of function evaluations, S is a positive integer greater than 0.

In ABC algorithm, the comparison of the new solution and the old solution is done by the fitness value. If the fitness value of the new solution is better than the fitness of the old solution, we select the new one and ignore the old solution. However, this method also has shortcomings [27]. In this paper, we directly use the objective function value for comparison and selection of the better solution.

1: Set the population size SN and the maximum number of functions evaluations $Max.FE$
2: Initialize the solutions $x_i (i = 1,2,\cdots,SN)$, and evaluate them
3: While(stopping criterion is not met, namely $FE < Max.FE$)
4: Generate candidate solution v_i for employed bees by Eq.(7) and evaluate them
5: Determine the solution of employed bees by greedy selection
6: Calculate the selection probability values P_i for solutions x_i by Eq.(4)
7: Generate candidate solution v_i for onlooker bees by Eq.(7) and evaluate them
8: Determine the solution of onlooker bees by greedy selection
9: if there exist an abandoned solution for the scout, replace it with a randomly produced solution x_i by Eq.(1)
10: Memorize the best solution achieved so far
11:End while ($FE = Max.FE$)

Figure 1: The main steps of ABCMSE

The main steps of ABCMSE are shown in Fig.1.

4 Experiments

4.1 Benchmark functions and parameter settings

To analyze and compare the performance and accuracy of the ABCMSE, a set of 26 scalable benchmark functions of dimensions $D=30$, 60 or 100 and a set of two functions of higher dimensions $D=100$, 200 or 300 are used in the experiments and are listed in Table 1-2. These functions have different properties such as unimodality, multimodality, separable and non-separable. These properties of the functions are given in column C of Table 1-2. In column C of Table 1-2, U shows that the function is unimodal, M shows that the function is multimodal, S shows that the function is separable and N shows that the function is non-separable. These unimodal functions are used to test the exploitation ability of the algorithm, and multimodal functions are used to test the exploration ability of the algorithm.

In order to testify the efficiency of ABCMSE, the experiment results are compared with the basic ABC algorithm. In all simulations, as the number of optimization parameters increases, we set the maximum number of function evaluations to be 150,000, 300,000 and 500,000 for each function, respectively. The population size of ABC and ABCMSE is 40 (i.e., $SN=20$), $\lim it$ is $SN*D$. In ABCMSE, $S=1$. All results reported in this section are obtained based on 30 independent runs.

4.2 Experimental results

The experimental results are shown in Table 3-7 in terms of the best, worst, mean and standard deviation (std) of the solutions obtained in the 30 independent runs by each algorithm. In order to compare the convergence rate of the ABC and

ABCMSE, we draw the convergence curves of eight tested functions in Fig.2.

Table 1: Benchmark functions used in experiments

Function name	Function	C	Search range
Sphere	$f_1(x)=\sum_{i=1}^{D}x_i^2$	US	[-100,100]
Elliptic	$f_2(x)=\sum_{i=1}^{D}(10^6)^{(i-1)/(D-1)}x_i^2$	UN	[-100,100]
SumSquares	$f_3(x)=\sum_{i=1}^{D}ix_i^2$	US	[-10,10]
SumPower	$f_4(x)=\sum_{i=1}^{D}\lvert x_i\rvert^{(i+1)}$	US	[-10,10]
Schwefel 2.22	$f_5(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	UN	[-10,10]
Schwefel 2.21	$f_6(x)=\max_i\{\lvert x_i\rvert,1\le i\le D\}$	UN	[-100,100]
Step	$f_7(x)=\sum_{i=1}^{D}\left(\lfloor x_i+0.5\rfloor\right)^2$	US	[-100,100]
Quartic	$f_8(x)=\sum_{i=1}^{D}ix_i^4$	US	[-1.28,1.28]
QuarticWN	$f_9(x)=\sum_{i=1}^{D}ix_i^4+\text{random}[0,1)$	US	[-1.28,1.28]
Rosenbrock	$f_{10}(x)=\sum_{i=1}^{D-1}[100(x_{i+1}-x_i^2)^2+(1-x_i)^2]$	UN	[-10,10]
Rastrigin	$f_{11}(x)=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$	MS	[-5.12,5.12]
Non-continuous rastrigin	$f_{12}(x)=\sum_{i=1}^{D}(y_i^2-10\cos(2\pi y_i)+10)$ $y_i=\begin{cases}x_i & \lvert x_i\rvert<\frac{1}{2}\\ \frac{round(2x_i)}{2} & \lvert x_i\rvert\ge\frac{1}{2}\end{cases}$	MS	[-5.12,5.12]
Griewank	$f_{13}(x)=\frac{1}{4000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})+1$	MN	[-600,600]
Schwefel2.26	$f_{14}(x)=418.98288727243369*D-\sum_{i=1}^{D}x_i\cdot\sin(\sqrt{\lvert x_i\rvert})$	MN	[-500,500]

Ackley	$f_{15}(x)=20+e-20\exp\{-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\}-\exp\{\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\}$	MN	[-32,32]

Table 2: Benchmark functions used in experiments

Function name	Function	C	Search range
Penalized1	$f_{16}(x)=\frac{\pi}{D}\left\{10\sin^2(\pi y_1)+\sum_{i=1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_D-1)^2\right\}+\sum_{i=1}^{D}u(x_i,10,100,4)$ $y_i=1+\frac{1}{4}(x_i+1),\ u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m & x_i>a\\ 0 & -a\le x_i\le a\\ k(-x_i-a)^m & x_i<-a\end{cases}$	MN	[-50,50]
Penalized2	$f_{17}(x)=\frac{1}{10}\left\{\sin^2(3\pi x_1)+\sum_{i=1}^{D-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]+(x_D-1)^2[1+\sin^2(2\pi x_D)]\right\}+\sum_{i=1}^{D}u(x_i,5,100,4)$	MN	[-50,50]
Alpine	$f_{18}(x)=\sum_{i=1}^{D}\lvert x_i\cdot\sin(x_i)+0.1\cdot x_i\rvert$	MS	[-10,10]
Levy	$f_{19}(x)=\sum_{i=1}^{D-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]+\sin^2(3\pi x_1)+\lvert x_D-1\rvert[1+\sin^2(3\pi x_D)]$	MN	[-10,10]
Weierstrass	$f_{20}(x)=\sum_{i=1}^{D}\left(\sum_{k=0}^{k_{\max}}[a^k\cos(2\pi b^k(x_i+0.5))]\right)-D\cdot\sum_{k=0}^{k_{\max}}[a^k\cos(2\pi b^k 0.5)]\quad a=0.5,b=3,k_{\max}=20$	MN	[-0.5, 0.5]
Schaffer	$f_{21}(x)=0.5+\frac{\sin^2\left(\sqrt{\sum_{i=1}^{D}x_i^2}\right)-0.5}{\left(1+0.001\sum_{i=1}^{D}x_i^2\right)^2}$	MN	[-100,100]
Hammelblau	$f_{22}(x)=\frac{1}{D}\sum_{i=1}^{D}(x_i^4-16x_i^2+5x_i)$	MS	[-5,5]
Michalewicz	$f_{23}(x)=-\sum_{i=1}^{D}\sin(x_i)\sin^{20}(\frac{ix_i^2}{\pi})$	MS	$[0,\pi]$
Shifted sphere	$f_{24}(x)=\sum_{i=1}^{D}z_i^2\quad z=x-o$	US	[-100,100]
Shifted rastrigin	$f_{25}(x)=\sum_{i=1}^{D}(z_i^2-10\cos(2\pi z_i)+10)\quad z=x-o$	MS	[-5.12,5.12]
Shifted griewank	$f_{26}(x)=\frac{1}{4000}\sum_{i=1}^{D}z_i^2-\prod_{i=1}^{D}\cos(\frac{z_i}{\sqrt{i}})+1\quad z=x-o$	MN	[-600,600]

Shifted ackley	$f_{27}(x)=20+e-20\exp\{-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}z_i^2}\}-\exp\{\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)\}\ \ z=x-o$	MN	[-32,32]
Shifted alpine	$f_{28}(x)=\sum_{i=1}^{D}\lvert z_i\cdot\sin(z_i)+0.1\cdot z_i\rvert \quad z=x-o$	MS	[-10,10]

Table 3: Performance comparisons of ABC and ABCMSE

Function	D	Algorithm	Best	Worst	Mean	Std
f_1	30	ABC	2.81E-16	7.34E-16	5.19E-16	8.21E-17
		ABCMSE	2.02E-122	2.36E-115	2.02E-116	5.35E-116
	60	ABC	9.71E-16	1.43E-15	1.22E-15	1.31E-16
		ABCMSE	1.39E-119	7.58E-113	3.31E-114	1.40E-113
	100	ABC	1.66E-15	2.75E-15	2.18E-15	2.53E-16
		ABCMSE	8.78E-119	6.56E-112	5.74E-113	1.61E-112
f_2	30	ABC	2.96E-16	7.29E-16	5.06E-16	1.05E-16
		ABCMSE	9.87E-119	3.42E-109	1.22E-110	6.13E-110
	60	ABC	7.61E-16	1.40E-15	1.14E-15	1.67E-16
		ABCMSE	4.55E-116	1.56E-106	5.24E-108	2.81E-107
	100	ABC	1.63E-15	2.51E-15	2.04E-15	2.10E-16
		ABCMSE	5.83E-114	1.78E-107	7.89E-109	3.28E-108
f_3	30	ABC	3.06E-16	6.79E-16	5.03E-16	9.05E-17
		ABCMSE	4.99E-123	3.12E-116	2.53E-117	7.29E-117
	60	ABC	8.35E-16	1.42E-15	1.19E-15	1.63E-16
		ABCMSE	2.54E-120	2.31E-114	2.13E-115	5.36E-115
	100	ABC	1.79E-15	2.53E-15	2.22E-15	2.24E-16
		ABCMSE	2.01E-118	4.84E-111	1.63E-112	8.68E-112
f_4	30	ABC	7.83E-18	4.57E-17	2.60E-17	1.04E-17
		ABCMSE	2.11E-136	1.40E-105	4.67E-107	2.51E-106
	60	ABC	1.13E-17	7.71E-17	4.58E-17	1.56E-17
		ABCMSE	1.32E-120	3.41E-97	1.75E-98	6.72E-98
	100	ABC	1.58E-17	9.55E-17	6.43E-17	1.83E-17
		ABCMSE	1.97E-113	3.28E-44	1.09E-45	5.88E-45
f_5	30	ABC	9.55E-16	1.58E-15	1.31E-15	1.36E-16
		ABCMSE	2.01E-64	1.63E-61	2.33E-62	3.14E-62
	60	ABC	2.53E-15	3.20E-15	2.84E-15	1.86E-16
		ABCMSE	4.07E-60	2.94E-58	4.37E-59	5.59E-59
	100	ABC	4.49E-15	5.64E-15	5.02E-15	2.75E-16
		ABCMSE	3.81E-58	1.04E-56	2.22E-57	2.22E-57
f_6	30	ABC	2.42E-01	1.99E+00	7.82E-01	3.50E-01
		ABCMSE	6.07E-03	2.51E-02	1.16E-02	4.83E-03

	60	ABC	4.02E+00	1.44E+01	1.10E+01	2.25E+00
		ABCMSE	1.22E+00	1.96E+00	1.51E+00	1.86E-01
	100	ABC	2.19E+01	3.18E+01	2.73E+01	2.38E+00
		ABCMSE	7.08E+00	9.68E+00	8.54E+00	6.96E-01

Table 4: Performance comparisons of ABC and ABCMSE

Function	D	Algorithm	Best	Worst	Mean	Std
f_7	30	ABC	0	0	0	0
		ABCMSE	0	0	0	0
	60	ABC	0	0	0	0
		ABCMSE	0	0	0	0
	100	ABC	0	0	0	0
		ABCMSE	0	0	0	0
f_8	30	ABC	1.03E-16	2.96E-16	2.18E-16	5.31E-17
		ABCMSE	9.68E-238	4.04E-221	1.35E-222	0
	60	ABC	3.22E-16	6.56E-16	4.87E-16	8.38E-17
		ABCMSE	6.45E-233	2.31E-221	9.05E-223	0
	100	ABC	7.58E-16	1.20E-15	9.62E-16	1.34E-16
		ABCMSE	4.78E-228	3.83E-218	1.28E-219	0
f_9	30	ABC	3.43E-02	6.02E-02	4.90E-02	8.52E-03
		ABCMSE	4.49E-03	2.95E-02	1.25E-02	4.77E-03
	60	ABC	8.09E-02	1.23E-01	9.68E-02	1.22E-02
		ABCMSE	1.56E-02	4.49E-02	3.40E-02	6.62E-03
	100	ABC	1.40E-01	1.90E-01	1.58E-01	1.55E-02
		ABCMSE	4.24E-02	7.04E-02	5.97E-02	6.49E-03
f_{10}	30	ABC	2.40E-03	1.32E-01	4.07E-02	3.88E-02
		ABCMSE	1.36E-04	7.44E+01	9.75E+00	2.17E+01
	60	ABC	7.59E-04	4.93E-01	9.36E-02	1.42E-01
		ABCMSE	3.17E-04	9.37E+01	1.74E+01	3.16E+01
	100	ABC	2.21E-03	1.11E+00	2.12E-01	3.17E-01
		ABCMSE	4.97E-03	1.59E+02	5.11E+01	4.88E+01
f_{11}	30	ABC	0	0	0	0
		ABCMSE	0	0	0	0
	60	ABC	0	0	0	0
		ABCMSE	0	0	0	0
	100	ABC	0	0	0	0
		ABCMSE	0	0	0	0
f_{12}	30	ABC	0	0	0	0

		ABCMSE	0	0	0	0
	60	ABC	0	0	0	0
		ABCMSE	0	0	0	0
	100	ABC	0	0	0	0
		ABCMSE	0	0	0	0

Table 5: Performance comparisons of ABC and ABCMSE

Function	D	Algorithm	Best	Worst	Mean	Std
	30	ABC	0	1.78E-09	6.46E-11	3.25E-10
		ABCMSE	0	4.70E-12	1.57E-13	8.43E-13
f_{13}	60	ABC	0	1.44E-15	2.26E-16	2.99E-16
		ABCMSE	0	1.09E-09	3.68E-11	1.97E-10
	100	ABC	0	9.99E-16	3.94E-16	2.92E-16
		ABCMSE	0	5.57E-14	1.86E-15	1.00E-14
	30	ABC	3.64E-12	2.31E-09	1.08E-10	4.95E-10
		ABCMSE	1.82E-12	3.64E-12	2.43E-12	8.57E-13
f_{14}	60	ABC	2.91E-11	1.90E-04	6.32E-06	3.40E-05
		ABCMSE	2.91E-11	2.91E-11	2.91E-11	0
	100	ABC	9.46E-11	1.16E-10	9.99E-11	5.62E-12
		ABCMSE	1.02E-10	1.31E-10	1.03E-10	5.22E-12
	30	ABC	3.26E-14	4.32E-14	3.72E-14	3.20E-15
		ABCMSE	1.83E-14	2.55E-14	2.23E-14	2.49E-15
f_{15}	60	ABC	7.16E-14	8.59E-14	7.84E-14	4.51E-15
		ABCMSE	4.68E-14	7.52E-14	5.80E-14	5.36E-15
	100	ABC	1.25E-13	1.57E-13	1.41E-13	7.95E-15
		ABCMSE	9.65E-14	1.28E-13	1.11E-13	7.42E-15
	30	ABC	3.29E-16	5.44E-16	4.94E-16	4.72E-17
		ABCMSE	1.57E-32	1.57E-32	1.57E-32	2.74E-48
f_{16}	60	ABC	9.44E-16	1.43E-15	1.19E-15	1.36E-16
		ABCMSE	7.85E-33	7.85E-33	7.85E-33	1.37E-48
	100	ABC	1.84E-15	2.53E-15	2.13E-15	2.11E-16
		ABCMSE	4.71E-33	4.71E-33	4.71E-33	2.74E-48
	30	ABC	3.08E-16	5.52E-16	4.62E-16	8.08E-17
		ABCMSE	1.35E-32	1.35E-32	1.35E-32	5.47E-48
f_{17}	60	ABC	9.12E-16	1.44E-15	1.19E-15	1.42E-16
		ABCMSE	1.35E-32	1.35E-32	1.35E-32	5.47E-48
	100	ABC	1.85E-15	2.55E-15	2.30E-15	1.85E-16
		ABCMSE	1.35E-32	1.35E-32	1.35E-32	5.47E-48

f_{18}	30	ABC	2.18E-13	1.98E-08	1.99E-09	4.89E-09
		ABCMSE	9.17E-69	8.24E-64	8.99E-65	2.15E-64
	60	ABC	3.81E-10	2.07E-06	1.26E-07	3.75E-07
		ABCMSE	2.70E-64	1.71E-60	2.37E-61	4.25E-61
	100	ABC	5.88E-09	8.08E-05	7.25E-06	1.63E-05
		ABCMSE	2.19E-60	6.11E-16	4.07E-17	1.52E-16

Table 6: Performance comparisons of ABC and ABCMSE

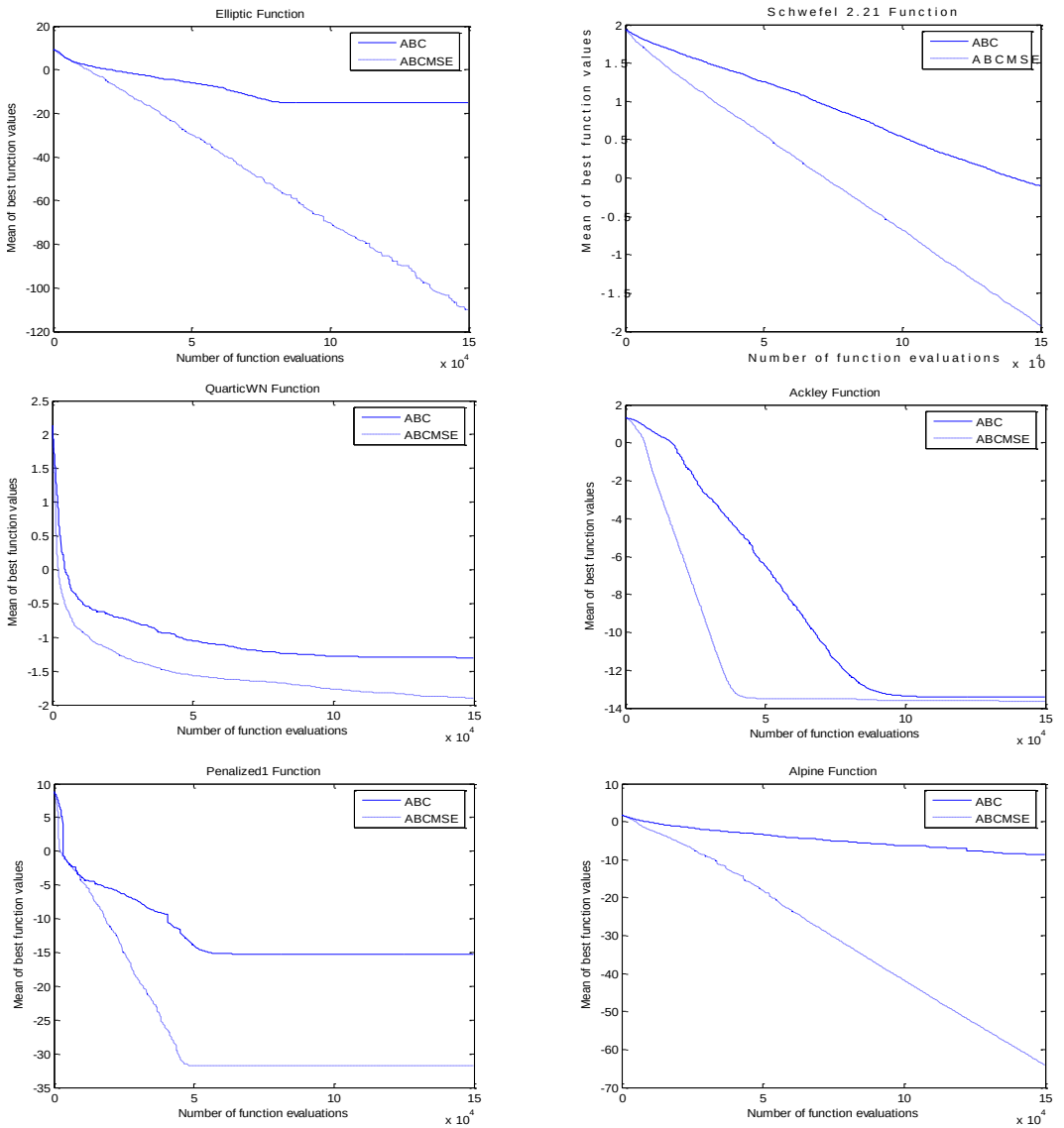
Function	D	Algorithm	Best	Worst	Mean	Std
f_{19}	30	ABC	2.79E-16	5.33E-16	4.24E-16	8.41E-17
		ABCMSE	1.35E-31	1.35E-31	1.35E-31	0
	60	ABC	7.73E-16	1.38E-15	1.08E-15	1.25E-16
		ABCMSE	1.35E-31	1.35E-31	1.35E-31	0
	100	ABC	1.21E-15	2.30E-15	1.98E-15	2.04E-16
		ABCMSE	1.35E-31	1.35E-31	1.35E-31	0
f_{20}	30	ABC	0	0	0	0
		ABCMSE	0	0	0	0
	60	ABC	0	2.84E-14	1.52E-14	1.03E-14
		ABCMSE	0	1.42E-14	2.84E-15	5.68E-15
	100	ABC	5.68E-14	1.14E-13	8.24E-14	2.36E-14
		ABCMSE	0	5.68E-14	3.13E-14	1.34E-14
f_{21}	30	ABC	2.28E-01	3.73E-01	3.30E-01	3.64E-02
		ABCMSE	7.82E-02	3.12E-01	2.13E-01	5.47E-02
	60	ABC	4.52E-01	4.90E-01	4.77E-01	8.57E-03
		ABCMSE	3.12E-01	4.72E-01	4.37E-01	3.31E-02
	100	4.95E-01	4.99E-01	4.97E-01	8.93E-04	4.95E-01
		4.80E-01	4.96E-01	4.93E-01	3.58E-03	4.80E-01
f_{22}	100	ABC	-7.83E+01	-7.83E+01	-7.83E+01	1.02E-09
		ABCMSE	-7.83E+01	-7.83E+01	-7.83E+01	2.52E-14
	200	ABC	-7.83E+01	-7.83E+01	-7.83E+01	3.84E-10
		ABCMSE	-7.83E+01	-7.83E+01	-7.83E+01	3.53E-07
	300	ABC	-7.83E+01	-7.83E+01	-7.83E+01	3.29E-11
		ABCMSE	-7.83E+01	-7.83E+01	-7.83E+01	2.51E-12
f_{23}	100	ABC	-9.67E+01	-9.55E+01	-9.60E+01	3.13E-01
		ABCMSE	-9.95E+01	-9.90E+01	-9.94E+01	1.07E-01
	200	ABC	-1.94E+02	-1.91E+02	-1.92E+02	4.17E-01
		ABCMSE	-1.99E+02	-1.99E+02	-1.99E+02	1.32E-01
	300	ABC	-2.89E+02	-2.87E+02	-2.88E+02	5.38E-01

		ABCMSE	-2.99E+02	-2.99E+02	-2.99E+02	1.08E-01
f_{24}	30	ABC	3.33E-16	7.60E-16	5.13E-16	8.14E-17
		ABCMSE	0	0	0	0
	60	ABC	9.82E-16	1.57E-15	1.24E-15	1.58E-16
		ABCMSE	0	0	0	0
	100	ABC	1.41E-15	2.75E-15	2.28E-15	2.50E-16
		ABCMSE	0	0	0	0

Table 7: Performance comparisons of ABC and ABCMSE

Function	D	Algorithm	Best	Worst	Mean	Std
f_{25}	30	ABC	0	0	0	0
		ABCMSE	0	0	0	0
	60	ABC	0	0	0	0
		ABCMSE	0	0	0	0
	100	ABC	0	0	0	0
		ABCMSE	0	0	0	0
f_{26}	30	ABC	0	3.63E-07	1.21E-08	6.51E-08
		ABCMSE	0	1.72E-02	1.40E-03	3.85E-03
	60	ABC	0	8.88E-16	2.41E-16	2.33E-16
		ABCMSE	0	2.21E-02	1.89E-03	4.83E-03
	100	ABC	0	1.89E-15	5.33E-16	4.37E-16
		ABCMSE	0	9.86E-03	3.29E-04	1.77E-03
f_{27}	30	ABC	2.90E-14	4.32E-14	3.54E-14	4.04E-15
		ABCMSE	1.84E-14	2.90E-14	2.21E-14	2.89E-15
	60	ABC	6.45E-14	8.94E-14	7.98E-14	6.81E-15
		ABCMSE	4.68E-14	6.45E-14	5.79E-14	4.83E-15
	100	ABC	1.25E-13	1.60E-13	1.42E-13	8.30E-15
		ABCMSE	8.94E-14	1.21E-13	1.08E-13	7.32E-15
f_{28}	30	ABC	4.36E-12	6.67E-07	5.14E-08	1.29E-07
		ABCMSE	4.86E-17	1.08E-14	1.83E-15	3.06E-15
	60	ABC	5.27E-09	9.64E-06	3.17E-06	1.02E-05
		ABCMSE	1.42E-16	2.28E-14	3.41E-15	5.46E-15
	100	ABC	1.05E-06	4.61E-04	8.12E-05	1.20E-04
		ABCMSE	2.69E-16	3.45E-14	4.23E-15	7.90E-15

As is shown in Table 3-7, the compared methods can find optimal solutions on functions f_7, f_{11}, f_{12} and f_{25} with $D=30$, 60 and 100. ABCMSE offers the higher accuracy on almost all the functions except functions f_{10}, f_{13} and f_{26} with $D=60,100$. As can be seen from Fig.2, Compare with ABC, ABCMSE has faster convergence speed and higher calculate accuracy. To sum up, the experimental results and Convergence curves clearly indicate that the ABCMSE is superior to ABC on almost all the functions. The superiority in terms of search ability and efficiency of ABCMSE should be attributed to an appropriate balance between exploration and exploitation.



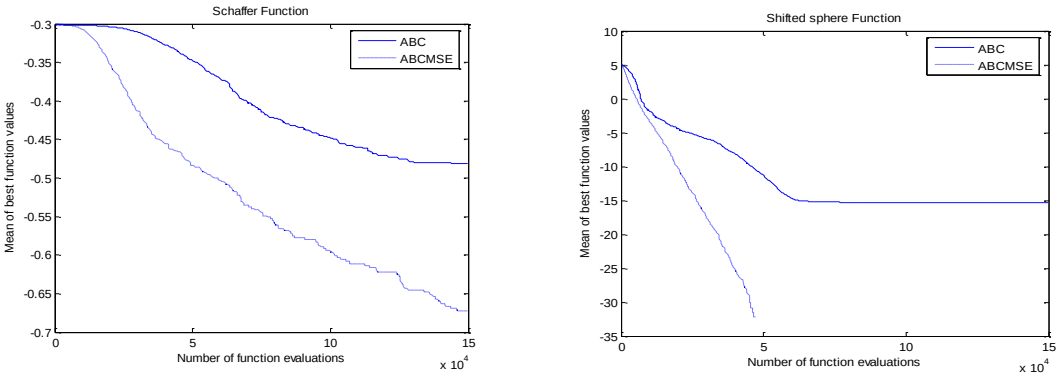


Figure 2: Convergence curves on the 8 test functions with $D = 30$

Table 8: Performance comparisons of GABC, MABC and ABCMSE on the 30 and 100 (f_{22}, f_{23}) dimensional functions

Function	GABC		MABC		ABCMSE	
	Mean	Std	Mean	Std	Mean	Std
f_1	4.62E-16	7.12E-17	9.43E-32	6.67E-32	2.02E-116	5.35E-116
f_2	3.62E-16	7.88E-17	3.66E-28	5.96E-28	1.22E-110	6.13E-110
f_3	4.55E-16	7.00E-17	2.10E-32	1.56E-32	2.53E-117	7.29E-117
f_4	1.64E-17	8.07E-18	2.70E-69	5.38E-69	4.67E-107	2.51E-106
f_5	1.35E-15	1.36E-16	2.40E-17	9.02E-18	2.33E-62	3.14E-62
f_6	2.18E-01	4.01E-02	1.02E+01	1.49E+00	1.16E-02	4.83E-03
f_7	0	0	0	0	0	0
f_8	1.21E-16	3.99E-17	1.45E-67	2.28E-67	1.35E-222	0
f_9	2.03E-02	5.74E-03	3.71E-02	8.53E-03	1.25E-02	4.77E-03
f_{10}	3.21E-01	8.21E-01	6.11E-01	4.55E-01	9.75E+00	2.17E+01
f_{11}	0	0	0	0	0	0
f_{12}	0	0	0	0	0	0
f_{13}	3.70E-17	5.32E-17	0	0	1.57E-13	8.43E-13
f_{14}	9.42E-02	5.16E-01	1.21E-13	4.53E-13	2.43E-12	8.57E-13
f_{15}	3.20E-14	3.36E-15	4.13E-14	2.17E-15	2.23E-14	2.49E-15
f_{16}	4.12E-16	8.36E-17	1.90E-32	3.70E-33	1.57E-32	2.74E-48
f_{17}	4.01E-16	8.19E-17	2.23E-31	1.46E-31	1.35E-32	5.47E-48
f_{18}	3.41E-09	1.13E-08	1.58E-16	2.48E-16	8.99E-65	2.15E-64

f_{19}	3.28E-16	5.03E-17	1.48E-31	2.30E-32	1.35E-31	0
f_{20}	0	0	0	0	0	0
f_{21}	2.66E-01	4.39E-02	2.95E-01	3.17E-02	2.13E-01	5.47E-02
f_{22}	-7.83E+01	2.94E-14	-7.83E+01	2.06E-06	-7.83E+01	2.52E-14
f_{23}	-9.94E+01	4.18E-02	-9.07E+01	5.03E-01	-9.94E+01	1.07E-01
f_{24}	4.38E-16	8.43E-17	0	0	0	0
f_{25}	0	0	0	0	0	0
f_{26}	3.33E-17	5.17E-17	0	0	1.40E-03	3.85E-03
f_{27}	3.20E-14	2.80E-15	4.92E-14	5.31E-15	2.21E-14	2.89E-15
f_{28}	6.65E-08	2.39E-07	1.38E-16	8.11E-17	1.83E-15	3.06E-15

Table 9: Performance comparisons of ABCBest1, ABCBset2 and ABCMSE on the 30 and 100 (f_{22}, f_{23}) dimensional functions

Function	ABCBest1		ABCBest2		ABCMSE	
	Mean	Std	Mean	Std	Mean	Std
f_1	3.11E-47	3.44E-47	5.96E-35	3.61E-35	2.02E-116	5.35E-116
f_2	5.35E-44	4.91E-44	1.70E-28	2.35E-28	1.22E-110	6.13E-110
f_3	6.50E-48	6.04E-48	5.55E-36	3.36E-36	2.53E-117	7.29E-117
f_4	1.77E-86	7.02E-86	3.00E-46	1.07E-45	4.67E-107	2.51E-106
f_5	2.10E-25	9.08E-26	1.36E-18	4.27E-19	2.33E-62	3.14E-62
f_6	2.18E+00	3.27E-01	3.55E+00	4.79E-01	1.16E-02	4.83E-03
f_7	0	0	0	0	0	0
f_8	2.63E-97	3.75E-97	3.10E-76	2.89E-76	1.35E-222	0
f_9	2.06E-02	4.75E-03	2.53E-02	4.67E-03	1.25E-02	4.77E-03
f_{10}	1.49E+01	2.87E+0	5.45E+00	8.40E+00	9.75E+00	2.17E+01
f_{11}	0	0	0	0	0	0
f_{12}	0	0	0	0	0	0
f_{13}	0	0	1.81E-08	6.29E-08	1.57E-13	8.43E-13
f_{14}	1.33E-12	8.18E-13	1.76E-12	3.32E-13	2.43E-12	8.57E-13
f_{15}	3.01E-14	2.91E-15	3.07E-14	3.43E-15	2.23E-14	2.49E-15
f_{16}	1.57E-32	5.57E-48	1.57E-32	5.57E-48	1.57E-32	2.74E-48
f_{17}	1.35E-32	5.57E-48	1.35E-32	5.57E-48	1.35E-32	5.47E-48

f_{18}	3.00E-16	8.99E-16	3.23E-14	9.14E-14	8.99E-65	2.15E-64
f_{19}	1.35E-31	6.68E-47	1.35E-31	6.68E-47	1.35E-31	0
f_{20}	4.74E-16	1.80E-15	9.47E-16	2.46E-15	0	0
f_{21}	2.39E-01	6.13E-02	2.81E-01	3.92E-02	2.13E-01	5.47E-02
f_{22}	-7.83E+01	6.68E-12	-7.83E+01	4.86E-09	-7.83E+01	2.52E-14
f_{23}	-9.57E+01	3.89E-01	-8.94E+01	4.72E-01	-9.94E+01	1.07E-01
f_{24}	0	0	0	0	0	0
f_{25}	0	0	0	0	0	0
f_{26}	8.81E-16	3.38E-15	1.46E-07	7.78E-07	1.40E-03	3.85E-03
f_{27}	2.89E-14	2.59E-15	3.01E-14	3.70E-15	2.21E-14	2.89E-15
f_{28}	1.50E-16	2.48E-16	1.33E-13	4.89E-13	1.83E-15	3.06E-15

Table 10: Performance comparisons of ABCVSS and ABCMSE on the 30 and 100 (f_{22}, f_{23}) dimensional functions

Function	ABCVSS		ABCMSE	
	Mean	Std	Mean	Std
f_1	1.53E-81	8.37E-81	2.02E-116	5.35E-116
f_2	4.82E-82	2.63E-81	1.22E-110	6.13E-110
f_3	3.19E-89	1.48E-88	2.53E-117	7.29E-117
f_4	5.55E-115	3.04E-11	4.67E-107	2.51E-106
f_5	7.89E-43	4.32E-42	2.33E-62	3.14E-62
f_6	4.08E-02	2.20E-02	1.16E-02	4.83E-03
f_7	0	0	0	0
f_8	3.25E-154	1.78E-15	1.35E-222	0
f_9	1.81E-02	5.27E-03	1.25E-02	4.77E-03
f_{10}	3.87E-01	1.54E+00	9.75E+00	2.17E+01
f_{11}	0	0	0	0
f_{12}	0	0	0	0
f_{13}	0	0	1.57E-13	8.43E-13
f_{14}	4.85E-13	8.18E-13	2.43E-12	8.57E-13
f_{15}	2.45E-14	4.00E-15	2.23E-14	2.49E-15
f_{16}	1.57E-32	5.57E-48	1.57E-32	2.74E-48

f_{17}	1.35E-32	5.57E-48	1.35E-32	5.47E-48
f_{18}	3.66E-44	1.93E-43	8.99E-65	2.15E-64
f_{19}	1.35E-31	6.68E-47	1.35E-31	0
f_{20}	0	0	0	0
f_{21}	2.84E-01	5.69E-02	2.13E-01	5.47E-02
f_{22}	-7.83E+01	3.02E-10	-7.83E+01	2.52E-14
f_{23}	-9.94E+01	8.84E-02	-9.94E+01	1.07E-01
f_{24}	0	0	0	0
f_{25}	0	0	0	0
f_{26}	0	0	1.40E-03	3.85E-03
f_{27}	2.53E-14	4.55E-15	2.21E-14	2.89E-15
f_{28}	7.49E-17	1.48E-17	1.83E-15	3.06E-15

Table 11: Performance comparisons of GABC, MABC and ABCMSE on the 60 and 200 (f_{22}, f_{23}) dimensional functions

Function	GABC		MABC		ABCMSE	
	Mean	Std	Mean	Std	Mean	Std
f_1	1.06E-15	1.21E-16	6.03E-29	4.31E-29	3.31E-114	1.40E-113
f_2	8.97E-16	9.29E-17	3.51E-25	2.72E-25	5.24E-108	2.81E-107
f_3	1.04E-15	1.27E-16	1.39E-29	8.84E-30	2.13E-115	5.36E-115
f_4	2.85E-17	1.01E-17	3.00E-62	3.87E-62	1.75E-98	6.72E-98
f_5	2.96E-15	1.85E-16	6.96E-16	1.20E-16	4.37E-59	5.59E-59
f_6	4.47E+00	6.09E-01	3.77E+01	3.14E+00	1.51E+00	1.86E-01
f_7	0	0	0	0	0	0
f_8	3.73E-16	6.67E-17	5.00E-62	9.38E-62	9.05E-223	0
f_9	5.43E-02	7.03E-03	1.14E-01	1.16E-02	3.40E-02	6.62E-03
f_{10}	3.30E+00	1.28E+0	1.51E+00	1.34E+00	1.74E+01	3.16E+01
f_{11}	0	0	0	0	0	0
f_{12}	0	0	0	0	0	0
f_{13}	2.47E-04	1.35E-03	0	0	3.68E-11	1.97E-10
f_{14}	3.97E+01	6.47E+0	3.56E-11	2.18E-12	2.91E-11	0
f_{15}	7.31E-14	5.57E-15	1.37E-13	1.24E-14	5.80E-14	5.36E-15

f_{16}	1.05E-15	1.21E-16	6.19E-31	3.62E-31	7.85E-33	1.37E-48
f_{17}	1.01E-15	1.28E-16	3.80E-29	1.87E-29	1.35E-32	5.47E-48
f_{18}	7.34E-07	1.70E-06	8.20E-16	4.69E-16	2.37E-61	4.25E-61
f_{19}	8.89E-16	8.73E-17	4.08E-30	2.58E-30	1.35E-31	0
f_{20}	9.00E-15	7.90E-15	9.94E-15	5.68E-15	2.84E-15	5.68E-15
f_{21}	4.62E-01	1.79E-02	4.84E-01	3.62E-03	4.37E-01	3.31E-02
f_{22}	-7.83E+01	4.89E-14	-7.83E+01	2.40E-07	-7.83E+01	3.53E-07
f_{23}	-1.96E+02	2.84E-01	-1.74E+02	9.91E-01	-1.99E+02	1.32E-01
f_{24}	1.01E-15	1.23E-16	5.61E-29	4.18E-29	0	0
f_{25}	0	0	0	0	0	0
f_{26}	6.66E-17	1.08E-16	0	0	1.89E-03	4.83E-03
f_{27}	7.54E-14	5.00E-15	2.00E-13	3.07E-14	5.79E-14	4.83E-15
f_{28}	1.24E-05	5.65E-05	9.71E-16	5.70E-16	3.41E-15	5.46E-15

Table 12: Performance comparisons of ABCBest1, ABCBset2 and ABCMSE on the 60 and 200 (f_{22}, f_{23}) dimensional functions

Function	ABCBest1		ABCBest2		ABCMSE	
	Mean	Std	Mean	Std	Mean	Std
f_1	3.92E-44	2.64E-44	4.82E-33	2.59E-33	3.31E-114	1.40E-113
f_2	1.70E-41	9.16E-42	5.86E-27	1.13E-26	5.24E-108	2.81E-107
f_3	2.06E-44	1.83E-44	9.10E-34	3.87E-34	2.13E-115	5.36E-115
f_4	8.74E-74	4.63E-73	7.53E-39	3.95E-38	1.75E-98	6.72E-98
f_5	8.48E-24	2.31E-24	1.58E-17	3.32E-18	4.37E-59	5.59E-59
f_6	2.10E+01	1.68E+0	2.40E+01	2.16E+00	1.51E+00	1.86E-01
f_7	0	0	0	0	0	0
f_8	4.65E-91	7.81E-91	6.76E-72	6.26E-72	9.05E-223	0
f_9	6.11E-02	8.89E-03	6.79E-02	9.38E-03	3.40E-02	6.62E-03
f_{10}	5.04E+01	5.46E+0	5.10E+01	3.77E+01	1.74E+01	3.16E+01
f_{11}	0	0	0	0	0	0
f_{12}	0	0	0	0	0	0
f_{13}	0	0	3.96E-09	2.04E-08	3.68E-11	1.97E-10
f_{14}	3.99E-11	3.64E-12	3.95E+00	2.16E+01	2.91E-11	0

f_{15}	6.93E-14	5.00E-15	7.47E-14	4.12E-15	5.80E-14	5.36E-15
f_{16}	7.85E-33	2.78E-48	7.85E-33	2.78E-48	7.85E-33	1.37E-48
f_{17}	1.35E-32	5.57E-48	1.35E-32	5.57E-48	1.35E-32	5.47E-48
f_{18}	5.29E-16	1.25E-15	2.23E-11	3.77E-11	2.37E-61	4.25E-61
f_{19}	1.35E-31	6.68E-47	1.41E-31	1.47E-32	1.35E-31	0
f_{20}	2.42E-14	8.47E-15	2.65E-14	8.94E-15	2.84E-15	5.68E-15
f_{21}	4.61E-01	1.15E-02	4.68E-01	9.17E-03	4.37E-01	3.31E-02
f_{22}	-7.83E+01	3.71E-11	-7.83E+01	1.76E-08	-7.83E+01	3.53E-07
f_{23}	-1.86E+02	6.01E-01	-1.76E+02	6.92E-01	-1.99E+02	1.32E-01
f_{24}	0	0	0	0	0	0
f_{25}	0	0	0	0	0	0
f_{26}	0	0	1.44E-08	7.17E-08	1.89E-03	4.83E-03
f_{27}	6.90E-14	4.82E-15	7.39E-14	3.54E-15	5.79E-14	4.83E-15
f_{28}	1.80E-16	1.17E-16	2.53E-10	1.17E-09	3.41E-15	5.46E-15

Table 13: Performance comparisons of ABCVSS and ABCMSE on the 60 and 200 (f_{22}, f_{23}) dimensional functions

Function	ABCVSS		ABCMSE	
	Mean	Std	Mean	Std
f_1	1.09E-83	5.01E-83	3.31E-114	1.40E-113
f_2	1.01E-82	5.54E-82	5.24E-108	2.81E-107
f_3	8.17E-86	4.47E-85	2.13E-115	5.36E-115
f_4	1.59E-118	8.70E-11	1.75E-98	6.72E-98
f_5	1.47E-45	7.00E-45	4.37E-59	5.59E-59
f_6	1.68E+00	4.05E-01	1.51E+00	1.86E-01
f_7	0	0	0	0
f_8	7.09E-171	0	9.05E-223	0
f_9	4.35E-02	7.69E-03	3.40E-02	6.62E-03
f_{10}	5.27E-01	1.18E+00	1.74E+01	3.16E+01
f_{11}	0	0	0	0
f_{12}	0	0	0	0
f_{13}	0	0	3.68E-11	1.97E-10

f_{14}	3.64E-11	0	2.91E-11	0
f_{15}	5.93E-14	7.65E-15	5.80E-14	5.36E-15
f_{16}	7.85E-33	2.78E-48	7.85E-33	1.37E-48
f_{17}	1.35E-32	5.57E-48	1.35E-32	5.47E-48
f_{18}	5.42E-47	2.02E-46	2.37E-61	4.25E-61
f_{19}	1.35E-31	6.68E-47	1.35E-31	0
f_{20}	0	0	2.84E-15	5.68E-15
f_{21}	4.72E-01	1.42E-02	4.37E-01	3.31E-02
f_{22}	-7.83E+01	6.84E-06	-7.83E+01	3.53E-07
f_{23}	-1.99E+02	6.38E-01	-1.99E+02	1.32E-01
f_{24}	0	0	0	0
f_{25}	0	0	0	0
f_{26}	0	0	1.89E-03	4.83E-03
f_{27}	5.97E-14	8.52E-15	5.79E-14	4.83E-15
f_{28}	2.86E-16	3.41E-16	3.41E-15	5.46E-15

Table 14: Performance comparisons of GABC, MABC and ABCMSE on the 100 and 300 (f_{22}, f_{23}) dimensional functions

Function	GABC		MABC		ABCMSE	
	Mean	Std	Mean	Std	Mean	Std
f_1	1.84E-15	1.72E-16	1.43E-27	8.12E-28	5.74E-113	1.61E-112
f_2	1.66E-15	1.72E-16	3.52E-24	2.47E-25	7.89E-109	3.28E-108
f_3	1.85E-15	2.00E-16	4.46E-28	2.08E-28	1.63E-112	8.68E-112
f_4	3.66E-17	9.63E-18	1.92E-48	3.42E-48	1.09E-45	5.88E-45
f_5	5.17E-15	2.24E-16	4.41E-15	1.50E-15	2.22E-57	2.22E-57
f_6	1.59E+01	1.55E+0	5.98E+01	1.60E+00	8.54E+00	6.96E-01
f_7	0	0	0	0	0	0
f_8	7.50E-16	9.05E-17	5.72E-60	5.32E-60	1.28E-219	0
f_9	9.47E-02	1.26E-02	2.31E-01	2.79E-02	5.97E-02	6.49E-03
f_{10}	2.40E+01	3.39E+0	1.98E+00	1.30E+00	5.11E+01	4.88E+01
f_{11}	0	0	0	0	0	0
f_{12}	0	0	0	0	0	0

f_{13}	1.26E-16	1.56E-16	0	0	1.86E-15	1.00E-14
f_{14}	1.20E+02	1.22E+0	1.19E-10	4.06E-12	1.03E-10	5.22E-12
f_{15}	1.32E-13	9.74E-15	3.56E-13	2.29E-14	1.11E-13	7.42E-15
f_{16}	1.90E-15	1.95E-16	1.89E-30	8.42E-31	4.71E-33	2.74E-48
f_{17}	1.85E-15	1.68E-16	1.81E-28	6.44E-29	1.35E-32	5.47E-48
f_{18}	1.92E-05	3.56E-05	5.83E-15	1.97E-15	4.07E-17	1.52E-16
f_{19}	1.70E-15	1.76E-16	8.49E-29	3.57E-29	1.35E-31	0
f_{20}	6.06E-14	1.79E-14	5.21E-14	6.69E-15	3.13E-14	1.34E-14
f_{21}	4.95E-01	1.87E-03	4.99E-01	1.75E-04	4.93E-01	3.58E-03
f_{22}	-7.83E+01	1.72E-02	-7.83E+01	1.84E-08	-7.83E+01	2.51E-12
f_{23}	-2.88E+02	6.19E-01	-2.63E+02	7.14E-01	-2.99E+02	1.08E-01
f_{24}	1.90E-15	1.96E-16	1.44E-27	9.16E-28	0	0
f_{25}	0	0	0	0	0	0
f_{26}	9.25E-17	1.09E-16	0	0	3.29E-04	1.77E-03
f_{27}	1.31E-13	7.35E-15	5.33E-13	7.58E-14	1.08E-13	7.32E-15
f_{28}	1.03E-05	2.50E-05	3.01E-15	1.39E-15	4.23E-15	7.90E-15

Table 15: Performance comparisons of ABCBest1, ABCBset2 and ABCMSE on the 100 and 300 (f_{22}, f_{23}) dimensional functions

Function	ABCBest1		ABCBest2		ABCMSE	
	Mean	Std	Mean	Std	Mean	Std
f_1	1.54E-42	8.93E-43	5.09E-32	2.03E-32	5.74E-113	1.61E-112
f_2	5.08E-40	3.02E-40	2.81E-26	1.61E-26	7.89E-109	3.28E-108
f_3	8.94E-43	7.34E-43	2.15E-32	1.24E-32	1.63E-112	8.68E-112
f_4	8.92E-60	4.00E-59	1.92E-29	6.05E-29	1.09E-45	5.88E-45
f_5	6.27E-23	1.09E-23	7.21E-17	1.36E-17	2.22E-57	2.22E-57
f_6	4.72E+01	2.29E+0	5.06E+01	2.67E+00	8.54E+00	6.96E-01
f_7	0	0	0	0	0	0
f_8	2.45E-88	2.81E-88	1.61E-69	1.73E-69	1.28E-219	0
f_9	1.30E-01	1.12E-02	1.42E-01	1.58E-02	5.97E-02	6.49E-03
f_{10}	5.81E+01	6.80E+0	1.18E+02	5.76E+01	5.11E+01	4.88E+01
f_{11}	0	0	0	0	0	0

f_{12}	0	0	0	0	0	0
f_{13}	0	0	2.94E-10	1.16E-09	1.86E-15	1.00E-14
f_{14}	1.22E-10	4.14E-12	1.32E+00	7.21E+00	1.03E-10	5.22E-12
f_{15}	1.27E-13	7.15E-15	1.39E-13	5.75E-15	1.11E-13	7.42E-15
f_{16}	4.71E-33	1.39E-48	4.71E-33	1.39E-48	4.71E-33	2.74E-48
f_{17}	1.35E-32	5.57E-48	2.18E-32	6.62E-33	1.35E-32	5.47E-48
f_{18}	1.83E-16	4.50E-16	8.44E-10	1.74E-09	4.07E-17	1.52E-16
f_{19}	1.35E-31	6.68E-47	2.42E-31	1.85E-31	1.35E-31	0
f_{20}	1.18E-13	2.70E-14	1.29E-13	2.33E-14	3.13E-14	1.34E-14
f_{21}	4.96E-01	8.28E-04	4.97E-01	6.58E-04	4.93E-01	3.58E-03
f_{22}	-7.83E+01	1.97E-12	-7.83E+01	1.26E-09	-7.83E+01	2.51E-12
f_{23}	-2.79E+02	7.32E-01	-2.67E+02	7.75E-01	-2.99E+02	1.08E-01
f_{24}	0	0	0	0	0	0
f_{25}	0	0	0	0	0	0
f_{26}	0	0	1.01E-12	5.23E-12	3.29E-04	1.77E-03
f_{27}	1.26E-13	8.48E-15	1.38E-13	6.90E-15	1.08E-13	7.32E-15
f_{28}	3.76E-16	3.77E-16	1.01E-09	2.12E-09	4.23E-15	7.90E-15

Table 16: Performance comparisons of ABCVSS and ABCMSE on the 100 and 300 (f_{22}, f_{23}) dimensional functions

Function	ABCVSS		ABCMSE	
	Mean	Std	Mean	Std
f_1	1.01E-83	5.52E-83	5.74E-113	1.61E-112
f_2	5.29E-75	2.80E-74	7.89E-109	3.28E-108
f_3	5.31E-85	2.68E-84	1.63E-112	8.68E-112
f_4	4.86E-109	2.66E-10	1.09E-45	5.88E-45
f_5	3.94E-40	2.12E-39	2.22E-57	2.22E-57
f_6	7.91E+00	1.32E+00	8.54E+00	6.96E-01
f_7	0	0	0	0
f_8	6.79E-166	0	1.28E-219	0
f_9	7.87E-02	1.16E-02	5.97E-02	6.49E-03
f_{10}	5.14E-01	1.05E+00	5.11E+01	4.88E+01

f_{11}	0	0	0	0
f_{12}	0	0	0	0
f_{13}	0	0	1.86E-15	1.00E-14
f_{14}	1.12E-10	3.67E-12	1.03E-10	5.22E-12
f_{15}	1.11E-13	9.93E-15	1.11E-13	7.42E-15
f_{16}	4.71E-33	1.39E-48	4.71E-33	2.74E-48
f_{17}	1.35E-32	5.57E-48	1.35E-32	5.47E-48
f_{18}	2.78E-17	1.17E-16	4.07E-17	1.52E-16
f_{19}	1.35E-31	6.68E-47	1.35E-31	0
f_{20}	1.89E-15	7.21E-15	3.13E-14	1.34E-14
f_{21}	4.98E-01	8.15E-04	4.93E-01	3.58E-03
f_{22}	-7.83E+01	6.85E-06	-7.83E+01	2.51E-12
f_{23}	-2.98E+02	1.20E+00	-2.99E+02	1.08E-01
f_{24}	0	0	0	0
f_{25}	0	0	0	0
f_{26}	0	0	3.29E-04	1.77E-03
f_{27}	1.15E-13	1.08E-14	1.08E-13	7.32E-15
f_{28}	3.48E-16	2.08E-16	4.23E-15	7.90E-15

In Table 8-16, ABCMSE is further compared with GABC [21], MABC [23], ABCBest1 [22], ABCBest2 [22] and ABCVSS [26], the number of maximum function evaluations is set to 150000 for 30 and 100 (f_{22}, f_{23}) dimensional functions, 300000 for 60 and 200 (f_{22}, f_{23}) dimensional functions and 500000 for 60 and 200 (f_{22}, f_{23}) dimensional functions. The results obtained by 30 independent runs are reported in Table 8-10 for 30-dimensional functions, Table 11-13 for 60-dimensional functions, Table 14-16 for 100-dimensional functions. In the comparison tables, the results of GABC、MABC、ABCBest1、ABCBest2 and ABCVSS are taken directly from [26].

As seen from the Tables 8-16, the ABCMSE algorithm outperforms the other ABC variants in most cases. Therefore, it can be concluded that ABCMSE is more efficient than these ABCs. In a word, the ABCMSE algorithm can improve bees'

searching abilities, prevent bees from falling into the local minimum, increase convergence speed and compute more efficiently.

5 Conclusion

In this paper, a novel artificial bee colony algorithm with mixed search equation is presented, called ABCMSE. In our method, two different solution search equations are mixed. Then, we use a parameter ω to take advantage of them and avoid the shortages of them. At the same time, the comparison and the selection of the new solution are changed from a fitness-based comparison to an objective-value based. The performance of ABCMSE was compared with the basic ABC algorithm and several classical versions of ABC algorithm using a set of benchmark functions. The results demonstrate that the new algorithm outperforms the basic ABC algorithm and other variants of the ABC algorithm in terms of solution quality and robustness for most of the experiments.

References

[1] M. Dorigo, V. Maniezzo, A. Colorni, The ant system: optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics Part B-Cybernetics*, **26**(1), (1996), 1-13.

[2] J. Kennedy, R.C. Eberhart, Particle swarm optimization, in: Proc. of IEEE International Conference on Neural Networks, Piscataway, USA, (1995), 1942-1948.

[3] X. L. Li, Z. J. Shao, J. X. Qian, An optimizing method based on autonomous animates: fish-swarm algorithm, *System Engineering-Theory & Practice*, **22**(11), (2002), 32-38.

[4] X. S. Yang, S. Deb, Engineering optimisation by cuckoo search, *International Journal of Mathematical Modelling & Numerical Optimisation*, **1**(4), (2010), 330-343.

[5] D. Karaboga, An idea based on honey bee swarm for numerical optimization, Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[6] B. Basturk, D. Karaboga, An artificial bee colony (ABC) algorithm for numeric function optimization, in: IEEE Swarm Intelligence Symposium, 2006.

[7] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, *Journal of Global Optimization*, **39**(3), (2007), 459-471.

[8] D. Karaboga, B. Basturk, On the performance of artificial bee colony (ABC) algorithm, *Applied Soft Computing*, **8**(1), (2008), 687-697.

[9] D. Karaboga, B. Akay, A comparative study of artificial bee colony algorithm, *Applied Mathematics and Computation*, **214**(1), (2009), 108-132.

[10] A. Singh, An artificial bee colony algorithm for the leaf-constrained minimum spanning tree problem, *Applied Soft Computing*, **9**(2), (2009), 625-631.

[11] D. Karaboga, C. Ozturk, A novel clustering approach: artificial bee colony (ABC) algorithm, *Applied Soft Computing*, **11**(1), (2011), 652-657.

[12] H. C. Tsai, Integrating the artificial bee colony and bees algorithm to face constrained optimization problems, *Information Sciences*, **258**(3), (2014), 80-93.

[13] D. Karaboga, S. Okdem, C. Ozturk, Cluster based wireless sensor network routing using artificial bee colony algorithm, *Wireless Networks*, **18**(7), (2012), 847-860.

[14] V.J. Manoj, E. Elias, Artificial bee colony algorithm for the design of multiplier-less nonuniform filter bank transmultiplexer, *Information Sciences*, **192**(6), (2012), 193-203.

[15] W. C. Yeh, T. J. Hsieh, Solving reliability redundancy allocation problems using an artificial bee colony algorithm, *Computers & Operations Research*,

38(11), (2011), 1465-1473.

[16] S.N. Omkar, J. Senthilnath, R. Khandelwal, G.N. Naik, S. Gopalakrishman, Artificial bee colony (ABC) for multi-objective design optimization of composite structures, *Applied Soft Computing*, **11**(1), (2011), 489-499.

[17] D. Karaboga, B. Akay, C. Ozturk, Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks, in: 4th International Conference MDAI, Kitakyushu, Japan, (2007), 318-329.

[18] W. Y. Szeto, Y. Wu, S. C. Ho, An artificial bee colony algorithm for the capacitated vehicle routing problem, *European Journal of Operational Research*, **215**(1), (2011), 126-135.

[19] F. Kang, J. J. Li, Z. Y. Ma, Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions, *Information Sciences*, **181**(16), (2011), 3508-3531.

[20] B. Alatas, Chaotic bee colony algorithms for global numerical optimization, *Expert System with Applications*, **37**(8), (2010), 5682-5687.

[21] G. Zhu, S. Kwong, Gbest-guided artificial bee colony algorithm for numerical function optimization, *Applied Mathematics and Computation*, **217**(7), (2010), 3166-3173.

[22] W. Gao, S. Liu, L. Huang, A global best artificial bee colony algorithm for global optimization, *Journal of Computational and Applied Mathematics*, **236**(11), (2012), 2741-2753.

[23] W. Gao, S. Liu, A modified artificial bee colony algorithm, *Computer & Operations Research*, **39**(3), (2012), 687-697.

[24] N. Imanian, M.E. Shiri, P. Moradi, Velocity based artificial bee colony algorithm for high dimensional continuous optimization problems, *Engineering Applications of Artificial Intelligence*, **36**(11), (2014), 148-163.

[25] H. Wang, Z. Wu, S. Rahnamayan, H. Sun, Y. Liu, J. Pan, Multi-strategy ensemble artificial bee colony algorithm, *Information Sciences*, **279**(9), (2014), 587-603.

[26] M. S. Kiran, H. Hakli, M. Gunduz, H. Uguz, Artificial bee colony algorithm with variable search strategy for continuous optimization, *Information Sciences*, **300**(1), (2015), 140-157.

[27] A. Banharnsakun, T. Achalakul, B. Sirinaovakul, The best-so-far selection in artificial bee colony algorithm, *Applied soft Computing*, **11**(2), (2011), 2888–2901.

[28] J. Qiu, J. Wang, D. Yang, J. Xie, A comparison of improved artificial bee colony algorithms based on differential evolution, *TELKOMNIKA Indonesian Journal of Electrical Engineering*, **11**(10), (2013), 5579-5587.

[29] K. Lei, Y. Qiu, Y. He, An effective particle swarm optimizer for solving complex functions with high dimensions, *Computer Science*, **33**(8), (2006), 202-205.