An Improved Artificial Bee Colony Algorithm for Solving Optimization Problems

Chun-Feng Wang, and Yong-Hong Zhang

Abstract—**The artificial bee colony (ABC) algorithm is a simple and effective optimization algorithm that can be applied to solve many real-world optimization problems. In this paper, an improved artificial bee colony algorithm, named IABC, is proposed. In this algorithm, a new search equation for onlooker bees is presented, in which the optimal and sub-optimal solutions are considered in the iteration process. In addition, for enhancing the global convergence, the initial population are generated by using a chaotic system and an opposition-based method. Furthermore, for improving the global convergence speed of our algorithm, a chaotic search is adopted in the best solution of the current iteration. Experimental results tested on 20 benchmark functions show that the IABC algorithm is far better than the basic ABC algorithm and other three improved ABC algorithms on the tested problems.**

Index Terms—**Artificial bee colony algorithm; Continuous optimization; Chaotic search; Opposition-based learning.**

I. Introduction

OPTIMIZATION can be defined as an issue of finding the best solutions to a problem under bounded circumstance. This paper considers the following global optimization problem:

$$\begin{array}{ll} \min & f(\mathbf{x}) \\ \text{s.t.} & \mathbf{x} \in X, \end{array} \tag{1}$$

where $\mathbf{x}$ is a continuous variable vector with domain $X \subset R^n$ defined by the bound constraint $l_j \leq x_j \leq u_j,\ j = 1, \cdots, n$. The function $f(\mathbf{x}) : X \rightarrow R$ is a continuous real-valued function.

For solving such problem, many optimization methods have been presented in the past years. These methods are generally divided into two groups:deterministic and stochastic algorithms. Most deterministic algorithms usually need gradient information, and these algorithms are ideal for unimodal functions have one global optimum, while they might be troublesome for multimodal functions having several local optimal solutions or functions that include flat region where the gradient is small. For overcoming the disadvantages of deterministic algorithms, many stochastic algorithms have been developed, include Biogeography-Based Optimization(BBO)[1], Bat Algorithm(BA)[2], Spider Monkey Optimization(SMO)[3], Genetic Algorithm(GA)[4,5], Differential Evolution(DE)[6], Particle Swarm Optimization(PSO)[7], Ant Colony Optimization(ACO)[8], Artificial Bee Colony (ABC)[9], Harmony Search (HS)[10], Cuckoo Search Algorithm (CSA)[11], Modified Immune Network Optimization Algorithm(MINA) [12], etc.

Among these stochastic algorithms, ABC algorithm is a relatively new meta-heuristic algorithm, was first proposed by Karboga in 2005[9]. The ABC uses a simple mechanism that imitates the behaviours of the honey bees to search for global optimal solutions. Because of its simplicity of implementation and capability to quickly converge to a reasonably good solution, the ABC has been successfully applied in solving many real-world problems[13-15], and has been paid more and more attention. A good survey study on ABC algorithm can be found in [16].

Although the standard ABC algorithm generally produces successful results in many applications, it has difficulties in keeping balance between exploration and exploitation. It is well known that both exploration and exploitation are necessary for a population-based optimization algorithm. However, by the solution search equation of ABC, we can observe that ABC algorithm is good at exploration, but poor at exploitation. In order to get a better performance for ABC algorithm, many variants of ABC have been developed. For example, based on PSO, Zhu and Kwong proposed an improved ABC algorithm named gbest-guide ABC (GABC) [17]. In this algorithm, the global best solution is placed into the solution search equation. By integrating ABC with self-adaptive guidance, Tuba et al. improved GABC algorithm[18]. Inspired by DE algorithm, Gao and Liu presented a new solution search equation for standard ABC[19]. For enhancing the exploitation capability of ABC, Gao et al. proposed an improved ABC[20]. In their algorithm, a modified search strategy is used to generate new food sources. For obtaining more powerful optimization algorithms, some researchers combined ABC algorithm with other heuristic techniques. For example, Bin and Qian presented a differential ABC algorithm for global numerical optimization[21]. Sharma and Pant used DE operators with standard ABC algorithm developed a hybrid algorithm [22]. By combing PSO and ABC, Hsieh et al. proposed a new hybrid algorithm[23]. Abraham et al. made a hybridization of ABC algorithm and DE strategy, and called this novel approach as hybrid differential ABC algorithm[24].

In this paper, for improving the exploitation of onlooker bees, the solution search equation is modified. In the new search equation, the information of the optimal and sub-optimal solutions is used to guide the search of new candidate solutions. Based on this search equation, an improved ABC algorithm is proposed, which is named IABC. In addition, for enhancing the global convergence, a chaotic system and an opposition-based method are utilized to generate the

Manuscript received Dec. 12, 2015; revised Feb. 28, 2016. This paper is supported by the National Natural Science Foundation of China (U1404105), the Key Scientific and Technological Project of Henan Province(142102210058); the Youth Science Foundation of Henan Normal University(2013qk02); Henan Normal University National Research Project to Cultivate the Funded Projects (01016400105).

Chun-Feng Wang is with College of Mathematics and Information Science, Henan Normal University, Xinxiang, 453007, PR China; Henan Engineering Laboratory for Big Data Statistical Analysis and Optimal Control, School of Mathematics and Information Sciences, Henan Normal University.

Yong-Hong Zhang is with College of Mathematics and Information Science, Henan Normal University,Xinxiang, 453007, PR China, e-mail:zhangyonghong09@126.com.

initial population. Furthermore, for improving the global convergence speed of our algorithm, a chaotic search is adopted in the best solution of the current iteration. To evaluate the performance of IABC algorithm, it is compared with ABC, GABC[17], COABC[25], ABC/best/1[20] over a testbed made up of 20 benchmark functions. The experimental results show that IABC has a better performance than the other four algorithms on all test functions in terms of best, worst, median, and standard deviation of the solutions obtained by each algorithm in 30 independent runs.

The remaining of the paper is arranged as follows. Section 2 describes the original ABC algorithm. The improved ABC algorithm (IABC) is presented in Section 3. The computational results are reported in Section 4. Finally, the conclusion is given in Section 6.

II. The original ABC algorithm

By the foraging behaviour of the colony, the artificial bees in ABC algorithm can be divided in three groups:employed bees, onlooker bees and scout bees.

In ABC algorithm, a food source and its nectar amount are defined as a possible solution and its fitness value, respectively. Since each employed bee is associated with one and only one food source, the number employed bees is equal to the number of food sources.

Assume that the search space is n-dimension, the position of the i-th food source can be expressed as a n-dimension vector $\mathbf{x}_i = (x_{i,1}, x_{i,2}, \cdots, x_{i,n})$, $i = 1, \cdots, SN$, SN is the number of food sources.

Generally, ABC algorithm contains four phases: initialization phase, employed bees phase, onlooker bees phase and scout bees phase.

Initialization phase: In this phase, the initial food sources can be generated randomly by the following equation:

$$x_{i,j} = x_j^l + rand(0,1) * (x_j^u - x_j^l), \tag{1}$$

where $i \in \{1, 2, \cdots, SN\}$, $j \in \{1, 2, \cdots, n\}$, x_j^l and x_j^u are the lower bound and upper bound for the jth dimension, respectively.

After initialization, compute the fitness $fit(\mathbf{x}_i)$ for each solution $\mathbf{x}_i$ by using the following equation (2):

$$fit(\mathbf{x}_i) = \begin{cases} \frac{1}{1+f(\mathbf{x}_i)}, & \text{if } f(\mathbf{x}_i) \geq 0, \\ 1 + abs(f(\mathbf{x}_i)), & \text{if } f(\mathbf{x}_i) < 0. \end{cases} \tag{2}$$

Employed bees phase: In this phase, each employed bee generates a new food source $\mathbf{v}_i$ in the neighborhood of its present position $\mathbf{x}_i$ by using the following equation:

$$v_{i,j} = x_{i,j} + \phi_{i,j}(x_{i,j} - x_{k,j}), \tag{3}$$

where $k \in \{1, 2, \cdots, SN\}$, $j \in \{1, 2, \cdots, n\}$ are selected randomly, and $k \neq i$, $\phi_{i,j}$ is a random number within the range [-1,1].

Once $\mathbf{v}_i$ is obtained, a greedy selection mechanism is employed between $\mathbf{v}_i$ and $\mathbf{x}_i$: if $f(\mathbf{v}_i) \leq f(\mathbf{x}_i)$, $\mathbf{v}_i$ will replace $\mathbf{x}_i$ and become a new member of the population, otherwise, $\mathbf{x}_i$ is retained.

Onlooker bees phase: In this phase, each onlooker bee selects a food source according to the probability based on its fitness value, which is defined by the following equation:

$$p_i = \frac{fit(\mathbf{x}_i)}{\sum\limits_{i=1}^{SN} fit(\mathbf{x}_i)}. \tag{4}$$

From (4), it is obvious that the higher the $fit(\mathbf{x}_i)$ is, the more probability that the ith food source is selected.

Once a food source $\mathbf{x}_i$ is selected, as in the case of the employed bees, a neighbour source $\mathbf{v}_i$ will be generated by using (3), and its fitness value $fit(\mathbf{v}_i)$ will be computed. Then, a greedy selection is applied between $\mathbf{x}_i$ and $\mathbf{v}_i$.

Scout bees phase: In this phase, if a food source $\mathbf{x}_i$ can not be improved further through a predetermined number of trail $limit$, then the source is assumed to be abandoned. If such an abandonment is detected, the related employed bee is converted to a scout bee, and produces a new food source randomly as follows:

$$x_{i,j} = x_j^l + rand(0,1) * (x_j^u - x_j^l),$$

where $j \in \{1, 2, \cdots, n\}$.

Based on the above explanation, the pseudo-code of the original ABC algorithm is given below.

Algorithm 1 ABC algorithm

```
1. Set parameters:$SN$, $maxcycle$, $limit$.
2. Use the (1) to initialize the position of the food sources $\{x_i \mid i = 1, \cdots, SN\}$, and
   evaluate the solutions. Set t=0.
3. While t  maxcycle do
4. // Employed bees phase
5. For each employed bee, use (3) to generate a new candidate solution $v_i$ in the
   neighbourhood of $x_i$, and evaluate it. Apply a greedy selection between $x_i$ and
   $v_i$.
6. Compute the fitness value for each solution by using (2), and compute its
   probability values $p_i$ by utilizing (4).
7. // Onlooker bees phase
8. For each onlooker bee, select a solution depending on $p_i$ values, generate a new
   candidate solution $v_i$ with (3), and evaluate it.
9. Apply a greedy selection between $x_i$ and $v_i$.
10. Memorize the best solution found so far.
11. //Scout bees phase
12. Determine the abandoned solution by utilizing the limit parameter value $limit$. If
    exists, replace it with a new solution for the scout bee by using (1).
13. $t$=$t$+1.
14. Until $t$=$maxcycle$
```

III. Improved artificial bee colony algorithm(IABC)

As it is pointed that, in population-based optimization algorithms, both exploration and exploitation process must be carried out together. However, ABC algorithm is good at exploration, but poor at exploitation. To improve the performance of ABC, one active research trend is to study its search equation.

Inspired by PSO, Zhu and Kwong presented the following solution search equation $GABC$[17]:

$$v_{i,j} = x_{i,j} + \phi_{i,j} * (x_{i,j} - x_{k,j}) + \psi_{i,j} * (x_{best,j} - x_{i,j}), \tag{5}$$

where $\psi_{i,j}$ is a uniform random number in [0,C]. In [20], based on the variant DE algorithm and the property of ABC, the solution search equation $ABC/best/1$ is modified as follows:

$$v_{i,j} = x_{best,j} + \phi_{i,j} * (x_{r_1,j} - x_{r_2,j}), \tag{6}$$

where r_1 and r_2 are integers randomly chosen from $\{1, 2, \cdots, SN\}$, and different from $r_1 \neq r_2 \neq i$; $\mathbf{x}_{best}$ is the best food source in the current population, and $j \in \{1, 2, \cdots, n\}$ is randomly chosen; $\phi_{i,j}$ is a random number in the range [-1,1]. In [25], a new mechanism $COABC$ is proposed at all onlooker bees are placed on the global best food source, which can described as follows:

$$v_{i,j} = x_{best,j} + \phi_j * (x_{best,j} - x_{k,j}), \tag{7}$$

where k is randomly chosen from $\{1, 2, \cdots, SN\}$, and $k \neq j$.

From (5)-(7), we can see that, only the information of the global solution $\mathbf{x}_{best}$ is used. However, in some practical problems, optimal solution not only depends on optimal experience, but also needs sub-optimal experience. Therefore, a new solution search equation for onlooker bees is proposed in this paper, which considers the information of optimal and sub-optimal solutions. And then, a highly efficient global optimization method, called IABC, is developed.

For further improving the performance of IABC, chaotic system and opposition-based learning method are applied to produce the initial population. Meanwhile, a chaotic search is adopted in the best solution of the current iteration to improve the global convergence speed of our algorithm.

A. Chaotic and opposition-based initialization

In evolutionary algorithms, population initialization is a crucial task. The reason is that it can affect the convergence speed and the quality of the final solution. Generally speaking, if no information about the solution is available, then random initialization is the most commonly used method to generate candidate solutions. But for enhancing the population diversity and solution quality of the initial population, a chaotic system[26] and an opposition-based method[27] are employed in this paper. The procedure is described as Algorithm 2.

```
Algorithm 2    Chaotic and opposition-based initialization
1. Set the population size SN, i=1,j=1.
2. // Chaotic system
3. For i=1 to SN do
4. Randomly initialize variable $ch_0 \in (0,1)$.
5. For j=1 to n do
6. $ch_j = \sin(\pi * ch_{j-1})$
7. $x_{i,j} = x_j^l + ch_j * (x_j^u - x_j^l)$
8. End for
9. End for
10. // Opposition-based learning method
11. For i=1 to SN do
12. For j=1 to n do
13. $ox_{i,j} = x_j^l + x_j^u - x_{i,j}$
14. End for
15. End for
16. Selecting SN fitter individual from the set $\{X \bigcup OX\}$ as the initial population.
```

B. Modified search equation for onlooker bees

Once a food source $\mathbf{x}_i$ is selected, for generate a new candidate food source $\mathbf{v}_i$, we present a new search equation $IABC$ for the onlooker bees by considering the information of the optimal solution $\mathbf{x}_{best}$ and sub-optimal solution $\mathbf{x}_{subbest}$ of the population in the process of iteration, which is defined as follows:

$$\begin{aligned} v_{i,j} = \omega * x_{best,j} + c_1 * \phi_j * (x_{best,j} - x_{i,j}) \\ + c_2 * \psi_j * (x_{subbest,j} - x_{i,j}), \end{aligned} \tag{8}$$

where ω is the inertia weight that controls the impact of the optimal solution at current iteration; $c_1,\ c_2$ are positive constant parameters; $\phi_j,\ \psi_j$ are uniform random numbers in the range [-1,1],$j = 1, 2, \cdots, n$.

Remark 1: In IABC algorithm, the parameter ω is updated dynamically, which is described by the following equation (9),

$$\omega = \omega_{min} + \frac{\omega_{max} - \omega_{min}}{maxcycle} * t, \tag{9}$$

where ω_{min} and ω_{max} represent the lower bound and upper bound of the ω; $maxcycle$ is the maximum number of iteration; t is the number of iterations.

Based on the above mentioned explanation, the pseudo code of the IABC algorithm is given in Algorithm 3.

```
Algorithm 3    IABC algorithm
1. Set parameters:SN, maxcycle, limit, $\omega_{\min}$, $\omega_{\max}$, $c_1$, $c_2$.
2. Use the (1) to initialize the position of the food sources $\{x_i \mid i = 1, \cdots, SN\}$, and
   evaluate the solutions. Set t=0.
3. While $t \le maxcycle$ do
4. // Employed bees phase
5. For each employed bee, use (3) to generate a new candidate solution $v_i$ in the
   neighbourhood of $x_i$, and evaluate it. Apply a greedy selection between $x_i$ and
   $v_i$.
6. Compute the fitness value for each solution by using (2), and compute its
   probability values $p_i$ by utilizing (4).
7. // Onlooker bees phase
8. For each onlloker bee, select a solution depending on $p_i$ values, generate a new
   candidate solution $v_i$ with (8), and evaluate it.
9. Apply a greedy selection between $x_i$ and $v_i$.
10. Memorize the best solution found so far.
11. //Scout bees phase
12. Determine the abandoned solution by utilizing the limit parameter value limit. If
    exists, replace it with a new solution for the scout bee by using (1).
13. By (10)-(13), to chaotic search in $x_{best}$, and update $x_{best}$ (if necessary).
14. t=t+1.
15. Until t=maxcycle
```

C. Chaotic search operator

To improve the global convergence of IABC, a chaotic search operator is adopted. Next, we give the details.

Let x_{best} is the best solution of the current iteration. Firstly, utilize the following equation (10) to generate chaotic variable ch_i:

$$ch_{i+1} = 4 * ch_i * (1 - ch_i),\ 1 \le i \le K, \tag{10}$$

where K is the length of chaotic sequence, $ch_0 \in (0,1)$ is a random number. Then map ch_i to a chaotic vector CH_i in the interval $[l,u]$:

$$CH_i = l + ch_i * (u - l),\ i = 1, \cdots, K, \tag{11}$$

where l and u are the lower bound and upper bound of variable x, respectively. Finally, a new candidate solution $\hat{x}_i$ is obtained by the following equation:

$$\hat{x}_i = (1 - \lambda) * x_{best} + \lambda * CH_i,\ i = 1, \cdots, K, \tag{12}$$

where λ is a shrinking factor, which is defined as follows:

$$\lambda = \frac{maxcycle - t + 1}{maxcycle}, \tag{13}$$

where $maxcycle$ is the maximum number of iterations, t is the number of iterations.

By (12) and (13), it can be seen that, λ will become smaller with the evolution generations increase, that is the local search range will become smaller with the process of evolution.

IV. Experimential results and discussion

A. Benchmark functions and parameter settings

In this section, to evaluate the performance of proposed IABC algorithm, it is applied to minimize a set of 20 scalable test functions, which are chosen from [28], and these benchmark functions with different characteristics, include dimension, initial range, formulations and properties, are listed briefly in Table I. These benchmark functions can be divided into two different groups:unimodal and multimodal. In order to comprehensively verify the effectiveness of IABC, its performance is compared with the initial ABC and other three stat-of-the-art ABC variants, i.e. GABC, COABC, ABC/best/1.

Functions $f_1 - f_{18}$ are all high-dimensional and scalable problems. Functions $f_1 - f_9$ are continuous and unimodal. Function f_{10} is a step function, which has one minimum and is discontinuous. Function f_{11} is a quartic function with noise. Functions $f_{12} - f_{18}$ are difficult multimodal functions where the number of local minima increases exponentially with the problem size. Functions $f_{19} - f_{20}$ are shifted functions, where z are $\mathbf{x} - \mathbf{o}$ and $\mathbf{x} - \mathbf{o} + \mathbf{1}$ for f_{19} and f_{20}, respectively. The proposed algorithm IABC and other four algorithms are coded in Matlab 7.0, and the experiments platform is a personal computer with Pentium 4, 3.06GHz CPU, 512M memory and Windows XP.

For all the algorithms, the stop criterion is that maximum number of iteration reaches $maxcycle$. Each benchmark function is independently run with every algorithm 30 times for comparison.

The parameters of algorithms are given as below. The common parameters:the population size $SN = 25$, the maximum number of iteration $maxcycle = 2500$, the $limit$ is 100. IABC settings: $\omega_{min} = 0.4$, $\omega_{max} = 0.9$, $c_1 = 0.01$, $c_2 = 0.01$, $K = 5$. COABC setting: $UTEB = 1$, $UTEB$ denotes the update times of employed bees.

TABLE I: Benchmark functions used in experiments.

Functions	Range	Optimal value
$f_1 = \sum_{i=1}^{n} x_i^2$	[-100,100]	0
$f_2 = \sum_{i=1}^{n} \mid x_i \mid + \prod_{i=1}^{n} \mid x_i \mid$	[-10,10]	0
$f_3 = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	[-100,100]	0
$f_4 = \max_i\{\mid x_i \mid, 1 \leq i \leq n\}$	[-100,100]	0
$f_5 = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	[-30,30]	0
$f_6 = \sum_{i=1}^{n} i x_i^2$	[-10,10]	0
$f_7 = \sum_{i=1}^{n} i x_i^4$	[-1.28,1.28]	0
$f_8 = \sum_{i=1}^{n} \mid x_i \mid^{i+1}$	[-1,1]	0
$f_9 = \sum_{i=1}^{n} (10^6)^{\frac{i-1}{n-1}} x_i^2$	[-100,100]	0
$f_{10} = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$	[-1.28,1.28]	0
$f_{11} = \sum_{i=1}^{n} i x_i^4 + random[0,1)$	[-1.28,1.28]	0
$f_{12} = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$	[-5.12,5.12]	0
$f_{13} = -20\exp(-0.2 * \sqrt{\sum_{i=1}^{n} x_i^2/n}) - \exp(\sum_{i=1}^{n} \cos(2\pi x_i/n) + 20 + e$	[-32,32]	0
$f_{14} = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}}) + 1$	[-600,600]	0
$f_{15} = 0.5 + \frac{sin^2(\sqrt{\sum_{i=1}^{n} x_i^2}) - 0.5}{(1 + 0.001(\sum_{i=1}^{n} x_i^2))^2}$	[-100,100]	0
$f_{16} = \frac{1}{n} \sum_{i=1}^{n} (x_i^4 - 16x_i^2 + 5x_i)$	[-5,5]	-78.3323
$f_{17} = \sum_{i=1}^{n} \mid x_i sin(x_i) + 0.1x_i \mid$	[-10,10]	0
$f_{18} = \sum_{i=1}^{n} (y_i^2 - 10\cos(2\pi y_i) + 10)$, $if \mid x_i \mid < \frac{1}{2}, y_i = x_i; else\ y_i = \frac{round(2x_i)}{2}$	[-5.12,5.12]	0
$f_{19} = \sum_{i=1}^{n} (\sum_{j=1}^{i} z_j)^2$	[-100,100]	0
$f_{20} = \sum_{i=1}^{n-1} [100(z_{i+1} - z_i^2)^2 + (z_i - 1)^2]$	[-30,30]	0

B. Comparisons and solution accuracy

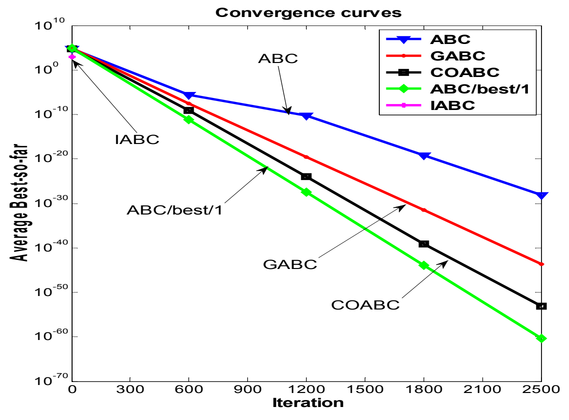
The best value(Best), the worst value(Worst), the mean value(Mean) and the standard deviation(SD) in 30 runs are calculated as the statistics for the performance measures. The comparison results are presented in Table II. Note that, the mean of solutions indicates the solution quality of the algorithms, and the standard deviation represents the stability of the algorithms.

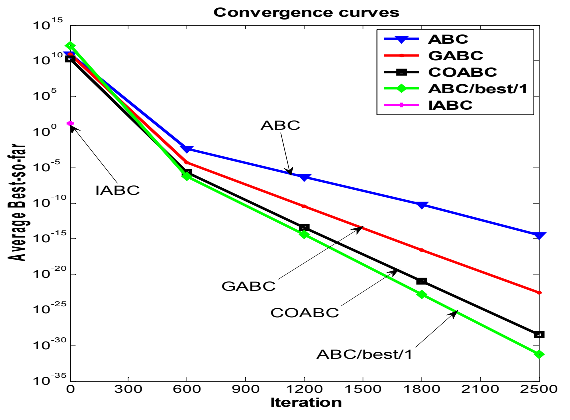
From Table II, it can be seen that the IABC performs better than the other four algorithms on all test functions.

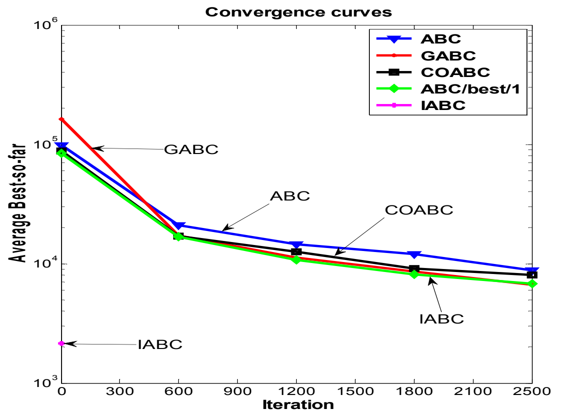
C. Comparisons on convergence speed

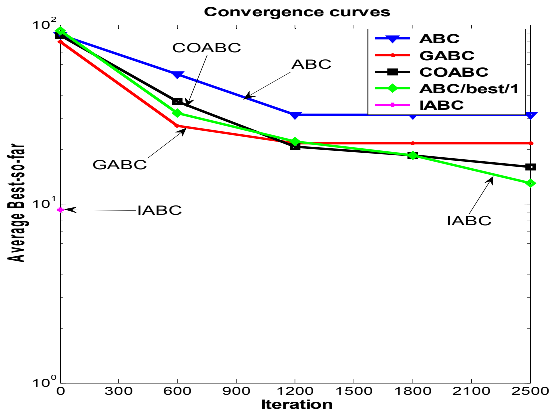
To compare the convergence speed of the five algorithms, the convergence graphs of ABC, GABC, COABC, ABC/best/1 and IABC are shown in Fig.1. In the figure, each curve represents the variation of best fitness over the iterations for a specific ABC. The terminal point of each curve therefore reveals the number of iterations for the trial that required the longest evolutionary process to achieve the global optimization.

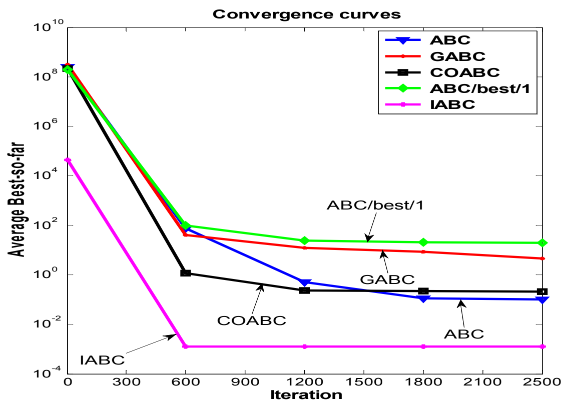
From Fig. 1, it can be seen that the convergence speed of IABC is more faster than the original ABC, and other three improved ABC algorithms.

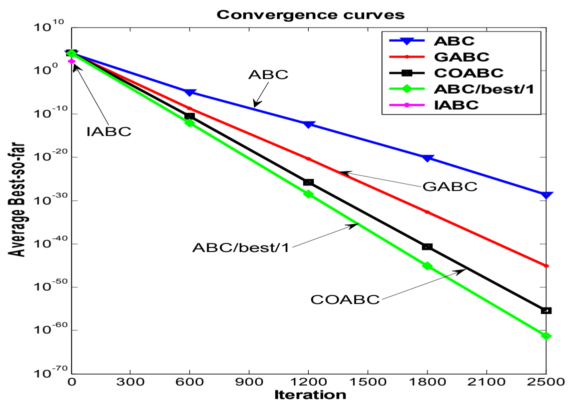

Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
ABC
IABC
ABC/best/1
GABC
COABC

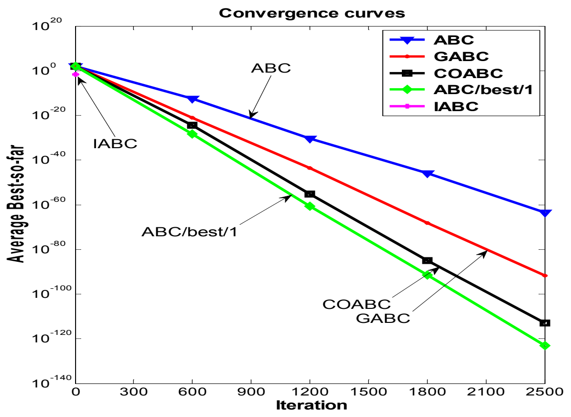

Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
ABC
IABC
GABC
COABC
ABC/best/1

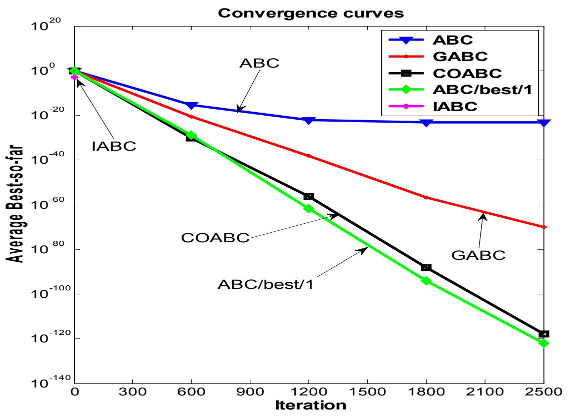

Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
GABC
ABC
COABC
IABC
IABC

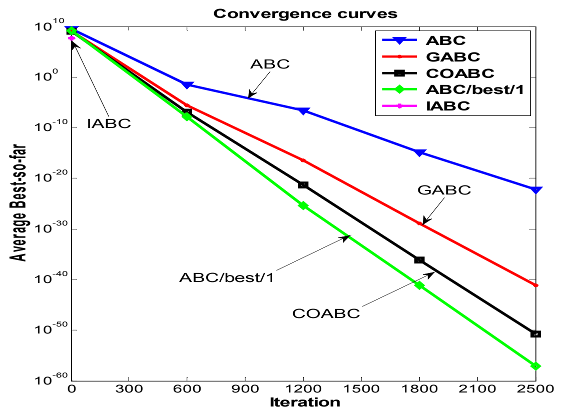

Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
COABC
ABC
GABC
IABC
IABC

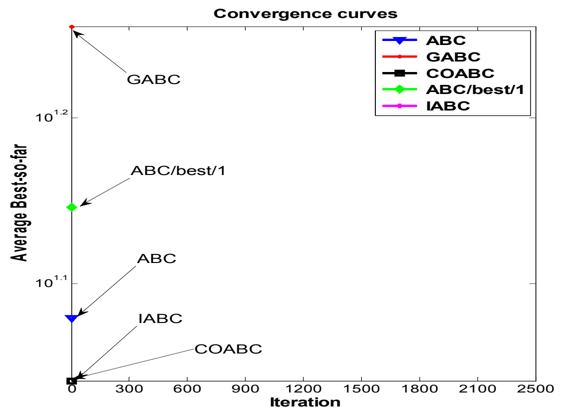

Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
ABC/best/1
GABC
COABC
ABC
IABC

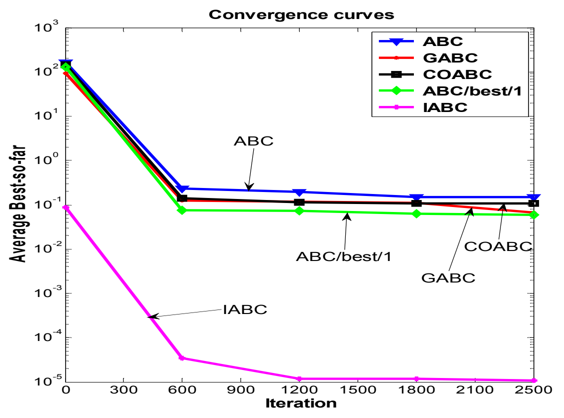

Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
ABC
IABC
GABC
ABC/best/1
COABC

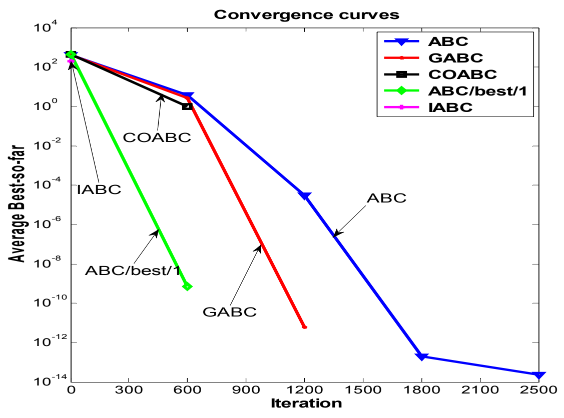

Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
ABC
IABC
ABC/best/1
COABC
GABC

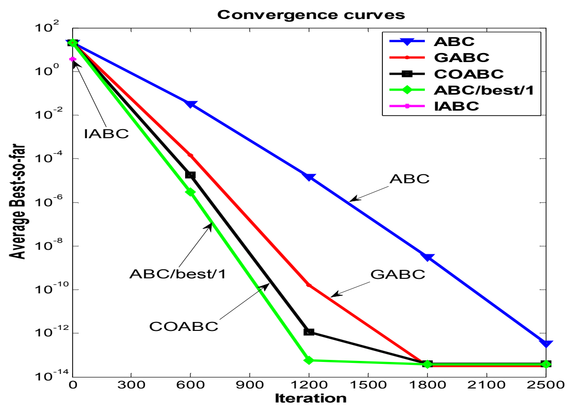

Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
ABC
IABC
COABC
GABC
ABC/best/1

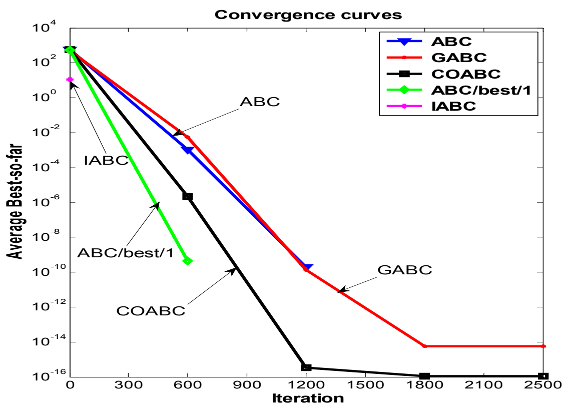

Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
10^{10}
10^{0}
10^{-10}
10^{-20}
10^{-30}
10^{-40}
10^{-50}
10^{-60}
0 300 600 900 1200 1500 1800 2100 2500

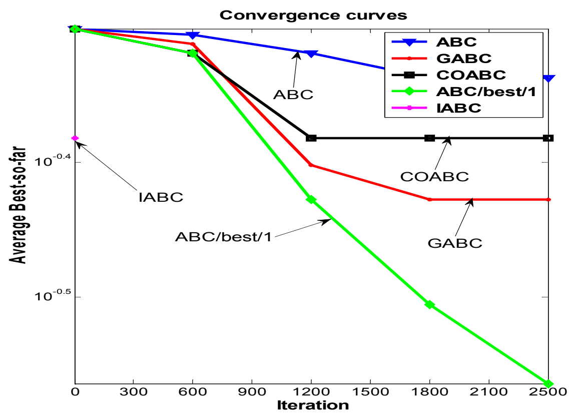

Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
$10^{1.2}$
$10^{1.1}$
0 300 600 900 1200 1500 1800 2100 2500

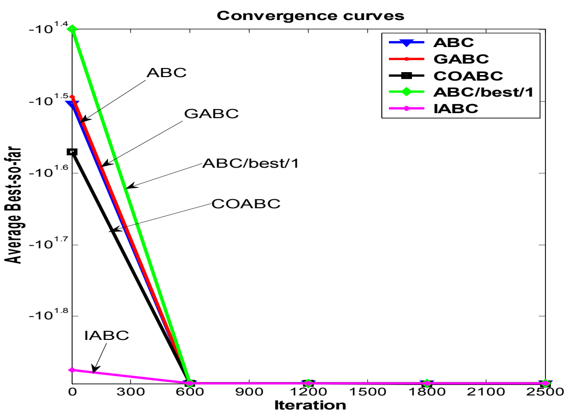

Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
10^{3}
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
0 300 600 900 1200 1500 1800 2100 2500


Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
10^{4}
10^{2}
10^{0}
10^{-2}
10^{-4}
10^{-6}
10^{-8}
10^{-10}
10^{-12}
10^{-14}
0 300 600 900 1200 1500 1800 2100 2500


Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
10^{2}
10^{0}
10^{-2}
10^{-4}
10^{-6}
10^{-8}
10^{-10}
10^{-12}
10^{-14}
0 300 600 900 1200 1500 1800 2100 2500


Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
10^{4}
10^{2}
10^{0}
10^{-2}
10^{-4}
10^{-6}
10^{-8}
10^{-10}
10^{-12}
10^{-14}
10^{-16}
0 300 600 900 1200 1500 1800 2100 2500


Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
$10^{-0.4}$
$10^{-0.5}$
0 300 600 900 1200 1500 1800 2100 2500


Convergence curves
ABC
GABC
COABC
ABC/best/1
IABC
Average Best-so-far
Iteration
$-10^{1.4}$
$-10^{1.5}$
$-10^{1.6}$
$-10^{1.7}$
$-10^{1.8}$
0 300 600 900 1200 1500 1800 2100 2500

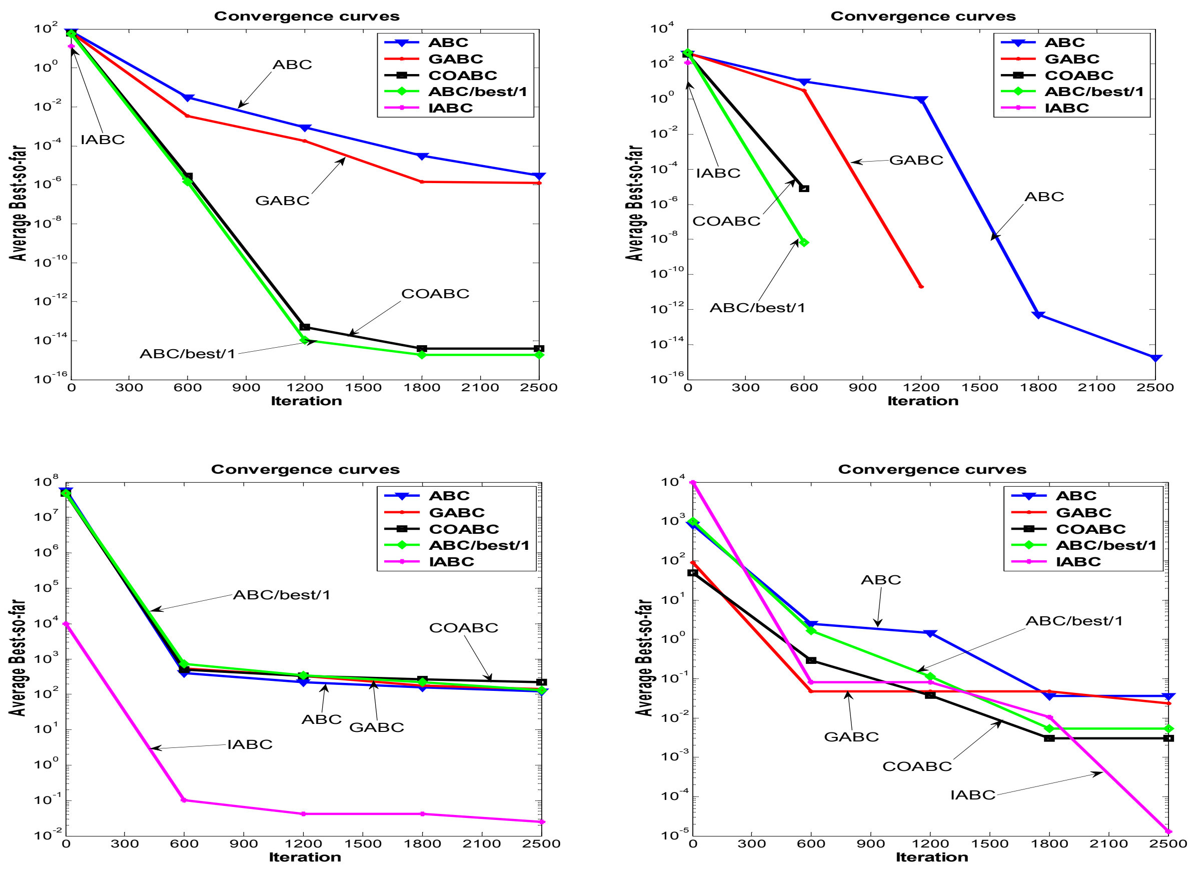


Fig. 1: Convergence Rates of Functions $f_1 - f_{20}$

V. Conclusion

In this paper, a new search equation for onlooker bees was proposed. After that, an improved artificial bee colony algorithm IABC was developed. The performance of IABC was compared with the standard ABC, and other three state-of-the-art algorithms GABC, COABC, ABC/best/1. The results showed that IABC presents promising results for considered problems.

In the future, the adaption of the parameters ω, c_1 and c_2 can be studied to improve the performance of IABC algorithm.

References

[1] D. Simon, Biogeography-based optimization, *IEEE Transactions on Evolutionary Computation*, 12:702-713,2008.

[2] X.S. Yang, A.H. Gandom, Bat algorithm: a novel approach for engineering oprimization, *Engineering Computations*, 29:464-483,2012.

[3] J. Bansal, H. Sharma, M. Clerc, Spider monkey optimization algorithm for numerical oprimization, *Memetic Computing*, 6:31-47, 2014.

[4] K. Deep, M. Thakur, A new mutation operator for real coded genetic algorithms, *Applied Mathematics and Computation*, 193:211-230, 2007.

[5] P. Kaelo, M.M. Ali, Integrated crossover rules in real coded genetic algorithms, *European Journal of Operational Research*, 176:60-76, 2007.

[6] R. Storn, K. Price, Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization*, 2:341-359, 1997.

[7] J. Kennedy, R. Eberhart, Particle swarm optimization, in *IEE Int. Conf. Neural Networks*, 1995.

[8] M. Dorigo, T. Stutzle, *Ant colony optimization*, MIT Press, Cambridege, MA,2004.

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

[10] Z. W. Geem, J.H. Kim, G.V. Loganathan, A new heuristic optimization algorithm:harmony search, *Simulation*, 76:60-68, 2001.

[11] S. A. Medjahed, T. A. Saadi, A. Benyettou, et al., Binary cuckoo search algorithm for band selection in hyperspectral image classification, *IAENG International Journal of Computer Science*, 42(3):183-191, 2015.

[12] H. Lu , K. Joarder, A modified immune network optimization algorithm, *IAENG International Journal of Computer Science*, 41(4):231-236, 2014.

[13] Z. Mustaffa, Y. Yusof , S.S. Kamaruddin, Enhanced artificial bee colony for training least squares support vector machines in commodity price forecasting, *Journal of Computational Science*, 5:196-205, 2014.

[14] Q.K. Pan, M.F. Tasgetiren,et al., "A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem," *Information Sciences*, 181:2455-2468,2011.

[15] V.J. Manoj, E. Elias, "Artificial bee colony algorithm for the design of multiplierless nonuniform filter bank transmultiplexer", *Information Sciences*, 192: 193-203,2012.

[16] D. Karaboga, B. Gorkemli, C. Ozturk, N. Karaboga, "A comprehensive survey: artificial bee colony (ABC) algorithm and applications," *Artificial Intelligence Review*, 42(1):21-57, 2012.

[17] G.P. Zhu, S. Kwong, "Gbest-guided artificial bee colony algorithm for numerical function optimization," *Applied Mathematics and Computation*, 217:3166-3173, 2010.

[18] M. Tuba, N. Bacanin, N. Stanarevic, "Guided artificial bee colony algorithm," in *Proceeding of the European Computing Conference(ECC11)*, 2011.

[19] W.F. Gao, S.Y. Liu, "A modified artificial bee colony algorithm," *Computers and Operations Research*, 39: 687-697, 2012.

[20] W.F. Gao, S.Y. Liu, L.L. Huang, "A global best artificial bee colony algorithm for global optimization," *Journal of Computational and Applied Mathematics*, 236:2741-2753, 2012.

[21] W. Bin, C.H. Qian, "Differential artificial bee colony algorithm for global numerical optimization," *Journal of Computers*, 6 (5):841-848, 2011.

TABLE II: IABC performance comparison with other four algorithms

Functions	ABC				GABC				COABC				ABC/best/1				IABC			
	Best	Worst	Mean	SD	Best	Worst	Mean	SD	Best	Worst	Mean	SD	Best	Worst	Mean	SD	Best	Worst	Mean	SD
f_1	2.48e-037	3.23e-036	1.36e-036	1.62e-036	1.53e-057	7.68e-057	4.10e-057	3.20e-057	9.81e-070	2.53e-069	1.96e-069	8.55e-070	2.85e-078	1.24e-076	4.54e-077	6.89e-077	0	0	0	0
f_2	8.21e-020	3.27e-019	1.72e-019	1.35e-019	1.09e-029	2.33e-029	1.70e-029	6.18e-030	8.46e-037	1.57e-036	1.15e-036	3.76e-037	1.32e-040	1.62e-040	1.44e-040	1.56e-041	0	0	0	0
f_3	3.41e+003	7.90e+003	5.66e+003	2.24e+003	4.55e+003	7.17e+003	6.04e+003	1.34e+003	4.28e+003	6.67e+003	5.77e+003	1.29e+003	5.91e+003	9.86e+003	7.66e+003	2.01e+003	0	0	0	0
f_4	2.98e+001	4.61e+001	3.60e+001	8.79e+000	1.33e+001	1.74e+001	1.53e+001	2.01e+000	1.53e+001	2.26e+001	1.87e+001	3.62e+000	8.98e+000	1.32e+001	1.08e+001	2.19e+000	0	0	0	0
f_6	1.67e-001	6.10e-001	3.73e-001	2.23e-001	2.84e-002	4.50e-001	2.29e-001	2.12e-001	7.40e-003	4.26e+000	1.54e+000	2.36e+000	1.15e-001	2.71e+001	1.01e+001	1.47e+001	3.56e-004	4.94e-004	4.17e-004	7.02e-005
f_7	2.30e-081	4.56e-079	1.62e-079	2.54e-079	8.03e-120	1.54e-118	9.64e-119	7.77e-119	2.46e-141	2.69e-139	9.36e-140	1.52e-139	2.09e-156	7.89e-155	2.82e-155	4.38e-155	0	0	0	0
f_8	1.89e-027	2.89e-023	9.65e-024	1.67e-023	7.36e-084	6.92e-074	2.30e-074	4.00e-074	2.07e-154	1.62e-143	5.56e-144	9.22e-144	1.32e-159	1.01e-151	3.39e-152	5.87e-152	0	0	0	0
f_9	9.72e-033	7.84e-032	3.66e-032	3.67e-032	6.95e-055	6.00e-054	2.95e-054	2.73e-054	2.66e-067	5.16e-066	1.94e-066	2.78e-066	5.45e-074	5.51e-073	3.15e-073	2.49e-073	0	0	0	0
f_{10}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
f_{11}	1.56e-001	2.51e-001	1.91e-001	5.23e-002	5.62e-002	9.02e-002	7.35e-002	1.69e-002	3.69e-002	8.55e-001	5.68e-002	2.54e-002	4.55e-002	7.05e-002	5.96e-002	1.28e-002	2.40e-006	8.93e-006	5.27e-006	3.33e-006
f_{12}	1.77e-015	6.64e-013	2.27e-013	3.78e-013	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
f_{13}	4.17e-014	6.30e-014	5.59e-014	1.23e-014	3.10e-014	4.17e-014	3.81e-014	6.15e-015	3.10e-014	3.81e-014	3.46e-014	3.55e-015	2.04e-014	3.10e-014	2.51e-014	5.42e-015	-8.81e-016	-8.81e-016	-8.81e-016	-8.81e-016
f_{14}	1.11e-016	4.94e-011	1.64e-011	2.85e-011	0	1.11e-016	3.70e-017	6.40e-017	0	2.22e-016	7.40e-017	1.28e-016	0	1.03e-013	3.43e-014	5.95e-014	0	0	0	0
f_{15}	4.41e-001	4.59e-001	4.53e-001	1.03e-002	3.45e-001	3.73e-001	3.64e-001	1.60e-002	3.73e-001	3.96e-001	3.88e-001	1.31e-002	2.72e-001	3.73e-001	3.19e-001	5.06e-002	0	0	0	0
f_{16}	-78.3323	-78.3323	-78.3323	0	-78.3323	-78.3323	-78.3323	0	-78.3323	-78.3323	-78.3323	0	-78.3323	-78.3323	-78.3323	0	-78.3323	-78.3323	-78.3323	0
f_{17}	3.72e-011	2.57e-007	1.17e-007	1.29e-007	5.88e-015	3.95e-010	1.33e-010	2.26e-010	2.17e-039	1.22e-015	4.07e-016	7.05e-016	7.87e-044	1.88e-015	7.03e-016	1.03e-015	0	0	0	0
f_{18}	0	8.88e-015	3.55e-015	4.69e-015	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
f_{19}	4.13e+003	5.65e+003	4.76e+003	7.94e+002	4.01e+003	6.92e+003	5.53e+003	1.46e+003	5.23e+003	5.70e+003	5.39e+003	2.66e+002	3.84e+003	4.82e+003	4.38e+003	4.93e+002	1.23e-004	8.67e-004	4.11e-004	3.99e-004
f_{20}	3.21e-001	7.66e-001	5.26e-001	2.24e-001	1.03e-002	7.88e+001	2.63e+001	4.54e+001	1.12e-001	3.01e+000	1.61e+000	1.45e+000	2.63e-002	5.39e+001	1.80e+001	3.10e+001	3.84e-006	9.39e-003	3.20e-003	5.36e-003

[22] T.K. Sharma, M. Pant, "Differential operators embedded artificial bee colonyalgorithm," *International Journal of Applied Evolutionary Computation*, 2 (3):1-14, 2011.

[23] T.J. Hsieh, H.F. Hsiao, W.C. Yeh, "Mining financial distress trend data usingpenalty guided support vector machines based on hybrid of particle swarmoptimization and artificial bee colony algorithm," *Neurocomputing*, 82 :196-206, 2012.

[24] A. Abraham, R.K. Jatoth, A. Rajasekhar, "Hybrid differential artificial bee colonyalgorithm," *Journal of Computional and Theoretical Nanoscience*, 9 (2):249-257, 2012.

[25] J. Luo, Q. Wang, X.H. Xiao, "A modified artificial bee colony algorithm based on converge-onlooker approach for global optimization," *Applied Mathematics and Computation*, 219:10253-10262, 2013.

[26] B. Alatas, "Chaotic bee colony algorithms for global numerical optimization," *Expert Systems with Applications*, 37:5682-5687,2010.

[27] S. Rahnamayan, et al., "Opposition-based differential evolution," *IEEE Transactions on Evolutionary Computation*, 12:64-79, 2008.

[28] P.N. Suganthan, N. Liang, J.J. Deb, et al., "Probelm definitions and evaluation criteria for the CEC 2005," *special sesson on real-parameter optimization*, Report no. KanGAL Report 2005.