



The University of  
**Nottingham**

UNITED KINGDOM · CHINA · MALAYSIA

Qin, Quande and Cheng, Shi and Zhang, Qingyu and Li, Li and Shi, Yuhui (2015) Artificial bee colony algorithm with time-varying strategy. *Discrete Dynamics in Nature and Society*, 2015 . pp. 1-17. ISSN 1607-887X

**Access from the University of Nottingham repository:**

<http://eprints.nottingham.ac.uk/47172/1/Artificial%20Bee%20Colony%20Algorithm%20with%20Time-Varying%20Strategy.pdf>

**Copyright and reuse:**

The Nottingham ePrints service makes this work by researchers of the University of Nottingham available open access under the following conditions.

This article is made available under the Creative Commons Attribution licence and may be reused according to the conditions of the licence. For more details see:  
<http://creativecommons.org/licenses/by/2.5/>

**A note on versions:**

The version presented here may differ from the published version or from the version of record. If you wish to cite this item you are advised to consult the publisher's version. Please see the repository url above for details on accessing the published version and note that access may require a subscription.

For more information, please contact [eprints@nottingham.ac.uk](mailto:eprints@nottingham.ac.uk)

## Research Article

# Artificial Bee Colony Algorithm with Time-Varying Strategy

Quande Qin,<sup>1,2</sup> Shi Cheng,<sup>3,4</sup> Qingyu Zhang,<sup>1,2</sup> Li Li,<sup>1</sup> and Yuhui Shi<sup>5</sup>

<sup>1</sup>Department of Management Science, College of Management, Shenzhen University, Shenzhen 518060, China

<sup>2</sup>Research Institute of Business Analytics & Supply Chain Management, Shenzhen University, Shenzhen 518060, China

<sup>3</sup>Division of Computer Science, University of Nottingham Ningbo China, Ningbo 315100, China

<sup>4</sup>International Doctoral Innovation Centre, University of Nottingham Ningbo China, Ningbo 315100, China

<sup>5</sup>Department of Electrical & Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China

Correspondence should be addressed to Shi Cheng; [shi.cheng@live.com](mailto:shi.cheng@live.com)

Received 29 October 2014; Revised 13 March 2015; Accepted 17 March 2015

Academic Editor: Ricardo López-Ruiz

Copyright © 2015 Quande Qin et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Artificial bee colony (ABC) is one of the newest additions to the class of swarm intelligence. ABC algorithm has been shown to be competitive with some other population-based algorithms. However, there is still an insufficiency that ABC is good at exploration but poor at exploitation. To make a proper balance between these two conflictive factors, this paper proposed a novel ABC variant with a time-varying strategy where the ratio between the number of employed bees and the number of onlooker bees varies with time. The linear and nonlinear time-varying strategies can be incorporated into the basic ABC algorithm, yielding ABC-LTVS and ABC-NTVS algorithms, respectively. The effects of the added parameters in the two new ABC algorithms are also studied through solving some representative benchmark functions. The proposed ABC algorithm is a simple and easy modification to the structure of the basic ABC algorithm. Moreover, the proposed approach is general and can be incorporated in other ABC variants. A set of 21 benchmark functions in 30 and 50 dimensions are utilized in the experimental studies. The experimental results show the effectiveness of the proposed time-varying strategy.

## 1. Introduction

Swarm intelligence concerns the design of intelligent optimization algorithms by taking inspirations from the collective behavior of social insects [1]. During the past decades, swarm intelligence has shown great success for solving complicated problems. The problems to be optimized by swarm intelligence algorithms do not need to be mathematically represented as continuous, convex, and/or differentiable functions; they can be represented in any form [2]. Artificial bee colony (ABC) algorithm developed by Karaboga in 2005 is a recent addition into this category [3–5].

ABC algorithm is inspired by the intelligent behavior of honey bees seeking quality food sources [3, 6–8]. In a short span of less than 10 years, ABC has already been demonstrated as a promising technique for solving global optimization problems [9, 10]. Numerical comparisons and

many engineering applications demonstrated that ABC could obtain good search results [11]. Due to its simplicity, flexibility, and outstanding performance, ABC has captured increasing interests of the swarm intelligence research community and been applied to many real-world areas, such as numerical optimization [6, 10], neural network training [7, 12], finance forecasting [13, 14], production scheduling [15–17], data clustering [18, 19], image segmentation [20, 21], service selection problem [22], and power system optimization [23].

However, like its counterpart population-based stochastic algorithms, it still has some inherent pitfalls [24–26]. The convergence speed of ABC is slower than those of the representative population-based stochastic algorithms, such as DE and PSO [27]. Moreover, ABC suffers from premature convergence while dealing with certain complicated problems [28]. Several ABC variants have been proposed with the aim of overcoming these pitfalls. These ABC variants

can be generally categorized into three groups: (1) first, the parameters of configuration tuning [29–31]; (2) second, the ABC algorithm hybridizing with other evolutionary optimization operators to enhance performance [10, 32, 33]; (3) third, the design of new learning strategy by modifying the search equation of the basic ABC algorithm [24, 25, 34].

It is recognized that exploration and exploitation are the two most important factors affecting a population-based optimization algorithm's performance [6, 35]. Exploration refers to the ability of a search algorithm to investigate the unknown regions in the search space in order to have high probability to discover good promising solutions. Exploitation, on the other hand, refers to the ability to concentrate the search around a promising region in order to refine a candidate solution [36, 37]. A good population-based optimization algorithm should properly balance these two conflictive objectives [38]. It is observed that ABC has good exploration ability but poor exploitation ability [24], which may impede ABC algorithm from proceeding towards a global optimum even though the population has not converged to a local optimum.

The colony of the ABC algorithm contains three groups of bees: employed bees, onlookers, and scouts [3]. In the employed bees phase of ABC algorithm, the algorithm focuses on the explorative search indicated by the solution updating scheme, which uses the current solution and a randomly chosen solution. A fitness-based probabilistic selection scheme is used in the onlooker phase, which indicates the exploitation tendency of the algorithm. In the original ABC algorithm, it is assumed that half of the colony consists of the employed bees, and the rest half consists of the onlookers [3]. In other words, the ratio of the employed and onlooker bees is the same, 1:1.

The basic ABC algorithm is easy to tune with few parameters but lacks effective and efficient ways to control the exploration ability and exploitation ability. In the present study, we propose an easy modification to the structure of the basic ABC algorithms in an attempt to balance their exploration and exploitation abilities. Compared with the basic ABC algorithm, we added a new parameter. The core idea of the modification is to design a proper mechanism, adjusting the ratio of the employed and onlooker bees with time. The proposed algorithms are called ABC with linear versus nonlinear time-varying strategy (ABC-LTVS and ABC-NTVS, resp.). The objective of this development is to enhance the global exploration in the early search and to encourage the solution to converge toward the global optima at the later search.

The remainder of this paper is organized as follows: Section 2 briefly introduces the basic ABC algorithm. The proposed algorithms, ABC-LTVS and ABC-NTVS, are elaborated in Section 3. In Section 4, comprehensive experimental studies are conducted on 21 benchmark functions in 30-dimension and 50-dimension problems to verify the effectiveness of the proposed algorithms. Finally, conclusions are presented in Section 5.

## 2. Basic ABC Algorithm

The ABC algorithm is a recently introduced optimization algorithm proposed by Karaboga [3, 39, 40], which is inspired by the intelligent foraging behavior of the honeybee swarm. In the ABC algorithm, there are two components: the foraging artificial bees and the food source [3, 39]. The position of a food source represents a possible solution. The nectar amount of a food source corresponds to the fitness of the associated solution. In the basic ABC algorithm, the colony of artificial bees contains three groups of bees: employed bees, onlookers, and scouts. The employed bees are responsible for searching available food sources and pass the food information to the onlooker bees [3, 40]. The onlookers select good sources from those found by the employed bees to further search the foods. When the fitness of the food sources is not improved through a predetermined number of cycles, denoted as *limit*, the food source is abandoned by its employed bee, and then, the employed bee becomes a scout and starts to search for a new food source in the vicinity of the hive.

The ABC algorithm consists of four phases: initialization, employed bees, onlooker bees, and scout bees [3, 39, 40]. In the initialization phase of the ABC, SN food source positions are randomly produced in the  $D$ -dimensional search space using the following equation [3, 6]:

$$v_{id} = l_d + r_1 (u_d - l_d), \quad (1)$$

where  $\vec{x}_i = [x_{i1}, x_{i2}, \dots, x_{iD}]$  is the  $i$ th food source,  $d \in \{1, 2, \dots, D\}$ ;  $i \in \{1, 2, \dots, \text{SN}\}$ , where SN denotes the number of food source;  $l_d$  and  $u_d$  are the lower and upper bounds for dimension  $d$ , respectively;  $r_1$  is a random number uniformly distributed within range  $[0, 1]$ .

After producing food sources and assigning them to the employed bees. There is only one employed bee on each food source. In the employed bee phase of ABC, each employed bee tries to find a better quality food source based on  $\vec{x}_i$ . A new trial food source, denoted as  $\vec{u}_i = [u_{i1}, u_{i2}, \dots, u_{iD}]$ , is calculated from the equation below [3, 6]:

$$u_{ij} = x_{ij} + \phi (x_{ij} - x_{kj}), \quad (2)$$

where  $j$  is a randomly generated whole number in the range  $[1, D]$ ;  $\phi$  is a random number uniformly distributed in the range  $[-1, 1]$ ; and  $k$  is the index of a randomly chosen food source satisfying  $k \neq i$ . After  $\vec{u}_i$  is obtained, it will be evaluated and compared to  $\vec{x}_i$ . If the fitness of  $\vec{u}_i$  is better than that of  $\vec{x}_i$ , the bee will forget the old food source  $\vec{u}_i$  and memorize the new one. Otherwise, it keeps  $\vec{x}_i$  in her memory. After all employed bees have finished their search, they share the nectar and position information of their food sources with the onlookers.

Each onlooker bee selects a food source of an employed bee to improve it. The roulette selection mechanism is performed by using (3) [3, 6]

$$P_i = \frac{\text{fit}(\vec{x}_i)}{\sum_{j=1}^{\text{SN}} \text{fit}(\vec{x}_j)}, \quad (3)$$

```

(1) Initialize the set of food sources  $\vec{x}_i, i = 1, 2, \dots, SN$ ;
(2) Evaluate each  $\vec{x}_i, i = 1, 2, \dots, SN$ ;
(3) while have not found "good enough" solution or not reached the predetermined
    maximum number of iterations do
(4)   for  $i = 1$  to  $SN$  do                                     /* Employed bees phase */
(5)     Generate  $\vec{u}_i$  with  $\vec{x}_i$  using (2);
(6)     Evaluate  $\vec{u}_i$ ;
(7)     if  $\text{fit}(\vec{u}_i) \geq \text{fit}(\vec{x}_i)$  then
(8)        $\vec{x}_i = \vec{u}_i$ ;
(9)   for  $i = 1$  to  $SN$  do                                     /* Onlooker bees phase */
(10)    Select an employed bee using (3);
(11)    Try to improve food source quality according to Steps (5)–(8);
(12)    Generate a new randomly food source for those does not improve with successive limit
        iterations                                     /* Scout bees phase */;
(13)    Memorize the best food source achieved so far;

```

ALGORITHM 1: The pseudocode of artificial bee colony algorithm.

where  $p_i$  is the probability of food source  $\vec{x}_i$  being selected by an onlooker bee and  $\text{fit}(\vec{x}_i)$  is the fitness of  $\vec{x}_i$ . The fitness of the food sources are defined as [3, 6]

$$\text{fit}(\vec{x}_i) = \begin{cases} \frac{1}{1 + f(\vec{x}_i)}, & f(\vec{x}_i) \geq 0, \\ 1 + |f(\vec{x}_i)|, & f(\vec{x}_i) < 0, \end{cases} \quad (4)$$

where  $f(\vec{x}_i)$  is the objective function value of  $\vec{x}_i$ . Once the onlooker has selected a food source, a new candidate food source can be obtained by (2). As in the employed bees, the greedy selection between these two food sources was performed.

If a food source,  $\vec{x}_i$ , cannot be improved for a predetermined number of cycles, referred to as *limit*, this food source is abandoned. Then, the scout produces a new food source randomly according to (1) to replace  $\vec{x}_i$ .

The detailed pseudocode of the ABC algorithm is shown in Algorithm 1 [3, 40].

### 3. The Proposed Algorithm

**3.1. Time-Varying Strategy.** During the process of the employed bees, each food source is assigned with one employed bee. The onlooker bees select the food source based on the fitness values, which is similar to "route wheel selection" in genetic algorithm. Due to this selection scheme, it is assured that the food sources with higher fitness value have more chance to be selected by the onlooker bees. It facilitates improving the quality of these food sources. Thus, the onlooker bees are more concentrated on the exploitation ability than the employed bees. In general, for population-based optimization algorithms, the exploration is typically preferred at the early stages of the search but is required to gradually give ways to exploitation in order to find the optimum solution efficiently [41]. In the ABC algorithm, the population size is fixed. With a large number of employed bees and relatively a small number of onlookers at the early search, it is helpful to move around the search space and enhance the exploration. On the other hand, the

small number of the employed bees and the large number of the onlookers allow the individuals to converge to the global optima in the later part of the optimization. In the basic ABC algorithm, the employed bees and onlooker bees are equal to each other; that is, the ratio between the number of the employed bees and the number of the onlookers is 1:1. Considering those concerns, we proposed a time-varying strategy, in which the ratio between the number of the employed bees and the number of the onlookers varies with time in this paper.

The total number of the colony size, employed bees, and onlooker denoted as  $NP$ ,  $NP_e$ , and  $NP_o$  respectively, and it holds that  $NP = NP_e + NP_o$ . The ratio between  $NP_e$  and  $NP_o$  is denoted as  $r_c$ ,  $r_c = NP_e/NP_o$ . A large value of  $r_c$  is conducive to global exploration. Conversely, it facilitates local exploitation for fine-tuning a local search. Intuitively, a linear-decreasing time-varying strategy (LTVS) may contribute to balance the exploration and exploitation during the entire search. A LTVS is proposed as the follows:

$$r_c = r_{\max} - (r_{\max} - r_{\min}) \times \frac{\text{fitc}}{\text{FEs}}, \quad (5)$$

where  $r_{\max}$  and  $r_{\min}$  are the maximum and minimum value of  $r_c$ , respectively;  $\text{fitc}$  is the current number of function evaluations; and  $\text{FEs}$  is the total number of function evaluations. The number of employed bees is set to the round value of  $r_c \times NP$ ,  $NP_e = \text{round}(r_c \times NP)$ , where  $\text{round}()$  is the round function with zero decimal place. If the number of employed bees  $NP_e$  is greater than the number of food source  $SN$ , each employed bee is assigned to one food source randomly in the first  $SN$  employed bees, and each of the rest of employed bees is placed at a randomly selected food source. When  $NP_e$  is smaller than  $SN$ , all employed bees randomly select food sources to search.

To investigate the dynamics of population distribution in basic ABC and ABC with LTVS, we herein take a typical multimodal function, Rastrigin function, with 2D dimension as an example. Figures 1 and 2 show the population distribution of Rastrigin function observed at various search phases of basic ABC and ABC with LTVS, respectively. It is

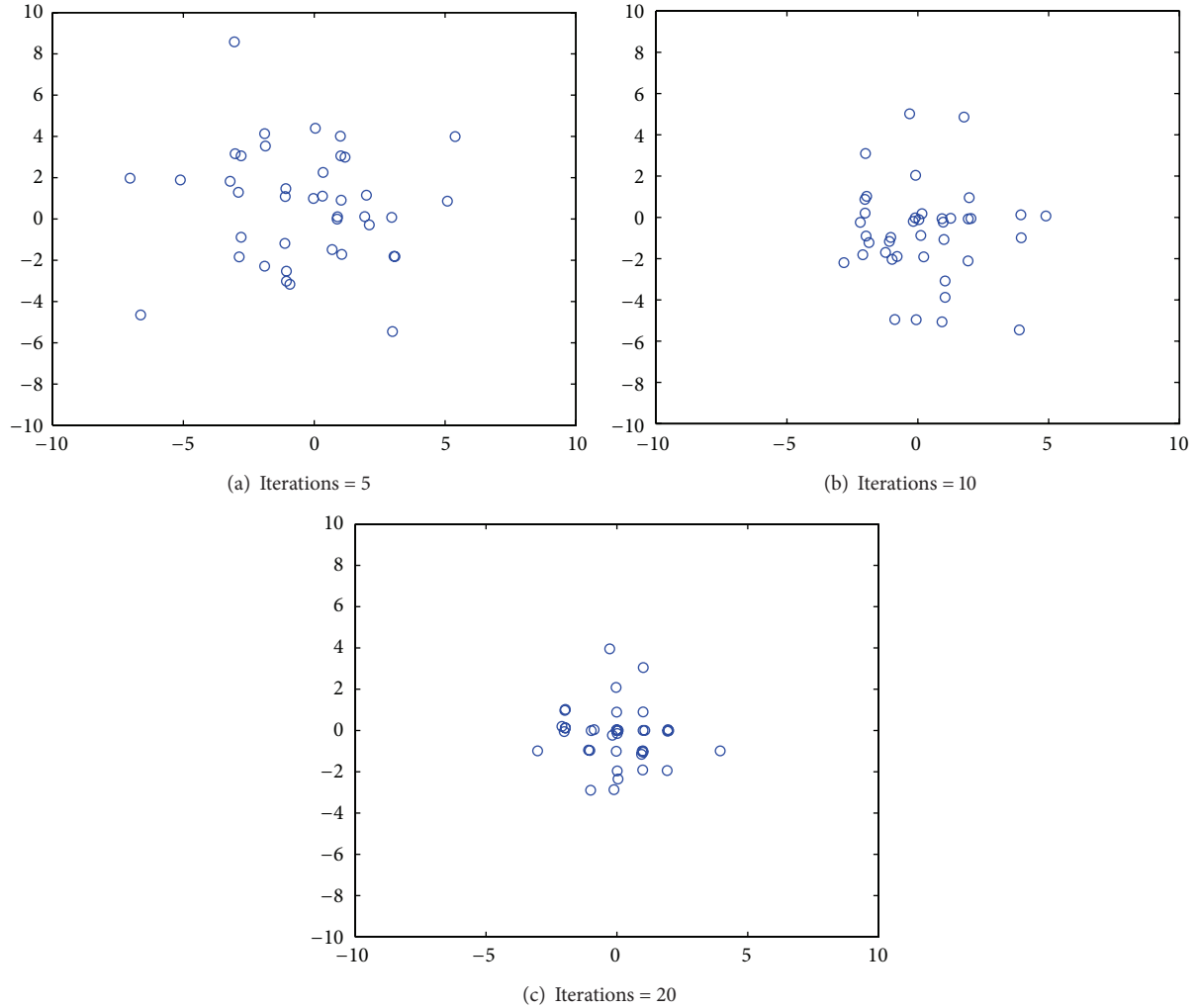


FIGURE 1: Population distribution observed at various stages in ABC.

noted that the value of  $r_{\max}$  and  $r_{\min}$  is set to 0.8 and 0.3 in ABC with LTVS in this experiment. Figures 1 and 2 show the population distribution at 5th, 10th, and 20th iteration when the 2-dimensional Rastrigin function was optimized by the original ABC and ABC with LTVS, respectively. Compared with the original ABC, we can obtain that the population in ABC with LTVS distributed in a wider range of search space at a relatively small iteration and gradually gathered around the global optimum. In other words, LTVS can explore in the search space in the early search stage, while converging to the global optimum with fast speed in the later search.

It would be interesting to know whether the nonlinear variation of  $r_c$  can enhance the performance of the ABC algorithm. In the preset study, we proposed a nonlinear-decreasing time-varying strategy (NTVS). This strategy is given as the following equation:

$$r_c = r_{\max} - (r_{\max} - r_{\min}) \times \left( \frac{\text{fitc}}{\text{FEs}} \right)^\alpha, \quad (6)$$

where  $\alpha$  is the nonlinear modulation index. Figure 3 shows typical variations of  $r_c$  with function evaluations for different

settings of  $\alpha$ . With  $\alpha = 1$ , this strategy becomes a special case of LTVS. With  $\alpha > 1$ ,  $r_c$  varied in a convex function manner. Compared with LTVS, it can be seen that  $r_c$  decreases in a relative-slow speed in the early search while in a faster speed in the later search. Conversely, with  $\alpha < 1$ ,  $r_c$  varied in a concave function manner.

**3.2. Parameters Tuning of Time-Varying Strategy.** In order to investigate the influence of  $r_{\max}$  and  $r_{\min}$  in LTVS, six combinations for setting the values of  $r_{\max}$  and  $r_{\min}$  are tested on 8 relevant functions including 3 typical unimodal functions,  $f_1$ ,  $f_2$ , and  $f_3$ , and 5 typical multimodal functions,  $f_6$ ,  $f_7$ ,  $f_8$ ,  $f_9$ , and  $f_{10}$ , with 30 dimensions. The colony size is set to 60, the maximum functions evaluations are  $7 \times 10^4$ , and the *limit* is set to 200 [42]. Experimental results of 25 independent trials are presented in Table 1. In Table 1, “Mean” indicates the mean of function values, “SD” stands for the standard deviation, and “Rank” refers to the performance level of the certain function in the six combinations of  $r_{\max}$  and  $r_{\min}$ . The best mean function value of six combinations is marked in boldface. From the rank results given in Table 2,

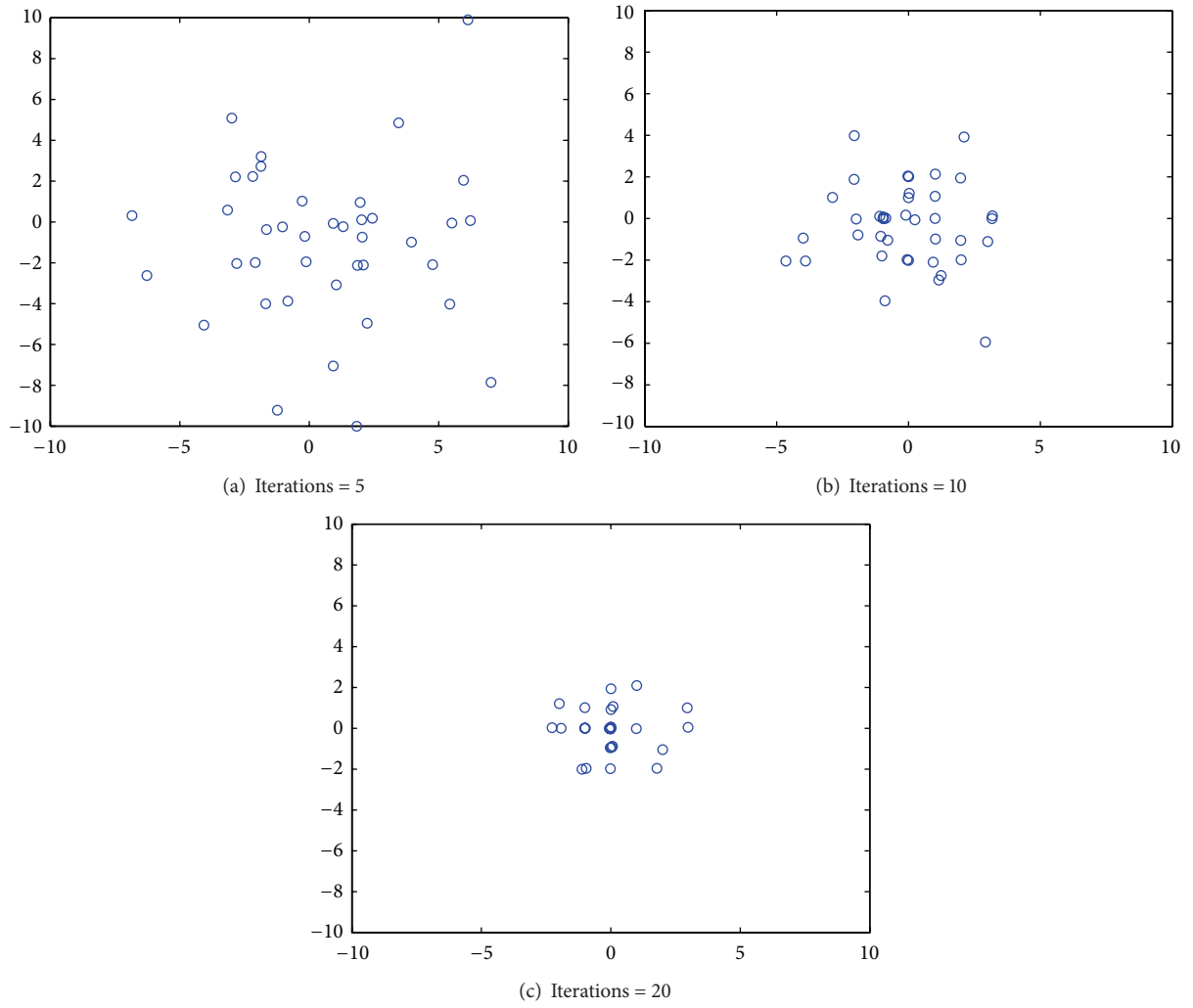


FIGURE 2: Population distribution observed at various stages in ABC with time-varying strategy.

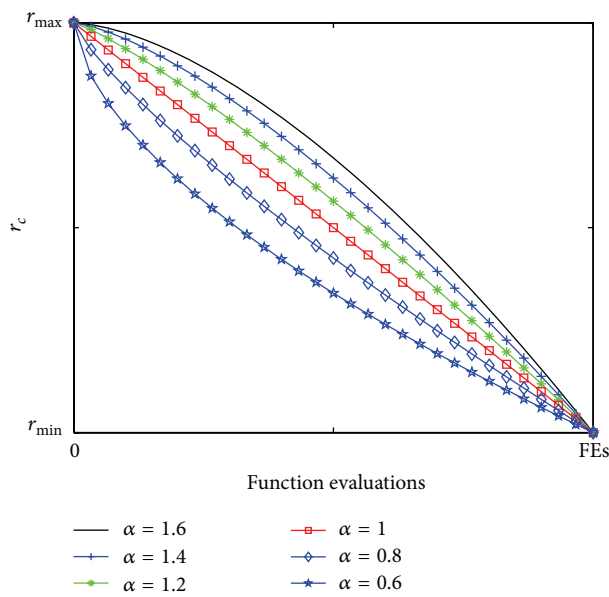


FIGURE 3: Variations of  $r_c$  with functions evaluations for different settings of  $\alpha$ .

TABLE 1: Experimental results of different combination of  $r_{\max}$  and  $r_{\min}$  in LTVS.

Parameter setting		$f_1$	$f_2$	$f_3$	$f_6$	$f_7$	$f_8$	$f_9$	$f_{10}$
$r_{\max} = 0.8$ $r_{\min} = 0.2$	Mean	<b>2.28e - 015</b>	<b>1.41e - 007</b>	<b>4.04e - 010</b>	1.3601	1.72e - 002	1.33e - 003	2.82e - 007	2.12e + 002
	SD	1.57e - 015	8.37e - 008	6.84e - 010	1.4594	8.59e - 002	4.63e - 003	2.04e - 007	8.10e + 001
	Rank	<b>1</b>	<b>1</b>	<b>1</b>	2	6	5	2	5
$r_{\max} = 0.8$ $r_{\min} = 0.3$	Mean	3.20e - 015	1.79e - 007	7.54e - 010	1.5393	6.82e - 003	6.85e - 007	3.24e - 007	2.10e + 002
	SD	2.59e - 015	7.97e - 008	1.41e - 009	1.7766	3.42e - 002	3.40e - 006	1.83e - 007	1.01e + 002
	Rank	2	2	2	5	5	2	4	4
$r_{\max} = 0.7$ $r_{\min} = 0.2$	Mean	1.34e - 014	3.38e - 007	7.28e - 009	1.3620	<b>1.73e - 006</b>	<b>6.48e - 010</b>	<b>2.62e - 007</b>	1.80e + 002
	SD	1.81e - 014	1.60e - 007	1.47e - 008	2.0555	5.34e - 006	1.72e - 009	1.39e - 007	7.84e + 001
	Rank	4	4	3	3	<b>1</b>	<b>1</b>	<b>1</b>	2
$r_{\max} = 0.7$ $r_{\min} = 0.3$	Mean	6.12e - 015	3.41e - 007	1.49e - 008	<b>1.2644</b>	1.93e - 003	3.02e - 004	3.35e - 007	2.19e + 002
	SD	4.41e - 015	1.64e - 007	4.59e - 008	1.1504	9.44e - 003	1.52e - 003	1.26e - 007	1.27e + 002
	Rank	3	3	4	<b>1</b>	4	3	5	6
$r_{\max} = 0.6$ $r_{\min} = 0.2$	Mean	4.86e - 014	6.36e - 007	8.76e - 008	1.4883	8.55e - 006	9.79e - 004	2.85e - 007	<b>1.67e + 002</b>
	SD	5.41e - 014	2.57e - 007	1.76e - 007	1.7028	3.02e - 005	2.77e - 003	1.51e - 007	1.02e + 002
	Rank	6	5	6	4	2	4	3	<b>1</b>
$r_{\max} = 0.6$ $r_{\min} = 0.3$	Mean	4.59e - 014	8.45e - 007	2.66e - 008	1.6454	1.33e - 003	1.43e - 003	4.17e - 007	1.83e + 002
	SD	3.49e - 015	4.49e - 007	5.57e - 008	1.6193	6.75e - 003	3.97e - 003	2.17e - 007	8.42e + 001
	Rank	5	6	5	6	3	6	6	3

TABLE 2: Rank results of the different combinations of  $r_{\max}$  and  $r_{\min}$  in LTVS.

	$r_{\max} = 0.8$ $r_{\min} = 0.2$	$r_{\max} = 0.8$ $r_{\min} = 0.3$	$r_{\max} = 0.7$ $r_{\min} = 0.2$	$r_{\max} = 0.7$ $r_{\min} = 0.3$	$r_{\max} = 0.6$ $r_{\min} = 0.2$	$r_{\max} = 0.6$ $r_{\min} = 0.3$
Average rank	2.875	3.25	2.375	3.625	3.875	5
Final rank	2	3	<b>1</b>	4	5	6

it shows that the settings of  $r_{\max} = 0.7$  and  $r_{\min} = 0.2$  are the best choice. Therefore, the parameters of  $r_{\max} = 0.7$  and  $r_{\min} = 0.2$  are used in LTVS. We try to obtain better performance with the NTVS of  $\alpha \neq 1$  having  $r_{\max}$  and  $r_{\min}$  kept fixed to their values obtained for the linear case. Table 3 shows the performance evaluation of the resultant system with different values of  $\alpha$ , keeping FEs =  $7 \times 10^4$ . From Tables 3 and 4, we can observe that the best result is obtained with  $\alpha = 1.2$ .

## 4. Experimental Study

**4.1. Benchmark Functions and Parameters Settings.** In order to test the proposed algorithm, a diverse set of 21 benchmark functions are used to conduct the experiments. These benchmark functions can be classified into three groups: Group 1, Group 2, and Group 3. The first five functions  $f_1$ - $f_5$  are unimodal functions in Group 1. The next group includes ten multimodal functions with many local optima which are used to test the global search capability in avoiding premature convergence. Note that  $f_6$  (Rosenbrock) is a unimodal problem in 2D or 3D search space but is a multimodal problem in

higher dimensions. Rotated and/or shifted functions belong to Group 3.  $f_{16}$  and  $f_{17}$  are rotated functions, in which the original variable  $\vec{x}$  is rotated by left multiplying the orthogonal matrix  $M$  [43],  $\vec{y} = M \times \vec{x}$ .  $M$  is used to increase the complexity of the function by changing separable functions to nonseparable functions without affecting the shape of the functions. The global optima of  $f_{18}$  and  $f_{19}$  are shifted to different numerical values for different dimensions ( $\vec{z} = \vec{x} - \vec{o}$ ), where  $\vec{o}$  is employed to shift the global optimal solution of the original function from the center of the search space to a new location.  $f_{20}$  and  $f_{21}$  are complicated which are shifted and rotated. We used  $\vec{x}^*$  to represent global optimum. In each benchmark function,  $\vec{x}^*$  and  $f(\vec{x}^*)$  represent global optimum and the corresponding function value, respectively. The function value of the best solution found by an algorithm in a run is denoted by  $f(\vec{x}_{\text{best}})$ . The error of this run is denoted as error =  $f(\vec{x}_{\text{best}}) - f(\vec{x}^*)$ . The parameters of all benchmark functions are described in Appendix.

To validate the effectiveness of the proposed time-varying strategy, experiments were conducted to compare ABC with ABC-LTVS and ABC-NTVS on 21 benchmark functions. In order to make a fair comparison, the colony size for all algorithms was set to 60. FEs are used as the stop criteria

TABLE 3: Experimental results of different setting of  $\alpha$  in NTVS.

Parameter setting	$f_1$	$f_2$	$f_3$	$f_6$	$f_7$	$f_8$	$f_9$	$f_{10}$
$\alpha = 1.6$								
Mean	$2.16e - 017$	$1.82e - 007$	$8.41e - 010$	1.3859	$1.49e - 006$	$1.73e - 003$	$3.10e - 007$	$2.03e + 002$
SD	$2.83e - 017$	$7.58e - 008$	$1.27e - 009$	1.1617	$6.63e - 006$	$4.82e - 003$	$1.15e - 007$	$1.06e + 002$
Rank	3	2	4	3	5	6	5	5
$\alpha = 1.4$								
Mean	$7.04e - 018$	$2.36e - 007$	<b><math>1.59e - 012</math></b>	1.6515	$8.40e - 010$	$6.48e - 004$	<b><math>4.78e - 009</math></b>	$1.41e + 002$
SD	$6.61e - 018$	$1.36e - 007$	$3.38e - 012$	1.6594	$2.33e - 009$	$2.35e - 003$	$2.83e - 009$	$1.05e + 002$
Rank	2	3	<b>1</b>	5	2	5	<b>1</b>	2
$\alpha = 1.2$								
Mean	<b><math>1.46e - 018</math></b>	<b><math>4.76e - 009</math></b>	$4.21e - 012$	1.5987	<b><math>6.25e - 010</math></b>	$1.01e - 005$	$5.58e - 009$	<b><math>9.60e + 001</math></b>
SD	$2.66e - 018$	$3.43e - 009$	$9.92e - 012$	1.4570	$1.61e - 009$	$4.69e - 005$	$2.94e - 009$	$8.84e + 001$
Rank	<b>1</b>	<b>1</b>	2	4	<b>1</b>	2	2	<b>1</b>
$\alpha = 1$								
Mean	$1.34e - 014$	$3.38e - 007$	$7.28e - 009$	1.3620	$1.73e - 006$	<b><math>6.48e - 010</math></b>	$2.62e - 007$	$1.80e + 002$
SD	$1.81e - 014$	$1.60e - 007$	$1.47e - 008$	2.0555	$5.34e - 006$	$1.72e - 009$	$1.39e - 007$	$7.84e + 001$
Rank	4	4	6	2	6	<b>1</b>	3	4
$\alpha = 0.8$								
Mean	$1.77e - 014$	$3.93e - 007$	$9.21e - 011$	1.0221	$1.75e - 007$	$3.633e - 004$	$2.68e - 007$	$2.09e + 002$
SD	$2.04e - 014$	$1.74e - 007$	$2.49e - 010$	1.0185	$4.57e - 007$	$1.82e - 003$	$1.47e - 007$	$9.24e + 001$
Rank	5	5	3	1	4	4	4	6
$\alpha = 0.6$								
Mean	$6.06e - 014$	$8.31e - 007$	$1.35e - 009$	1.9156	$1.90e - 008$	$3.14e - 004$	$3.63e - 007$	$1.45e + 002$
SD	$7.52e - 014$	$3.58e - 007$	$2.45e - 009$	2.0691	$7.87e - 007$	$1.64e - 003$	$2.17e - 007$	$9.87e + 001$
Rank	6	6	5	6	3	3	6	3

TABLE 4: Rank results of different setting of  $\alpha$  in NTVS.

	$\alpha = 1.6$	$\alpha = 1.4$	$\alpha = 1.2$	$\alpha = 1.0$	$\alpha = 0.8$	$\alpha = 0.6$
Average rank	4.125	2.625	1.75	3.75	4	4.75
Final rank	5	2	<b>1</b>	3	4	6

for all algorithms and it is set to be  $7 \times 10^4$  for solving 30D problems and  $1.2 \times 10^5$  for solving 50D problems, respectively. The parameter, *limit*, is set to 200 [42]. All experiments on each benchmark function were run 25 times independently.

**4.2. Experimental Results for 30D Problems.** The comparative results obtained by ABC, ABC-LTVS, and ABC-NTVS are presented in Table 5. The best mean results on each problem among all algorithms are given in bold. In order to determine whether the results obtained by ABC-LTVS and ABC-NTVS are statistically different from the results generated by ABC algorithms, a two-tailed *t*-test with 48 degrees of freedom is used at a significant level of 0.05. Values of “1,” “0,” and “-1” in columns “ $h_1$ ” and “ $h_2$ ” in Table 5, respectively, denote that ABC-LTVS and ABC-NTVS perform significantly better than, almost the same as, and significantly worse than ABC algorithm. In order to give visualized comparisons of the involved algorithms, the convergence graphs of the best and mean function values for each ABC algorithm regarding all

benchmark functions are shown in Figures 4, 5, and 6. In these figures, each curve represents the variation of mean value of error over the FEs for a specific ABC algorithm.

For solving unimodal functions, ABC-NTVS achieves the highest solution accuracy on  $f_1$ ,  $f_2$ ,  $f_3$ , and  $f_5$ , and ABC-LTVS obtains the best solution on  $f_4$ . ABC-LTVS and ABC-NTVS perform significantly better than ABC algorithm on  $f_1$ ,  $f_2$ , and  $f_3$ . On multimodal problems, there are many local minima and it is not easy to find the global optima. ABC-LTVS and ABC-NTVS are shown to offer better performance than ABC algorithm on these problems. ABC-LTVS can find the best solution on  $f_6$ ,  $f_8$ , and  $f_{13}$  and ABC-NTVS performs the best on  $f_7$ ,  $f_9$ ,  $f_{10}$ ,  $f_{11}$ , and  $f_{12}$ . ABC, ABC-LTVS, and ABC-NTVS obtain the similar performance on  $f_{14}$ . The performance of ABC-LTVS and ABC-NTVS are significantly better than that of ABC algorithm on these multimodal problems except ABC-NTVS for solving  $f_8$ .

For most of optimization algorithms, their performance will sharply decrease when solving the shifted and/or rotated



TABLE 5: Experimental results for 30-dimension problem.

Function	ABC		ABC-LTVS		ABC-NTVS		$h_1$	$h_2$
	Mean	SD	Mean	SD	Mean	SD		
$f_1$	$7.36e - 010$	$5.41e - 010$	$1.34e - 014$	$1.81e - 014$	<b>1.46e - 018</b>	$2.66e - 018$	1	1
$f_2$	$1.26e - 004$	$7.27e - 005$	$3.38e - 007$	$1.60e - 007$	<b>4.76e - 009</b>	$3.43e - 009$	1	1
$f_3$	$4.37e + 000$	$4.05e + 000$	$7.28e - 009$	$1.47e - 009$	<b>4.21e - 012</b>	$9.92e - 012$	1	1
$f_4$	0.1463	0.0309	<b>0.1385</b>	0.0307	0.1409	0.0412	0	0
$f_5$	$5.32e + 002$	$7.20e + 001$	$5.66e + 002$	$5.30e + 001$	<b>5.29e + 002</b>	$7.51e + 001$	0	0
$f_6$	3.4901	3.5885	<b>1.3620</b>	2.0555	1.5987	1.4570	1	1
$f_7$	$6.39e - 001$	$7.12e - 001$	$1.73e - 006$	$5.34e - 006$	<b>6.25e - 010</b>	$1.61e - 009$	1	1
$f_8$	$1.23e - 003$	$3.52e - 003$	<b>6.48e - 010</b>	$1.72e - 009$	$1.01e - 005$	$4.69e - 005$	0	0
$f_9$	$5.79e - 005$	$2.52e - 005$	$2.62e - 007$	$1.39e - 007$	<b>5.58e - 009</b>	$2.94e - 009$	1	1
$f_{10}$	$3.62e + 002$	$1.01e + 002$	$1.80e + 002$	$7.84e + 001$	<b>9.60e + 001</b>	$8.84e + 001$	1	1
$f_{11}$	1.3891	0.7813	0.1919	0.3751	<b>0.0421</b>	0.1998	1	1
$f_{12}$	$5.02e - 005$	$5.74e - 005$	$1.54e - 006$	$3.83e - 006$	<b>8.17e - 007</b>	$1.81e - 006$	1	1
$f_{13}$	$1.64e - 003$	$3.33e - 003$	<b>6.65e - 005</b>	$5.28e - 005$	$7.51e - 005$	$1.12e - 004$	1	1
$f_{14}$	$1.42e - 008$	$6.02e - 010$	<b>1.37e - 008</b>	0	<b>1.37e - 008</b>	0	1	1
$f_{15}$	$1.30e - 010$	$8.08e - 011$	$3.96e - 015$	$3.52e - 015$	<b>2.11e - 016</b>	$1.59e - 016$	1	1
$f_{16}$	47.5969	29.6407	45.9323	27.7604	<b>42.1560</b>	23.4833	0	0
$f_{17}$	80.7968	12.1854	80.0519	10.6073	<b>77.9708</b>	15.6648	0	0
$f_{18}$	6.5476	5.8738	<b>1.7831</b>	1.9028	2.2656	2.2663	1	1
$f_{19}$	0.6250	0.6287	0.0399	0.1990	<b>5.24e - 005</b>	$2.62e - 004$	1	1
$f_{20}$	<b>73.2093</b>	16.1789	76.5697	16.2155	73.4801	12.5874	0	0
$f_{21}$	$8.40e - 005$	$8.07e - 005$	<b>7.74e - 006</b>	$1.40e - 005$	$9.23e - 006$	$3.07e - 005$	1	1

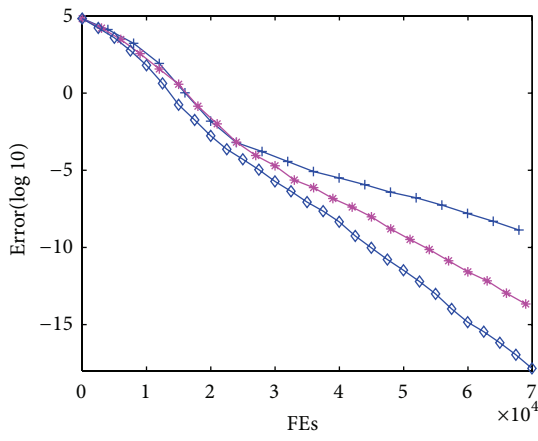
problems. Table 5 shows the results on the rotated and/or shifted functions. The results appear that these three ABC algorithms are affected. We can observe that ABC-LTVS and ABC-NTVS still obtain relatively good performance. ABC-LTVS and ABC-NTVS performs significantly better than ABC algorithms on  $f_{18}$ ,  $f_{19}$ , and  $f_{21}$ . With respect to the stability of algorithms, ABC-LTVS and ABC-NTVS show the good stability as compared to ABC algorithms. The standard deviations of solutions found by ABC-LTVS and ABC-NTVS are small for the most functions. On the whole, ABC-LTVS and ABC-NTVS exhibit the accurate convergence precision on almost all the benchmark, which indicates the effectiveness of the proposed time-varying strategy. Moreover, experimental results demonstrate that ABC-NTVS slightly outperforms ABC-LTVS.

**4.3. Experimental Results for 50D Problems.** The experiments conducted on 50D problems and the results for solving unimodal, multimodal, and shift and/or rotate problems are presented in Table 6. As the convergence graphs of 50D are similar to the 30D problems and space limitation, they are not given. Compared with ABC algorithm, ABC-LTVS and ABC-NTVS can still obtain high-quality solutions under 50D problems, which can be seen from Table 6. The meaning of column " $h_1$ " and " $h_2$ " in Table 6 is the same as the Table 5. It is noted that ABC-NTVS, from the mean of the results, performs worse than ABC on  $f_8$ , but not statistically significantly. According the  $t$ -tests results, ABC-LTVS performs significantly better than ABC algorithm on

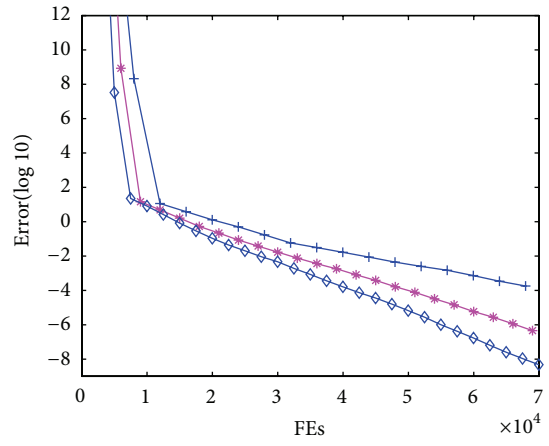
all benchmark functions except  $f_4$ ,  $f_{16}$ , and  $f_{17}$ , so does ABC-NTVS except  $f_4$ ,  $f_8$ , and  $f_{17}$ .

The experiments conducted on 50D problems and the results for solving unimodal, multimodal, and shift and/or rotate problems are presented in Table 6. As the convergence graphs of 50D are similar to the 30D problems and space limitation, they are not given. Compared with ABC algorithm, ABC-LTVS and ABC-NTVS can still obtain high-quality solutions under 50D problems, which can be seen from Table 6. The meaning of column " $h_1$ " and " $h_2$ " in Table 6 is the same as the Table 5. According to the  $t$ -tests results, ABC-LTVS performs significantly better than ABC algorithm on all benchmark functions except  $f_5$ ,  $f_{16}$ , and  $f_{17}$ , so does ABC-NTVS except  $f_5$ ,  $f_8$ , and  $f_{17}$ .

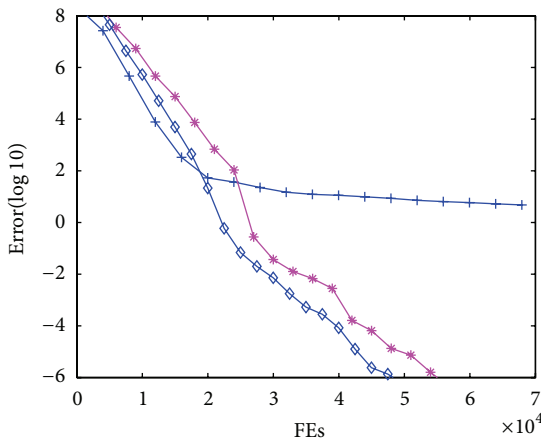
**4.4. GABC with Time-Varying Strategy.** Inspired by particle swarm optimization (PSO), Zhu and Kwong [24] proposed a popular ABC variant, called Gbest-guided artificial bee colony (GABC). GABC incorporates the information of global best (gbest) position into (2). The experimental results have shown that GABC algorithm outperforms the basic ABC algorithm. To test the effect of the time-varying strategy, we applied the proposed LTVS and NTVS into GABC algorithm, yielding GABC-LTVS and GABC-NTVS algorithms, respectively. Experiments are conducted on 30D and 50D benchmark functions to test whether the proposed time-varying strategy is effective in GABC algorithm. The parameters setting of GABC algorithm is in accordance with the original reference except the colony



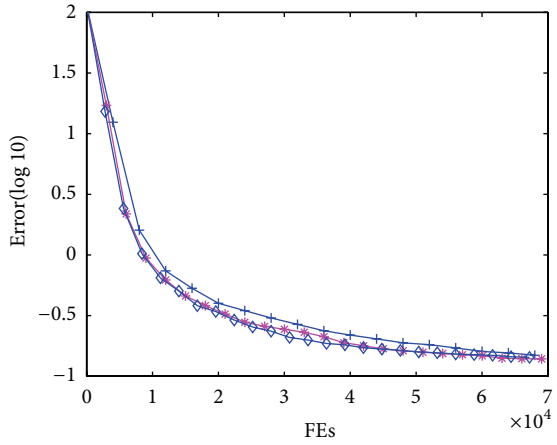
(a) Sphere  $f_1$



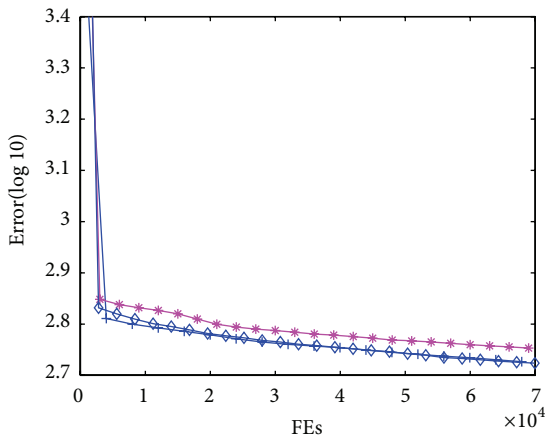
(b) Schwefel's P2.22  $f_2$



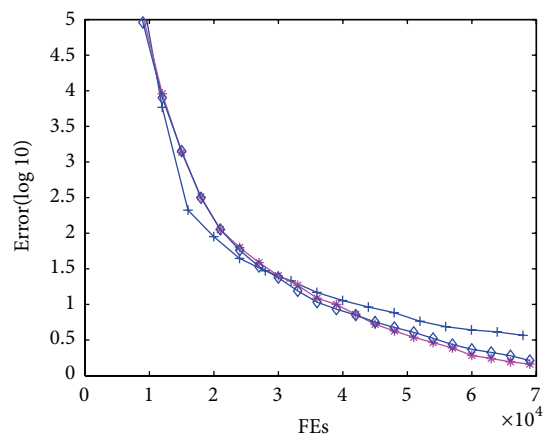
(c) Elliptic  $f_3$



(d) Noise  $f_4$



(e) Zakharov  $f_5$



(f) Rosenbrock  $f_6$

—+— ABC  
 —\*— ABC-LTVS  
 —◇— ABC-NTVS

—+— ABC  
 —\*— ABC-LTVS  
 —◇— ABC-NTVS

FIGURE 4: Convergence curves of ABC variants solving unimodal functions  $f_1 - f_5$  and multimodal function Rosenbrock  $f_6$ .

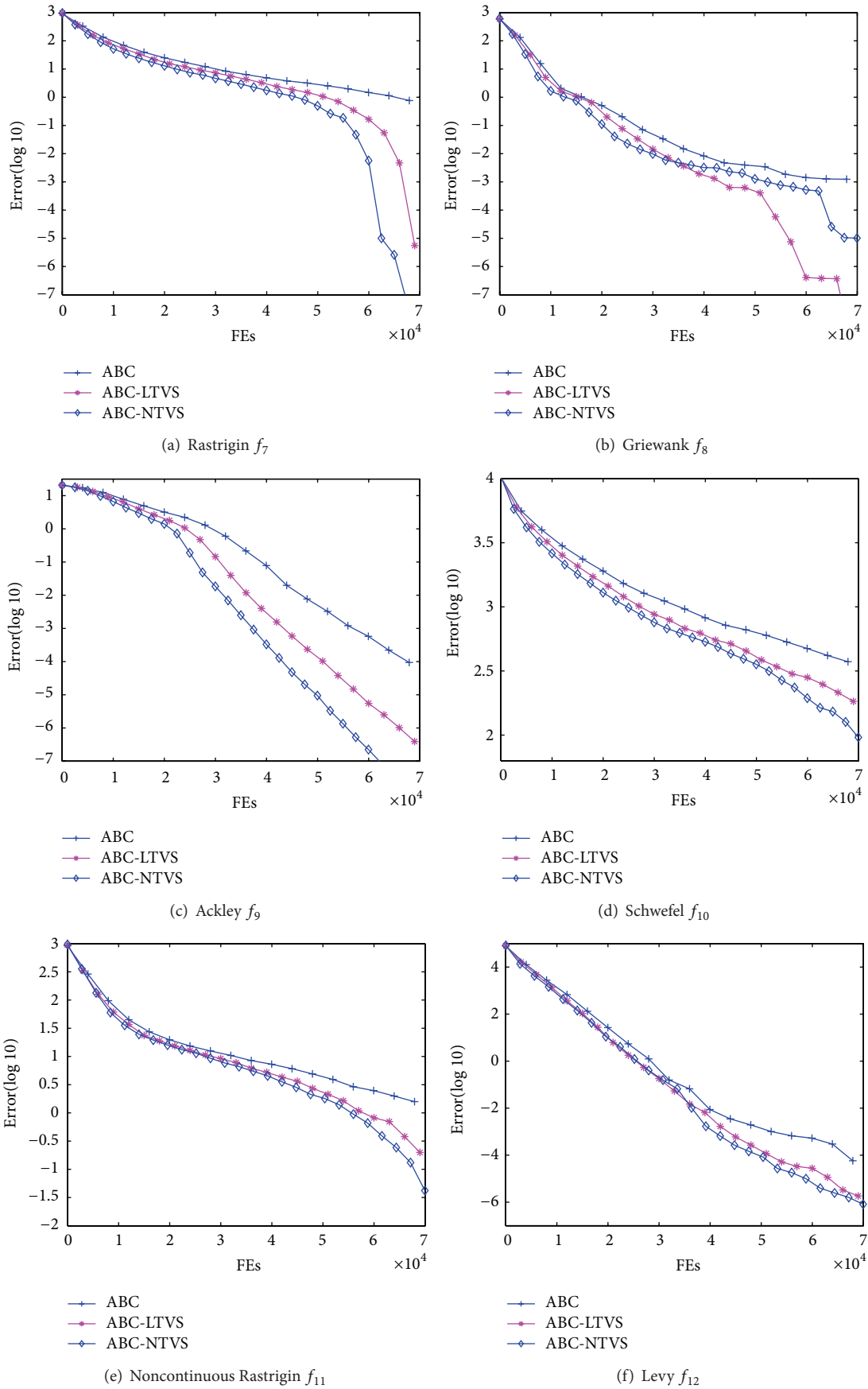


FIGURE 5: Continued.

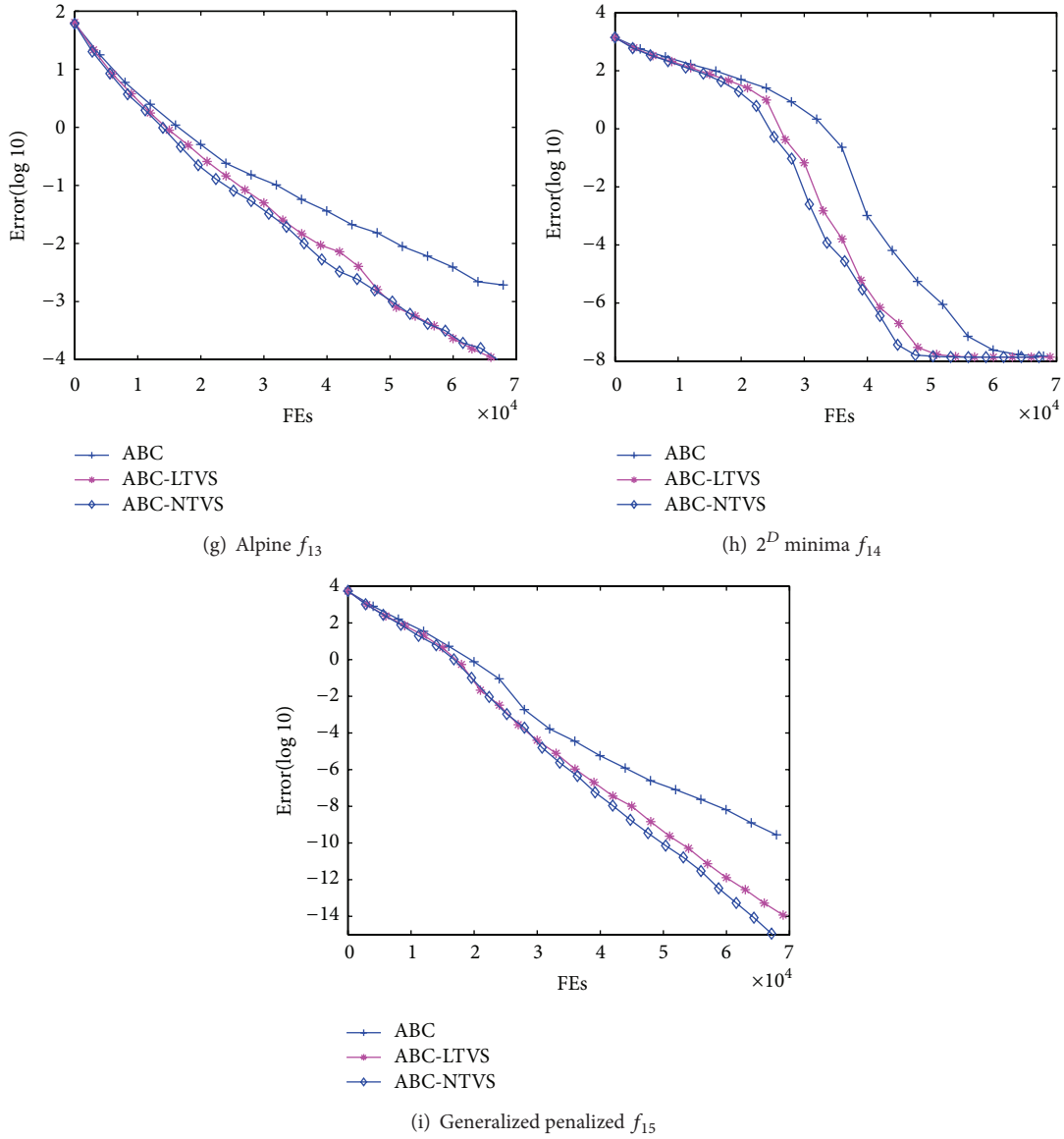


FIGURE 5: Convergence curves of ABC variants solving multimodal functions  $f_7$ - $f_{15}$ .

size and FEs, the settings of which is same as the previous experiment.

Tables 7 and 8 give the experimental results on 30D and 50D problems, respectively. The best results on each problem among these three GABC algorithms are shown in bold. In addition, the columns " $h_1$ " and " $h_2$ " in Tables 7 and 8 are used to determine the statistical significance of the difference obtained by GABC with those yielded by the GABC-LTVS and GABC-NTVS using two-tailed  $t$ -tests, respectively. According to the results of  $t$ -tests shown in Tables 7 and 8, GABC-LTVS and GABC-NTVS can find more accurate solutions, which are significantly better than those of GABC on about half of all benchmark functions regardless of problem dimensions.

## 5. Conclusions

In order to make a balance between the exploration and exploitation in ABC algorithm, a time-varying strategy has been developed. The proposed strategy is implemented through making the ratio between the number of employed bees and the numbers of onlooker bees vary with time. We have developed the two types of time-varying strategies of LTVS and NTVS. Moreover we have examined and fine-tuned the parameters settings of LTVS and NTVS for better performance. Comprehensive experiments have demonstrated the effectiveness of the proposed time-varying strategy in ABC and GABC algorithms. The modifications proposed in the present work are general enough to be

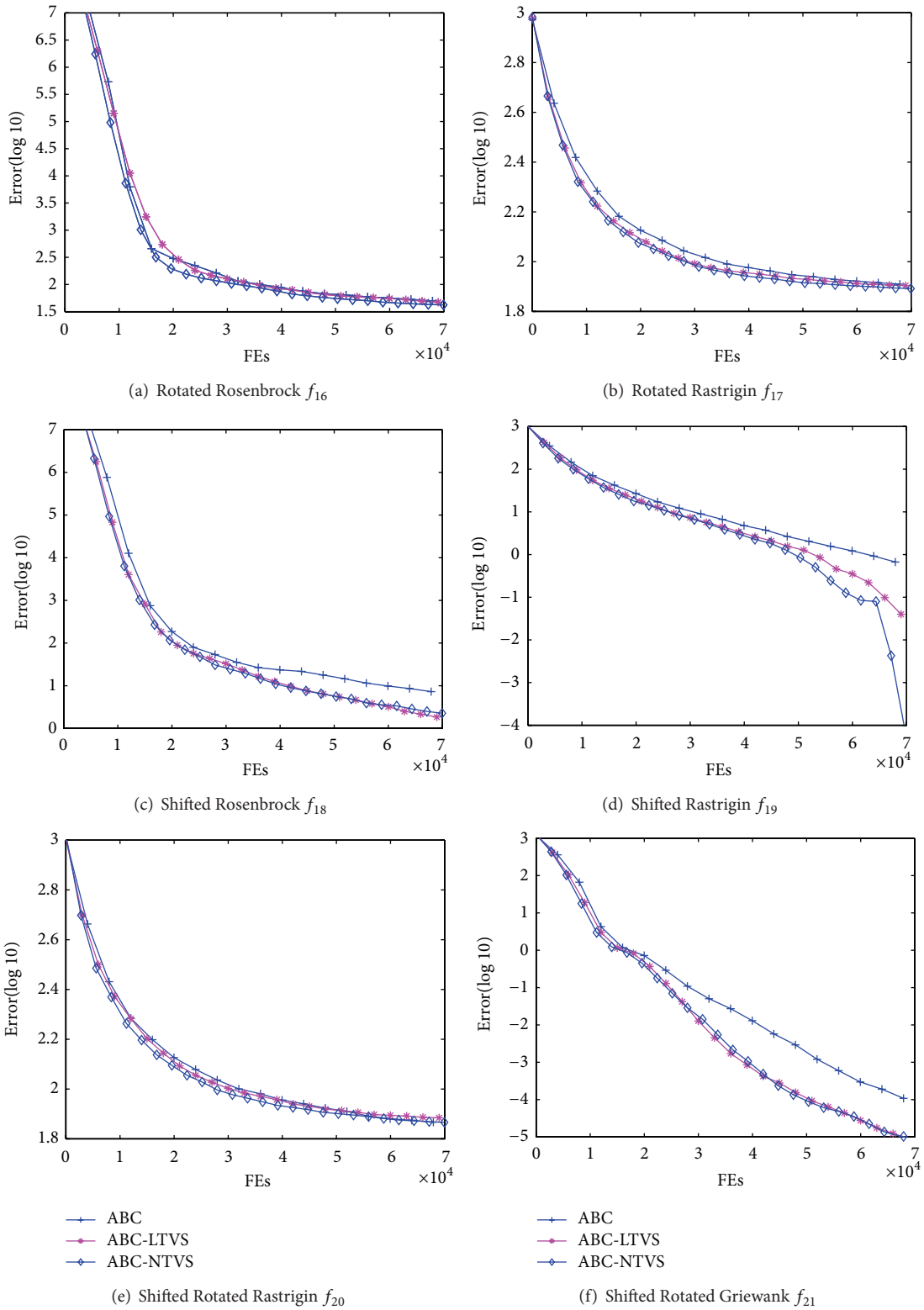


FIGURE 6: Convergence curves of ABC variants solving rotated and/or shifted functions  $f_{16}-f_{21}$ .

TABLE 6: Experimental results for 50-dimension problem.

Function	ABC		ABC-LTVS		ABC-NTVS		$h_1$	$h_2$
	Mean	SD	Mean	SD	Mean	SD		
$f_1$	$6.88e - 010$	$3.54e - 010$	$9.35e - 015$	$7.24e - 015$	<b>5.28e - 016</b>	$5.79e - 016$	1	1
$f_2$	$1.59e - 004$	$9.94e - 005$	$4.22e - 007$	$2.01e - 007$	<b>7.14e - 008</b>	$2.67e - 008$	1	1
$f_3$	$4.57e - 005$	$6.75e - 005$	$8.82e - 008$	$3.58e - 007$	<b>3.51e - 010</b>	$4.42e - 010$	1	1
$f_4$	0.3606	0.0579	<b>0.3132</b>	0.0691	0.3214	0.0657	1	1
$f_5$	$1.11e + 003$	$1.09e + 002$	$1.09e + 003$	$9.64e + 001$	<b>1.07e + 003</b>	$8.08e + 001$	0	0
$f_6$	8.2319	6.6280	<b>1.3564</b>	1.8842	1.3761	1.8716	1	1
$f_7$	1.0794	0.7609	0.0979	0.2829	<b>0.0865</b>	0.2755	1	1
$f_8$	$2.45e - 008$	$3.16e - 008$	<b>1.42e - 009</b>	$5.88e - 009$	$3.23e - 004$	$1.54e - 003$	1	0
$f_9$	$4.78e - 005$	$1.49e - 005$	$2.25e - 007$	$9.33e - 008$	<b>5.01e - 008</b>	$2.47e - 008$	1	1
$f_{10}$	$7.83e + 002$	$1.75e + 002$	$3.97e + 002$	$9.89e + 001$	<b>3.45e + 002</b>	$1.78e + 002$	1	1
$f_{11}$	2.7273	0.9297	0.3388	0.4789	<b>0.3029</b>	0.4551	1	1
$f_{12}$	$4.16e - 005$	$7.77e - 005$	$9.34e - 007$	$2.42e - 006$	<b>2.55e - 007</b>	$6.02e - 007$	1	1
$f_{13}$	$4.52e - 003$	$4.91e - 003$	$5.76e - 004$	$7.16e - 004$	<b>4.73e - 004</b>	$5.83e - 004$	1	1
$f_{14}$	$2.37e - 008$	$1.33e - 009$	<b>2.29e - 008</b>	$1.40e - 012$	<b>2.29e - 008</b>	$1.07e - 012$	1	1
$f_{15}$	$1.57e - 010$	$1.41e - 010$	$4.61e - 015$	$4.07e - 015$	<b>3.53e - 016</b>	$4.77e - 016$	1	1
$f_{16}$	95.9988	40.0144	82.0768	32.2224	<b>72.1410</b>	26.1076	0	1
$f_{17}$	142.9558	23.9951	143.1200	18.8582	<b>142.5459</b>	20.4493	0	0
$f_{18}$	5.9388	4.7166	<b>1.6944</b>	1.7353	1.9040	1.8484	1	1
$f_{19}$	1.5057	0.8212	0.1243	0.3400	<b>0.0400</b>	0.1990	1	1
$f_{20}$	152.5097	18.9577	<b>141.0244</b>	20.0979	146.2151	17.2185	1	0
$f_{21}$	$1.30e - 004$	$2.39e - 004$	$6.84e - 006$	$8.96e - 006$	<b>4.41e - 006</b>	$5.53e - 006$	1	1

TABLE 7: Experimental results for 30D problem of GABC algorithms.

Function	GABC		GABC-LTVS		GABC-NTVS		$h_1$	$h_2$
	Mean	SD	Mean	SD	Mean	SD		
$f_1$	$7.03e - 016$	$1.29e - 016$	$1.16e - 025$	$9.49e - 026$	<b>1.71e - 027</b>	$1.21e - 027$	1	1
$f_2$	$6.21e - 009$	$2.30e - 008$	$5.73e - 013$	$2.18e - 013$	<b>6.69e - 014</b>	$3.64e - 014$	1	1
$f_3$	$1.34e - 012$	$1.07e - 012$	$2.76e - 021$	$3.38e - 021$	<b>1.26e - 023</b>	$1.711e - 023$	1	1
$f_4$	0.0692	0.0188	0.0652	0.0144	<b>0.0602</b>	0.0155	0	0
$f_5$	$5.23e + 002$	$6.31e + 001$	$5.20e + 002$	$7.81e + 001$	<b>5.08e + 002</b>	$6.66e + 001$	0	0
$f_6$	6.4906	10.3348	2.7541	3.9329	<b>2.5028</b>	2.4833	0	0
$f_7$	$8.62e - 011$	$1.58e - 010$	$7.11e - 017$	$3.55e - 016$	<b>0</b>	0	1	1
$f_8$	$7.89e - 004$	$3.87e - 003$	<b>4.78e - 012</b>	$2.13e - 011$	$1.03e - 005$	$5.16e - 005$	0	0
$f_9$	$2.07e - 009$	$7.48e - 010$	$9.53e - 013$	$3.34e - 013$	<b>2.01e - 013</b>	$4.74e - 014$	1	1
$f_{10}$	59.7836	70.9355	<b>23.8017</b>	59.1745	25.5825	48.3538	0	0
$f_{11}$	$3.76e - 009$	$1.02e - 008$	$6.11e - 015$	$2.12e - 014$	<b>1.42e - 016</b>	$4.92e - 016$	0	0
$f_{12}$	$5.71e - 010$	$1.38e - 009$	$1.51e - 013$	$6.85e - 013$	<b>4.58e - 016</b>	$8.36e - 016$	1	1
$f_{13}$	$2.58e - 005$	$3.20e - 005$	$1.11e - 005$	$1.82e - 005$	<b>8.89e - 006</b>	$2.03e - 005$	0	1
$f_{14}$	<b>1.37e - 008</b>	$2.23e - 013$	<b>1.37e - 008</b>	$2.32e - 013$	<b>1.37e - 008</b>	$2.17e - 013$	0	0
$f_{15}$	$6.39e - 016$	$1.15e - 016$	$4.72e - 026$	$4.33e - 026$	<b>6.54e - 028</b>	$6.60e - 028$	1	1
$f_{16}$	<b>38.0229</b>	23.1492	41.9666	26.0778	38.1041	23.7575	0	0
$f_{17}$	55.5550	8.2793	53.8867	9.6732	<b>51.0331</b>	9.6130	0	0
$f_{18}$	7.4043	14.2994	3.7490	4.2342	<b>2.4815</b>	2.9901	0	1
$f_{19}$	$3.24e - 008$	$1.59e - 007$	$4.26e - 016$	$1.18e - 015$	<b>7.11e - 017</b>	$3.55e - 016$	1	1
$f_{20}$	51.3741	7.0974	<b>50.2909</b>	11.1007	54.0969	10.6401	0	0
$f_{21}$	$3.97e - 004$	$2.04e - 003$	<b>3.68e - 007</b>	$6.51e - 007$	$1.11e - 006$	$4.09e - 006$	1	1

TABLE 8: Experimental results for 50D problem of GABC algorithms.

Function	GABC		GABC-LTVS		GABC-NTVS		$h_1$	$h_2$
	Mean	SD	Mean	SD	Mean	SD		
$f_1$	$1.29e - 015$	$1.40e - 016$	$1.67e - 025$	$1.44e - 025$	<b><math>1.79e - 027</math></b>	$1.64e - 027$	1	1
$f_2$	$7.72e - 009$	$2.30e - 009$	$6.28e - 013$	$1.76e - 013$	<b><math>6.54e - 014</math></b>	$2.14e - 014$	1	1
$f_3$	$3.09e - 012$	$2.46e - 012$	$6.32e - 021$	$8.53e - 021$	<b><math>1.02e - 023</math></b>	$8.18e - 024$	1	1
$f_4$	0.1694	0.0252	0.1647	0.0294	<b>0.1627</b>	0.0296	0	0
$f_5$	$1.04e + 003$	$7.26e + 001$	$1.05e + 003$	$9.29e + 001$	<b><math>1.02e + 003</math></b>	$1.05e + 002$	0	0
$f_6$	12.3236	23.0585	<b>1.7470</b>	1.9037	3.3437	6.2364	1	0
$f_7$	$3.12e - 008$	$1.53e - 007$	$7.11e - 017$	$3.55e - 016$	<b>0</b>	0	0	0
$f_8$	$9.09e - 013$	$2.77e - 012$	$8.41e - 015$	$4.18e - 014$	<b><math>2.68e - 015</math></b>	$1.33e - 014$	0	0
$f_9$	$2.52e - 009$	$9.94e - 010$	$1.04e - 012$	$3.32e - 013$	<b><math>2.39e - 013</math></b>	$5.46e - 014$	1	1
$f_{10}$	162.7082	100.3839	<b>52.5115</b>	90.8550	80.6078	101.0140	1	1
$f_{11}$	$4.99e - 008$	$9.96e - 008$	$1.93e - 014$	$6.27e - 014$	<b><math>1.99e - 015</math></b>	$8.51e - 015$	1	1
$f_{12}$	$3.61e - 010$	$5.26e - 010$	$1.26e - 014$	$2.24e - 014$	<b><math>3.51e - 016</math></b>	$5.18e - 016$	1	1
$f_{13}$	$1.71e - 004$	$2.75e - 004$	$4.09e - 005$	$5.81e - 005$	<b><math>2.02e - 005</math></b>	$3.10e - 005$	1	1
$f_{14}$	<b><math>2.29e - 008</math></b>	$2.07e - 013$	<b><math>2.29e - 008</math></b>	$2.24e - 013$	<b><math>2.29e - 008</math></b>	$1.51e - 013$	0	0
$f_{15}$	$1.17e - 015$	$1.73e - 016$	$9.38e - 026$	$1.05e - 025$	<b><math>7.58e - 028</math></b>	$4.85e - 028$	1	1
$f_{16}$	79.5737	43.0285	87.9010	38.3788	<b>74.1063</b>	42.5826	0	0
$f_{17}$	101.0561	19.2130	<b>97.5846</b>	14.9214	98.3341	12.7543	0	0
$f_{18}$	6.7201	10.6406	<b>3.1402</b>	4.8951	4.1251	12.6738	1	0
$f_{19}$	$2.22e - 010$	$4.99e - 010$	$4.89e - 013$	$2.40e - 012$	<b>0</b>	0	1	1
$f_{20}$	107.1083	13.3490	106.0348	15.2274	<b>104.4232</b>	13.6487	0	0
$f_{21}$	$1.32e - 005$	$4.76e - 005$	<b><math>1.94e - 007</math></b>	$2.30e - 007$	$1.57e - 006$	$4.50e - 006$	0	0

applied in other state-of-the-art ABC variants to further improve the search performance.

## Appendix

### A. Benchmark Functions

#### A.1. Unimodal Functions

(1) Sphere function

$$f_1(\vec{x}) = \sum_{i=1}^D x_i^2, \quad -100 \leq x_i \leq 100, \quad \vec{x}^* = \{0\}^D. \quad (\text{A.1})$$

(2) Schwefel's function P2.22

$$f_2(\vec{x}) = \sum_{i=1}^D |x_i| + \prod_{i=1}^D |x_i|, \quad -100 \leq x_i \leq 100, \quad \vec{x}^* = \{0\}^D. \quad (\text{A.2})$$

(3) Elliptic function

$$f_3(\vec{x}) = \sum_{i=1}^D (10^6)^{(i-1)/(D-1)} x_i^2, \quad -100 \leq x_i \leq 100, \quad \vec{x}^* = \{0\}^D. \quad (\text{A.3})$$

(4) Noise function

$$f_4(\vec{x}) = \sum_{i=1}^D i x_i^4 + \text{random}[0, 1), \quad -1.28 \leq x_i \leq 1.28, \quad \vec{x}^* = \{0\}^D. \quad (\text{A.4})$$

(5) Zakharov function

$$f_5(\vec{x}) = \sum_{i=1}^D x_i^2 + \left( \sum_{i=1}^D 0.5 i x_i \right)^2 + \left( \sum_{i=1}^D 0.5 i x_i \right)^4, \quad -10 \leq x_i \leq 10, \quad \vec{x}^* = \{0\}^D. \quad (\text{A.5})$$

#### A.2. Multimodal Functions

(6) Rosenbrock function

$$f_6(\vec{x}) = \sum_{i=1}^{D-1} \left[ 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right], \quad -30 \leq x_i \leq 30, \quad \vec{x}^* = \{1\}^D. \quad (\text{A.6})$$

(7) Rastrigin function

$$f_7(\vec{x}) = \sum_{i=1}^D (x_i^2 - 10 \cos(2\pi x_i) + 10), \quad -10 \leq x_i \leq 10, \quad \vec{x}^* = \{0\}^D. \quad (\text{A.7})$$

(8) Griewank function

$$f_8(\vec{x}) = \frac{1}{4000} \sum_{i=1}^D x_i^2 - \prod_{i=1}^D \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1, \quad (A.8)$$

$$-600 \leq x_i \leq 600, \quad \vec{x}^* = \{0\}^D.$$

(9) Ackley function

$$f_9(\vec{x}) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2}\right) - \exp\left(\frac{1}{D} \sum_{i=1}^D \cos 2\pi x_i\right) + 20 + e, \quad -32 \leq x_i \leq 32, \quad \vec{x}^* = \{0\}^D. \quad (A.9)$$

(10) Schwefel function

$$f_{10}(\vec{x}) = 418.9829 \times D - \sum_{i=1}^D \left(x_i \sin\left(\sqrt{|x_i|}\right)\right), \quad (A.10)$$

$$-500 \leq x_i \leq 500, \quad \vec{x}^* = \{420.96\}^D.$$

(11) Noncontinuous Rastrigin function

$$f_{11}(\vec{x}) = \sum_{i=1}^D \left(y_i^2 - 10 \cos(2\pi y_i) + 10\right), \quad (A.11)$$

$$-10 \leq x_i \leq 10, \quad \vec{x}^* = \{0\}^D,$$

where

$$y_i = \begin{cases} x_i, & |x_i| < 0.5, \\ \frac{\text{round}(2x_i)}{2}, & |x_i| \geq 0.5, \end{cases} \quad (A.12)$$

for  $i = 1, 2, \dots, D$ .

(12) Levy function

$$f_{12}(\vec{x}) = \sum_{i=1}^{D-1} (x_i - 1)^2 \left[1 + 10 \sin^2(3\pi x_{i+1})\right] + \sin^2(3\pi x_i) + |x_D - 1| \left[1 + 10 \sin^2(3\pi x_D)\right], \quad (A.13)$$

$$-50 \leq x_i \leq 50, \quad \vec{x}^* = \{1\}^D.$$

(13) Alpine function

$$f_{13}(\vec{x}) = \sum_{i=1}^D |x_i \sin x_i + 0.1x_i|, \quad (A.14)$$

$$-10 \leq x_i \leq 10, \quad \vec{x}^* = \{0\}^D.$$

(14)  $2^D$  minima

$$f_{14}(\vec{x}) = 78.332331408 \times D - \sum_{i=1}^D (x_i^4 - 16x_i^2 + 5x_i), \quad -5 \leq x_i \leq 5, \quad \vec{x}^* = \{-2.9035\}^D. \quad (A.15)$$

(15) Generalized penalized

$$f_{15}(\vec{x}) = \frac{\pi}{D} \left\{ 10 \sin^2(\pi y_1) + \sum_{i=1}^{D-1} (y_i - 1)^2 \times \left[1 + 10 \sin^2(\pi y_{i+1})\right] + (y_D - 1)^2 \right\} + \sum_{i=1}^D u(x_i, 10, 100, 4) \quad (A.16)$$

$$-50 \leq x_i \leq 50, \quad \vec{x}^* = \{1\}^D,$$

where

$$y_i = 1 + \frac{1}{4}(x_i + 1),$$

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a, \\ 0, & -a < x_i < a, \\ k(-x_i - a)^m, & x_i < -a. \end{cases} \quad (A.17)$$

### A.3. Shifted and Rotated Functions

(16) Rotated Rosenbrock function

$$f_{16}(\vec{x}) = \sum_{i=1}^{D-1} \left[100(y_{i+1} - y_i^2)^2 + (y_i - 1)^2\right], \quad (A.18)$$

$$\vec{y} = M \times \vec{x}, \quad -10 \leq x_i \leq 10, \quad \vec{x}^* = \{0\}^D.$$

(17) Rotated Rastrigin function

$$f_{17}(\vec{x}) = \sum_{i=1}^D \left[y_i^2 - 10 \cos(2\pi y_i) + 10\right], \quad (A.19)$$

$$\vec{y} = M \times \vec{x}, \quad -5.12 \leq x_i \leq 5.12, \quad \vec{x}^* = \{0\}^D.$$

(18) Shifted Rosenbrock function

$$f_{18}(\vec{x}) = \sum_{i=1}^{D-1} \left[100(z_{i+1} - z_i^2)^2 + (z_i - 1)^2\right], \quad (A.20)$$

$$\vec{z} = \vec{x} - \vec{\delta} + 1, \quad -10 \leq x_i \leq 10, \quad \vec{x}^* = \{0\}^D.$$

(19) Shifted Rastrigin function

$$f_{19}(\vec{x}) = \sum_{i=1}^{D-1} \left[z_i^2 - 10 \cos(2\pi z_i) + 10\right], \quad (A.21)$$

$$\vec{z} = \vec{x} - \vec{\delta}, \quad -5.12 \leq x_i \leq 5.12, \quad \vec{x}^* = \{0\}^D.$$



(20) Shifted Rotated Rastrigin function

$$f_{20}(\vec{x}) = \sum_{i=1}^{D-1} \left[ z_i^2 - 10 \cos(2\pi z_i) + 10 \right], \quad (\text{A.22})$$

$$\vec{z} = (\vec{x} - \vec{\delta}) \times M, \quad -5.12 \leq x_i \leq 5.12, \quad \vec{x}^* = \{0\}^D.$$

(21) Shifted Rotated Griewank function

$$f_{21}(\vec{x}) = \frac{1}{4000} \sum_{i=1}^D z_i^2 - \prod_{i=1}^D \cos\left(\frac{z_i}{\sqrt{i}}\right) + 1, \quad (\text{A.23})$$

$$\vec{z} = (\vec{x} - \vec{\delta}) \times M, \quad -600 \leq x_i \leq 600, \quad \vec{x}^* = \{0\}^D.$$

## Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

## Acknowledgments

This work is partially supported by National Natural Science Foundation of China under Grant nos. 71240015, 71402103, and 61273367; National Science Foundation of SZU under Grant 836; the Foundation for Distinguished Young Talents in Higher Education of Guangdong, China, under Grant 2012WYM\_0116; the MOE Youth Foundation Project of Humanities and Social Sciences at Universities in China under Grant 13YJC630123; The Youth Foundation Project of Humanities and Social Sciences in Shenzhen University under grant 14QNFC28; and Ningbo Science & Technology Bureau (Science and Technology Project no. 2012B10055).

## References

- [1] J. Kennedy, R. Eberhart, and Y. Shi, *Swarm Intelligence*, Morgan Kaufmann, Boston, Mass, USA, 2001.
- [2] Y. Shi, "An optimization algorithm based on brainstorming process," *International Journal of Swarm Intelligence Research*, vol. 2, no. 4, pp. 35–62, 2011.
- [3] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Tech. Rep., Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.
- [4] D. Karaboga and B. Akay, "A survey: algorithms simulating bee swarm intelligence," *Artificial Intelligence Review*, vol. 31, no. 1–4, pp. 61–85, 2009.
- [5] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga, "A comprehensive survey: artificial bee colony (ABC) algorithm and applications," *Artificial Intelligence Review*, vol. 42, no. 1, pp. 21–57, 2014.
- [6] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.
- [7] D. Karaboga, B. Akay, and C. Ozturk, "Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks," in *Modeling Decisions for Artificial Intelligence*, V. Torra, Y. Narukawa, and Y. Yoshida, Eds., vol. 4617 of *Lecture Notes in Computer Science*, pp. 318–329, Springer, Berlin, Germany, 2007.
- [8] Q. Qin, S. Cheng, L. Li, and Y. Shi, "Artificial bee colony algorithm: a survey," *CAAI Transactions on Intelligent Systems*, vol. 9, no. 2, pp. 127–135, 2014.
- [9] W. Gao and S. Liu, "Improved artificial bee colony algorithm for global optimization," *Information Processing Letters*, vol. 111, no. 17, pp. 871–882, 2011.
- [10] M. S. Kiran and M. Gündüz, "A recombination-based hybridization of particle swarm optimization and artificial bee colony algorithm for continuous optimization problems," *Applied Soft Computing Journal*, vol. 13, no. 4, pp. 2188–2203, 2013.
- [11] D. Karaboga and B. Akay, "A comparative study of artificial Bee colony algorithm," *Applied Mathematics and Computation*, vol. 214, no. 1, pp. 108–132, 2009.
- [12] C. Ozturk and D. Karaboga, "Hybrid artificial bee colony algorithm for neural network training," in *Proceedings of the IEEE Congress of Evolutionary Computation (CEC '11)*, pp. 84–88, IEEE, June 2011.
- [13] T.-J. Hsieh, H.-F. Hsiao, and W.-C. Yeh, "Forecasting stock markets using wavelet transforms and recurrent neural networks: an integrated system based on artificial bee colony algorithm," *Applied Soft Computing Journal*, vol. 11, no. 2, pp. 2510–2525, 2011.
- [14] T.-J. Hsieh, H.-F. Hsiao, and W.-C. Yeh, "Mining financial distress trend data using penalty guided support vector machines based on hybrid of particle swarm optimization and artificial bee colony algorithm," *Neurocomputing*, vol. 82, pp. 196–206, 2012.
- [15] R. Zhang, S. Song, and C. Wu, "A hybrid artificial bee colony algorithm for the job shop scheduling problem," *International Journal of Production Economics*, vol. 141, no. 1, pp. 167–178, 2013.
- [16] A. Alvarado-Iniesta, J. L. Garcia-Alcaraz, M. I. Rodriguez-Borbon, and A. Maldonado, "Optimization of the material flow in a manufacturing plant by use of artificial bee colony algorithm," *Expert Systems with Applications*, vol. 40, no. 12, pp. 4785–4790, 2013.
- [17] Z. Cui and X. Gu, "An improved discrete artificial bee colony algorithm to minimize the makespan on hybrid flow shop problems," *Neurocomputing*, vol. 148, pp. 248–259, 2015.
- [18] C. Zhang, D. Ouyang, and J. Ning, "An artificial bee colony approach for clustering," *Expert Systems with Applications*, vol. 37, no. 7, pp. 4761–4767, 2010.
- [19] X. Yan, Y. Zhu, W. Zou, and L. Wang, "A new approach for data clustering using hybrid artificial bee colony algorithm," *Neurocomputing*, vol. 97, pp. 241–250, 2012.
- [20] M.-H. Horng, "Multilevel thresholding selection based on the artificial bee colony algorithm for image segmentation," *Expert Systems with Applications*, vol. 38, no. 11, pp. 13785–13791, 2011.
- [21] M. He, K. Hu, Y. Zhu, L. Ma, H. Chen, and Y. Song, "Hierarchical artificial bee colony optimizer with divide-and-conquer and crossover for multilevel threshold image segmentation," *Discrete Dynamics in Nature and Society*, vol. 2014, Article ID 941534, 22 pages, 2014.
- [22] C. Zhang and B. Zhang, "A hybrid artificial bee colony algorithm for the service selection problem," *Discrete Dynamics in Nature and Society*, vol. 2014, Article ID 835071, 13 pages, 2014.
- [23] K. Ayan and U. Kiliç, "Artificial bee colony algorithm solution for optimal reactive power flow," *Applied Soft Computing Journal*, vol. 12, no. 5, pp. 1477–1482, 2012.
- [24] 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.

- [25] A. Banharnsakun, T. Achalakul, and B. Sirinaovakul, "The best-so-far selection in artificial bee colony algorithm," *Applied Soft Computing Journal*, vol. 11, no. 2, pp. 2888–2901, 2011.
- [26] Z.-A. He, C. Ma, X. Wang et al., "A modified artificial bee colony algorithm based on search space division and disruptive selection strategy," *Mathematical Problems in Engineering*, vol. 2014, Article ID 432654, 14 pages, 2014.
- [27] W.-F. Gao and S.-Y. Liu, "A modified artificial bee colony algorithm," *Computers & Operations Research*, vol. 39, no. 3, pp. 687–697, 2012.
- [28] 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.
- [29] A. Alizadegan, B. Asady, and M. Ahmadpour, "Two modified versions of artificial bee colony algorithm," *Applied Mathematics and Computation*, vol. 225, pp. 601–609, 2013.
- [30] B. Akay and D. Karaboga, "A modified artificial bee colony algorithm for real-parameter optimization," *Information Sciences*, vol. 192, pp. 120–142, 2012.
- [31] K. Diwold, A. Aderhold, A. Scheidler, and M. Middendorf, "Performance evaluation of artificial bee colony optimization and new selection schemes," *Memetic Computing*, vol. 3, no. 3, pp. 149–162, 2011.
- [32] F. Kang, J. Li, and Z. Ma, "Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions," *Information Sciences*, vol. 181, no. 16, pp. 3508–3531, 2011.
- [33] B. Alatas, "Chaotic bee colony algorithms for global numerical optimization," *Expert Systems with Applications*, vol. 37, no. 8, pp. 5682–5687, 2010.
- [34] 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.
- [35] Q. Qin, S. Cheng, Q. Zhang, Y. Wei, and Y. Shi, "Multiple strategies based orthogonal design particle swarm optimizer for numerical optimization," *Computers & Operations Research*, vol. 60, pp. 91–110, 2015.
- [36] S. Cheng, Y. Shi, and Q. Qin, "Promoting diversity in particle swarm optimization to solve multimodal problems," in *Neural Information Processing*, B.-L. Lu, L. Zhang, and J. Kwok, Eds., vol. 7063 of *Lecture Notes in Computer Science*, pp. 228–237, Springer, Berlin, Germany, 2011.
- [37] S. Cheng, Y. Shi, and Q. Qin, "Population diversity of particle swarm optimizer solving single and multi-objective problems," *International Journal of Swarm Intelligence Research*, vol. 3, no. 4, pp. 23–60, 2012.
- [38] S. Cheng, *Population diversity in particle swarm optimization: definition, observation, control, and application [Ph.D. dissertation]*, Department of Electrical Engineering and Electronics, University of Liverpool, 2013.
- [39] D. Karaboga and B. Akay, "A modified Artificial Bee Colony (ABC) algorithm for constrained optimization problems," *Applied Soft Computing Journal*, vol. 11, no. 3, pp. 3021–3031, 2011.
- [40] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 687–697, 2008.
- [41] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.
- [42] T. K. Sharma and M. Pant, "Enhancing the food locations in an artificial bee colony algorithm," *Soft Computing*, vol. 17, no. 10, pp. 1939–1965, 2013.
- [43] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.



# Hindawi

Submit your manuscripts at  
<http://www.hindawi.com>

