Soft Comput
DOI 10.1007/s00500-014-1549-5

METHODOLOGIES AND APPLICATION

Gaussian bare-bones artificial bee colony algorithm

Xinyu Zhou · Zhijian Wu · Hui Wang · Shahryar Rahnamayan

Abstract As a relatively new global optimization technique, artificial bee colony (ABC) algorithm becomes popular in recent years for its simplicity and effectiveness. However, there is still an inefficiency in ABC regarding its solution search equation, which is good at exploration but poor at exploitation. To overcome this drawback, a Gaussian bare-bones ABC is proposed, where a new search equation is designed based on utilizing the global best solution. Furthermore, we employ the generalized opposition-based learning strategy to generate new food sources for scout bees, which is beneficial to discover more useful information for guiding search. A comprehensive set of experiments is conducted on 23 benchmark functions and a real-world optimization problem to verify the effectiveness of the proposed approach. Some well-known ABC variants and state-of-the-art evolutionary algorithms are used for comparison. The experimental results show that the proposed approach offers higher solution quality and faster convergence speed.

Keywords Swarm intelligence · Artificial bee colony · Solution search equation · Bare-bones technique · Generalized opposition-based learning

Communicated by V. Loia.

X. Zhou (✉)
School of Computer and Information Engineering, Jiangxi Normal University, Nanchang 330022, China
e-mail: xyzhou@whu.edu.cn

Z. Wu
State Key Laboratory of Software Engineering, School of Computer, Wuhan University, Wuhan 430072, China
e-mail: zhijianwu@whu.edu.cn

H. Wang
School of Information Engineering, Nanchang Institute of Technology, Nanchang 330099, China
e-mail: huiwang@whu.edu.cn

S. Rahnamayan
Department of Electrical, Computer and Software Engineering, University of Ontario Institute of Technology (OUIT), 2000 Simcoe Street North, Oshawa, ON L1H 7K4, Canada
e-mail: shahryar.rahnamayan@uoit.ca

1 Introduction

A wide variety of real-word problems can be converted into optimization problems and then be solved with optimization techniques. Unfortunately, many of these problems are often characterized as non-convex, discontinuous or non-differentiable, thus it is difficult to deal with them with traditional optimization algorithms. Evolutionary algorithms (EAs), as a powerful tool, are playing an increasingly important role in solving such kind of problems nowadays. EAs are stochastic search algorithms that simulate the evolutionary process of selection, variation and genetics in nature (Bäck 1996). The most prominent EAs proposed in the literatures are genetic algorithm (GA) (Tang et al. 1996), evolution strategy (ES) (Beyer and Schwefel 2002; Auger and Hansen 2005), particle swarm optimization (PSO) (Kennedy and Eberhart 1995; Eberhart and Shi 2001), ant colony optimization (ACO) (Dorigo and Di Caro 1999), differential evolution (DE) (Storn and Price 1997; Neri and Tirronen 2010; Das and Suganthan 2011), artificial bee colony (ABC) algorithm (Karaboga 2005), and so on.

In this paper, we focus on ABC algorithm, proposed by Karaboga (2005) and Karaboga and Basturk (2007). Like PSO and ACO that mimic the collective intelligent behavior of social insect swarms, ABC algorithm simulates the foraging behavior of a honeybee swarm. A recent comparative study (Karaboga and Akay 2009) has shown that the

Published online: 11 December 2014

performance of ABC is competitive to that of some popular EAs such as PSO and DE. Due to its simplicity yet effectiveness, ABC has been widely used to solve variant real-world optimization problems, such as symbolic regression (Karaboga et al. 2012b), neural network (Yeh and Hsieh 2012; Garro et al. 2011), and vehicle routing problem (Szeto et al. 2011).

However, similar to other EAs, ABC also tends to suffer from the problem of poor convergence. The possible reason is that the solution search equation which is used to generate new candidate solutions, has good exploration capability but poor exploitation capability (Zhu and Kwong 2010), and thereby it causes the problem of slow convergence speed. Therefore, how to enhance the exploitation of ABC for fast convergence is a challenging research topic. Recently, there is an increasing research trend in the ABC community (Zhu and Kwong 2010; Banharnsakun et al. 2011; Gao and Liu 2011, 2012; Gao et al. 2012, 2013a; Li et al. 2012) that the global best solution of current population is used in the solution search equation as a source of good information to guide the search behavior. For example, as one of the most representative works of this trend, Zhu and Kwong (2010) proposed a global best (gbest)-guided ABC (GABC) algorithm based on the inspiration of PSO. In GABC, the information of the gbest solution is incorporated into the solution search equation of ABC to improve the exploitation. However, it is necessary to note that although the information of the best solution is beneficial to enhance the exploitation, some side effects can also be easily triggered such as the algorithm becomes too greedy or not reliable, if the mechanism of utilizing the information is not well designed.

Following this active research trend, to improve ABC's exploitation and also keep its reliability, we propose a Gaussian bare-bones ABC (GBABC) algorithm based on the utilization of the global best solution. In GBABC, a new Gaussian bare-bones search equation is designed to generate new candidate solutions instead of the old one. An important feature of this new search equation is that positions of new food sources are sampled through a Gaussian distribution with dynamical mean value and variance value. In addition to the new search equation, another modification is suggested in the scout bee phase of GBABC. In the basic ABC, the discarded food sources in the scout bee phase are replaced with randomly generated candidate solutions, however, this may cause a problem that the already obtained search experience would be lost. Hence, to preserve search experience for scout bees, the generalized opposition-based learning (GOBL) strategy is employed to generate new food sources. To verify the performance of GBABC, a comprehensive set of experiments is conducted on 23 well-known benchmark functions, including shifted and/or rotated types, and a real-world optimization problem. The experimental results demonstrate the effectiveness of GBABC in solving complex numerical optimization problems when compared with other algorithms.

The rest of this paper is organized as follows. In Sect. 2, the related works are briefly reviewed, including the basic ABC and some improved ABCs. In Sect. 3, we present the proposed approach in detail. Section 4 presents and discusses the experimental results. Finally, the conclusion is drawn in Sect. 5.

2 Related works

2.1 Basic ABC

The ABC algorithm simulates the intelligent foraging behavior of a honeybee swarm. In ABC, the colony of artificial bees consists of three different kinds of bees: employed bees, onlooker bees and scout bees (Karaboga et al. 2012a). First, employed bees are responsible for exploring food sources near the hive, and then, after all the employ bees finish their searching, they will share the information with the onlooker bees about the nectar amounts and the positions of food sources on the dance area. Second, after evaluating the information from the employed bees, the onlooker bees decide to select a good part of food sources for further exploitation. More nectar amounts a food source owns, higher probability of being selected by onlooker bees it has. Last, if a food source is exhausted, its associated employed bee transforms into a scout bee, and then start to randomly search for a new source in the vicinity of the hive. Note that the number of onlookers equals to that of employed bees.

Similar to other EAs, ABC also starts with an initial population of SN randomly generated food sources. Each food source $X_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$ corresponds to a candidate solution to the optimization problem, and D denotes the problem dimension size. After initialization, the above-described search process of ABC can be divided into three phases as follows.

- Employed bee phase
  In this phase, each employed bee generates a new food source $V_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,D})$ in the neighborhood of its parent position $X_i = (x_{i,1,i,2}, \ldots, x_{i,D})$ using the following solution search equation.

$$v_{i,j} = x_{i,j} + \phi_{i,j} \cdot (x_{i,j} - x_{k,j}) \tag{1}$$

  where $k \in \{1, 2, \ldots, SN\}$ and $j \in \{1, 2, \ldots, D\}$ are randomly chosen indexes; k has to be different from i; $\phi_{i,j}$ is a random number in the range $[-1, 1]$. If the new food source V_i is better than its parent X_i, then X_i is replaced with V_i.
- Onlooker bee phase

After receiving the information from the employed bees, the onlooker bees begin to select food sources for exploitation. The probability of selecting a food source depends on its nectar amount, which can be calculated as follows.

$$p_i = \frac{\text{fit}_i}{\sum_{j=1}^{SN} \text{fit}_j} \tag{2}$$

where p_i is the selection probability and fit_i is the fitness value of the ith food source. Once an onlooker bee completes its selection, it would also produce a modification on its chosen food source using Eq. (1). As in the case of the employed bees, the greedy selection method is employed to retain a better one from the old food source and the modified food source as well.

- Scout bee phase
  If the fitness value of a food source keeps unchanged for at least *limit* times, this food source is considered to be exhausted, where *limit* is a control parameter and has predefined value. It is worth to note that, except for some common control parameters also shared with other EAs such as the population size, *limit* is the only single-specific parameter in ABC. Under this case, the food source has to be abandoned, and a new food source is generated to replace it as follows.

$$x_{i,j} = a_j + \text{rand}_j \cdot (b_j - a_j) \tag{3}$$

where $[a_j, b_j]$ is the boundary constraint for the jth dimensional variable, and rand_j is a uniformly distributed random number within the range [0, 1].

2.2 Improved ABCs

Attracted by the simplicity and efficiency of ABC, many researchers around the world have devoted their significant efforts to the improvements on ABC. As a result, a number of ABC variants have been proposed. In this subsection, we briefly review these improved ABCs, and categorize them into twofold. The first category focuses on the improvements on the solution search equation, while the second one is the hybridization of ABC with other search operators (Gao et al. 2013a). It is necessary to emphasize that our work falls in both of the categories.

The first category includes the following representative works of which the contributions are mainly the improvements on the solution search equation. In Zhu and Kwong (2010), inspired by PSO, Zhu and Kwong proposed a gbest-guided ABC (GABC) algorithm which incorporates the information of the global best solution into the solution search equation to improve the exploitation. Accordingly, the improved solution search equation contains a new added term and it is called the gbest term. The reported experimental results show that GABC outperforms the basic ABC on many used test functions. In TSai et al. (2009), inspired by the concept of Newtonian law of universal gravitation, TSai and Pan et al. proposed an interactive ABC (IABC). In IABC, the universal gravitations between the onlooker bees and the selected employed bees are exploited, and thus the solution search equation used for the onlooker bees is reformed to take into account the factor of universal gravitation. In Rajasekhar et al. (2011b), instead of the original solution search equation, Rajasekhar and Abraham et al. used a Lévy mutation to generate new food sources in the neighborhood of the global best solution of current population. Akay and Karaboga (2012) investigated the effects of two factors on the performance of ABC: frequency of the perturbation and magnitude of the perturbation. Consequently, two new parameters are introduced to control the both factors, and then a modified ABC algorithm is proposed. Note that the control parameter of controlling the frequency of perturbation is the same as the crossover probability of other EAs in essence. Based on the best-so-far solutions, Banharnsakun et al. (2011) proposed an improved ABC variant. In their approach, the search direction of onlooker bees is biased using the best-so-far solutions-based method, to accelerate the convergence process.

Inspired by the mutation strategies of DE, Gao and Liu et al. developed several different versions of improved search equations in their literatures (Gao and Liu 2011, 2012; Gao et al. 2012). In Gao and Liu (2011), they proposed two improved versions of solution search equations, namely ABC/best/1 and ABC/rand/1, which correspond to DE's mutation strategies DE/best/1 and DE/rand/1. Furthermore, they introduced a selective probability parameter to control the frequency of using these two improved search equations for a better balance between the exploration and exploitation. In Gao et al. (2012), compared the performance of two different ABC variants which, respectively, employ the ABC/best/1 and ABC/best/2 search equations. The experimental results show that ABC/best/1 greatly outperforms ABC/best/2. In Gao and Liu (2012), by employing the same ABC/best/1 search equation, Gao and Liu proposed a modified ABC (MABC) algorithm. In MABC, the framework is different from that of the basic ABC, which excludes the probabilistic selection scheme and scout bee phase. In Das et al. (2013) proposed a novel variant of ABC called fitness learning-based ABC with proximity stimuli (F*l*ABCps). In F*l*ABCps, the solution search equation updates more than one dimension to produce a new candidate solution based on the Rechenbergs 1/5th mutation rule, and the knowledge of the top $q\%$ food sources is also used in the improved solution search

equation. Recently, Bansal et al. (2014) proposed a self-adaptive ABC (SAABC) algorithm to balance the exploration and exploitation abilities of ABC. In SAABC, a self-adaptive step size mechanism is introduced to fine-tune the control parameters of the solution update strategy, and the parameter *limit* is also set in an adaptive manner. In Karaboga and Gorkemli (2014), proposed a quick ABC (qABC) algorithm to improve tshe performance of ABC in terms of local search ability. In qABC, the behavior of the onlooker bees is modified to focusing on exploiting the neighborhood food source.

The second category is reviewed as follows. Kang et al. (2009) proposed a hybrid simplex ABC algorithm by combining Nelder–Mead simplex search (NMSS) method with ABC. Because the NMSS method is a local descent algorithm, to some extent, it can be considered as a local exploitation tool in their proposed approach. Similarly, Kang et al. (2011) also developed a Rosenbrock ABC algorithm, where the Rosenbrock method is modified for multimodal optimization problems and then introduced into ABC as a local exploitation tool as well. In Alatas (2010), Alatas made two modifications for ABC based on the concept of chaotic maps. The first modification is to use the chaotic maps to initialize the colony, and the second one is to use the chaotic search for the scout bees to generate new random food sources. Rajasekhar et al. (2011a, b) proposed two versions of mutated ABC variants based on the Sobol and Lévy distributions. The reported experimental results show the superiority of the mutation operations, especially on high dimensional problems. Also based on the Lévy distribution, Sharma et al. (2013) incorporated the Lévy flight random walk into ABC as a local search operator. In Bansal et al. (2013), proposed a Memetic ABC (MeABC) by combing a local search phase, where golden section search (GSS) approach is used to control the step size. El-Abd (2012) introduced the concept of GOBL into ABC which uses the GOBL strategy in the initialization and generation jumping phases. Gao et al. (2013a) used the orthogonal experimental design to form an orthogonal learning strategy for ABC to discover more useful information from the search experiences. In Gao et al. (2013b), to improve the exploitation of ABC, Gao and Liu et al. adopted the Powell's method as a local search tool. In this way, ABC and Powell's method have complementary advantages. In the procedure of initializing population, Sharma and Pant (2013) designed a novel method to initialize the positions of food sources, which are located on intermediate positions between the uniformly generated random numbers and random numbers generated by opposition-based learning (OBL). The corresponding algorithm is called intermediate ABC (I-ABC).

In this subsection, we only presented a brief overview of some related works, interested readers can refer to the good survey of ABC in Karaboga et al. (2012a).

3 Gaussian bare-bones ABC (GBABC)

3.1 Gaussian bare-bones search equation

Previously, we briefly reviewed some representative ABCs in Sect. 2.2. Among these improved ABCs, a considerable part of research efforts is focused on modifying the solution search equation with the global best solution of current population. One of the representative works of this type is GABC developed by Zhu and Kwong (2010). In GABC, the information of gbest is utilized to guide the search of new food sources, and the modified solution search equation in GABC is described as follows:

$$v_{i,j} = x_{i,j} + \phi_{i,j} \cdot (x_{i,j} - x_{k,j}) + \psi_{i,j} \cdot (y_j - x_{i,j}) \tag{4}$$

where the third term in the right-hand side of Eq. (4) is a new added term called gbest term, y_j is the jth element of the global best solution, $\psi_{i,j}$ is a uniform random number within $[0, C]$, and C is a nonnegative constant and is suggested to set to 1.5.

Although this new search equation has been shown superior to the original one, its mechanism of utilizing the gbest can still cause inefficiency to the search ability of the algorithm and slow down convergence speed. Because the guidance of the last two terms may be in opposite directions, and this can cause an "oscillation" phenomenon (Zhan et al. 2011; Gao et al. 2013a). For the clearness and easiness of understanding, we first simplify Eq. (4) as Eq. (5) by moving the first term $x_{i,j}$ and two weight components $\phi_{i,j}$ and $\psi_{i,j}$ as follows:

$$v_{i,j} = (x_{i,j} - x_{k,j}) + (y_j - x_{i,j}) \tag{5}$$

In Eq. (5), we consider the following case for a minimization problem to demonstrate the oscillation phenomenon. At first, X_i locates between X_k and Y (gbest), and the distance between X_i and Y is farther than the one between X_i and X_k, as shown in Fig. 1a, and then X_i will move toward Y for its larger pull. However, the distance between X_i and X_k will increase while X_i moving toward Y, as shown in Fig. 1b. In this case, X_i will be puzzled in deciding where to stay, and thus the oscillation would occur (Zhan et al. 2011; Gao et al. 2013a).

To overcome such issue, we design a Gaussian bare-bones search equation inspired by the concept of BBPSO (Kennedy 2003). It is well known that PSO is a popular swarm intelligence-based algorithm which simulates the behavior of birds flocking and fish schooling (Kennedy and Eberhart 1995). In PSO, the flying trajectory of each particle is affected by its personal best position (pbest) and its neighborhood's best position (gbest). Some theoretical studies (Clerc and Kennedy 2002; van den Bergh and Engelbrecht 2006)

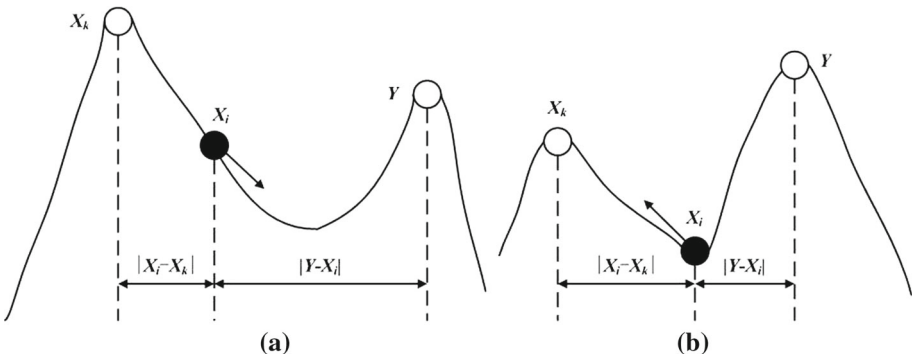

Fig. 1 Oscillation phenomenon in GABC. **a** X_i moves toward Y. **b** X_i moves toward X_k

provided an analysis about particle's trajectory that each particle is able to converge to a stable point, that is,

$$\lim_{G\to+\infty} X_i^G = \frac{c_1 \cdot \text{pbest}_i^G + c_2 \cdot \text{gbest}}{c_1 + c_2} \tag{6}$$

where G is the generation counter, c_1 and c_2 are two learning factors.

Based on PSO's convergence behavior, in BBPSO, Kennedy (2003) proposed to eliminate the velocity formula and update particle's position according to the following equation.

$$X_i^{G+1} = N\left(\frac{\text{pbest}_i^G + \text{gbest}}{2}, \left|\text{pbest}_i^G - \text{gbest}\right|\right) \tag{7}$$

where $N(\cdot)$ represents a Gaussian distribution with mean $(\text{pbest}_i^G + \text{gbest})/2$ and variance $\left|\text{pbest}_i^G - \text{gbest}\right|$. Furthermore, to speed up convergence, Kennedy (2003) proposed a modified BBPSO using an alternative mechanism as follows.

$$x_{i,j}^{G+1} = \begin{cases} N\left(\frac{\text{pbest}_{i,j}^G + \text{gbest}_j}{2}, \left|\text{pbest}_{i,j}^G - \text{gbest}_j\right|\right) & \text{if rand}_j > 0.5 \\ \text{pbest}_{i,j}^G & \text{otherwise} \end{cases} \tag{8}$$

where rand_j is a random value within [0, 1] for the jth dimension. Note that this mechanism provides a 50 % chance that the jth dimension of particle X_i is derived from its personal best position (Wang et al. 2013).

Inspired by this mechanism of updating particle's position, a more efficient search equation can be designed to accelerate ABC's convergence by elaborately using the global best solution. Therefore, we propose a new Gaussian bare-bones search equation for ABC to generate positions of new food sources, that is,

$$v_{i,j}^G = \begin{cases} N\left(\frac{x_{i,j}^G + x_{\text{best},j}}{2}, \left|x_{i,j}^G - x_{\text{best},j}\right|\right) & \text{if rand}_j \le CR \\ x_{i,j} & \text{otherwise} \end{cases} \tag{9}$$

where X_{best} is the global best solution of current population, and CR is a new introduced parameter which controls how many elements in expectation can be derived from its parent X_i for V_i. Since there is only one dimension of X_i to be updated for V_i in the original search equation, the introduction of CR is helpful to inherit more information from X_{best} to enhance the exploitation. The setting of CR will be discussed in the next Sect. 4.1.

In Eq. (9), new candidate solutions are generated in the search space formed by the current solution and the global best solution. As a result, the positions of these new candidate solutions will be located around the center position between X_i and X_{best}. And the search behavior will gradually turn to exploitation from exploration. At the initial evolutionary stage, the search behavior focuses on exploration due to the large variance (initially, X_i will be far away from X_{best}). With increasing the number of generations, the variance becomes smaller (X_i will approach to X_{best}), so the search behavior will turn to exploitation. Compared with the original search equation Eq. (1) and the modified one of GABC Eq. (4), the Gaussian bare-bones search equation Eq. (9) has two obvious advantages. First, since Eq. (1) is good at exploration but poor at exploitation, it may cause poor convergence. However, Eq. (9) takes advantages of the global best solution to guide the search of new candidate solutions, which is beneficial to enhance the exploitation. Second, in Eq. (4), the last two terms $(x_{i,j} - x_{k,j})$ and $(y_j - x_{i,j})$ may be in opposite directions, and thereby this may result in an "oscillation" phenomenon and cause inefficiency to the search ability of the algorithm. But Eq. (9) only uses a Gaussian distribution to generate positions, which can be considered as one single term, therefore, it is easily capable to avoid the oscillation phenomenon and maximize the search ability of the algorithm.

From the above explanation, it is clear that the new designed solution search equation described by Eq. (9) is better at exploitation than the original one. However, it may also run the risk of reducing the exploration of ABC. In other words, a contradiction is arising that the original solution search equation is good at exploration but may results in slow convergence, while the new one is good at exploitation but may cause premature convergence. It is well known that

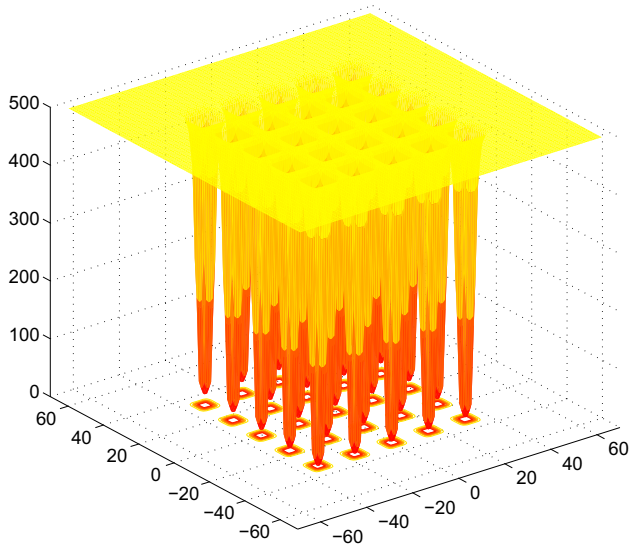

Fig. 2 3-D plot of the Shekel's Foxholes function

both the exploration and exploitation capabilities are necessary for EAs. In practice, however, these two capabilities contradict each other. To achieve good performances on problem optimizations, they should be well balanced. From the principle of labor division of artificial bees in ABC, we can observe that the employed bees are designed to explore new food sources, while the onlooker bees are responsible for exploiting those explored food sources. Therefore, following this basic idea, we employ a simple but efficient method to address this contradiction that the employed bees still use the original solution search equation for generating new candidate solutions, but the onlooker bees use the new one. Under this case, our approach attempts to balance the exploitation and exploration.

To better illustrate the difference of search behaviors between the Gaussian bare-bones search equation and the original one, the two-dimensional Shekel's Foxholes function shown in Fig. 1 is employed as a case study. This function has 24 distinct local minima and one global minimum $f(-32, -32) = 0.998004$ in the range $[65.536, 65.536]^2$, its detailed definition can be found in Yao's literature (Yao et al. 1999). In this case study, both the basic ABC and its modified version with the Gaussian bare-bones search equation for onlooker bees (ABC-BB for short), are used to solve the Shekel's Foxholes problem, respectively. The population distributions of ABC and ABC-BB at different generations are plotted. Both ABC and ABC-BB have the same number of food sources $SN = 30$, and $CR = 0.3$ for ABC-BB. Figure 2 shows the contour plots of the Shekel's Foxholes function,

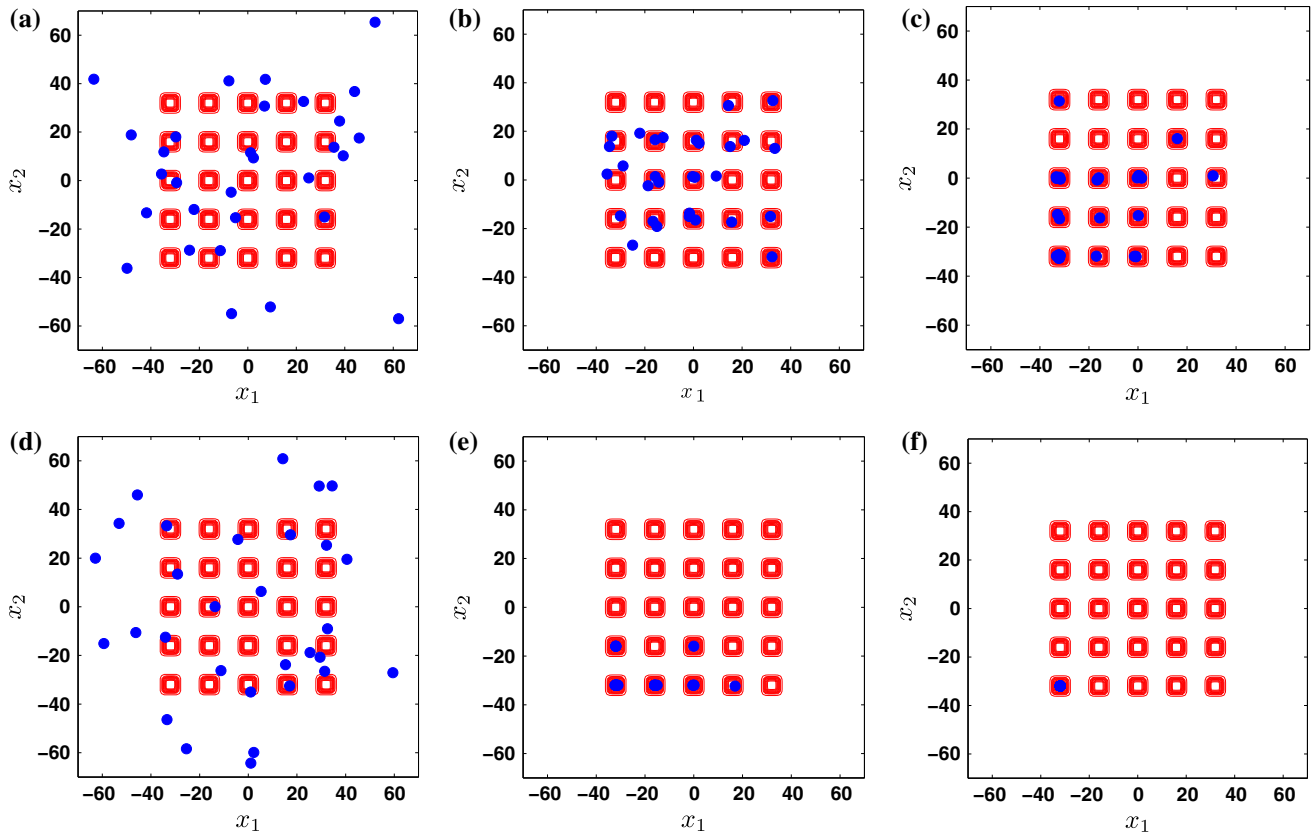

Fig. 3 Population distributions of ABC and ABC-BB at different generations. **a** ABC at the first generation. **b** ABC at the 15th generation. **c** ABC at the 30th generation. **d** ABC-BB at the first generation. **e** ABC-BB at the 15th generation. **f** ABC-BB at the 30th generation

and Fig. 3 shows the population distributions of ABC and ABC-BB at the first, 15th and 30th generations. As seen, at the beginning of evolution, the food sources of both ABC and ABC-BB almost cover the entire search space, respectively. However, the search regions of ABC-BB converge faster than those of ABC as the evolution proceeds. It implies that ABC-BB can converge to the global optimum rapidly with the help of Gaussian bare-bones search equation.

3.2 Modified scout bee phase

In the original scout bee phase, if a food source cannot be improved further for at least *limit* times, it is considered to be exhausted and would be abandoned. As a result, a new food source is generated in a random manner by the scout bee to replace the exhausted one. In this case, however, it may cause a problem that the already obtained search experiences would be lost, because the exhausted food source may contain more useful information than the new random one for guidance of search process. Hence, we introduce the GOBL strategy into the scout bee phase, which is beneficial to preserve the search experiences for the efficiency of the algorithm. The GOBL strategy is an enhanced version of the concept of opposition-based learning (Tizhoosh 2005; Rahnamayan et al. 2008). The main idea behind GOBL is that when evaluating a candidate solution X to a given problem, simultaneously computing its *opposite solution* $\breve{X}$ can provide a higher chance for $\breve{X}$ to be closer to the global optimum than a random candidate solution (Wang et al. 2011).

Let $X_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$ be a candidate solution to an optimization problem of D-dimension, and its opposite solution $\breve{X} = (\breve{x}_1, \breve{x}_2, \ldots, \breve{x}_D)$ is defined by:

$$\breve{x}_j = k \cdot (da_j + db_j) - x_j \tag{10}$$

where $k \in U(0, 1)$ is a generalized coefficient. $[da_j, db_j]$ is the dynamic boundary constraint for the jth dimensional variable, which is defined as follows.

$$\begin{cases} da_j = \min(x_{i,j}) \\ db_j = \max(x_{i,j}) \end{cases} \tag{11}$$

After using GOBL, however, an opposite candidate solution may jump out of the initial box-constraint $[a_j, b_j]$, this implies that GOBL fails to generate a feasible candidate solution. To avoid this case, the non-feasible opposite solution is replaced with a random one using Eq. (3). To be specific, first, the opposite candidate solution of an exhausted food source is generated by Eq. (10), and a random candidate solution generated by Eq. (3) is also provided at the same time. And then the better one between the opposite candidate solution and the random candidate solution is picked out to replace the exhausted food source. This procedure can be formulated as follows.

$$X_i^{G+1} = \begin{cases} \breve{X}_i & \text{if } f(\breve{X}_i) \le f(V_i) \\ V_i & \text{otherwise} \end{cases} \tag{12}$$

where V_i represents a new random candidate solution. Note that there is only one food source can be abandoned at each generation in basic ABC, while no such limitation exists in our approach. In other words, if more than one food source have not been improved for at least *limit* times, all of them can be abandoned. In this way, it is helpful to avoid being trapped into local optimal when solving complex problems.

3.3 Pseudocode of GBABC

Compared with basic ABC, GBABC makes two modifications. First, in the onlooker bee phase, the original solution search equation is replaced with the new deigned Gaussian bare-bones one. Second, in the scout bee phase, the GOBL strategy is employed to generate new food sources for the scouts except for the random food sources. The pseudocode of GBABC is described in Algorithm 1, where FEs is the number of used fitness function evaluations, and $MaxFEs$, as the stopping criterion, is the maximal number of fitness function evaluations. $trial_i$ records the unchanged times of X_i's fitness value.

4 Experimental verification

4.1 Benchmark functions

To verify the performance of our approach, a set of 23 benchmark functions is used in the following experiments. The first 13 test functions are well-known scalable problems (Yao et al. 1999; Wang et al. 2013). Among these problems, F01–F04 are unimodal functions, and F05 is the Rosenbrock function which is multimodal when $D > 3$ (Shang and Qiu 2006). F06 is a step function which has one minimum and is discontinuous, while F07 is a noisy quartic function. F08–F13 are multimodal functions with many local minima. The remaining 10 functions (F14–F23) are shifted and/or rotated types taken from the CEC 2005 competition (Suganthan et al. 2005). Brief descriptions of these test functions are summarized in Table 1. The detailed definitions of functions F01–F13 can be found in Yao et al. (1999) and Wang et al. (2013), while those of functions F14–F23 can be found in Suganthan et al. (2005).

4.2 Adjusting the parameter CR

In GBABC, only one additional parameter CR is introduced whose value may affect the performance. Therefore, in the

Algorithm 1 Pseudocode of GBABC

Randomly generate SN candidate solutions $\{X_i \mid i = 1, 2, \cdots, SN\}$ as food sources;
$FEs = SN$;
while $FEs \leq MaxFEs$ **do**
  /* Employed bee phase */
  for $i = 1$ to SN **do**
    Generate a new candidate solution V_i according to Eq. (1);
    if $f(V_i) < f(X_i)$ **then**
      Replace X_i with V_i;
      $trial_i$=0;
    else
      $trial_i = trial_i + 1$;
    end if
    $FEs = FEs + 1$;
  end for
  /* Onlooker bee phase */
  Calculate the probability p_i according to Eq. (2);
  for $i = 1$ to SN **do**
    Choose a food source X_j from the current population P by the roulette wheel selection mechanism;
    Generate a new candidate solution V_j according to Eq. (9);
    if $f(V_j) < f(X_j)$ **then**
      Replace X_j with V_j;
      $trial_j$=0;
    else
      $trial_j = trial_j + 1$;
    end if
    $FEs = FEs + 1$;
  end for
  /* Scout bee phase */
  for $i = 1$ to SN **do**
    if $trial_i > limit$ **then**
      $trial_i$=0;
      Generate an opposite solution of X_i, $\breve{X}_i$, according to Eq. (10);
      Randomly generate a new candidate solution V_i according to Eq. (3);
      if $f(\breve{X}_i) < f(V_i)$ **then**
        Replace X_i with $\breve{X}_i$;
      else
        Replace X_i with V_i;
      end if
      $FEs = FEs + 2$;
    end if
  end for
end while

Table 1 The 23 benchmark functions used in the experiments, where D is the dimension of the functions, and X^* is the global optimum of the function

Functions	Name	Search range	$f(X^*)$
F01	Sphere	$[-100, 100]$	0
F02	Schwefel 2.22	$[-10, 10]$	0
F03	Schwefel 1.2	$[-100, 100]$	0
F04	Schwefel 2.21	$[-100, 100]$	0
F05	Rosenbrock	$[-30, 30]$	0
F06	Step	$[-100, 100]$	0
F07	Quartic	$[-1.28, 1.28]$	0
F08	Schwefel 2.26	$[-500, 500]$	$-418.98 \cdot D$
F09	Rastrigin	$[-5.12, 5.12]$	0
F10	Ackley	$[-32, 32]$	0
F11	Griewank	$[-600, 600]$	0
F12	Penalized 1	$[-50, 50]$	0
F13	Penalized 2	$[-50, 50]$	0
F14	Shifted sphere	$[-100, 100]$	-450
F15	Shifted Schwefel 1.2	$[-100, 100]$	-450
F16	Shifted rotated high conditioned elliptic	$[-100, 100]$	-450
F17	Shifted Schwefel 1.2 with noise in fitness	$[-100, 100]$	-450
F18	Schwefel 2.6 with global optimum on bounds	$[-100, 100]$	-310
F19	Shifted Rosenbrock	$[-100, 100]$	390
F20	Shifted rotated Griewank without bounds	[0, 600]	-180
F21	Shifted rotated Ackley with global optimum on bounds	$[-32, 32]$	-140
F22	Shifted Rastrigin	$[-5, 5]$	-330
F23	Shifted rotated Rastrigin	$[-5, 5]$	-330

subsection, we investigate different CR values to select the best one for maximizing the performance of GBABC. The available CR values are in the range $[0.1, 0.9]$ in steps of 0.1, i.e., there are nine different choices for CR in total. All the functions are tested at $D = 30$, the maximal number of fitness function evaluations ($MaxFEs$) is set to $5{,}000 \cdot D$ for F01–F13, and $10{,}000 \cdot D$ for F14–F23 according to the suggestions in Suganthan et al. (2005). The number of food sources SN and $limit$ are set to 30 and 100, respectively. Each test function is run 30 times, and the mean error ($f(X) - f(X^*)$, X^* is the global optimum) and standard deviation values are recorded.

To simplify the final results, Table 2 only presents the relatively good values of CR which can yield better results than others. From the results, it is clear that a higher value of CR is more suitable for solving unimodal problems (e.g., the results on F01, F02 and F04), while a lower one is better for multimodal problems (e.g., the results on F07, F08 and F09). The reason may be that a higher value of CR can make the new candidate solution inherit more information from the global best solution, which is helpful to enhance the exploitation for solving unimodal problems. But a lower value of CR is better at exploration for solving multimodal problems.

Table 2 The results of GBABC with different *CR* values

Functions	$CR = 0.1$ Mean error ± std dev	$CR = 0.3$ Mean error ± std dev	$CR = 0.5$ Mean error ± std dev	$CR = 0.8$ Mean error ± std dev
F01	3.15E−46 ± 2.09E−46	2.63E−49 ± 5.27E−49	3.91E−51 ± 1.26E−50	1.56E−51 ± 1.85E−51
F02	2.51E−25 ± 2.13E−25	2.53E−31 ± 2.69E−31	1.85E−33 ± 2.41E−33	1.11E−35 ± 1.20E−35
F03	1.25E+02 ± 8.59E+01	4.48E+01 ± 4.77E+01	2.44E+01 ± 5.81E+01	3.23E+01 ± 3.29E+01
F04	1.67E−02 ± 8.36E−03	1.39E−07 ± 1.88E−07	2.21E−07 ± 4.79E−07	1.28E−07 ± 2.92E−07
F05	5.00E+00 ± 7.15E+00	3.63E+00 ± 3.40E+00	8.08E+00 ± 8.17E+00	1.12E+01 ± 1.76E+01
F06	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00
F07	7.05E−05 ± 6.04E−05	9.82E−05 ± 7.06E−05	1.51E−04 ± 1.04E−04	2.35E−04 ± 1.95E−04
F08	3.82E−04 ± 3.27E−13	3.82E−04 ± 4.54E−13	1.58E+01 ± 4.03E+01	1.98E+01 ± 4.41E+01
F09	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00	8.23E−15 ± 2.40E−14	6.67E−02 ± 2.49E−01
F10	4.44E−16 ± 0.00E+00	4.44E−16 ± 0.00E+00	4.44E−16 ± 0.00E+00	4.44E−16 ± 0.00E+00
F11	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00	0.00E+00 ± 0.00E+00
F12	1.57E−32 ± 5.47E−48	1.57E−32 ± 5.47E−48	1.57E−32 ± 5.47E−48	1.57E−32 ± 5.47E−48
F13	1.35E−32 ± 5.47E−48	1.35E−32 ± 5.47E−48	1.35E−32 ± 5.47E−48	1.91E−31 ± 8.96E−31
F14	1.00E−13 ± 2.40E−14	6.44E−14 ± 1.93E−14	6.82E−14 ± 2.27E−14	7.58E−14 ± 2.68E−14
F15	1.44E+03 ± 3.76E+02	2.94E+02 ± 1.11E+02	3.86E+01 ± 5.96E+01	9.48E−01 ± 1.08E+00
F16	5.03E+06 ± 1.60E+06	2.80E+06 ± 1.02E+06	2.95E+06 ± 1.28E+06	3.40E+06 ± 1.27E+06
F17	2.02E+04 ± 2.69E+03	7.38E+03 ± 1.46E+03	1.47E+03 ± 5.07E+02	9.73E+01 ± 7.54E+01
F18	4.21E+03 ± 5.55E+02	2.43E+03 ± 4.72E+02	2.82E+03 ± 6.80E+02	3.49E+03 ± 5.71E+02
F19	2.82E+01 ± 6.13E+01	3.64E+01 ± 4.94E+01	8.56E+01 ± 1.27E+02	1.09E+02 ± 1.09E+02
F20	8.84E−03 ± 9.58E−03	1.86E−02 ± 1.63E−02	1.62E−02 ± 1.37E−02	1.50E−02 ± 1.39E−02
F21	2.08E+01 ± 7.29E−02	2.09E+01 ± 8.62E−02	2.09E+01 ± 6.50E−02	2.09E+01 ± 5.94E−02
F22	9.28E−14 ± 2.74E−14	6.63E−14 ± 2.12E−14	8.53E−14 ± 2.84E−14	9.85E−14 ± 3.26E−14
F23	1.39E+02 ± 1.83E+01	1.28E+02 ± 1.77E+01	9.19E+01 ± 1.87E+01	8.08E+01 ± 1.55E+01

To select the best value of CR at a statistical level, the Friedman test is conducted to obtain average rankings according to the suggestions in García et al. (2009, 2010). Table 3 shows the average rankings of GBABC with different CR values. As seen, $CR = 0.3$ achieves the best average ranking. Therefore, in the following experiments, the parameter CR is set to 0.3.

Table 3 Average rankings of GBABC with different *CR* values, and the best value is shown in boldface

CR values	Average rankings
$CR = 0.1$	2.78
CR = 0.3	**2.26**
$CR = 0.5$	2.35
$CR = 0.8$	2.61

4.3 Effects of each modification

Compared with basic ABC, we make two modifications in GBABC, i.e., the Gaussian bare-bones search equation and modified scout bee phase. To investigate how much these two modifications make contribution to improving the performance of the algorithm, respectively, three relative variants are used for verification. They are listed as follows.

- ABC without any modification, i.e., basic ABC.
- ABC with the Gaussian bare-bones search equation for onlooker bees (ABC-BB for short).
- ABC with the modified scout bee phase (ABC-GOBL for short).

Therefore, there are three different algorithms are used to compare with GBABC, i.e., the basic ABC, ABC-BB and ABC-GOBL. For a fair comparison, all the competitive algorithms have the same parameter settings, i.e., the number of food sources SN is set to 30, and $limit$ equals to 100. The stopping criterion is the same as in Sect. 4.2. Each algorithm is run 30 times per function, and the mean error and standard deviation values are recorded. Table 4 presents the final results of these four algorithms. The best values are marked in boldface.

From the comparison of ABC-BB with ABC, it can be seen that ABC-BB outperforms ABC on the majority of test functions, and it only loses on F05 and F19. For the

Table 4 The results achieved by the original ABC, ABC-BB, ABC-GOBL and GBABC at $D = 30$

Functions	ABC Mean error ± std dev	ABC-BB Mean error ± std dev	ABC-GOBL Mean error ± std dev	GBABC Mean error ± std dev
F01	5.01E−37 ± 1.11E−36	**9.35E−50 ± 1.31E−49**	4.11E−37 ± 6.90E−37	2.63E−49 ± 5.27E−49
F02	1.01E−19 ± 5.54E−20	4.33E−31 ± 5.88E−31	8.70E−20 ± 4.82E−20	**2.53E−31 ± 2.69E−31**
F03	6.21E+03 ± 1.06E+03	2.58E+03 ± 8.25E+02	2.30E+03 ± 1.96E+03	**4.48E+01 ± 7.18E+01**
F04	2.58E+01 ± 6.91E+00	1.66E−02 ± 4.03E−03	7.76E−02 ± 4.98E−02	**1.39E−07 ± 1.88E−07**
F05	6.44E−01 ± 7.06E−01	1.25E+01 ± 5.27E+00	**3.83E−01 ± 9.18E−01**	3.63E+00 ± 3.40E+00
F06	**0.00E+00 ± 0.00E+00**	**0.00E+00 ± 0.00E+00**	**0.00E+00 ± 0.00E+00**	**0.00E+00 ± 0.00E+00**
F07	2.91E−01 ± 8.41E−02	1.74E−02 ± 7.49E−03	**9.23E−05 ± 6.89E−05**	9.82E−05 ± 7.06E−05
F08	**3.82E−04 ± 1.20E−12**	**3.82E−04 ± 4.70E−13**	**3.82E−04 ± 4.04E−12**	**3.82E−04 ± 4.54E−13**
F09	4.75E−14 ± 2.12E−13	**0.00E+00 ± 0.00E+00**	4.86E−15 ± 1.40E−14	**0.00E+00 ± 0.00E+00**
F10	6.01E−14 ± 1.02E−14	1.89E−14 ± 3.60E−15	3.17E−15 ± 1.50E−15	**4.44E-16 ± 0.00E+00**
F11	3.26E−05 ± 1.75E−04	**0.00E+00 ± 0.00E+00**	2.46E−09 ± 1.31E−08	**0.00E+00 ± 0.00E+00**
F12	**1.57E−32 ± 4.63E−34**	**1.57E−32 ± 5.47E−48**	**1.57E−32 ± 5.47E−48**	**1.57E−32 ± 5.47E−48**
F13	**1.35E−32 ± 5.47E−48**	**1.35E−32 ± 5.47E−48**	**1.37E−32 ± 9.05E−34**	**1.35E−32 ± 5.47E−48**
F14	1.48E−13 ± 2.78E−14	6.44E−14 ± 1.93E−14	**1.50E−14 ± 3.11E−14**	6.44E−14 ± 1.93E−14
F15	6.50E+03 ± 1.12E+03	**2.63E+02 ± 1.32E+02**	3.83E+03 ± 1.27E+03	2.94E+02 ± 1.11E+02
F16	9.54E+06 ± 2.46E+06	**2.55E+06 ± 9.35E+05**	8.26E+06 ± 1.48E+06	2.80E+06 ± 1.02E+06
F17	3.74E+04 ± 5.34E+03	**4.76E+03 ± 1.04E+03**	2.84E+04 ± 2.66E+03	7.38E+03 ± 1.46E+03
F18	9.22E+03 ± 1.04E+03	2.47E+03 ± 5.05E+02	9.05E+03 ± 1.09E+03	**2.43E+03 ± 4.72E+02**
F19	5.62E+01 ± 6.87E+01	6.98E+01 ± 9.13E+01	3.73E+01 ± 3.39E+01	**3.64E+01 ± 4.94E+01**
F20	1.59E−02 ± 7.76E−03	1.52E−02 ± 1.60E−02	**1.45E−02 ± 9.83E−03**	1.86E−02 ± 1.63E−02
F21	2.09E+01 ± 6.42E−02	2.09E+01 ± 5.68E−02	**2.07E+01 ± 8.14E−02**	2.09E+01 ± 8.62E−02
F22	6.46E−13 ± 1.52E−12	7.96E−14 ± 2.78E−14	2.54E−13 ± 3.29E−13	**6.63E−14 ± 2.12E−14**
F23	2.40E+02 ± 3.16E+01	**1.18E+02 ± 2.20E+01**	1.37E+02 ± 2.58E+01	1.28E+02 ± 1.77E+01

Rosenbrock problem (F05) and its shifted version (F19), it is pointed out in Shang and Qiu (2006) that this problem is multimodal when $D > 3$, and its global optimum is inside a long narrow parabolic-shaped flat valley. A more explorative algorithm, e.g., the basic ABC, usually performs better on it. However, on the mutlimodal Rastrigin problem (F09), the performance of ABC-BB is much better than that of ABC. The overall performance of ABC-BB shows that the Gaussian bare-bones search equation is more effective than the original one. For ABC-GOBL, its performance is not worse than that of ABC on any test function. Specifically, the results of ABC-GOBL on F04, F07 and F11 are much better than those of ABC. The comparison of ABC-GOBL with ABC demonstrates that the modified scout bee phase can enhance the performance of ABC. By combining these two modifications, GBABC obtains the best performance among the involved four algorithms. Specifically, on F03, F04 and F10, only one single modification can hardly achieve good performance, but much improvement can be obtained by making the two modifications work together. The experimental results and comparisons verify that the new search equation and the GOBL strategy indeed help ABC with them perform better than ABC with either or neither one on most of the test functions. Furthermore, the Friedman test is conducted to compare the above four algorithms at a statistical level, and Table 5 shows their average rankings. As seen, the best average ranking is achieved by GBABC, and the other three algorithms can be sorted as follows: ABC-BB, ABC-GOBL and ABC.

Table 5 Average rankings of ABC, ABC-BB, ABC-GOBL and GBABC, and the best value is shown in boldface

Algorithms	Average rankings
ABC	3.26
ABC-BB	2.24
ABC-GOBL	2.65
GBABC	**1.85**

4.4 Comparison with other ABCs

This subsection presents a comparative study of GBABC with GABC and GOABC at both $D = 30$ and 50. In addition, the basic ABC is also included as a baseline. GABC Zhu and Kwong (2010) is a gbest-guided ABC variant, which

Table 6 The results achieved by ABC, GABC, GOABC and GBABC at $D = 30$

Functions	ABC Mean error ± std dev	GABC Mean error ± std dev	GOABC Mean error ± std dev	GBABC Mean error ± std dev
F01	5.01E−37 ± 1.11E−36†	1.69E−47 ± 1.60E−47†	1.00E−50 ± 2.12E−50‡	2.63E−49 ± 5.27E−49
F02	1.01E−19 ± 5.54E−20†	1.26E−24 ± 7.72E−25†	4.28E−31 ± 4.53E−31†	2.53E−31 ± 2.69E−31
F03	6.21E+03 ± 1.06E+03†	1.03E+04 ± 3.21E+03†	3.95E+02 ± 7.98E+02†	4.48E+01 ± 7.18E+01
F04	2.58E+01 ± 6.91E+00†	1.96E+01 ± 4.54E+00†	1.38E+00 ± 4.43E−01†	1.39E−07 ± 1.88E−07
F05	6.44E−01 ± 7.06E−01‡	2.30E+01 ± 2.83E+01$^{\approx}$	2.87E+01 ± 1.78E−01$^{\approx}$	3.63E+00 ± 3.40E+00
F06	0.00E+00 ± 0.00E+00$^{\approx}$	0.00E+00 ± 0.00E+00$^{\approx}$	0.00E+00 ± 0.00E+00$^{\approx}$	0.00E+00 ± 0.00E+00
F07	2.91E−01 ± 8.41E−02†	7.32E−02 ± 1.91E−02†	2.04E−02 ± 1.23E−02†	9.82E−05 ± 7.06E−05
F08	3.82E−04 ± 1.20E−12$^{\approx}$	3.82E−04 ± 3.27E−13$^{\approx}$	5.00E+02 ± 3.60E+02†	3.82E−04 ± 4.54E−13
F09	4.75E−14 ± 2.12E−13†	0.00E+00 ± 0.00E+00$^{\approx}$	9.47E−01 ± 3.20E+00†	0.00E+00 ± 0.00E+00
F10	6.01E−14 ± 1.02E−14†	4.07E−14 ± 4.03E−15†	4.14E−14 ± 1.18E−14†	4.44E−16 ± 0.00E+00
F11	3.26E−05 ± 1.75E−04†	1.85E−13 ± 9.78E−13†	4.76E−03 ± 8.80E−03†	0.00E+00 ± 0.00E+00
F12	1.57E−32 ± 4.63E−34$^{\approx}$	1.57E−32 ± 4.63E−34$^{\approx}$	2.52E−07 ± 7.95E−07†	1.57E−32 ± 5.47E−48
F13	1.35E−32 ± 5.47E−48$^{\approx}$	1.35E−32 ± 5.47E−48$^{\approx}$	6.44E−03 ± 1.41E−02†	1.35E−32 ± 5.47E−48
F14	1.48E−13 ± 2.78E−14†	1.04E−13 ± 2.12E−14†	1.01E−06 ± 3.23E−06†	6.44E−14 ± 1.93E−14
F15	6.50E+03 ± 1.12E+03†	1.07E+04 ± 2.35E+03†	3.86E+03 ± 4.65E+03†	2.94E+02 ± 1.11E+02
F16	9.54E+06 ± 2.46E+06†	1.46E+07 ± 4.67E+06†	6.95E+06 ± 3.54E+06†	2.80E+06 ± 1.02E+06
F17	3.74E+04 ± 5.34E+03†	3.71E+04 ± 6.44E+03†	3.55E+04 ± 8.12E+03†	7.38E+03 ± 1.46E+03
F18	9.22E+03 ± 1.04E+03†	7.14E+03 ± 9.41E+02†	1.03E+04 ± 2.12E+03†	2.43E+03 ± 4.72E+02
F19	5.62E+01 ± 6.87E+01$^{\approx}$	2.10E+01 ± 2.84E+01$^{\approx}$	1.56E+03 ± 6.02E+03†	3.64E+01 ± 4.94E+01
F20	1.59E−02 ± 7.76E−03$^{\approx}$	5.99E−02 ± 4.73E−02†	1.37E+00 ± 2.80E−01†	1.86E−02 ± 1.63E−02
F21	2.09E+01 ± 6.42E−02$^{\approx}$	2.09E+01 ± 4.90E−02$^{\approx}$	2.05E+01 ± 1.67E−01‡	2.09E+01 ± 8.62E−02
F22	6.46E−13 ± 1.52E−12†	9.85E−14 ± 2.51E−14†	5.98E−01 ± 9.57E−01†	6.63E−14 ± 2.12E−14
F23	2.40E+02 ± 3.16E+01†	1.51E+02 ± 4.16E+01†	3.00E+02 ± 5.30E+01†	1.28E+02 ± 1.77E+01
$w/l/t$	15/1/7	15/0/8	19/2/2	–

incorporates the information of global best solution into the solution search equation to improve the exploitation of ABC. The reported experimental results show that GABC is very promising. GOABC (El-Abd 2012) is a generalized opposition-based ABC variant, which also uses the GOBL strategy. Similar to other opposition-based algorithms such as ODE (Rahnamayan et al. 2008) and GOPSO (Wang et al. 2011), in GOABC the GOBL is considered as an independent unit and used to generate opposite solutions of the whole population. However, differs from GOABC, our approach only uses the GOBL in the scout bee phase for the discarded food sources.

For a fair comparison, the same settings of common parameters are used for the above competitive algorithms, i.e., $SN = 30$, $limit = 100$, $MaxFEs = 5000 \cdot D$ for F01–F13 and $MaxFEs = 10{,}000 \cdot D$ for F14–F23. For other specific parameters, $C = 1.5$ for GABC, and $JR = 0.3$ for GOABC, which are the same as in the original literatures. Each algorithm is run 30 times per function, and the mean error and standard deviation values are presented in Tables 6 and 7 for $D = 30$ and 50, respectively. Moreover, to compare the significance between two algorithms, the paired Wilcoxon signed-rank test is used. The Wilcoxon signed-rank test is a non-parametric statistical hypothesis test, which can be used as an alternative to the paired t test when the results cannot be assumed to be normally distributed (García et al. 2009, 2010). "†", "‡" and "≈" indicate our approach is, respectively, better than, worse than, and similar to that of its competitor according to the Wilcoxon signed-ranked test at $\alpha = 0.05$. The competition results are summarized as "$w/l/t$", which denotes that our approach wins on w functions, loses on l functions, and ties on t functions, compared with its competitor.

For $D = 30$, from the results presented in Table 6, it is clear that GBABC performs significantly better than the other three algorithms on the majority of test functions. To be specific, GBABC outperforms ABC on 15 out of 23 test functions, while ABC only achieves better result on the Rosenbrock problem (F05). On this problem, ABC is the winner among the four algorithms. For GABC, it seems not to be better than GBABC on any test function. Note that on the Rastrigin problem (F09), both GABC and GBABC achieve

Table 7 The results achieved by ABC, GABC, GOABC and GBABC at $D = 50$

Functions	ABC Mean error ± std dev	GABC Mean error ± std dev	GOABC Mean error ± std dev	GBABC Mean error ± std dev
F01	2.90E−36 ± 2.86E−36†	1.14E−45 ± 2.18E−45‡	1.80E−52 ± 4.33E−52‡	1.15E−40 ± 1.33E−40
F02	2.37E−19 ± 1.28E−19†	1.72E−23 ± 6.70E−24†	1.15E−32 ± 1.94E−32‡	9.97E−29 ± 5.88E−29
F03	2.11E+04 ± 1.77E+03†	3.00E+04 ± 4.32E+03†	9.40E+02 ± 2.81E+03$^{\approx}$	5.94E+01 ± 6.19E+01
F04	5.77E+01 ± 3.32E+00†	5.11E+01 ± 4.20E+00†	5.83E+00 ± 8.16E−01†	2.46E−11 ± 3.51E−11
F05	7.16E−01 ± 9.67E−01‡	4.53E+01 ± 4.42E+01$^{\approx}$	4.86E+01 ± 1.91E−01$^{\approx}$	4.92E+01 ± 1.86E+01
F06	0.00E+00 ± 0.00E+00$^{\approx}$	0.00E+00 ± 0.00E+00$^{\approx}$	0.00E+00 ± 0.00E+00$^{\approx}$	0.00E+00 ± 0.00E+00
F07	4.68E−01 ± 7.04E−02†	2.29E−01 ± 3.68E−02†	1.23E−02 ± 7.86E−03†	9.08E−05 ± 7.19E−05
F08	6.36E−04 ± 2.49E−09$^{\approx}$	6.36E−04 ± 2.54E−12$^{\approx}$	6.12E+02 ± 2.63E+02†	6.36E−04 ± 6.73E−06
F09	6.61E−11 ± 2.53E−10†	0.00E+00 ± 0.00E+00$^{\approx}$	2.91E−03 ± 1.57E−02†	0.00E+00 ± 0.00E+00
F10	1.22E−13 ± 2.11E−14†	8.27E−14 ± 8.11E−15†	8.00E−14 ± 1.77E−14v	4.44E−16 ± 0.00E+00
F11	4.18E−09 ± 2.02E−08†	6.84E−15 ± 3.46E−14†	3.53E−03 ± 7.98E−03†	0.00E+00 ± 0.00E+00
F12	1.14E−32 ± 2.15E−33†	9.42E−33 ± 2.74E−48$^{\approx}$	3.36E−07 ± 9.53E−07†	9.42E−33 ± 2.74E−48
F13	1.58E−32 ± 2.88E−33†	1.35E−32 ± 5.47E−48$^{\approx}$	1.05E−02 ± 2.26E−02†	1.35E−32 ± 2.21E−34
F14	2.77E−13 ± 3.81E−14†	2.54E−13 ± 2.84E−14†	2.96E−06 ± 1.13E−05†	1.35E−13 ± 2.74E−14
F15	4.74E+04 ± 8.30E+03†	3.44E+04 ± 7.35E+03†	1.49E+04 ± 9.19E+03†	5.61E+03 ± 1.01E+03
F16	4.30E+07 ± 9.96E+06†	3.04E+07 ± 8.24E+06†	9.20E+06 ± 3.46E+06†	7.11Ev06 ± 2.68E+06
F17	1.01E+05 ± 1.12E+04†	1.03E+05 ± 9.32E+03†	8.99E+04 ± 1.68E+04†	3.33E+04 ± 3.56E+03
F18	1.99E+04 ± 1.58E+03†	2.01E+04 ± 1.66E+03†	2.05E+04 ± 1.81E+03†	5.47E+03 ± 7.62E+02
F19	1.21E+02 ± 5.32E+01†	3.76E+01 ± 3.89E+01$^{\approx}$	6.31E+02 ± 1.38E+03†	7.17E+01 ± 7.63E+01
F20	1.58E−01 ± 8.04E−02†	4.32E−02 ± 3.08E−02†	2.18E+00 ± 5.86E−01†	1.39E−02 ± 1.39E−02
F21	2.11E+01 ± 4.02E−02†	2.11E+01 ± 3.56E−02†	2.05E+01 ± 1.51E−01‡	2.10E+01 ± 8.13E−02
F22	2.56E−13 ± 8.53E−14†	2.43E−13 ± 5.07E−14†	2.65E−01 ± 7.37E−01†	1.63E−13 ± 2.84E−14
F23	3.95E+02 ± 5.49E+01†	4.04E+02 ± 7.89E+01†	7.26E+02 ± 8.32E+01†	1.61E+02 ± 1.52E+01
$w/l/t$	20/1/2	16/1/6	17/3/3	–

Table 8 Average rankings of ABC, GABC, GOABC and GBABC at both $D = 30$ and $D = 50$, and the best value is shown in boldface

Algorithms	Average rankings	
	$D = 30$	$D = 50$
ABC	2.93	3.07
GABC	2.50	2.46
GOABC	2.98	2.89
GBABC	**1.59**	**1.58**

Table 9 Parameter settings for ODE, GOPSO, modified BBPSO and MGBDE

Algorithms	Parameter settings
ODE	$NP = 100$, $F = 0.5$, $CR = 0.9$, $JR = 0.3$
GOPSO	$NP = 40$, $\omega = 0.729$, $c_1 = c_2 = 1.496$, $JR = 0.3$
modified BBPSO	$NP = 50$
MGBDE	$NP = 100$, $F = 0.5$, $CR = 0.9$

the global optimum which are much better than other two algorithms. Although GOABC wins GBABC on three test functions, GBABC beats it on the other 19 test functions, especially on the multimodal functions. For $D = 50$, with respect to the four algorithms, although their overall performance deteriorates with the growth of the dimensionality of the problem, GBABC consistently gets significantly better results than its competitors. Specifically, compared with ABC, our approach wins on from 15 test functions at $D = 30$ to 20 test functions at $D = 50$. The Friedman test is also conducted to obtain the average rankings for both $D = 30$ and 50, and Table 8 gives the final results. As seen, GBABC achieves the best average ranking. It means that GBABC is the best one among the four algorithms.1

4.5 Comparison with some related EAs

To further investigate the performance of GBABC, in this subsection, we compare GBABC with four related EAs, including two DE variants and two PSO variants which are listed as follows:

- opposition-based DE (ODE) (Rahnamayan et al. 2008);
- generalized opposition-based PSO (GOPSO) (Wang et al. 2011);

Table 10 The results achieved by ODE, GOPSO, modified BBPSO, MGBDE and GBABC at $D = 30$

Func.	ODE Mean error ± std dev	GOPSO Mean error ± std dev	Modified BBPSO Mean error ± std dev	MGBDE Mean error ± std dev	GBABC Mean error ± std dev
F01	2.40E−11 ± 1.28E−10†	2.49E−44 ± 1.27E−43†	3.09E−48 ± 8.93E−48†	5.73E−38 ± 1.81E−37†	2.63E−49 ± 5.27E−49
F02	3.93E−11 ± 1.49E−10†	1.08E−20 ± 3.10E−20†	1.13E−32 ± 1.28E−32‡	6.25E−23 ± 1.45E−22†	2.53E−31 ± 2.69E−31
F03	1.42E−01 ± 3.39E−01‡	3.62E+03 ± 1.60E+03†	6.98E−01 ± 6.09E−01‡	1.60E+02 ± 9.00E+01†	4.48E+01 ± 7.18E+01
F04	6.19E+00 ± 1.88E+00†	7.20E−03 ± 6.89E−03†	4.26E−04 ± 4.22E−04†	4.95E−04 ± 5.14E−04†	1.39E−07 ± 1.88E−07
F05	6.51E+01 ± 4.36E+01†	2.42E+01 ± 1.33E+00$^{\approx}$	3.80E+01 ± 2.90E+01$^{\approx}$	2.52E+01 ± 1.78E+01$^{\approx}$	3.63E+00 ± 3.40E+00
F06	1.23E+02 ± 1.12E+02†	0.00E+00 ± 0.00E+00$^{\approx}$	0.00E+00 ± 0.00E+00$^{\approx}$	0.00E+00 ± 0.00E+00$^{\approx}$	0.00E+00 ± 0.00E+00
F07	2.78E−02 ± 2.23E−02†	7.81E−03 ± 3.94E−03†	5.72E−03 ± 1.65E−03†	4.79E−03 ± 1.80E−03†	9.82E−05 ± 7.06E−05
F08	4.36E+03 ± 1.68E+03†	8.68E+03 ± 4.16E+02†	9.82E+02 ± 2.96E+02†	5.42E+02 ± 2.02E+02†	3.82E−04 ± 4.54E−13
F09	3.95E+01 ± 2.49E+01†	1.69E+02 ± 3.92E+01†	1.82E+01 ± 5.42E+00†	5.57E+00 ± 2.58E+00†	0.00E+00 ± 0.00E+00
F10	1.08E+00 ± 9.51E−01†	1.55E−14 ± 4.27E−15†	8.50E−15 ± 2.42E−15†	7.31E−15 ± 8.86E−16v	4.44E−16 ± 0.00E+00
F11	1.09E−01 ± 3.68E−01†	6.57E−03 ± 8.15E−03†	3.04E−03 ± 5.07E−03†	5.75E−04 ± 2.18E−03†	0.00E+00 ± 0.00E+00
F12	7.95E−02 ± 1.22E−01†	3.46E−03 ± 1.86E−02†	3.46E−03 ± 1.86E−02†	1.57E−32 ± 5.47E−48$^{\approx}$	1.57E−32 ± 5.47E−48
F13	6.43E−01 ± 2.19E+00†	1.46E−03 ± 3.73E−03$^{\approx}$	1.10E−03 ± 3.30E−03†	1.35E−32 ± 5.47E−48v	1.35E−32 ± 5.47E−48
F14	6.54E−04 ± 3.52E−03†	7.58E−14 ± 3.06E−14†	5.68E−14 ± 0.00E+00‡	4.93E−14 ± 1.93E−14‡	6.44E−14 ± 1.93E−14
F15	1.64E−01 ± 7.22E−01‡	3.12E+03 ± 2.94E+03†	4.28E−04 ± 1.20E−03‡	5.29E+00 ± 5.69E+00‡	2.94E+02 ± 1.11E+02
F16	6.59E+05 ± 2.48E+05‡	2.00E+07 ± 1.79E+07†	1.68E+06 ± 7.77E+05$^{\approx}$	6.79E+06 ± 3.16E+06†	2.80E+06 ± 1.02E+06
F17	2.75E+00 ± 6.02E+00‡	6.43E+03 ± 2.61E+03‡	1.30E+02 ± 8.13E+01‡	1.24E+02 ± 7.71E+01‡	7.38E+03 ± 1.46E+03
F18	2.77E+03 ± 6.33E+02†	9.24E+03 ± 1.76E+03†	3.69E+03 ± 8.96E+02†	2.04E+03 ± 4.69E+02$^{\approx}$	2.43E+03 ± 4.72E+02
F19	7.03E+05 ± 1.94E+06†	4.77E+01 ± 6.48E+01†	8.02E+01 ± 8.17E+01†	2.98E+01 ± 2.67E+01‡	3.64E+01 ± 4.94E+01
F20	2.20E−02 ± 1.64E−02†	2.62E+00 ± 7.04E+00†	1.33E−02 ± 9.53E−03$^{\approx}$	1.14E−02 ± 1.25E−02$^{\approx}$	1.86E−02 ± 1.63E−02
F21	2.09E+01 ± 5.82E−02$^{\approx}$	2.09E+01 ± 8.89E−02$^{\approx}$	2.10E+01 ± 4.06E−02†	2.09E+01 ± 5.54E−02$^{\approx}$	2.09E+01 ± 8.62E−02
F22	3.14E+01 ± 1.25E+01†	7.13E+01 ± 5.39E+01†	2.00E+01 ± 5.01E+00†	5.94E+00 ± 2.84E+00†	6.63E−14 ± 2.12E−14
F23	5.92E+01 ± 2.62E+01‡	2.18E+02 ± 1.66E+01†	7.67E+01 ± 3.91E+01‡	1.62E+02 ± 2.14E+01†	1.28E+02 ± 1.77E+01
$w/l/t$	17/5/1	18/1/4	13/6/4	12/4/7	–

Table 11 Average rankings of ODE, GOPSO, modified BBPSO, MGBDE and GBABC at $D = 30$, and the best value is shown in boldface

Algorithms	Average rankings
ODE	3.80
GOPSO	4.07
modified BBPSO	2.70
MGBDE	2.35
GBABC	**2.09**

- modified bare-bones PSO (modified BBPSO) (Kennedy 2003);
- modified Gaussian bare-bones DE (MGBDE) (Wang et al. 2013).

Both ODE and GOPSO use the (generalized) opposition-based learning strategy, while modified BBPSO and MGBDE use the bare-bones technique. For a fair comparison, the control parameters of these competitive EAs are set to the suggested values offered by their corresponding literatures. Table 9 shows the control parameter settings in detail. The stopping criterion is the same as previous subsections. Each algorithm is run 30 times per function, and the mean error and standard deviation values are given in Table 10.

With respect to the overall performance, from Table 10, we can see that GBABC obtains significantly better results on the majority of test functions compared with its competitors. To be specific, ODE wins on 5 test functions compared with GBABC, but on other 17 test functions GBABC performs better. Note that ODE performs the best on the test functions F16 and F17 among the five algorithms. The reasons may be twofold: first, ODE employs the classic "DE/rand/1" mutation scheme, which has no bias to any special search direction and can choose new search directions in a random manner. Therefore, it is sufficient to search the complex shifted and rotated fitness landscape. Second, the OBL strategy employed by ODE can provide more chances of finding the global optimum, which implies that more regions can be searched by ODE. For GOPSO, it outperforms GBABC on only one test function. Modified BBPSO performs better

Table 12 Comparison between GBABC and DEs on optimizing 30-dimensional problems

Functions	$MaxFEs$	DE Mean error ± std dev	jDE Mean error ± std dev	JADE Mean error ± std dev	SaDE Mean error ± std dev	GBABC Mean error ± std dev
Sphere	150,000	9.80E−14 ± 8.40E−14	1.46E−28 ± 1.78E−28	2.69E−56 ± 1.41E−55	3.28E−20 ± 3.63E−20	**2.63E−49 ± 5.27E−49**
Schwefel 2.22	200,000	1.60E−09 ± 1.10E−09	9.02E−24 ± 6.01E−24	3.18E−25 ± 2.05E−24	3.51E−25 ± 2.74E−25	**5.84E−42 ± 3.81E−42**
Schwefel 1.2	500,000	1.16E−13 ± 1.73E−13	1.93E−56 ± 7.02E−56	**6.11E−81 ± 1.62E−80**	1.54E−14 ± 4.56E−14	1.10E+00 ± 6.34E−01
Schwefel 2.21	500,000	2.44E−14 ± 1.64E−13	1.34E−09 ± 5.71E−10	5.29E−14 ± 2.05E−14	**6.39E−27 ± 8.27E−27**	8.71E−25 ± 8.59E−25
Rosenbrock	2000,000	2.10E+00 ± 1.50E+00	1.30E+01 ± 1.40E+01	3.2E−01 ± 1.10E+00	2.10E+01 ± 7.80E+00	**9.66E−02 ± 1.79E−01**
Step	10,000	4.70E+03 ± 1.20E−03	6.13E+02 ± 1.72E+02	5.62E+00 ± 1.87E+00	5.07E+01 ± 1.34E+01	**6.00E−01 ± 8.00E−01**
Quartic	300,000	4.70E−03 ± 1.20E−03	3.35E−03 ± 8.68E−04	6.14E−04 ± 2.55E−04	4.86E−03 ± 5.21E−04	**7.46E−05 ± 2.03E−05**
Schwefel 2.26	100,000	5.90E+03 ± 1.10E+03	**1.70E−10 ± 1.71E−10**	2.62E−04 ± 3.59E−04	1.13E−08 ± 1.08E−08	3.82E−04 ± 7.28E−13
Rastrigin	100,000	1.80E+02 ± 1.30E+01	3.32E−04 ± 6.39E−04	1.33E−01 ± 9.74E−02	2.43E+00 ± 1.60E+00	**0.00E+00 ± 0.00E+00**
Ackley	50,000	1.10E−01 ± 3.90E−02	2.37E−04 ± 7.10E−05	**3.35E−09 ± 2.84E−09**	3.81E−06 ± 8.26E−07	1.91E−07 ± 5.17E−08
Griewank	50,000	2.00E−01 ± 1.10E−01	7.29E−06 ± 1.05E−05	1.57E−08 ± 1.09E−07	2.52E−09 ± 1.24E−08	**1.78E−10 ± 2.29E−10**
Penalized 1	50,000	1.20E−02 ± 1.00E−02	7.03E−08 ± 5.74E−08	1.67E−15 ± 1.02E−14	8.25E−12 ± 5.12E−12	**4.20E−16 ± 6.59E−16**
Penalized 2	50,000	7.50E−02 ± 3.80E−02	1.80E−05 ± 1.42E−05	1.87E−10 ± 1.09E−09	1.93E−09 ± 1.53E−09	**2.76E−15 ± 3.16E−15**
$w/l/t$		13/0/0	12/1/0	11/2/0	12/1/0	–

Table 13 Comparison between GBABC and PSOs on optimizing 30-dimensional problems

Functions	FIPS Mean error ± std dev	HPSO-TVAC Mean error ± Std dev	CLPSO Mean error ± std dev	OLPSO Mean error ± std dev	GBABC Mean error ± std dev
Sphere	2.42E−13 ± 1.73E−13	2.83E−33 ± 3.19E−33	1.58E−12 ± 7.70E−13	4.12E−54 ± 6.34E−54	**3.23E−67 ± 3.17E−67**
Schwefel 2.22	2.76E−08 ± 9.04E−09	9.03E−20 ± 9.58E−20	2.51E−08 ± 5.84E−09	9.85E−30 ± 1.01E−29	**5.84E−42 ± 3.81E−42**
Rosenbrock	2.51E+01 ± 5.10E−01	2.39E+01 ± 2.65E+01	1.13E+01 ± 9.85E−00	2.15E+01 ± 2.99E+01	**1.70E+00 ± 1.55E+00**
Quartic with noise	4.24E−03 ± 1.28E−03	9.82E−02 ± 3.26E−02	5.85E−03 ± 1.11E−03	1.16E−02 ± 4.10E−03	**1.31E−04 ± 6.95E−05**
Schwefel 2.26	9.93E+02 ± 5.09E+02	1.59E+03 ± 3.26E+02	**3.82E−04 ± 1.28E−05**	3.84E+02 ± 2.17E+02	**3.82E−04 ± 8.91E−13**
Rastrigin	6.51E+01 ± 1.33E+01	9.43E−00 ± 3.48E−00	9.09E−05 ± 1.25E−04	1.07E−00 ± 9.92E−01	**0.00E+00 ± 0.00E+00**
Ackley	2.33E−07 ± 7.19E−08	7.29E−14 ± 3.00E−14	3.66E−07 ± 7.57E−08	7.98E−15 ± 2.03E−15	**4.44E−16 ± 0.00E+00**
Griewank	9.01E−12 ± 1.84E−11	9.75E−03 ± 8.33E−03	9.02E−09 ± 8.57E−09	4.83E−03 ± 8.63E−03	**0.00E+00 ± 0.00E+00**
Penalized 1	1.96E−15 ± 1.11E−15	2.71E−29 ± 1.88E−28	6.45E−14 ± 3.70E−14	1.59E−32 ± 1.03E−33	**1.57E−32 ± 5.47E−48**
Penalized 2	2.70E−14 ± 1.57E−14	2.79E−28 ± 2.18E−28	1.25E−12 ± 9.45E−12	4.39E−04 ± 2.20E−03	**1.35E−32 ± 5.47E−48**
Shifted Rosenbrock	4.25E+02 ± 2.54E+01	4.94E+02 ± 9.65E+01	4.03E+02 ± 1.35E+01	4.25E+02 ± 3.48E+01	**3.91E+02 ± 1.92E+00**
Shifted Rastrigin	−2.46E+02 ± 2.21E+01	−3.18E+02 ± 5.75E+00	**−3.30E+02 ± 3.39E−05**	−3.28E+02 ± 1.04E+00	**−3.30E+02 ± 3.60E−14**
$w/l/t$	12/0/0	12/0/0	10/0/2	12/0/0	–

than GBABC on six test functions, while GBABC outperforms it on other 13 test functions. It is impressive that modified BBPSO achieves the best results on the unimodal test function F02 (Schwefel 2.22 problem) compared with other four algorithms. The possible reason is that modified BBPSO provides the new candidate solution a 50 % chance to inherit the elements from its personal best position, and thus it also has good exploitation capability. Compared with MGBDE, GBABC wins on 12 test functions and loses on other 4 ones.

Table 11 shows the average rankings of the five algorithms based on Friedman test. As seen, GBABC achieves the best average ranking. The remaining algorithms can be sorted by their average rankings into the following order: MGBDE, modified BBPSO, ODE and GOPSO.

4.6 Comparison with some state-of-the-art EAs

In this subsection, the proposed GBABC is further compared with some state-of-the-art EAs which include different DE, PSO and ABC variants, they are: (1) basic DE (Storn and Price 1997), self-adapting control parameters DE (jDE) (Brest et al. 2006), adaptive DE with optional external archive (JADE) (Zhang and Sanderson 2009), DE with strategy adaptation (SaDE) (Qin et al. 2009); (2) fully informed PSO (FIPS) (Mendes et al. 2004), self-organizing hierarchical PSO with time-varying acceleration coefficients (HPSO-TVAC) (Ratnaweera et al. 2004), comprehensive learning PSO (CLPSO) (Liang et al. 2006) and orthogonal learning PSO (OLPSO) (Zhan et al. 2011); (3) memetic ABC (MeABC) (Bansal et al. 2013) and opposition-based lévy flight ABC (OBLFABC) (Sharma et al. 2013). Table 12 presents the comparison between GBABC and DEs on optimizing 30-dimensional functions. All the results of these DEs are based on the reports in the literatures (Zhang and Sanderson 2009; Gong et al. 2011). From the results, we can see that GBABC performs the best on the majority of test functions. Specifically, basic DE seems not to be better than GBABC on any function. Both jDE and SaDE only win GBABC on one function, while GBABC beats them on the remaining 12 functions. Compared with JADE, GBABC does better on 11 functions. The compared results of GBABC and PSOs are reported in Table 13. The results of PSOs are all directly derivated from the literature (Zhan et al. 2011), and the suggested stopping criterion $MaxFEs$ equals to 200,000. As seen, for FIPS, HPSO-TVAC and OLPSO, none of them seems to perform better than GBABC on any used test function. GBABC works better than CLPSO on 10 out of 12 functions except on Schwefel 2.26 and Shifted Rastrigin on which both two algorithms have the same performance. Table 14 presents the comparison among MeABC, OBLFABC, and GBABC on optimizing 30-dimensional functions. The stopping criteria for these three algorithms are either maximum number of function evaluations (which is set to be 200,000) is reached or the acceptable error (mentioned in Table 14) has been achieved. The number of run is 100. All the results of MeABC and OBLFABC are based on the reports in the literature (Bansal et al. 2013). The success rate (SR) and average number of function evaluations (AFE) are also reported in the Table 14. From this table, we can see that GBABC outperforms MeABC and OBLFABC both on 4 out of 5 test functions, while MeABC and OBLFABC achieve better results on the Shifted Rosenbrock problem. As for the SR, however, all the three algorithms fail to get satisfied results on the Shifted schwefel problem. In short, the whole comparisons demonstrates that GBABC shows superior performance in terms of the quality of the final solutions.

4.7 Application to a real-world optimization problem

In this section, we further investigate the performance of our approach over a real-world problem, i.e., the frequency modulated (FM) sound wave synthesis problem (Das and Suganthan 2010; Bansal et al. 2013). The FM sound wave synthesis plays an important role in several modern music systems. It provides a simple and efficient method creating complex sound timbres. The parameter optimization of a FM synthesizer is a six-dimensional optimization problem where the vector to be optimized is $X = \{a_1, w_1, a_2, w_2, a_3, w_3\}$ of the sound wave given in Eq. (13). The problem is to generate a sound [generated by Eq. (13)] similar to the object sound [generated by Eq. (14)]. The formulas for the estimated sound wave and the target sound are given as follows:

$$y(t) = a_1 \sin(w_1 t\theta + a_2 \sin(w_2 t\theta) + a_3 \sin(w_3 t\theta))) \quad (13)$$

$$y_0(t) = 1.0 \sin(0.5 t\theta + 1.5 \sin(4.8 t\theta) + 2.0 \sin(4.9 t\theta))) \quad (14)$$

where $\theta = 2\pi/100$ and $X_i \in [-6.4, 6.35]$.

The goal of this problem is to minimize the sum of squared errors between the estimated sound and the target sound, as given by Eq. (15). This problem is a highly complex multimodal one having strong epistasis, with minimum value $f(X) = 0$. $f(X)$ can be expressed as follows:

$$f(x) = \sum_{t=0}^{100} (y(t) - y_0(t))^2 \quad (15)$$

In this experiment, the basic ABC, GABC, GOABC and GBABC are applied to solve this problem. The parameter settings for all these four algorithms are kept the same as described in Sect. 4.4. The stopping criterion *MaxFEs* is set to 200,000. Each algorithm is run 30 times, and the mean and standard deviation values are reported in Table 15. From the results, it is clear that GBABC performs better than other three algorithms in terms of the quality of the final solutions.

Table 14 Comparison among MeABC, OBLFABC, and GBABC on optimizing 30-dimensional problems

Functions	Acceptable error	Measure	MeABC	OBLFABC	GBABC
Quartic	1.00E+00	Mean error	9.20E+00	9.59E+00	**9.00E−01**
		Std dev	4.02E−01	5.14E−01	**7.41E−02**
		AFE	200,017.89	200,030.13	2,740
		SR	0	0	100
Shifted Rosenbrock	1.00E−01	Mean error	**1.03E−01**	6.40E−01	3.91E+02
		Std dev	**7.83E−02**	2.87E+00	1.92E+00
		AFE	103,949.03	62,464.38	200,000
		SR	97	88	0
Shifted sphere	1.00E−05	Mean error	7.86E−06	7.72E−06	**5.11E−06**
		Std dev	1.92E−06	2.29E−06	**9.28E−07**
		AFE	5,535.34	6,718.35	4,300
		SR	100	100	100
Shifted Rastrigin	1.00E−02	Mean error	8.23E+01	9.01E+01	**7.10E−03**
		Std dev	1.24E+01	1.16E+01	**2.84E−03**
		AFE	200,012.07	200,031	70,200
		SR	0	0	100
Shifted schwefel	1.00E−05	Mean error	1.05E+04	1.12E+04	**1.32E+03**
		Std dev	3.33E+03	3.07E+03	**5.76E+02**
		AFE	200,025.38	200,032.45	200,000
		SR	0	0	0

Table 15 The results of ABC, GABC, GOABC and GBABC on the FM sound wave synthesis problem, and the best value is shown in boldface

Algorithms	Mean	Std dev
ABC	4.30E+00	4.41E+00
GABC	3.45E+00	4.98E+00
GOABC	1.26E+01	5.36E+00
GBABC	**2.01E+00**	**3.47E+00**

This paper only presents a comparative study on the FM sound wave synthesis problem. More applications to real-world problems will be investigated in the future work (Das and Suganthan 2010; Sharma et al. 2012; Bansal et al. 2013).

5 Conclusions

This paper presents a Gaussian bare-bones ABC (GBABC) algorithm, which aims to improve the exploitation of ABC. In GBABC, the solution search equation of onlooker bees is replaced with a new one—Gaussian bare-bones search equation. The Gaussian bare-bones search equation uses a Gaussian distribution to sample the search space between the current solution and the global best solution of current population. During the evolution of the algorithm, the search behavior of this new search equation gradually varies from exploration to exploitation. It is beneficial to balance the exploitation and exploration capabilities. Furthermore, to preserve the search experiences of the scout bees, the GOBL strategy is employed to generate opposite solutions of the discarded food sources in the scout bee phase. A comprehensive set of experiments is conducted on 23 benchmark functions and a real-world optimization problem to verify the performance of our approach, and some other well-known ABCs and state-of-the-art EAs are used for comparision. The experimental results demonstrate that our approach achieves better performance in term of solution accuracy and convergence speed.

Acknowledgments This work was supported by the National Natural Science Foundation of China (Grant Nos. 61305150, 61364025, and 61462045), the Science and Technology Plan Projects of Jiangxi Provincial Education Department (Nos. GJJ13729 and GJJ14747), the Science and Technology Foundation of Jiangxi Province (No. 20142BAB217020), and the Humanity and Social Science Foundation of Ministry of Education of China (No. 13YJCZH174).

References

Akay B, Karaboga D (2012) A modified artificial bee colony algorithm for real-parameter optimization. Inf Sci 192:120–142

Alatas B (2010) Chaotic bee colony algorithms for global numerical optimization. Exp Syst Appl 37(8):5682–5687

Auger A, Hansen N (2005) A restart cma evolution strategy with increasing population size. Proc IEEE Congr Evolut Comput, IEEE 2:1769–1776

Bäck T (1996) Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms, vol 996. Oxford University Press, Oxford

Banharnsakun A, Achalakul T, Sirinaovakul B (2011) The best-so-far selection in artificial bee colony algorithm. Appl Soft Comput 11(2):2888–2901

Bansal JC, Sharma H, Arya K, Nagar A (2013) Memetic search in artificial bee colony algorithm. Soft Comput 17(10):1911–1928

Bansal JC, Sharma H, Arya K, Deep K, Pant M (2014) Self-adaptive artificial bee colony. Optimization 63(10):1513–1532

Beyer HG, Schwefel HP (2002) Evolution strategies—a comprehensive introduction. Nat Comput 1(1):3–52

Brest J, Greiner S, Boskovic B, Mernik M, Zumer V (2006) Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans Evolut Comput 10(6):646–657

Clerc M, Kennedy J (2002) The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans Evolut Comput 6(1):58–73

Das S, Suganthan P (2010) Problem definitions and evaluation criteria for cec 2011 competition on testing evolutionary algorithms on real world optimization problems. Jadavpur University, Nanyang Technological University, Kolkata

Das S, Suganthan PN (2011) Differential evolution: a survey of the state-of-the-art. IEEE Trans Evolut Comput 15(1):4–31

Das S, Biswas S, Kundu S (2013) Synergizing fitness learning with proximity-based food source selection in artificial bee colony algorithm for numerical optimization. Appl Soft Comput 13(12):4676–4694

Dorigo M, Di Caro G (1999) Ant colony optimization: a new meta-heuristic. In: Proceedings of IEEE congress on evolutionary computation, IEEE, vol 2

Eberhart RC, Shi Y (2001) Particle swarm optimization: developments, applications and resources. Proc IEEE Congr Evolut Comput, IEEE 1:81–86

El-Abd M (2012) Generalized opposition-based artificial bee colony algorithm. In: IEEE congress on evolutionary computation, IEEE, pp 1–4

Gao W, Liu S (2011) Improved artificial bee colony algorithm for global optimization. Inf Process Lett 111(17):871–882

Gao W, Liu S (2012) A modified artificial bee colony algorithm. Comput Oper Res 39(3):687–697

Gao W, Liu S, Huang L (2012) A global best artificial bee colony algorithm for global optimization. J Comput Appl Math 236(11):2741–2753

Gao W, Liu S, Huang L (2013a) A novel artificial bee colony algorithm based on modified search equation and orthogonal learning. IEEE Trans Cybern 43(3):1011–1024

Gao W, Liu S, Huang L (2013b) A novel artificial bee colony algorithm with powell's method. Appl Soft Comput 13(9):3763–3775

García S, Molina D, Lozano M, Herrera F (2009) A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization. J Heuristics 15(6):617–644

García S, Fernández A, Luengo J, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Inf Sci 180(10):2044–2064

Garro BA, Sossa H, Vázquez RA (2011) Artificial neural network synthesis by means of artificial bee colony (ABC) algorithm. In: IEEE congress on evolutionary computation, IEEE, pp 331–338

Gong W, Cai Z, Ling CX, Li H (2011) Enhanced differential evolution with adaptive strategies for numerical optimization. IEEE Trans Syst Man Cybern Part B: Cybern 41(2):397–413

Kang F, Li J, Xu Q (2009) Structural inverse analysis by hybrid simplex artificial bee colony algorithms. Comput Struct 87(13):861–870

Kang F, Li J, Ma Z (2011) Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions. Inf Sci 181(16):3508–3531

Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Technical repprt TR06, Erciyes University

Karaboga D, Akay B (2009) A comparative study of artificial bee colony algorithm. Appl Math Comput 214(1):108–132

Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Optim 39(3):459–471

Karaboga D, Gorkemli B (2014) A quick artificial bee colony (qABC) algorithm and its performance on optimization problems. Appl Soft Comput 23:227–238

Karaboga D, Gorkemli B, Ozturk C, Karaboga N (2012a) A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artif Intell Rev 42(1):21–57

Karaboga D, Ozturk C, Karaboga N, Gorkemli B (2012b) Artificial bee colony programming for symbolic regression. Inf Sci 209(20):1–15

Kennedy J (2003) Bare bones particle swarms. In: IEEE swarm intelligence symposium, IEEE, pp 80–87

Kennedy J, Eberhart R (1995) Particle swarm optimization. IEEE Int Conf Neural Netw 4:1942–1948

Li G, Niu P, Xiao X (2012) Development and investigation of efficient artificial bee colony algorithm for numerical function optimization. Appl Soft Comput 12(1):320–332

Liang JJ, Qin AK, Suganthan PN, Baskar S (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evolut Comput 10(3):281–295

Mendes R, Kennedy J, Neves J (2004) The fully informed particle swarm: simpler, maybe better. IEEE Trans Evolut Comput 8(3):204–210

Neri F, Tirronen V (2010) Recent advances in differential evolution: a survey and experimental analysis. Artif Intell Rev 33(1):61–106

Qin AK, LHuang V, Suganthan PN (2009) Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans Evolut Comput 13(2):398–417

Rahnamayan S, Tizhoosh HR, Salama MMA (2008) Opposition-based differential evolution. IEEE Trans Evolut Comput 12(1):64–79

Rajasekhar A, Abraham A, Pant M (2011a) Design of fractional order pid controller using sobol mutated artificial bee colony alogrithm. In: International conference on hybrid intelligent systems, IEEE, pp 151–156

Rajasekhar A, Abraham A, Pant M (2011b) Levy mutated artificial bee colony algorithm for global optimization. In: IEEE international conference on systems, man, and cybernetics, IEEE, pp 655–662

Ratnaweera A, Halgamuge SK, Watson HC (2004) Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans Evolut Comput 8(3):240–255

Shang Y, Qiu Y (2006) A note on the extended Rosenbrock function. Evolut Comput 14(1):119–126

Sharma H, Bansal JC, Arya K (2012) Fitness based differential evolution. Memet Comput 4(4):303–316

Sharma H, Bansal JC, Arya K (2013) Opposition based lévy flight artificial bee colony. Memet Comput 5(3):213–227

Sharma TK, Pant M (2013) Enhancing the food locations in an artificial bee colony algorithm. Soft Comput 17(10):1939–1965

Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11(4):341–359

Suganthan PN, Hansen N, Liang JJ, Deb K, Chen YP, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization. Technical report, Nanyang Technological University, Singapore

Szeto W, Wu Y, Ho SC (2011) An artificial bee colony algorithm for the capacitated vehicle routing problem. Eur J Oper Res 215(1):126–135

Tang KS, Man K, Kwong S, He Q (1996) Genetic algorithms and their applications. IEEE Signal Process Mag 13(6):22–37

Tizhoosh HR (2005) Opposition-based learning: a new scheme for machine intelligence. In: International conference on Computational intelligence for modelling, control and automation, IEEE, pp 695–701

TSai PW, Pan JS, Liao BY, Chu SC (2009) Enhanced artificial bee colony optimization. Int J Innov Comput Inf Control 5(12):5081–5092

van den Bergh F, Engelbrecht AP (2006) A study of particle swarm optimization particle trajectories. Inf Sci 176(8):937–971

Wang H, Wu Z, Rahnamayan S, Liu Y, Ventresca M (2011) Enhancing particle swarm optimization using generalized opposition-based learning. Inf Sci 181(20):4699–4714

Wang H, Rahnamayan S, Sun H, Omran MG (2013) Gaussian bare-bones differential evolution. IEEE Trans Cybern 43(2):634–647

Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evolut Comput 3(2):82–102

Yeh WC, Hsieh TJ (2012) Artificial bee colony algorithm-neural networks for s-system models of biochemical networks approximation. Neural Comput Appl 21(2):365–375

Zhan Z, Zhang J, Li Y, Shi Y (2011) Orthogonal learning particle swarm optimization. IEEE Trans Evolut Comput 15(6):832–847

Zhang J, Sanderson AC (2009) JADE: adaptive differential evolution with optional external archive. IEEE Trans Evolut Comput 13(5):945–958

Zhu G, Kwong S (2010) Gbest-guided artificial bee colony algorithm for numerical function optimization. Appl Math Comput 217(7):3166–3173