Noname manuscript No.
(will be inserted by the editor)

Artificial Bee Colony Algorithm with Directed Scout

Radhwan A.A. Saleh · Rustu Akay

Received: date / Accepted: date

Abstract As a relatively new model, the Artificial Bee Colony Algorithm (ABC) has shown impressive success in solving optimization problems. Nevertheless, its efficiency is still not satisfactory for some complex optimization problems. This paper has modified ABC and its other recent variants to improve its performance by modify the scout phase. This modification enhances its exploitation ability by intensifying the regions in the search space, which probably includes reasonable solutions. **The experiments were performed on the CEC2014 benchmark suite, CEC2015 benchmark functions, and three real-life problems: pressure vessel design problem, tension and compression spring design problem, and Frequency-Modulated (FM) problem. And the proposed modification was applied to basic ABC, Gbest-Guided ABC, Depth First Search ABC, and Teaching Learning Based ABC, and they were compared with their modified counterparts.** The results have shown that our modification can successfully increase the performance of the original versions. Moreover, the proposed modified algorithm was compared with the state-of-the-art optimization algorithms, and it produced competitive results.

Keywords Global optimization, population-based meta-heuristic, artificial bee colony, scout bees, engineering optimization.

Radhwan A.A. Saleh
Institute of Natural and Applied Sciences, Erciyes University, Kayseri, Turkey
E-mail: 206113002@kocaeli.edu.tr

R. Akay
Department of Mechatronics Engineering, Erciyes University, Kayseri, Turkey
E-mail: akay@erciyes.edu.tr

1 Introduction

Most of the real-world problems are non-convex, non-differential, or do not have implicit equations. For conventional gradient-based optimization approaches, solving these problems is rather tricky [61]. Meta-heuristics have shown superior performance in solving these problems as a practical solution and have attracted researchers because they are simple; thus, scientists from different fields can quickly learn and apply them to solve their problems. They are also flexible; therefore, they do not need any specific modification in their structure to use them in any application. They are derivative-free; consequently, they are highly optimized for real problems with costly or unknown derivative data. Finally, they can avoid local optima because of their stochastic nature, making them a good option for real problems [43]. These characteristics motivated the scientists to simulate various natural concepts, introduce new meta-heuristics, hybridize two or more, or enhance the current meta-heuristics.

The P-meta-heuristic algorithms in the literature are application dependent; that is, one algorithm might show promising results in one specific application but might show bad results in other applications [55]. This fact motivates researchers to continue developing novel P-meta-heuristic algorithms or at least modifying and improving the available ones. **Accordingly, in this work, we are going to introduce a novel version of P-meta-heuristic algorithms which is based on one of the most common P-meta-heuristic algorithms, called Artificial Bee Colony (ABC).**

Recently, ABC algorithm, initially developed in 2005 by D. Karaboga [29], has proved its efficiency in solving many optimization problems in different applications [33]. **Indeed, it is one of the best algorithms of**

population-based meta-heuristics with citation exceeds 6813 in google scholar. Its simple structure, low number of parameters, and ease of implementation have attracted many researchers to use its versions on different application. Although the performance of ABC in solving complex optimization problems is competitive compared to other population-based optimizer algorithms, which is due to its good exploration, it is still not good enough because of its poor exploitation [25]. That is why many researchers tried to modify the original ABC and proposed new versions. **Despite the high performance of the modified versions of ABC in literature, they may be improved furthermore if we directed the scout search to exploit the promising search area. It is worthy to mention that most ABC versions have focused on their modifications on employed and onlooker bee phases as we will see in the following review.**

In both employed and onlooker bee phases of ABC, only one of the parent solution is modified. This update mechanism during the search along the current axis ignores the other axes, which leads to weak exploitation [2]. Besides, in the scout phase, the abandoned food source is replaced by a new random source, which confirms that the ABC search equation and structure focus more on exploration. At the same time, the effective meta-heuristic depends heavily on a good compromise between exploration and exploitation. Therefore, the main challenge is to improve the performance of the ABC algorithm in balancing exploration and exploitation of the solution search equation. **Hence, improving ABC performance has become an active research topic.**

The versions of ABC could be classified into three categories as the following: versions that keep the framework of ABC but modify its search equations, versions that modify both the search equations and the framework of ABC, and versions that hybrid ABC with other algorithms.

According to the versions which keep the framework of ABC but modify its search equations, the reason behind the weakness of exploitation is the search equations of ABC. Since the update during the search is along the current axis and ignores the other axes [2]. As a result, many versions of ABC just have modified its search equations as the following versions. Inspired by PSO search equations, a gbest-guided ABC (GABC) was introduced. This new version utilized the knowledge obtained from the best global solution to improve the speed of convergence in the search equation [62]. In [2], two control parameters are presented and added to ABC to control the magnitude and frequency of perturbation. In [51], the interactive ABC was used by including the principle of universal gravitation in the search solution equation. In [16] two search equations were designed to maintain an excellent balance between the exploration and exploitation of ABC. In this work, a control parameter was developed in which two search equations, focusing respectively on exploitation and exploration, are combined with balancing exploitation with exploration (called MABC). A Gaussian ABC was provided using a parameter to control gaussian and uniform distributions frequency [47]. PS-MEABC inspired the search equation, which uses the best solution and other good solutions from PSO [57]. Another way of using the best global knowledge in the onlooker phase has been developed [41]. The results of the experiments show that, in some cases, the he proposed approach beats the other approaches.

According to the versions which modify both the search equations and the framework of ABC, not only the search equations are the reason for weakness but also the searching strategy needs to focus more on exploitation. For example, in [15], a parameter was introduced to balance the search equations and introduced an incoming search strategy. [17] suggested a new search equation similar to the Genetic Algorithm crossover procedure (named CABG). Besides, the orthogonal learning method was integrated with three different versions of ABC to find out more valuable information during the searching process. The simulation results show that CABG is competitive and effective. The Gaussian search equation has been proposed to provide solutions for the onlooker phase. This equation exploits the hidden information found in the best solution to improve the exploitation [14]. Then they developed two new equations to act as search equations for both onlooker and employee phases and used Powell's method to enhance the exploitation [19]. A quick ABC (qABC), which employs a novel search equation for the onlooker phase, was introduced to exploit the neighbors of the best solution [32]. The directional information was appended to ABC to create a new search strategy to construct a candidate solution based on the previous directional information (called dABC) [35]. A self-adaptive local search strategy is implemented with ABC to exploit the searching area near the best solution, which leads to accelerating ABC convergence [24]. A depth-first search (DFS) framework, which assigns more computing resources to higher quality food sources, and two search equations, which are used to exploit the elite solutions, are added to enhance the performance of ABC (called DFSABC_elite). It was re-

ported that the suggested framework could speed up the convergence rate, and DFSABC_elite is more robust than the compared algorithms [9]. In [52], the mechanism used to select the food sources in the onlooker phase of ABC is replaced by a new mechanism that depends on neighborhood radius in a ring topology. Furthermore, with the use of the opposition-based learning and neighborhood radius concept, the scout phase is strengthened. In [60], MGABC used the advantages of multi-elite guidance to increase the exploitative ability of ABC without affecting its explorative ability. For this purpose, two novel solution search equations are used instead of the original ones in employed and onlooker bee phases, respectively, furthermore a modified neighborhood search operator is developed.

According to the versions which hybrid ABC with other algorithms, the success hybrid version combines the advantages of its components. Thus, the success hybrid version of ABC should increase its exploitation ability. For example, Rosenbrock ABC, referred to as RABC, wherein ABC do the exploration phase, and the rotational path approach completes the exploitation phase. In [56], the chaotic mapping technique is used in the initialization and scout phase to increase ABC's performance and quality (called ERABC). In [27] Hooke–Jeeves pattern search is combined with ABC, where the exploitation is carried out by pattern search. Its results showed that it is promising about convergence speed, the accuracy of the solution, and efficiency. The ABC algorithm with a multi-strategy ensemble called MEABC is proposed to solve optimization problems with various features. It includes a pool of different resolution search strategies that competes with each other during the search process to produce offspring [53]. Also, five multiple update strategies were combined with ABC to enhance its performance [34]. The Differential Evolution strategy is also hybridized with ABC, and the performance of this hybridization, called DE-ABC, is better than the performance of both DE and ABC [1]. Hybridization between ABC and PSO is produced to share valuable information between them. Two information-sharing processes are added between PSO and ABC, and the proposed hybridized method shows competitive results [50]. To obtain the parallel value of GA calculation and the speed convergence of ABC hybridization between GA and ABC is produced by exchanging knowledge between the bee colony population and the GA population [59]. Levy flight and opposition-based learning methods are used to improve the performance of ABC by enhancing its exploitation in different works [48], [49], [46]. Another hybridization of ABC with teaching-learning based optimization (TLBO) is produced, called TLABC, where ABC exploration combines with TLBO exploitation and enhances the performance of both ABC and TLBO [7].

Being motivated by the fact that there is not an optimal version of ABC among the previous versions, in this paper, not only do we focus on improving the performance of the ABC algorithm, but also its most promising versions. So, the contributions of this paper are summarized as follows.

- The indicator parameters are introduced for each solution to improve the exploitation ability of the ABC algorithm.
- A new local search equation is proposed to get more robust performance and enhance the convergence speed of the ABC algorithm.
- Different well-known ABC variants such as GABC, DFSABC, and TLABC are also used to validate the efficiency of the proposed idea.

The rest of this paper has the following structure. The original ABC is briefly discussed in Section 2, and related works on improved ABCs are also introduced. The structure of the proposed algorithms is defined in Section 3 based on the proposed modifications. Section 4 presents simulation tests and comparisons with other algorithms. The conclusion of the paper is in Section 5.

2 Artificial Bee Colony Algorithm

The ABC algorithm is a kind of optimization algorithm inspired by honey bees' wobble dance and intelligent behavior. In ABC, the solution imitates the food source, and the solution's fitness imitates the quantity of nectar in each source of food. The ABC algorithm splits the bee foraging activities into three stages. The first half of the colony comprises employed bees responsible for randomly looking for a portion of better food in the vicinity of the respective parent food source, then transmit food source information to onlooker bees. The second part of the colony comprises onlooker bees that use the information coming from onlooker bees to check for better food sources. If a predetermined number of evaluations (limited) do not improve a food source's quality, this source of food is abandoned by its employed bee, which then becomes a scout bee and starts to look for a new random source of food. In ABC, three control parameters are used: The number of employed equals the number of onlooker bees, which equals the number of food sources, limit, and the maximum number of cycles. The main steps of ABC are given in Algorithm (1) below [31]

Algorithm 1 ABC Algorithm Main Steps

1: Initialization.
2: Employed bee phase.
3: Onlooker bee phase.
4: Scout bee phase.
5: Memorizing the best solution.

2.1 Initialization

The first step of any meta-heuristic algorithm is the initialization step. Eq. (1) is used to generate a random population of food sources (SN). Where x^j_{max} is the upper bound and x^j_{min} is the lower bound of the jth food source (solution).

$$x^j_i = x^j_{min} + rand(0,1)(x^j_{max} - x^j_{min}) \quad (1)$$

2.2 Employed Bee Phase

Eq. (2) is used to change the existing solution in this phase where $k \in [2, SN]$ and $j \in [2, D]$ are randomly chosen indices, where SN is the number of food sources, D is the dimension of the solution, and k is not equal to i. The current solution is updated and replaced by employed bees based on the fitness value of a new solution determined by Eq. (3) .

$$v^j_i = x^j_i + rand(0,1)(x^j_i - x^j_k) \quad (2)$$

$$Fit(i) = \begin{cases} \frac{1}{1+f(i)}, \ if \ f(i) \geq 0 \\ 1 + |f(i)|, \ if \ f(i) < 0) \end{cases} \quad (3)$$

2.3 Onlooker Bee Phase

The task of the onlooker bees starts with the end of the employed bee task. In this phase, every onlooker bee will fly to a food source according to the quality information calculated from the food resources provided by the employee bees. Eq. (4) calculates the probability of the quality of those sources. The higher the fitness value is, the greater the chance of selection.

$$P_i = \frac{Fit(i)}{\sum_{i=1}^{N} Fit(i)} \quad (4)$$

2.4 Scout Bee Phase

If that source of food is not changed for a limited number of iterations, it is presumed that a food source is exhausted. After that, the employed bee becomes a scout, and a randomly new food source in the search area is generated using Eq. (2).

3 Modified Versions of Artificial Bee Colony Algorithm Used in the Study

3.1 Gbest-Guided ABC (GABC)

It is one of ABC modified versions that adjusted the search equation of ABC to implement the global best solution of PSO to direct the search for new candidate solutions to achieve better exploitation. In this version, the search equation Eq. (2) is replaced by Eq. (5) to guarantee the improvement of the exploitation of ABC [62].

$$v^j_i = x^j_i + rand(0,1)(x^j_i - x^j_k) + rand(0,C)(x^j_{gbest} - x^j_i) \quad (5)$$

Where C is a nonnegative constant that its value controls the balance between the exploration and the exploitation and x^j_{gbest} is the jth parameter of the global best solution in the population. When the value of C decreases the exploration increases, while when its value increases, the exploitation increases. However, this constant should be chosen carefully because if its value is too high, both the exploration and the exploitation may decrease.

3.2 Depth First Search ABC (DFSABC)

It is common knowledge that ABC is one of the most common swarm algorithms due to its high exploration. However, sadly, its search equation (Eq. (2)) makes it have slow convergence. Consequently, in [9], a good balance between exploration and exploitation, in addition to the depth-first-search framework were proposed. There are two modifications to ABC introduced in this version, which are:

- Two different new search equations replace the search equation used in the employed and onlooker bee phase of ABC. Consequently, the exploitation ability has increased.
- In the onlooker bee phase, just the best solutions (elite solutions) are used among the overall population. Since more attention is paid to elite solutions regions, more improvement in exploitation will be noted

These two modifications guarantee the improvement of ABC exploitation and its overall efficiency. Since DFSABC_elite is another version of ABC, it has the same phases with some changes in employed and onlooker phases. Consequently, in this section, we will discuss both the employed and onlooker phases in detail.

Employed Bee Phase At the beginning of this phase, the elite solutions (P food sources usually 10% or 20% of the population) are chosen, and a flag parameter Flag1 is set to 1. Where Flag1 is equal to 1, for each employed bee, a random source of food x_r is selected and a new solution x_{new} is generated in the neighborhood of that employed bee by Eq. (6).

$$x_j^i = x_j^b + rand(0,1)(x_j^b - x_j^k) \tag{6}$$

where $j \in [2, D]$ and x_j^i is the j^{th} dimension of the i^{th} new solution in each iteration, x_j^b is the j^{th} dimension of b^{th} solution, and the b^{th} solution is a randomly chosen solution from the elite solutions, x_j^k is the j^{th} dimension of k^{th} solution, and the k^{th} solution is chosen randomly from the whole current population where k, b, and r are different from each other. The old food source is replaced by the new one if and only if its objective value is better than the old one and the failure value of the i^{th} food source will be reset to 0, and Flag1 resets to 0 (This means that the next employed bee in the same iteration will also search in the same food source as Flag1 is not 1). Otherwise, the old food source will also be maintained, the failure value of the i^{th} food source will be increased by 1, and Flag1 resets to 1 (this means that the next employed bee is looking for another random source of food because Flag1 is 1).

Onlooker Bee Phase At the beginning of this phase, a flag parameter Flag2 is set to 1. Where Flag1 is equal to 1, for each onlooker bee a random source of food x_b is selected from the elite solutions, and a new solution x_{new} is generated in the neighborhood of that onlooker bee by Eq. (7).

$$x_j^i = \frac{1}{2}(x_j^b + x_j^{best}) + rand(0,1)(x_j^b - x_j^k) \tag{7}$$

Where X_j^{best} is the best solution among all the population and is different from X_j^b. The old food source is replaced by the new one if and only if its objective value is better than the old one and the failure value of the i^{th} food source will be reset to 0, and Flag2 resets to 0 (This means that the next onlooker bee in the same iteration will also search in the same food source as Flag2 is not 1). Otherwise, the old food source will also be maintained, the failure value of the i^{th} food source will be increased by 1, and Flag2 resets to1 (this means that the next employed bee is looking for another random source of food because Flag2 is 1).

3.3 Teaching-Learning Based ABC (TLABC)

TLABC is a hybridization between TLBO and ABC algorithm, which combines the advantages of both (the exploration of ABC and the exploitation of TLBO). It effectively employs three hybrid search phases as follows [7].

Teaching Based Employed Bee Phase Here each employed bee uses a hybrid of TLBO and mutation operator of differential evolution to search for a new food source, which can develop the variety of search tendencies extraordinarily and upgrade the search ability of TLABC. Eq. (8) is the search equation used in this phase.

$$u_{i,d} = \begin{cases} x_{i,d}^{old} + r_2.(x_{teacher,d} - T_F.x_{mean,d}), \ if \ r_1 \leq 0.5 \\ x_{r_1,d} + F.(x_{r_2,d} - x_{r_3,d}), \ otherwise \end{cases} \tag{8}$$

Where i is the current solution and $d \in [1, D]$; r_1,r_2,and $r_3 \in [1, SN]$; $rand_1$ and $rand_2$ are normally distributed random numbers; $TF = round(1 + rand * 1)$ and F is a scale factor in $[0, 1]$.

Learning Based Onlooker Bee Phase In this stage, an onlooker bee chooses a food source to search out as indicated by the selection probability, which it is determined to utilize Eq. (4). After that, the onlooker bee finds out new food sources using the TLBO's learning strategy expressed in Eq. (9)

$$u_s = \begin{cases} x_s + rand \cdot (x_s - x_j), \ if \ f(x_s) \leq f(x_j) \\ x_s + rand \cdot (x_j - x_s), \ if \ f(x_j) > f(x_s) \end{cases} \tag{9}$$

Where $j \in [1, SN]$ and $j \neq s$.

Generalized Oppositional Scout Bee Phase In this stage, if a nourishment source cannot be improved further for a specific period, it is viewed as depleted and would be relinquished. At that point, an arbitrary candidate solution and the generalized oppositional solution of it are created. The best solution for them is utilized rather than the old depleted nourishment source. Eq. (1) and Eq. (10) are the equations used in this phase, respectively.

$$x_{ind}^{GO} = k \cdot (a_j + b_j) - x_{ind} \tag{10}$$

Where k is a random number in $[0, 1]$, $a_j = max(X)$, and $b_j = min(X)$.

4 Proposed Intensification in the ABC Algorithm

In the scout phase of the original ABC and its other versions, the abandoned food source is replaced by a new random source or its oppositional, leading to increased exploration. However, ABC and its versions already have high exploration. So in this work, we focused

on using the scout phase in exploitation instead. Accordingly, to achieve good exploitation, two steps have to be taken under consideration: first, we must know where the fertile area to be exploited is. Then we can exploit this area by using an excellent local search equation.

The first step is completed by introducing indicator parameters for each solution, which increases in each evaluation if there is an improvement in that solution as it is introduced in (Eq. (11).). The idea here is that the solution will continue to improve in the fertile area, unlike the useless area.

$$h_i = \begin{cases} h_i + 1; fitness(x_{new}) \geq fitness(x_i) \\ h_i; otherwise \end{cases} \quad (11)$$

Where h_i is the indicator of the ith solution. It is clear from (Eq. (11).) that each solution has its own indicator which increases whenever there is enhancing of its fitness. As a result, at the early stages of searching, we may have many promising searching areas. However, our strategy is to visit them one by one from the highest fertility area to the least. By doing so we have directed the scout bees searching to be just in the most promising areas.

The second step is completed by exploiting the neighbor of the solution with the maximum indicator (the fertile area). Here we focus on replacing the scout phase from the exploration phase to the exploitation phase by replacing the abandoned food source with a new solution in the fertile area. To achieve this scenario, the proposed search equation (Eq. (12).) is used, where a solution is created around the fertile area of the search rather than the old depleted nourishment source.

In the second term of (Eq. (12).), the absolute value of the difference between the two solutions x_g and x_h is calculated. Using the absolute value here is to guarantee that the direction of approximation is towards x_g. We need the new solution to converge towards x_g because x_g is always nearer than x_h to the global solution except when $x_g = x_h$. This structure is incorporated in the original ABC and other well-known versions of ABC, such as GABC, DFSABC_elite, and TLABC. Therefore, the respective amended ABCs are referred to as ABC/ds, GABC/ds, DFSABC/ds and TLABC/ds, respectively.

$$x_{ind} = \frac{x_g + x_h}{2} + rand.|x_g - x_h| \quad (12)$$

Where x_{ind} is the new solution generated in the fertile area, x_g is the current best solution, rand is a normally distributed random number in [0, 1], and x_h is the solution with maximum indicator value. The abundant solution in the scout phase will be replaced by x_{ind} to increase the exploitation. To sum up, one of the following four cases might occur during the scout phase searching process.

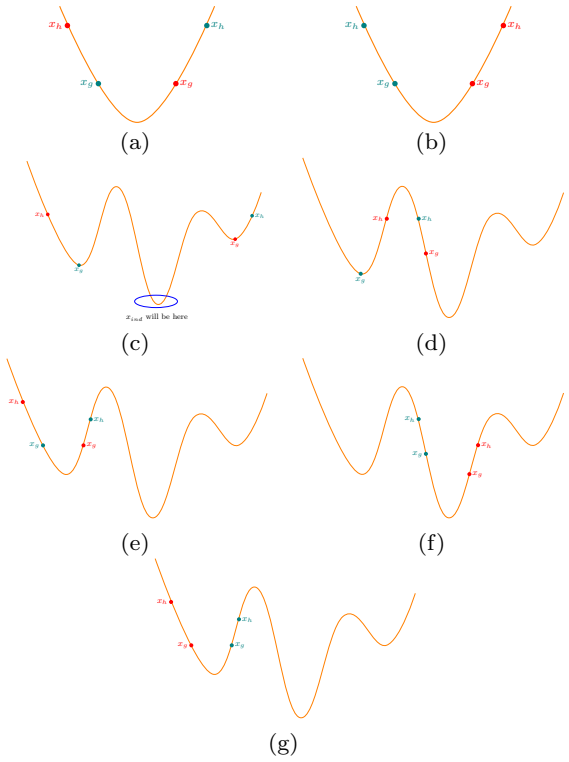

Fig. 1 Scout phase searching process in the proposed model

1. When x_g and x_h are in opposite directions around a convex as is shown in Fig. 1(a), in this case, the new solution x_{ind} will be somewhere near the global solution of this convex.
2. When x_g and x_h are in the same direction as a convex, as shown in Fig. 1(b), in this case, the new solution x_{ind} will be somewhere between them. Moreover we ensured that the number of solutions in this area increased, which guarantees increased exploitation.
3. When x_g and x_h are in opposite directions but the searching area is not convex, four possible scenarios might occur as the following:
   (a) When x_g and x_h are around the global convex, the new solution x_{ind} will be somewhere near the global convex as is shown in Fig. 1(c).
   (b) When x_g and x_h are around a vertex, the new solution x_{ind} will be somewhere around this vertex, as shown in Fig. 1(d). Furthermore, this is when our proposed model will not positively or negatively affect the actual algorithm performance.
   (c) When x_g and x_h are around local convex, the new solution x_{ind} will be somewhere around this local convex, as shown in Fig. 1(e). However, in this scenario, this area will be exploited very

well. In the subsequent iterations, x_{ind} will be its local solution. **It is worth here of mentioning the following important points: First, after each update in the scout phase, the maximum indicator will be updated in order to indicate another promising solution, and another fertile area will occur. Second, this process not only guarantees that the searching area will be in all fertile areas, but it guarantees the searching process will follow one of the scenarios in figures Fig. 1(c) or Fig. 1(d) where x_g is already the local optima. Finally, since the ABC is a stochastic-based algorithm, our idea is stochastic based too. However, following the logic of the proposed idea, avoiding stacking in a local optimum has a high likelihood but needs more iterations.**

4. When x_g and x_h are in the same directions, but the searching area is not convex. Here, two possible scenarios might occur as the following:
   (a) When x_g and x_h are in the same direction on the global convex, shown in Fig. 1(f), the new solution x_{ind} will be somewhere near the global solution. Moreover we ensured that the number of solutions in this area increased, which guarantees increased exploitation.
   (b) When x_g and x_h are in the same direction on a local convex, shown in Fig. 1(g), the new solution x_{ind} will be somewhere near this local solution. However, in this scenario, this area will be exploited very well. In the subsequent iterations, x_g will be its local solution and the 3rd scenario of the 3rd case will occur.

However, the exploration of ABC is already high, and we here make some trade-offs between exploration and exploitation by generating solutions near the best solution instead of randomly. We improved the exploitation of the algorithm and achieved an acceptable balance between exploration and exploitation. Besides, we add an indicator of the fertile area, which is the other modification we added. The fertile area could be discovered by using this indicator for each solution, which will increase whenever the solution increases. Thus the solution which its indicator is the highest is indeed in the fertile area

The ABC/ds, GABC/ds, DFSABC/ds, and TLABC/ds pseudo-codes are shown in Algorithm (2), Algorithm (3), Algorithm (4), and Algorithm (5), respectively.

Algorithm 2 ABC with Directed Scout (ABC/ds)

Initialize the population by Eq. (1),
Calculate the fitness values of the population,
repeat
  for $i = 1 \dots SN$ **do**
    Generate a new solution x_{new} using Eq. (2),
    Proposed solution change phase **(Algorithm (6))**
  end for
  Determine the nectar amounts of food sources by Eq. (4
  for $i = 1 \dots SN$ **do**
    Using the roulette method, some onlooker bees will be selected to move onto the food sources according to the selection probability of Eq. (4,
    Generate a new solution x_{new} using Eq. (2),
    Proposed solution change phase **(Algorithm (6))**
  end for
  Proposed scout bee phase **(Algorithm (7))**
  Memorized the best solution
until (Cycle=MaxCycle or termination criteria are met)

Algorithm 3 Gbest-Guided ABC with Directed Scout (GABC/ds)

Initialize the population by Eq. (1),
Calculate the fitness values of the population,
repeat
  for $i = 1 \dots SN$ **do**
    Generate a new solution x_{new} using Eq. (5),
    Proposed solution change phase **(Algorithm (6))**
  end for
  Determine the nectar amounts of food sources by Eq. (4
  for $i = 1 \dots SN$ **do**
    Using the roulette method, some onlooker bees will be selected to move onto the food sources according to the selection probability of Eq. (4,
    Generate a new solution x_{new} using Eq. (5),
    Proposed solution change phase **(Algorithm (6))**
  end for
  Proposed scout bee phase **(Algorithm (7))**
  Memorized the best solution
until (Cycle=MaxCycle or termination criteria are met)

Algorithm 4 Depth First Search ABC with Directed Scout (DFSABC/ds)

Initialize the population by Eq. (1),
Calculate the fitness values of the population,
repeat
  set flag1=1, flag2=1
  for $i = 1 \dots SN$ **do**
    if $(flag1 == 1)$ **then**
      Choose a random food source x_r from the population
    end if
    Generate a new solution x_{new} using Eq. (6),
    if $fitness(x_{new}) \geq fitness(x_r)$ **then**
      x_r is replaced by x_{new}
      Set $failure(r) = 0$, $indicator(r) = indicator(r) + 1$, $flag1 = 0$
    else
      Set $failure(r) = failure(r) + 1$, $flag1 = 1$
    end if
  end for
  for $i = 1 \dots SN$ **do**
    if $(flag2 == 1)$ **then**
      Choose a random food source x_b from the elite solutions
    end if
    Generate a new solution x_{new} using Eq. (7),
    if $fitness(x_{new}) \geq fitness(x_b)$ **then**
      x_b is replaced by x_{new}
      Set $failure(b) = 0$, $indicator(r) = indicator(b) + 1$, $flag2 = 0$
    else
      Set $failure(b) = failure(b) + 1$, $flag1 = 1$
    end if
  end for
  Proposed scout bee phase **(Algorithm (7))**
  Memorized the best solution
until (Cycle=MaxCycle or termination criteria are met)

Algorithm 5 Teaching-Learning Based ABC with Directed Scout (TLABC/ds)

Initialize the population by Eq. (1),
Calculate the fitness values of the population,
repeat
  for $i = 1 \dots SN$ **do**
    Generate a new solution x_{new} using Eq. (8),
    Proposed solution change phase **(Algorithm (6))**
  end for
  Determine the nectar amounts of food sources by Eq. (4
  for $i = 1 \dots SN$ **do**
    Using the roulette method, some onlooker bees will be selected to move onto the food sources according to the selection probability of Eq. (4,
    Generate a new solution x_{new} using Eq. (9),
    Proposed solution change phase **(Algorithm (6))**
  end for
  Proposed scout bee phase **(Algorithm (7))**
  Memorized the best solution
until (Cycle=MaxCycle or termination criteria are met)

Algorithm 6 The procedure of proposed solution change phase

if $fitness(x_{new}) \geq fitness(x_i)$ **then**
  x_i is replaced by x_{new}
  Set $failure(i) = 0$, $indicator(i) = indicator(i) + 1$,
else
  Set $failure(i) = failure(i) + 1$
end if

Algorithm 7 The procedure of proposed scout bee phase

Find the solution with maximum failure value.
if $max(failure) \geq limit$ **then**
  Find the food source with maximum improvement indicator
  Update its value by applying the update Eq. (12)
  Create its generalized oppositional sol. using Eq. (10)
  The best solution of them is utilized rather than the old depleted nourishment source.
end if

4.1 Time complexity of the Proposed Models

The complexity of the proposed models is determined based on the benchmark functions of CEC2014. In compliance with CEC2014 directives. $T0$, $T1$ and $T2$ parameters are the same as the CEC2014 parameters. As defined in [38], $T0$ is the time calculated by running the following test problem:

$for\ i = 1 : 1000000$
  $x = 0.55 + (double)i;\ \ x = x + x;\ \ x = x/2;$
  $x = x * x;\ \ x = sqrt(x);\ \ x = log(x);$
  $x = exp(x);\ \ x = x/(x + 2);$
$end;$

$T1$ is the time needs to evalute function f_{18} for 200000 times and $T2$ is the time taken by the optimization algorithm to solve function f_{18} with 200000 function evaluations. Execute calculation of $T2$ 5 times and get 5 $T2$ values ($\widehat{T}2 = Mean(T2)$). The complexity of the algorithm is reflected by $(\widehat{T}2 - T1)/T0$

Table 1 shows the complexity of the four proposed models against their original versions. It is clear from Table 1 that the complexity of the proposed models is very close to the original ones except for the DFSABC/ds which has lower complexity than DFSABC.

Table 1 Complexity results on CEC2014 benchmark

Algorithm	D		
	10	30	50
ABC	1.84E+01	1.87E+01	1.98E+01
ABC/ds	1.88E+01	1.95E+01	2.16E+01
GABC	1.96E+01	1.90E+01	2.24E+01
GABC/ds	1.88E+01	1.92E+01	2.05E+01
DFSABC	2.19E+01	2.31E+01	2.59E+01
DFSABC/ds	9.27E+00	1.11E+01	1.02E+01
TLABC	3.70E+01	4.08E+01	4.25E+01
TLABC/ds	3.71E+01	4.11E+01	4.35E+01

5 Experimental Results

In this paper, we divided the discussion of the results into three subsections. In the first subsection, we used CEC2014 benchmark functions to show the performance of the proposed models. Then, in the second subsection, we used CEC2015 benchmark functions to show the performance of the proposed models in 15 challenging, expensive problems. In the third and final subsection, the validity of the proposed models is checked on some common real-life problems.

5.1 Proposed Models for solving CEC2014 Benchmark functions

In this section, the validity of the proposed optimization algorithms is checked on CEC2014 benchmark problems [40]. This benchmark collection comprises 30 unconstrained (unimodal, multimodal, hybrid, and composite) optimization problems of varying degrees of difficulty. Table 2 shows the summary of the four types of functions presented in the CEC2014 benchmark.

5.1.1 Experiments configuration

The configuration of running the experiments is the same as that proposed in [40]as the following: each experiment was run 51 times. The stopping criteria were $10^4 * Dimension$, the search space range was [-100, 100], the dimensions were $D = 30$ and $D = 50$, and the initialization within the search space was uniform random.

All the comparison algorithms ABC, GABC, DFSABC, and TLABC, used the same values for common parameters. The population size (NP), the number of food sources (SN), and the limit were set to 100, 50, and SN*D for all algorithms [30][10][35]. For GABC, nonnegative constant (C) was set to 1.5 [62]. For DFSABC, the elite solutions (P) was set to 0.1 * SN [9]. Table shows the control parameters for all algorithms used in this paper.

In this section compares results between ABC, ABC/ds, GABC, GABC/ds, DFSABC, DFSABC/ds, TLABC,

Table 2 Information of 30 CEC2014 benchmark functions

Type	Function	f_{min}
Unimodal	f_1: Rotated High Conditioned Elliptic	100
	f_2: Rotated Bent Cigar	200
	f_3: Rotated Discus	300
Multimodal	f_4: Shifted and Rotated Rosenbrock's	400
	f_5: Shifted and Rotated Ackley's	500
	f_6: Shifted and Rotated Weierstrass	600
	f_7: Shifted and Rotated Griewank's	700
	f_8: Shifted Rastrigin's	800
	f_9: Shifted and Rotated Rastrigin's	900
	f_{10}: Shifted Schwefel's	1000
	f_{11}: Shifted and Rot. Schwefel's	1100
	f_{12}: Shifted and Rot. Katsuura	1200
	f_{13}: Shifted and Rot. HappyCat	1300
	f_{14}: Shifted and Rot. HGBat	1400
	f_{15}: Shf. and Rot. Exp.Griewank's Ros.	1500
	f_{16}: Shifted and Rot. Exp.Scaffer's f_6	1600
Hybrid	f_{17}: Hybrid Function 1 (N=3)	1700
	f_{18}: Hybrid Function 2 (N=3)	1800
	f_{19}: Hybrid Function 3 (N=4)	1900
	f_{20}: Hybrid Function 4 (N=4)	2000
	f_{21}: Hybrid Function 5 (N=5)	2100
	f_{22}: Hybrid Function 6 (N=5)	2200
Composite	f_{23}: Composition Function 1 (N=5)	2300
	f_{24}: Composition Function 2 (N=3)	2400
	f_{25}: Composition Function 3 (N=3)	2500
	f_{26}: Composition Function 4 (N=5)	2600
	f_{27}: Composition Function 5 (N=5)	2700
	f_{28}: Composition Function 6 (N=5)	2800
	f_{29}: Composition Function 7 (N=3)	2900
	f_{30}: Composition Function 8 (N=3)	3000

Table 3 Control parameters of algorithms

Algorithm	Parameter settings
ABC/ds	SN = 50, limit =SN × D
GABC/ds	SN = 50, limit =SN × D, C = 1.5
TLABC/ds	SN = 50, limit =SN × D
DFSABC/ds	SN = 50, limit =SN × D, P = 0.1 × SN, r = 1/p
ABC	SN = 50, limit =SN × D
GABC	SN = 50, limit =SN × D, C = 1.5
TLABC	SN = 50, limit =SN × D
DFSABC	SN = 50, limit =SN × D, P = 0.1 × SN, r = 1/p
DPABC	SN = 50, limit =SN × D, m = 3.5, minSNCP = 20
DPGABC	SN = 50, limit =SN × D, C = 1.5, m = 3.5, minSNCP = 20
DPCABC	SN = 50, limit =SN × D, m = 3.5, minSNCP = 20
CABC	SN = 50, limit =SN × D
OCABC	SN = 50, limit =SN × D
ARABC	SN = 50, limit =SN × D
LX-BBO	Mutation Probability= 0.005, ymin = 0.1, ymax = 1, (E/I) = 1, a = 0, b = 0.5 and k = 0.95
B-BBO	Mutation Probability= 0.005
OHDA	member_selection = 100%, Dim_to_move = 10, Local_radius = 0.05 or 0.1, Tumbling = 3, L = [0.5, 0.7, 1] × EDa
fb-TSA	ST = 0.1

and TLABC/ds on CEC2014 benchmarks for 30 and 50 dimension were shown in Table 4 and Table 5, respectively. The average absolute error value of the objective function value for the test functions is shown in these tables. The better results in the tables are highlighted with bold letters.

The tables above show that the results of the models are very close to each other. Consequently, some statistical tests should be applied to prove that there is a significant difference between them. Thus, the Wilcoxon sum rank test is used to check whether there is a significant difference between the proposed models and the original versions.

However, the pair-wise Wilcoxon test results are shown also in Tables 4 and 5 for 30 and 50 dimensions respectively. This test is used to show the significant difference between the original algorithms and their proposed modified versions. The symbols +, = and - mean that the proposed model is significantly superior, similar or inferior to the original version, respectively, according to the Wilcoxon rank sum test at $\alpha = 0.05$ significance level.

It is noticed from Tables 4 and 5 that

- ABC/ds is better significantly compared to ABC in 27 functions out of 30 for dimension 30.
- ABC/ds is better significantly compared to ABC in 26 functions out of 30 for dimension 50.
- GABC/ds is better significantly compared to GABC in 23 functions out of 30 for dimension 30.
- GABC/ds is better significantly compared to GABC in 24 functions out of 30 for dimension 50.
- DFSABC/ds is better significantly compared to DFSABC in 20 functions out of 30 for dimension 30.
- DFSABC/ds is better significantly compared to DFSABC in 19 functions out of 30 for dimension 50.
- TLABC/ds is better significantly compared to TLABC in 7 functions out of 30 for dimension 30.
- TLABC/ds is better significantly compared to TLABC in 6 functions out of 30 for dimension 50.

It is seen that the modification we proposed has a different effect on improving the performance of the original versions. Consequently, we can conclude that the proposed modification affects the performance of ABC more than GABC, DFSABC, and TLABC. And it affects the performance of GABC more than DFSABC, and TLABC and so on. This is due to the variety of the power of the exploitation of the original versions. Since our proposed modification improves the exploitation of ABC versions, it has a more negligible effect on TLABC due to its high exploitation (coming from the TLBO algorithm, which is one of the best local searching optimization algorithms). However, it is clear that the proposed modification has enhanced the performance of TLABC algorith in solving the hybrid problems.

We can also conclude from Table 4 that, among the four original algorithms and the four modified versions, the GABC/ds algorithm succeeded in achieving the best results in 9 for 30-dimensional problems. In contrast, ABC/ds, DFSABC/ds, TLABC/ds, TLABC, DFSABC, ABC, and GABC algorithms succeeded in achieving the best results in 8, 7, 6, 4, 2, 1, and 1 problems respectively. In problem f_{24}, the best and same result was achieved by TLABC, and TLABC/ds. In problem f_{26}, the best and same result was achieved by ABC, ABC/ds, GABC, GABC/ds, and DFSABC/ds. We can

Table 4 Comparison results of the proposed algorithms and their original algorithms for 30-dimensional CEC2014 benchmark problems in terms of average error value.

F	ABC	ABC/ds		GABC	GABC/ds		DFSABC	DFSABC/ds		TLABC	TLABC/ds	
f_1	1.09E+07	5.14E+06	+	1.08E+07	2.98E+06	+	6.33E+06	5.76E+06	=	1.84E+05	**1.41E+05**	+
f_2	1.56E+02	4.39E+01	+	7.36E+01	1.34E-03	+	5.80E+02	1.26E+03	=	3.17E-06	**4.75E-12**	+
f_3	5.74E+02	3.49E+02	+	4.21E+02	2.35E+02	+	9.57E+02	1.11E+03	=	3.18E+01	**1.08E-03**	+
f_4	2.89E+01	**1.06E+01**	+	1.68E+01	2.48E+01	=	2.25E+01	4.49E+01	-	4.13E+01	3.66E+01	=
f_5	2.04E+01	2.03E+01	+	2.04E+01	2.03E+01	+	2.06E+01	**2.00E+01**	+	2.10E+01	2.09E+01	=
f_6	1.50E+01	8.59E+00	+	1.36E+01	**6.58E+00**	+	1.28E+01	1.06E+01	+	1.14E+01	1.22E+01	-
f_7	2.76E-05	3.94E-06	+	1.09E-06	1.13E-07	+	**6.35E-08**	3.38E-04	=	2.90E-02	3.06E-02	=
f_8	1.14E-13	1.14E-13	=	1.14E-13	1.14E-13	=	**0.00E+00**	1.95E-02	=	2.64E+01	4.37E+01	-
f_9	8.50E+01	3.51E+01	+	5.09E+01	2.95E+01	+	5.36E+01	**1.98E+01**	+	4.75E+01	5.07E+01	=
f_{10}	1.41E+00	**7.22E-01**	+	1.27E+00	1.59E+00	=	1.16E+00	1.84E+00	-	1.91E+02	6.37E+02	-
f_{11}	2.23E+03	1.55E+03	+	2.24E+03	**1.43E+03**	+	3.14E+03	1.59E+03	+	1.78E+03	1.83E+03	-
f_{12}	4.11E-01	2.49E-01	+	5.32E-01	**8.57E-02**	+	9.89E-01	1.61E-01	+	2.43E+00	2.40E+00	=
f_{13}	2.50E-01	**1.25E-01**	+	2.42E-01	1.28E-01	+	4.14E-01	1.72E-01	+	2.92E-01	2.90E-01	=
f_{14}	2.01E-01	8.61E-02	+	1.83E-01	**7.82E-02**	+	3.98E-01	1.23E-01	+	2.57E-01	2.53E-01	=
f_{15}	1.15E+01	3.93E+00	+	7.45E+00	3.34E+00	+	8.59E+00	**2.95E+00**	+	7.27E+00	7.87E+00	=
f_{16}	1.07E+01	9.83E+00	+	1.03E+01	**8.90E+00**	+	1.12E+01	9.08E+00	+	1.13E+01	1.07E+01	+
f_{17}	2.93E+06	1.28E+06	+	2.04E+06	9.37E+05	+	1.15E+07	1.40E+06	+	9.12E+04	**3.74E+04**	+
f_{18}	1.08E+04	**9.69E+02**	+	8.03E+03	4.21E+03	+	1.72E+03	2.41E+03	=	1.83E+03	1.88E+03	=
f_{19}	7.61E+00	7.01E+00	+	7.05E+00	6.01E+00	+	7.89E+00	**5.59E+00**	+	9.04E+00	9.07E+00	=
f_{20}	5.03E+03	9.89E+02	+	2.53E+03	8.95E+02	+	2.01E+04	**8.85E+02**	+	4.38E+03	2.62E+03	+
f_{21}	4.67E+05	2.37E+05	+	3.94E+05	2.00E+05	+	1.20E+06	2.60E+05	+	3.80E+04	**1.77E+04**	+
f_{22}	2.97E+02	**2.14E+02**	+	3.19E+02	2.44E+02	+	7.19E+02	1.91E+02	+	1.57E+02	1.53E+02	=
f_{23}	3.15E+02	3.15E+02	=	3.15E+02	3.15E+02	=	3.16E+02	3.15E+02	+	**2.68E+02**	3.15E+02	-
f_{24}	2.27E+02	2.10E+02	+	2.24E+02	2.02E+02	+	2.27E+02	2.14E+02	+	**2.00E+02**	**2.00E+02**	=
f_{25}	2.08E+02	2.07E+02	+	2.07E+02	2.05E+02	+	2.08E+02	2.07E+02	+	**2.00E+02**	2.02E+02	-
f_{26}	**1.00E+02**	**1.00E+02**	=	**1.00E+02**	**1.00E+02**	=	1.35E+02	**1.00E+02**	+	**1.00E+02**	1.04E+02	=
f_{27}	4.11E+02	**4.04E+02**	+	4.08E+02	**4.04E+02**	+	4.29E+02	**4.04E+02**	+	4.05E+02	5.69E+02	-
f_{28}	9.61E+02	8.16E+02	+	8.26E+02	**7.92E+02**	+	8.73E+02	8.36E+02	+	8.39E+02	9.45E+02	-
f_{29}	1.14E+03	**1.05E+03**	+	1.55E+03	2.84E+03	-	1.37E+03	2.48E+03	-	1.40E+03	1.33E+03	=
f_{30}	4.67E+03	2.43E+03	+	3.13E+03	**2.06E+03**	+	2.70E+03	3.39E+03	-	2.34E+03	2.51E+03	=
G1($f_1 - f_3$)			3-0-0			3-0-0			0-3-0			3-0-0
G2($f_4 - f_{16}$)			12-0-1			10-3-1			9-2-2			1-8-4
G3($f_{17} - f_{22}$)			6-0-0			6-0-0			5-1-0			3-3-0
G4($f_{23} - f_{30}$)			6-2-0			5-2-1			6-0-2			0-4-4
Total			27-2-1			23-5-2			20-6-4			7-15-8

Table 5 Comparison results of the proposed algorithms and their original algorithms for 50-dimensional CEC2014 benchmark problems in terms of average error value.

F	ABC	ABC/ds		GABC	GABC/ds		DFSABC	DFSABC/ds		TLABC	TLABC/ds	
f_1	1.91E+07	5.37E+06	+	1.50E+07	5.08E+06	+	3.28E+07	5.13E+06	+	4.91E+05	**4.51E+05**	=
f_2	1.83E+03	**1.65E+03**	+	7.18E+03	6.03E+03	=	2.45E+03	2.95E+03	=	4.36E+03	5.45E+03	=
f_3	6.94E+03	3.62E+03	+	5.43E+03	**2.93E+03**	+	7.18E+03	7.72E+03	=	6.49E+04	5.83E+03	+
f_4	6.29E+01	**5.48E+01**	=	6.50E+01	6.38E+01	=	8.58E+01	8.00E+01	=	8.22E+01	9.55E+01	-
f_5	2.05E+01	2.04E+01	+	2.06E+01	2.03E+01	+	2.07E+01	**2.01E+01**	+	2.11E+01	2.11E+01	=
f_6	3.33E+01	1.80E+01	+	3.04E+01	1.62E+01	+	3.59E+01	**1.13E+01**	+	3.02E+01	3.01E+01	=
f_7	8.73E-03	5.76E-04	+	9.17E-04	**3.45E-04**	+	8.27E-04	1.42E-03	-	7.73E-02	4.48E-02	=
f_8	3.56E-09	**1.14E-13**	+	**1.14E-13**	**1.14E-13**	=	5.85E-02	1.95E-02	=	9.79E+01	1.17E+02	-
f_9	1.95E+02	7.28E+01	+	1.28E+02	7.11E+01	+	1.18E+02	**4.15E+01**	+	1.33E+02	1.38E+02	=
f_{10}	4.17E+00	2.10E+00	+	4.45E+00	4.03E+00	=	**1.48E+00**	9.21E+00	-	1.09E+03	2.35E+03	-
f_{11}	4.83E+03	**3.59E+03**	+	5.07E+03	3.66E+03	+	6.55E+03	4.01E+03	+	4.66E+03	4.13E+03	=
f_{12}	5.06E-01	1.31E-01	+	6.15E-01	**7.18E-02**	+	1.14E+00	1.54E-01	+	3.32E+00	3.37E+00	=
f_{13}	3.37E-01	**1.73E-01**	+	2.94E-01	1.74E-01	+	4.96E-01	1.97E-01	+	4.84E-01	5.36E-01	-
f_{14}	2.53E-01	1.05E-01	+	2.22E-01	**8.61E-02**	+	4.16E-01	1.31E-01	+	3.14E-01	3.25E-01	=
f_{15}	2.81E+01	8.39E+00	+	1.92E+01	8.17E+00	+	2.30E+01	**6.00E+00**	+	2.43E+01	3.45E+01	-
f_{16}	1.95E+01	1.82E+01	+	1.91E+01	1.75E+01	+	2.05E+01	**1.73E+01**	+	2.10E+01	2.01E+01	+
f_{17}	7.75E+06	1.39E+06	+	6.53E+06	1.49E+06	+	2.28E+07	2.08E+06	+	1.01E+05	**6.96E+04**	+
f_{18}	1.57E+04	2.18E+03	+	6.99E+03	1.82E+03	+	1.01E+03	2.46E+03	-	**9.09E+02**	1.00E+03	=
f_{19}	1.79E+01	**1.31E+01**	+	1.66E+01	1.63E+01	+	1.69E+01	1.46E+01	+	1.67E+01	1.68E+01	=
f_{20}	1.86E+04	5.79E+03	+	1.46E+04	**3.15E+03**	+	4.37E+04	5.45E+03	+	1.87E+04	1.31E+04	+
f_{21}	3.66E+06	8.99E+05	+	2.99E+06	1.02E+06	+	8.74E+06	7.78E+05	+	1.28E+05	**6.70E+04**	+
f_{22}	8.48E+02	5.18E+02	+	9.16E+02	6.35E+02	+	1.70E+03	6.01E+02	+	4.70E+02	**4.69E+02**	=
f_{23}	3.44E+02	3.44E+02	=	3.44E+02	3.43E+02	+	3.44E+02	3.44E+02	=	**2.81E+02**	3.44E+02	-
f_{24}	2.58E+02	2.56E+02	+	2.57E+02	2.10E+02	+	2.60E+02	2.44E+02	=	**2.00E+02**	2.20E+02	-
f_{25}	2.17E+02	2.13E+02	+	2.15E+02	2.11E+02	+	2.16E+02	2.14E+02	+	**2.00E+02**	2.01E+02	-
f_{26}	**1.00E+02**	**1.00E+02**	=	**1.00E+02**	**1.00E+02**	=	1.42E+02	**1.00E+02**	+	1.65E+02	1.79E+02	-
f_{27}	**5.00E+02**	6.69E+02	=	9.49E+02	7.82E+02	+	1.06E+03	7.90E+02	+	6.10E+02	1.14E+03	-
f_{28}	1.69E+03	1.21E+03	+	1.25E+03	1.16E+03	+	1.31E+03	1.26E+03	+	**1.13E+03**	1.49E+03	-
f_{29}	1.63E+03	1.54E+03	+	2.13E+03	8.78E+03	-	1.61E+03	5.15E+03	-	1.59E+03	**1.27E+03**	+
f_{30}	1.17E+04	**8.94E+03**	+	1.02E+04	9.07E+03	+	9.66E+03	9.81E+03	=	1.11E+04	1.20E+04	-
G1($f_1 - f_3$)			3-0-0			2-1-0			1-2-0			1-2-0
G2($f_4 - f_{16}$)			12-1-0			10-3-0			9-2-2			1-7-5
G3($f_{17} - f_{22}$)			6-0-0			6-0-0			5-0-1			3-3-0
G4($f_{23} - f_{30}$)			5-3-0			6-1-1			4-3-1			1-0-7
Total			26-4-0			24-5-1			19-7-4			6-12-12

conclude from Table 5 that the ABC/ds algorithm succeeded in achieving the best results in 8 problems. In contrast, GABC/ds, DFSABC/ds, TLABC/ds, TLABC, ABC, GABC, and DFSABC algorithms succeeded in achieving the best results in 7, 6, 5, 5, 2, 2, and 1 problem, respectively. These results show the improvement in the performance of the original algorithm.

5.1.2 Wilcoxon sum rank test results

From the results presented in Tables 4 and 5, we can see that GABC/ds is the best version to compare its success against the other versions. Tables 6 and 7 compare the number of success of GABC/ds against the other versions of ABC on the CEC2014 benchmark for 30 and 50 dimensions, respectively. The results in Tables 6 and 7 compare GABC/ds version with other versions as a paired comparison where the (26-3-1) cell means that GABC/ds is better than ABC in 26 problems, they

have the same result in three problems, and worse in one problem. We can conclude from these two tables that the proposed version called GABC/ds is better than ABC, GABC, DFSABC, TLABC, ABC/ds, DFSABC/ds, and TLABC/ds.

Since GABC/ds is a meta-heuristic algorithm, the Wilcoxon sum rank test was used to validate the results statistically and show a significant difference between the proposed model (GABC/ds) with the other versions of ABC. It is evident in Tables 6 and 7 that there are significant differences in most cases, such that the significant difference between GABC/ds and the other versions of ABC is demonstrated. The symbols +, = and - mean that the proposed model is significantly superior, similar or inferior to the GABC/ds version, respectively, according to the Wilcoxon rank sum test at $\alpha = 0.05$ significance level.

Table 6 Results of Wilcoxon sum rank test between GABC/ds and the other seven versions of ABC on CEC2014 functions for 30 dimensions.

F	GABC/ds vs						
	ABC	GABC	DFSABC	TLABC	ABC/ds	DFSABC/ds	TLABC/ds
f_1	+	+	=	+	+	+	-
f_2	+	+	=	+	+	-	-
f_3	+	+	=	+	=	-	-
f_4	+	=	-	=	=	+	=
f_5	+	+	+	=	=	-	+
f_6	+	+	+	-	+	+	+
f_7	+	+	=	=	+	+	=
f_8	=	-	=	-	=	+	+
f_9	+	+	+	=	+	-	+
f_{10}	+	=	-	-	=	=	+
f_{11}	+	+	+	-	=	=	+
f_{12}	+	+	+	=	+	+	+
f_{13}	+	+	+	=	=	+	+
f_{14}	+	+	+	=	+	+	+
f_{15}	+	+	+	=	+	-	+
f_{16}	+	+	+	+	+	=	+
f_{17}	+	+	+	+	+	=	-
f_{18}	+	+	=	=	-	-	-
f_{19}	+	+	+	=	+	-	+
f_{20}	+	+	+	+	=	=	+
f_{21}	+	+	+	+	+	=	-
f_{22}	+	+	+	=	=	-	-
f_{23}	=	+	+	-	=	=	=
f_{24}	+	+	+	-	+	+	-
f_{25}	+	+	+	-	+	+	-
f_{26}	=	+	+	=	=	=	+
f_{27}	+	+	+	-	=	=	+
f_{28}	+	+	+	-	+	+	+
f_{29}	+	-	-	=	-	=	-
f_{30}	+	+	-	=	+	+	+
G1($f_1 - f_3$)	3-0-0	3-0-0	0-3-0	3-0-0	2-1-0	1-0-2	0-0-3
G2($f_4 - f_{16}$)	11-1-1	10-2-1	9-2-2	1-8-4	7-6-0	7-3-3	11-2-0
G3($f_{17} - f_{22}$)	6-0-0	6-0-0	5-1-0	3-3-0	3-2-1	0-3-3	2-0-4
G4($f_{23} - f_{30}$)	6-2-0	7-0-1	6-0-2	0-3-5	4-3-1	4-4-0	4-1-3
Total	26-3-1	26-2-2	20-6-4	6-14-9	16-12-2	12-10-8	17-3-10

5.1.3 Convergence trajectory and box plot comparison

The convergence curves on some test functions of all compared algorithms above are shown in Fig. 2 and Fig. 3 for dimensions 30 and 50 respectively. Convergence curves are drawn by an average value of 51 independent runs. It is evident from these two figures that the modified versions converge faster than the original versions in most cases. Although the modified versions track the original version's convergence rate initially,

Table 7 Results of Wilcoxon sum rank test between GABC/ds and the other seven versions of ABC on CEC2014 functions for 50 dimensions.

F	GABC/ds vs						
	ABC	GABC	DFSABC	TLABC	ABC/ds	DFSABC/ds	TLABC/ds
f_1	+	+	+	-	=	=	-
f_2	-	=	-	=	-	-	=
f_3	+	+	+	+	+	+	=
f_4	=	=	+	=	=	+	+
f_5	+	+	+	+	+	-	+
f_6	+	+	+	+	+	-	+
f_7	+	+	+	+	=	+	+
f_8	+	=	+	+	+	+	+
f_9	+	+	+	+	=	-	+
f_{10}	=	=	-	+	-	+	+
f_{11}	+	+	+	+	=	+	+
f_{12}	+	+	+	+	+	+	+
f_{13}	+	+	+	+	=	+	+
f_{14}	+	+	+	+	+	+	+
f_{15}	+	+	+	+	=	-	+
f_{16}	+	+	+	+	+	=	+
f_{17}	+	+	+	-	=	=	-
f_{18}	+	+	-	-	=	=	-
f_{19}	+	+	+	+	-	-	=
f_{20}	+	+	+	+	+	+	+
f_{21}	+	+	+	-	=	=	-
f_{22}	+	+	+	-	-	=	-
f_{23}	+	+	+	-	+	+	=
f_{24}	+	+	+	-	+	+	+
f_{25}	+	+	+	-	+	+	-
f_{26}	=	=	+	+	=	=	+
f_{27}	-	+	+	-	=	=	+
f_{28}	+	+	+	-	+	+	+
f_{29}	-	-	-	-	-	=	-
f_{30}	+	+	+	+	=	+	+
G1($f_1 - f_3$)	2-0-1	2-1-0	2-0-1	1-1-1	1-1-1	1-1-1	0-2-1
G2($f_4 - f_{16}$)	11-2-0	10-3-0	12-0-1	12-1-0	6-6-1	8-1-4	13-0-0
G3($f_{17} - f_{22}$)	6-0-0	5-1-0	5-0-1	2-0-4	1-3-2	1-4-1	1-1-4
G4($f_{23} - f_{30}$)	5-1-2	7-0-1	7-0-1	2-0-6	4-3-1	5-3-0	5-1-2
Total	24-3-3	24-5-1	26-0-4	17-2-11	12-13-5	15-9-6	19-4-7

they become faster after some iterations in most cases. This is due to the dependency of our modifications on the scout phase of ABC versions. And the searching process does not enter to scout phase before some iterations (when failure exceeds the limit value). It is worth mentioning here that our modified versions are susceptible to limit value.

The box plot of some test functions of all compared algorithms above is shown in Fig. 4 and Fig. 5 for dimensions 30 and 50, respectively. It can be found that the objective distributions of the proposed methods are smaller than the orjinal methods, demonstrating their robust performances in the test functions.

5.1.4 Comparison with other existing evolutionary algorithms

In this section, the performance of the proposed algorithm GABC/ds has been compared to that of state-of-the-art algorithms. Table 8 shows the average error of the 51 runs in objective function value. It is noticed in Table 8 that compared to other meta-heuristic algorithms, the proposed algorithm GABC/ds produces very competitive results. Moreover, Table 9 compares the proposed GABC/ds with other algorithms as a paired comparison where the result shown in the cell which is in row 6 and column 2 (21-0-9) means that GABC/ds is better than GSA in 21 problems, they do not have the same results, and worse in 9 problems. The analysis of Tables 8 and 9 shows that, in comparison to other meta-heuristic algorithms, the proposed GABC/ds al-

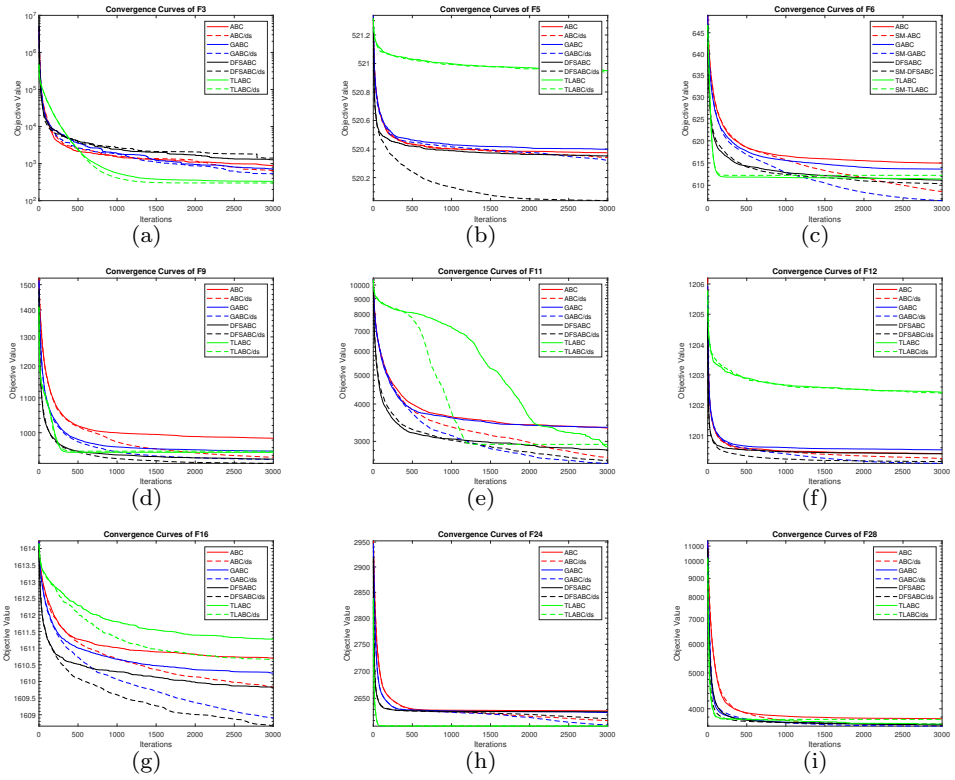

Fig. 2 Convergence curves of algorithms for several 30-variable CEC2014 benchmark functions

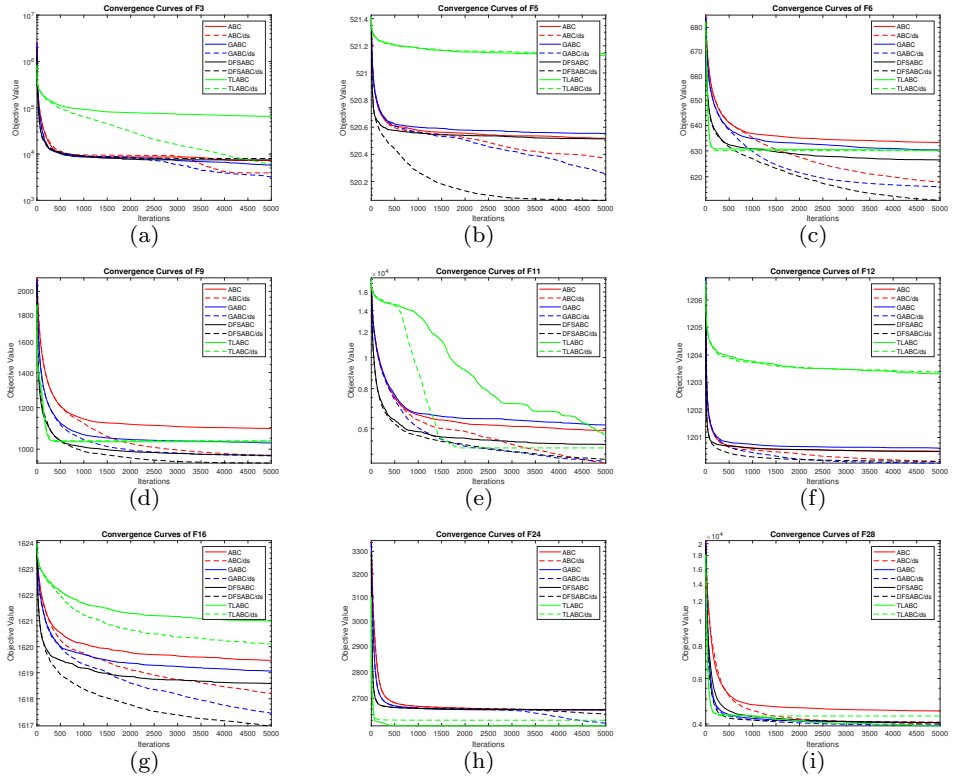

Fig. 3 Convergence curves of algorithms for several 50-variable CEC2014 benchmark functions

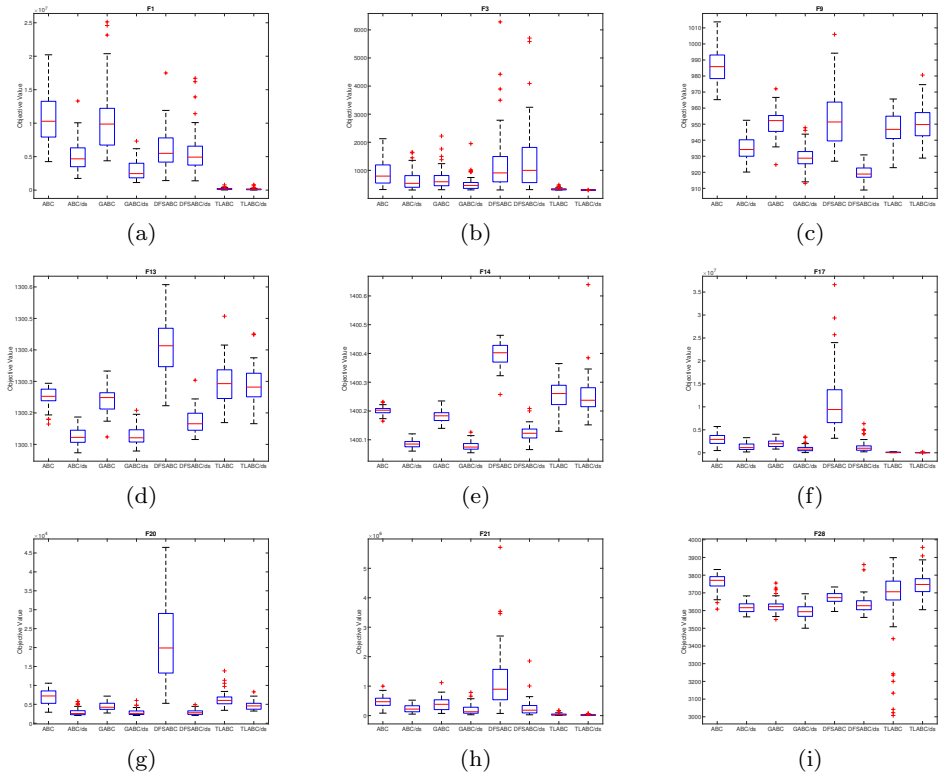

Fig. 4 Box plot of algorithms for several 30-variable CEC2014 benchmark functions

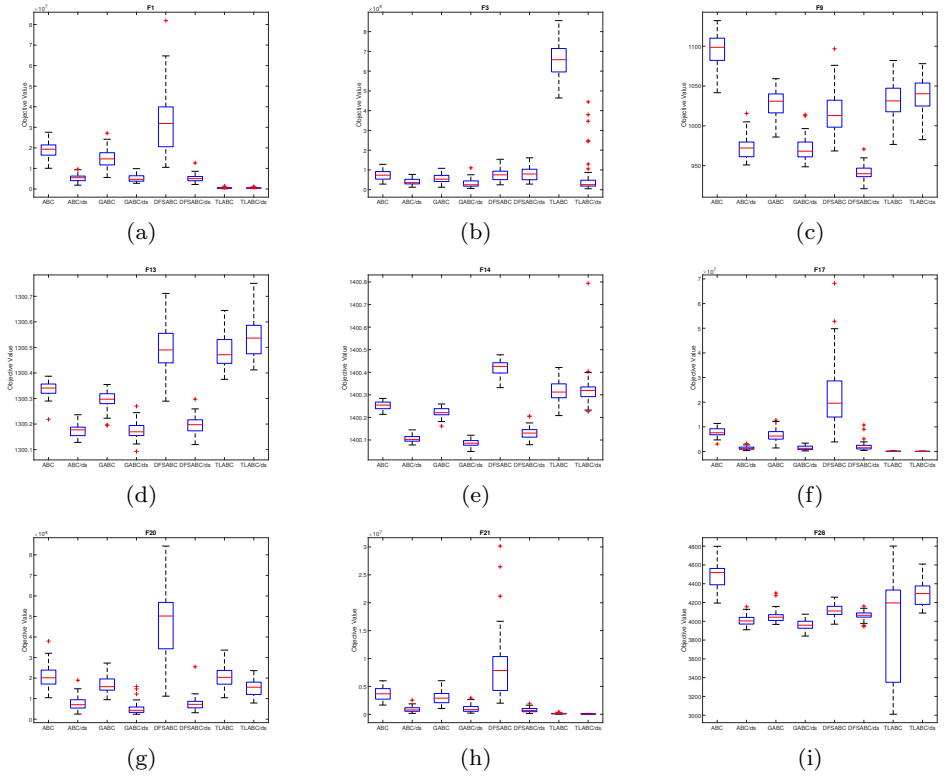

Fig. 5 Box plot of algorithms for several 50-variable CEC2014 benchmark functions

gorithm produces very competitive results. **Furthermore, as it is suggested in Table 9, the proposed GABC/ds algorithm shows high performance in solving multimodal problems since it has defeated all other algorithms in solving multimodal problems.**

5.2 Proposed Models for solving CEC2015 Benchmark functions

Based on the experiments mentioned above, the proposed modified ABC versions in the CEC2014 benchmark problems are better than the original versions. Nevertheless, as the No Free Lunch Theorems show, every pair of algorithms A and B is equally effective on all of the issues.If an algorithm performs well on any kind of problem, it will inevitably do poorly on the other problems. Consequently, for further tests of the effectiveness and efficiency of the proposed modified versions of ABC, we tested them on 15 CEC2015 test problems [37] with D10 and D30. Then, we compared our modified versions of ABC with the original versions and some of the state of the art ABC versions. For a fair comparison, the experiments setting are the same as the setting in [12] (maxFES=10000*D and 51 independent runs).

It is noticed from Tables 10 and 11 that the proposed models have achieved the best results on most cases. These results confirm the conclusion that we made from the results mentioned above, on CEC2014 benchmark functions, about improving the efficiency of ABC original versions when our proposed modification was added to them. It is also emphasized that the modification we proposed has different effects on improving the original versions' performance. Consequently, we can conclude that the proposed modification affects the performance of ABC more than GABC, DFSABC, and TLABC. And it affects the performance of GABC more than DFSABC, and TLABC, and so on. This is due to the variety of the power of the exploitation of the original versions. Since our proposed modification improves the exploitation of ABC versions, it has a more negligible effect on TLABC due to its high exploitation compare with ABC or GABC (from the TLBO algorithm, which is one of the best local searching optimization algorithms).

From the results presented in Tables 10 and 11, we can see that ABC/ds is the best version of CEC2015 benchmark functions to compare its success against the other versions. Tables 12 and 13 compare the success of ABC/ds against the other versions of ABC on the CEC2015 benchmark for 10 and 30 dimensions respectively. The results in Tables 12 and 13 compare ABC/ds version with other versions as a paired comparison where the result shown in the cell which is in row 6 and column 2 in Tables 12 (12-1-2) means that ABC/ds is better than ABC in 12 problems, they have the same result in one problem, and worse in two problems. We can conclude from these two tables that the proposed version called ABC/ds is better than ABC, GABC, DFSABC, TLABC, GABC/ds, DFSABC/ds, TLABC/ds, DPABC, DPGABC, CABC, DPCABC, ARABC, OCABC, and HFPSO. **Results that are shown in Tables 12 and 13 indicate to the large amount of the improvement that happened when our proposed idea was implemented with ABC algorithm. ABC/ds shows not only high performance on specific kind of CEC2015 benchmark functions, but on all kinds of problems. Nonetheless, it is shown in Table 12 that no one of our proposed models achieves the best results on f_3, f_4, and f_5. Accordingly, we need to discuss why this failure occurs.**

In benchmark CEC2015, the problems from f_3 to f_9 are multinomial problems that are non-separable too. However, just f_3, f_4, and f_5 problems that have more specific characteristics such as "continuous but differentiable only on a set of points = f_3", "Local optima's number is huge and second better local optimum is far from the global optimum = f_4", and "Continuous everywhere yet differentiable nowhere = f_5". If we focused on those characteristics we can find that they can affect negatively our idea performance as the following:

- **Our idea depends on the indicator of the fertile area which its value increases when there is a slope. According to our suggestion, we can find enhancement of the fitness of the solutions in the areas that are inclined more than other areas. As a result, the indicators of the solutions in these areas will be larger than the others and they will be exploited first. This makes our model slower than the original models and sometimes gets worse results. For this reason our model has bad results on f_3 and f_5.**
- **Whenever there are many local optima, there will be many fertile areas. Therefore our model will need to exploit many areas which lead to slowing down finding the global optimal solution. For this reason our model has bad results on f_4.**

Table 8 Comparison results of the proposed GABC/ds algorithm and state of the art algorithms in terms of average error in objective function value for 30 dimension of CEC2014 test problems

F	GSA [22,23,44]	CS [22,23,58]	LX-BBO [22,20]	B-BBO [22,20]	SOS [22,23,8]	RW-GWO [22]	GWO [22]	OHDA [21]	fb_TSA[26]	BFL-PSO [6]	SDPSO [42]	GABC/ds
f_1	1.57E+07	2.29E+05	1.01E+07	6.50E+06	1.95E+07	8.02E+06	3.32E+07	4,40E+06	8,12E+06	2.92E+06	**2.41E+03**	2.98E+06
f_2	8.89E+03	1.39E+02	5.34E+04	2.36E+04	3.61E+09	2.23E+05	1.01E+09	6,03E+01	7,27E+03	1.15E+04	2.07E+00	**1.34E-03**
f_3	7.32E+04	1.05E+04	1.64E+04	6.04E+03	2.38E+04	3.16E+02	2.85E+04	9,31E+01	1,61E+04	1.62E+01	**2.06E+00**	2.35E+02
f_4	3.68E+02	7.07E+01	9.99E+01	1.03E+02	4.14E+02	3.41E+01	1.94E+02	9,91E+01	9,20E+01	8.08E+01	**3.82E+00**	2.48E+01
f_5	2.00E+01	2.02E+01	**3.06E+00**	3.74E+00	2.07E+01	2.05E+01	2.10E+01	2,00E+01	2,10E+01	2.03E+01	2.00E+01	2.03E+01
f_6	2.88E+01	2.48E+01	1.69E+01	2.00E+01	2.48E+01	9.84E+00	1.16E+01	2,95E+01	3,00E+00	**1.17E+00**	3.40E+01	6.58E+00
f_7	1.99E-04	7.45E-02	1.76E-01	7.82E-02	3.38E+01	2.53E-01	7.67E+00	5,55E-02	**0,00E+00**	3.38E-04	1.54E-02	1.13E-07
f_8	1.40E+02	7.12E+01	5.53E+01	4.71E-01	8.10E+01	4.38E+01	6.50E+01	1,33E+02	3,60E+01	8.12E-01	1.59E+02	**1.14E-13**
f_9	1.65E+02	1.78E+02	7.66E+01	9.11E+01	1.73E+02	6.33E+01	8.54E+01	1,99E+02	6,70E+01	**2.17E+01**	2.38E+02	2.95E+01
f_{10}	3.35E+03	2.02E+03	1.26E+04	6.69E+03	2.40E+03	9.61E+02	1.80E+03	3,13E+03	1,35E+03	1.93E+01	2.45E+03	**1.59E+00**
f_{11}	4.06E+03	4.49E+03	1.23E+04	6.72E+03	4.48E+03	2.68E+03	2.90E+03	4,18E+03	6,35E+03	1.50E+03	3.11E+03	**1.43E+03**
f_{12}	9.10E-02	8.11E-01	1.11E-02	1.11E-02	7.37E-01	5.45E-01	2.12E+00	4,82E-01	**0,00E+00**	3.03E-01	2.62E-01	8.57E-02
f_{13}	4.03E-01	4.17E-01	6.55E-01	6.75E-01	6.85E-01	2.80E-01	3.74E-01	4,66E-01	**0,00E+00**	1.12E-01	3.21E-01	1.28E-01
f_{14}	2.30E-01	5.18E-01	6.20E-01	3.93E-01	8.39E+00	4.23E-01	7.49E-01	3,05E-01	**0,00E+00**	2.17E-01	1.86E-01	7.82E-02
f_{15}	1.26E+01	1.32E+01	1.55E+01	1.88E+01	2.56E+02	8.81E+00	2.06E+01	6,03E+01	1,00E+01	3.83E+00	6.24E+00	**3.34E+00**
f_{16}	1.48E+01	1.23E+01	1.08E+01	1.07E+01	1.20E+01	1.03E+01	1.09E+01	1,26E+01	1,00E+01	**8.16E+00**	1.19E+01	8.90E+00
f_{17}	7.29E+05	1.24E+05	1.46E+06	1.28E+06	5.59E+06	5.71E+05	6.28E+05	2,88E+05	6,09E+05	3.79E+05	**4.39E+04**	9.37E+05
f_{18}	3.86E+02	1.42E+03	2.90E+03	8.22E+02	4.86E+05	6.52E+03	5.27E+05	2,94E+03	3,10E+02	1.12E+03	**1.89E+02**	4.21E+03
f_{19}	1.57E+02	1.18E+01	5.19E+03	7.81E+03	4.07E+01	1.14E+01	2.56E+01	1,93E+01	1,00E+01	**4.39E+00**	1.06E+01	6.01E+00
f_{20}	8.24E+04	**1.36E+02**	2.61E+04	1.63E+04	1.59E+04	6.27E+02	1.31E+04	1,13E+03	1,32E+04	6.36E+02	5.08E+03	8.95E+02
f_{21}	1.79E+05	**1.67E+03**	1.11E+06	1.23E+06	7.86E+05	2.58E+05	4.97E+05	6,76E+04	1,48E+05	6.62E+04	2.43E+04	2.00E+05
f_{22}	9.51E+02	3.11E+02	1.88E+03	1.68E+02	5.45E+02	2.08E+02	2.50E+02	5,53E+02	2,10E+02	**9.72E+01**	2.41E+02	2.44E+02
f_{23}	**2.00E+02**	3.44E+02	4.11E+02	3.43E+02	3.42E+02	3.15E+02	3.28E+02	3,16E+02	3,20E+02	3.15E+02	3.14E+02	3.15E+02
f_{24}	2.01E+02	2.21E+02	1.48E+04	3.41E+04	2.36E+02	**2.00E+02**	2.00E+02	2,54E+02	2,20E+02	2.24E+02	2.27E+02	2.02E+02
f_{25}	**2.00E+02**	2.09E+02	5.29E+02	6.54E+02	2.15E+02	2.04E+02	2.11E+02	2,24E+02	2,10E+02	2.05E+02	2.01E+02	2.05E+02
f_{26}	1.69E+02	1.01E+02	**2.13E+00**	3.64E+01	1.01E+02	1.00E+02	1.00E+02	1,00E+02	1,00E+02	1.02E+02	1.00E+02	1.00E+02
f_{27}	7.69E+02	4.18E+02	**1.96E+02**	3.05E+02	4.96E+02	4.09E+02	4.33E+02	5,69E+02	3,40E+02	3.69E+02	4.03E+02	4.04E+02
f_{28}	7.65E+02	9.13E+02	1.94E+03	2.12E+03	1.32E+03	4.34E+02	9.14E+02	5,03E+02	9,10E+02	8.14E+02	**4.07E+02**	7.92E+02
f_{29}	**2.00E+02**	2.01E+02	1.98E+07	3.09E+07	2.16E+04	2.14E+02	2.90E+05	2,16E+02	1,05E+03	1.34E+03	2.07E+02	2.84E+03
f_{30}	2.31E+04	3.59E+03	6.96E+06	1.38E+07	3.73E+04	6.69E+02	2.98E+04	1,00E+03	2,13E+03	2.27E+03	**4.11E+02**	2.06E+03

Table 9 Comparison the number of success of GABC/ds against the state of the art algorithms on the CEC2014 benchmark.

GABC/ds	G1(1-3)	G2(4-16)	G3(17-22)	G4(23-30)	Total
GSA [22,23,44]	3-0-0	12-0-1	3-0-3	3-0-5	21-0-9
CS [22,23,58]	2-0-1	12-0-1	2-0-4	7-0-1	23-0-7
LX-BBO [22,20]	3-0-0	11-0-2	5-0-1	6-0-2	25-0-5
B-BBO [22,20]	3-0-0	11-0-2	4-0-2	6-0-2	24-0-6
SOS [22,23,8]	3-0-0	13-0-0	6-0-0	8-0-0	30-0-0
RW-GWO [22]	3-0-0	13-0-0	3-0-3	1-2-5	20-2-8
GWO [22]	3-0-0	13-0-0	5-0-1	6-1-1	27-1-2
OHDA [21]	2-0-1	12-0-1	3-0-3	4-1-3	21-1-8
fb_TSA[26]	3-0-0	8-0-5	2-0-4	5-1-2	18-1-11
BFL-PSO [6]	1-0-2	9-0-4	0-0-6	4-2-2	14-2-14
SDPSO [42]	1-0-2	11-0-2	2-0-4	1-1-6	15-1-14

5.3 Proposed Models for solving engineering design problems

Three engineering design issues are used to see how effective the proposed algorithms are in solving practical optimization problems. These issues are commonly utilized in literature and all researchers can easily find their mathematical equations and explanations.

5.3.1 Pressure vessel design problem

This is a common problem in the literature [5] that aims to minimize the overall cost of a cylindrical material, shaping, and welding as in Fig 6. The problem is mathematically formulated by Eq. (13). In this equation, four design parameters need to be optimized. Two of them are continuous (L and R), and two are integers that are multiples of 0.0625 (T_s and T_h).

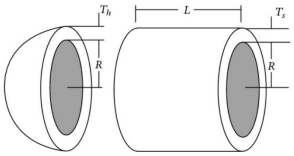

Fig. 6 Sketch map of the pressure vessel design problem

$$
\begin{aligned}
\min\ & f(x) = 0.6224x_1x_2x_3 + 1.7781x_2x_3^2 + \\
& \quad 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\
& x = (x_1, x_2, x_3, x_4) = (T_s, T_h, R, L) \\
& g_1(x) = -x_1 + 0.0193x_3 \le 0 \\
& g_2(x) = -x_2 + 0.00954x_3 \le 0 \\
& g_3(x) = -\pi x_3^2 x_4 - \tfrac{4}{3}\pi x_3^3 + 1296000 \le 0 \\
& g_4(x) = x_4 - 240 \le 0 \\
& 0 \le x_1, x_2 \le 99 \\
& 10 \le x_3, x_4 \le 200
\end{aligned}
\tag{13}
$$

Table 14 presents the experiment settings and results of the other optimization algorithms which are directly obtained from related papers [3,36,22]. To make the comparison fair, the same stopping criteria and the number of runs is chosen as in the study presented in [3] (100000 evaluations and 100 independent runs). The results presented in Table 14 confirm the efficiency of the proposed modified versions of ABC against their original versions on solving pressure vessel design problems. Moreover, the TLABC/ds, and DFSABC/ds versions succeeded in finding one of the minimum bests results (5885.332774) at (0.778169, 0.384649, 40.319619, 200.000000). Besides, the TLABC/ds finds the best mean and standard deviation among all compared algorithms in Table 14.

Table 10 Comparison results of the proposed algorithms against the other versions of ABC for 10-dimensional CEC2015 benchmark problems in terms of average error value.

F	ABC	DPABC [12]	GABC [12]	DPGABC [12]	CABC [18]	DPCABC [12]	DFSABC [11]	ARABC [12]	OCABC [18]	TLABC [7]	HFPSO [13]	ABC/ds	GABC/ds	DFSABC/ds	TLABC/ds
f_1	7.29E+05	1.11E+06	6.61E+05	1.20E+06	1.05E+06	1.43E+06	5.17E+05	8.78E+05	2.46E+05	2.29E+04	3.63E+05	6.05E+05	3.01E+05	6.85E+05	**1.67E+04**
f_2	1.27E+03	**3.61E+02**	4.79E+03	1.29E+03	4.73E+03	3.10E+03	4.43E+03	2.37E+03	4.44E+03	5.46E+03	6.72E+03	6.76E+02	4.91E+03	4.77E+03	5.96E+03
f_3	1.88E+01	1.87E+01	1.94E+01	1.85E+01	1.88E+01	1.83E+01	1.87E+01	**1.82E+01**	1.82E+01	1.92E+01	3.20E+02	1.89E+01	2.00E+01	1.86E+01	1.92E+01
f_4	8.93E+00	8.49E+00	5.30E+00	4.95E+00	5.85E+00	4.53E+00	4.00E+00	4.92E+00	**2.03E+00**	5.07E+00	4.39E+02	3.50E+00	3.03E+00	2.34E+00	5.11E+00
f_5	1.90E+02	1.64E+02	1.53E+02	1.22E+02	2.17E+02	1.58E+02	7.13E+01	1.45E+02	1.23E+02	**6.47E+01**	1.35E+03	1.37E+02	1.08E+02	1.02E+02	6.74E+01
f_6	3.38E+03	5.30E+03	3.10E+03	6.10E+03	7.40E+03	7.20E+03	2.69E+03	5.54E+03	1.25E+03	1.76E+02	3.80E+03	1.49E+03	2.48E+03	3.50E+03	**1.17E+02**
f_7	5.51E+01	5.07E+01	5.38E+01	4.58E+01	6.06E+01	4.81E+01	3.72E+01	5.50E+01	2.48E+01	6.15E-01	7.03E+02	3.18E-01	**2.10E-01**	3.31E-01	5.23E-01
f_8	9.95E+03	3.34E+04	4.66E+03	3.28E+04	8.04E+03	3.33E+04	9.50E+03	1.00E+04	2.55E+03	3.97E+01	4.83E+03	2.21E+03	1.65E+03	9.72E+03	**2.56E+01**
f_9	8.97E+01	9.66E+01	1.00E+02	1.00E+02	1.00E+02	1.00E+02	1.00E+02	1.00E+02	1.00E+02	1.00E+02	1.03E+03	**8.64E+01**	1.00E+02	1.00E+02	1.00E+02
f_{10}	1.45E+03	4.71E+03	1.96E+03	4.00E+03	2.70E+03	8.66E+03	1.64E+03	2.03E+03	7.65E+02	2.90E+02	3.94E+03	9.75E+02	5.54E+02	1.34E+03	**2.66E+02**
f_{11}	7.09E+01	1.36E+02	8.23E+01	1.23E+02	8.87E+01	1.44E+02	1.06E+02	8.40E+01	**5.87E+01**	1.49E+02	1.43E+03	1.40E+02	6.99E+01	2.73E+02	2.07E+02
f_{12}	1.03E+02	1.03E+02	1.02E+02	1.02E+02	1.03E+02	1.03E+02	1.02E+02	1.03E+02	1.02E+02	**1.01E+02**	1.31E+03	1.01E+02	1.01E+02	1.02E+02	1.01E+02
f_{13}	3.06E+02	3.05E+02	3.06E+02	3.05E+02	3.07E+02	3.07E+02	3.06E+02	3.06E+02	3.06E+02	3.06E-02	1.30E+03	**1.72E-02**	2.22E-02	3.06E-02	3.09E-02
f_{14}	4.49E+02	4.63E+02	8.56E+02	5.40E+02	1.94E+03	1.12E+03	1.64E+03	1.08E+03	2.58E+03	2.84E+03	5.86E+03	**3.92E+02**	1.03E+03	4.30E+03	4.70E+03
f_{15}	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	1.60E+03	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**

Table 11 Comparison results of the proposed algorithms against the other versions of ABC for 30-dimensional CEC2015 benchmark problems in terms of average error value.

F	ABC	DPABC [12]	GABC [12]	DPGABC [12]	CABC [18]	DPCABC [12]	DFSABC [11]	ARABC [12]	OCABC [18]	TLABC [7]	HFPSO [13]	ABC/ds	GABC/ds	DFSABC/ds	TLABC/ds
f_1	2.86E+06	2.69E+06	2.67E+06	2.69E+06	4.73E+06	3.72E+06	2.45E+06	5.20E+06	1.95E+06	7.08E+04	1.46E+06	1.59E+06	1.41E+06	2.73E+06	**6.52E+04**
f_2	1.73E+03	**4.11E+02**	3.50E+03	1.32E+03	4.23E+03	2.51E+03	1.35E+03	1.34E+03	9.18E+02	1.58E+03	4.95E+03	7.18E+02	3.05E+03	1.50E+03	1.37E+03
f_3	2.02E+01	**2.01E+01**	2.02E+01	2.01E+01	2.03E+01	2.02E+01	2.01E+01	2.02E+01	2.05E+01	2.09E+01	3.20E+02	2.03E+01	2.03E+01	2.01E+01	2.09E+01
f_4	7.65E+01	7.35E+01	5.11E+01	4.91E+01	5.03E+01	4.42E+01	4.20E+01	4.29E+01	2.50E+01	5.38E+01	5.54E+02	3.23E+01	2.71E+01	**1.88E+01**	5.11E+01
f_5	2.01E+03	1.88E+03	1.81E+03	1.80E+03	2.22E+03	1.87E+03	1.67E+03	1.80E+03	2.52E+03	**1.42E+03**	4.08E+03	1.59E+03	1.55E+03	1.54E+03	1.72E+03
f_6	1.22E+06	9.77E+05	1.11E+06	1.28E+06	1.81E+06	2.34E+06	8.10E+05	2.15E+06	3.42E+05	5.47E+04	1.68E+05	2.61E+05	3.21E+05	4.14E+05	**2.86E+04**
f_7	7.35E+00	6.45E+00	8.33E+00	6.70E+00	1.03E+01	7.86E+00	8.12E+00	8.94E+00	7.24E+00	6.52E+00	7.21E+02	**5.94E+00**	8.35E+00	6.91E+00	6.90E+00
f_8	2.20E+05	3.02E+05	1.99E+05	3.74E+05	3.15E+05	3.46E+05	2.49E+05	3.69E+05	7.83E+04	3.24E+04	6.56E+04	7.77E+04	1.22E+05	9.74E+04	**2.37E+04**
f_9	1.05E+02	1.04E+02	1.04E+02	1.04E+02	1.04E+02	1.04E+02	1.04E+02	1.04E+02	1.04E+02	**1.03E+02**	1.05E+03	1.04E+02	1.03E+02	1.08E+02	1.03E+02
f_{10}	6.24E+05	6.14E+05	7.36E+05	6.64E+05	1.09E+06	1.23E+06	5.93E+05	1.06E+06	2.80E+05	3.03E+04	1.10E+05	4.35E+05	2.65E+05	5.73E+05	**1.75E+04**
f_{11}	3.26E+02	3.26E+02	3.26E+02	3.26E+02	3.39E+02	3.43E+02	3.25E+02	3.46E+02	3.52E+02	7.06E+02	1.80E+03	**3.08E+02**	3.09E+02	3.51E+02	7.22E+02
f_{12}	1.06E+02	1.06E+02	1.06E+02	1.06E+02	1.06E+02	1.06E+02	**1.05E+02**	1.06E+02	1.06E+02	1.05E+02	1.33E+03	1.05E+02	1.05E+02	1.05E+02	1.08E+02
f_{13}	3.11E+02	3.11E+02	2.83E+02	2.83E+02	2.80E+02	2.79E+02	2.71E+02	2.68E+02	2.78E+02	2.94E-02	1.30E+03	**2.33E-02**	2.56E-02	2.63E-02	3.03E-02
f_{14}	3.08E+04	3.07E+04	3.14E+04	3.15E+04	3.17E+04	3.16E+04	3.18E+04	3.16E+04	3.23E+04	3.37E+04	3.46E+04	**3.05E+04**	3.13E+04	3.19E+04	3.38E+04
f_{15}	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	1.00E+02	1.60E+03	1.00E+02	**1.00E+02**	**1.00E+02**	1.00E+02

Table 12 Comparison of the number of success of ABC/ds against the other versions of ABC on the CEC2015 benchmark for 10 dimensions

ABC/ds	G1(1-2)	G2(3-5)	G3(6-8)	G4(9-15)	Total
ABC	2-0-0	2-0-1	3-0-0	6-1-0	13-1-0
GABC	2-0-0	2-0-1	3-0-0	5-1-1	12-1-2
DFSABC	2-0-0	2-0-1	3-0-0	4-1-2	11-1-3
TLABC	1-0-1	2-0-1	1-0-2	3-1-3	7-1-7
GABC/ds	1-0-1	0-0-3	3-0-0	3-1-3	7-1-7
DFSABC/ds	2-0-0	0-0-3	3-0-0	6-1-0	11-1-3
TLABC/ds	1-0-1	3-0-0	1-0-2	5-1-2	10-0-5
DPABC	1-0-1	2-0-1	3-0-0	6-1-0	12-1-2
DPGABC	2-0-0	2-0-1	3-0-0	6-1-0	13-1-1
CABC	2-0-0	3-0-0	3-0-0	6-1-0	14-1-0
DPCABC	2-0-0	2-0-1	3-0-0	6-1-0	13-1-1
ARABC	2-0-0	2-0-1	3-0-0	6-1-0	13-1-1
OCABC	2-0-0	2-0-1	3-0-0	5-1-1	12-1-2
HFPSO [13]	1-0-1	3-0-0	3-0-0	6-0-0	14-0-1

Table 13 Comparison of the number of success of ABC/ds against the other versions of ABC on the CEC2015 benchmark for 30 dimensions

ABC/ds	G1(1-2)	G2(3-5)	G3(6-8)	G4(9-15)	Total
ABC	2-0-0	2-0-1	3-0-0	5-1-1	12-1-2
GABC	2-0-0	3-0-0	3-0-0	5-1-1	13-1-1
DFSABC	1-0-1	1-0-2	3-0-0	5-1-1	10-1-4
TLABC	1-0-1	2-0-1	1-0-2	5-1-1	9-1-5
GABC/ds	1-0-1	1-0-2	1-0-2	4-1-2	7-1-7
DFSABC/ds	2-0-0	0-0-3	3-0-0	6-1-0	11-1-3
TLABC/ds	1-0-1	2-0-1	1-0-2	6-1-0	10-0-5
DPABC	1-0-1	2-0-1	3-0-0	5-1-1	11-1-3
DPGABC	2-0-0	1-0-2	3-0-0	5-1-1	11-1-3
CABC	2-0-0	2-0-1	3-0-0	5-1-1	12-1-2
DPCABC	2-0-0	2-0-1	3-0-0	6-1-0	13-1-1
ARABC	2-0-0	2-0-1	3-0-0	5-1-1	12-1-2
OCABC	1-0-1	0-0-3	2-0-1	4-1-2	7-1-7
HFPSO [13]	1-0-1	3-0-0	1-0-2	6-0-1	11-0-4

5.3.2 Tension and compression spring design problem

This problem aims to minimize the weight, subject to restrictions on shear stress, deflect, surge frequency, external diameter and design variables, tension, and compression spring as illustrated in Fig 7 [28]. The problem is mathematically formulated by Eq. (14). The design parameters that need to be optimized in this problem are wire diameter (x_1), mean coil diameter (x_2), and the number of active coils (x_3).

Table 14 Statistical results of different methods for the pressure vessel design problem

Method	Worst	Mean	Best	Std.
GA3	6308,497000	6293,843200	6288,744500	7,410000
GA4	6469,322000	6177,253300	6059,946300	130,000000
CPSO	6363,804100	6147,133200	6061,077700	86,400000
HPSO	6288,677000	6099,932300	6059,714300	86,200000
NM-PSO	5960,055700	5946,790100	5930,313700	9,160000
G-QPSO	7544,492500	6440,378600	6059,720800	448,000000
QPSO	8017,281600	6440,378600	6059,720900	479,000000
PSO	14076,324000	8756,680300	6693,721200	1490,000000
CDE	6371,045500	6085,230300	6059,734000	43,000000
UPSO	N/A	9032,550000	6544,270000	995,000000
PSO-DE	N/A	6059,714000	6059,714000	N/A
ABC	N/A	6245,308000	6059,714000	205,000000
(+)-ES	N/A	6379,938000	6059,701600	210,000000
TLBO	N/A	6059,714300	6059,714300	N/A
MBA	6392,506200	6200,647700	5889,321600	160,000000
EPSO	7315,675200	6254,180400	5885,338300	424,000000
HFPSO	6940,824900	6140,590500	5885,332900	197,000000
GWO	N/A	N/A	6136,660000	N/A
RW-GWO	N/A	N/A	6059,736000	N/A
ABC/ds	5989,578922	5908,436787	5886,543058	21,933888
GABC/ds	5885,353308	5885,333689	5885,332774	0,002671
DFSABC/ds	7319,000702	5909,465773	5885,332774	143,718183
TLABC/ds	5885,332774	5885,332774	5885,332774	0,000000

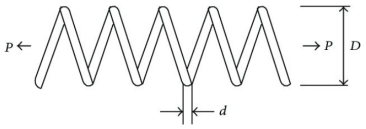

Fig. 7 Sketch map of the tension and compression spring design problem

Table 15 Statistical results of different methods for the tension and compression spring design problem

Method	Worst	Mean	Best	Std.
GA3	0,012822	0,012769	0,012705	0,000039
GA4	0,012973	0,012742	0,012681	0,000059
CPSO	0,015116	0,013568	0,012721	0,000842
HPSO	0,012924	0,012730	0,012675	0,000520
NM-PSO	0,012719	0,012707	0,012665	0,000016
G-QPSO	0,012633	0,012631	0,012630	0,000001
QPSO	0,017759	0,013524	0,012665	0,001270
PSO	0,018127	0,013854	0,012669	0,001340
CDE	0,071802	0,019555	0,012857	0,011600
UPSO	0,012790	0,012703	0,012670	0,000027
PSO-DE	0,012666	0,012665	0,012665	0,000000
ABC	0,012738	0,012669	0,012665	0,000013
(+)-ES	0,012665	0,012665	0,012665	0,000000
TLBO	0,012665	0,012665	0,012665	0,000000
MBA	0,016717	0,012923	0,012669	0,000590
EPSO	N/A	0,022940	0,013120	0,007200
HFPSO	N/A	0,013165	0,012689	0,000390
GWO	N/A	N/A	0,012675	N/A
RW-GWO	N/A	N/A	0,012674	N/A
ABC/ds	0,013160	0,012754	0,012670	0,000071
GABC/ds	0,012795	0,012694	0,012667	0,000024
DFSABC/ds	0,013558	0,012840	0,012674	0,000155
TLABC/ds	0,012702	0,012669	0,012665	0,000007

$$
\begin{aligned}
&\min\ f(x) = (x_3+2)x_2x_1^2 \\
&g_1(x) = 1 - \tfrac{x_2^3x_3}{71785x_1^4} \le 0 \\
&g_2(x) = \tfrac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \tfrac{1}{5108x_1^2} - 1 \le 0 \\
&g_3(x) = 1 - \tfrac{140.45x_1}{x_2^2x_3} \le 0 \\
&g_4(x) = \tfrac{x_1+x_2}{1.5} - 1 \le 0 \\
&0.05 \le x_1 \le 2 \\
&0.25 \le x_2 \le 1.30 \\
&2 \le x_3 \le 15
\end{aligned}
\tag{14}
$$

Table 15 presents the experiment settings and results of the other optimization algorithms directly obtained from related papers [22,4,45,54]. The results shown in Table 15 are concluded from 100 independent runs. These results concluded that the proposed modified versions are efficient and competitive in solving tension and compression spring design problem.

5.3.3 Frequency-Modulated (FM) problem

In some contemporary music systems, FM sound wave synthesis has a significant role. The FM synthesizer optimization problem is to produce a target-like sound. The problem is a highly complex multimodal with six-dimensional and a minimum value of $f(x) = 0$ [22]. The problem is mathematically formulated by Eq. (15). The decision vector that needs to be optimized is [a_1, a_2, a_3, w_1, w_2, w_3].

$$
\begin{aligned}
&\min\ f(x) = \textstyle\sum_{t=0}^{100}(y(t) - y_0(t))^2 \\
&y(t) = a_1 sin(w_1t\theta + a_2 sin(w_2t\theta + a_3 sin(w_3t\theta))) \\
&y_0(t) = (1.0)si(5.0)t\theta - (1.5)sin(4.8)t\theta + (2.0)sin(4.9)t\theta \\
&\theta = 2\pi/100 \ and\ parameter\ range\ is\ [-6.4,\ 6.35]
\end{aligned}
\tag{15}
$$

Table 16 Statistical results of different methods for the frequency-modulated problem

Method	Worst	Mean	Best	Std.
RW-GWO	18,791700	8,871170	0,006001	6,948362
GWO	25,163300	20,038300	1,931100	5,917430
CLPSO	14,080000	3,820000	0,007000	23,530000
CPSOH	42,520000	27,080000	3,450000	60,610000
TRIBES-D	22,240000	14,680000	2,220000	4,570000
G-CMA-ES	55,090000	38,750000	3,326000	16,770000
ABC	18,681163	12,764227	1,020556	3,772660
ABC/ds	21,368424	12,028569	0,004893	5,321962
GABC/ds	19,189378	11,664234	0,001799	5,639880
DFSABC/ds	21,253264	10,344264	0,000036	6,301305
TLABC/ds	20,210335	3,902105	0,000000	6,437108

Table 16 presents the experiment settings and results of the other optimization algorithms, which are directly obtained from related papers [5,3,36,22,39]. The results shown in Table 16 are concluded from 30 independent runs with 30000 maximum number of evaluations. The results in Table 16 indicate that the proposed modified versions can also solve the FM problem. Indeed, these results concluded that although the proposed modified versions did not find the optimal solution to this problem, they found results better than the original versions except the TLABC. However, the best mean result in the FM problem among the four proposed versions is achieved by TLABC/ds.

6 Discussion and Conclusion

This paper suggests a modification that can be applied to ABC versions and enhance their performances. To check the performance of the proposed modified versions, some well-known practical engineering design problems besides the CEC2014 and CEC2015 benchmark functions are used. In CEC2014 and CEC2015 benchmark functions experiments, 10, 30, and 50 variables are used. The proposed modified versions are compared to the original version of ABC, some common versions of ABC, and some state-of-the-art meta-heuristic algorithms. The overall results of this modification enhanced the performance of the original versions. Moreover, based on results gained from the experiments of CEC2014 benchmark functions, GABC/ds can provide highly competitive performance compared with ABC, GABC, DFSABC, TLABC, ABC/ds, DFSABC/ds, TLABC/ds, GSA, CS, LX-BBO, B-BBO, SOS, GWO, and RW-GWO. Furthermore, the other proposed version ABC/ds can provide higher competitive performance than ABC, GABC, DFSABC, TLABC, GABC/ds, DFSABC/ds, TLABC/ds based on the results gained from the experiments of CEC2015 benchmark functions. We can summarize our results according to the type of optimization problems as the following. First, proposed modified ver-

sions of the ABC algorithm have a superior exploitation ability on the unimodal functions. Second, the results of multimodal functions verified the efficient exploration ability of the proposed modified versions. Third, the results on the hybrid and composite functions were strong evidence of the superiority at avoiding the local optimal. Finally, the convergence analysis of these versions against their original versions verified an improvement in convergence. Moreover, the real-life problems have shown that the proposed modified versions of ABC have high efficiency in unknown search areas.

For future studies, this proposed modification can also be applied to the other meta-heuristics to improve their efficiency. Multi-objective and binary versions of the proposed algorithm can be investigated. Furthermore, we will integrate the proposed model with machine learning techniques such as Artificial Neural Networks and Support Vector Machine to optimize their parameters.

Declarations

Funding No funding was received for conducting this study.

Conflicts of interest/Competing interests The authors have no conflicts of interest to declare that are relevant to the content of this article.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors

Availability of data and material Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Code availability Codes written during the current study can be obtained from the relevant author upon reasonable request.

Authors' contributions Radhwan A.A. Saleh: Conceptualization of this study, Methodology, Software, Data Curation, Visualization, Writing - Original draft preparation. Rüştü Akay: Supervision, Methodology, Visualization, Writing - Reviewing and Editing, Investigation.

References

1. Abraham, A., Jatoth, R.K., Rajasekhar, A.: Hybrid differential artificial bee colony algorithm. Journal of Computational and Theoretical Nanoscience **9**(2), 249–257 (2012)
2. Akay, B., Karaboga, D.: A modified artificial bee colony algorithm for real-parameter optimization. Information Sciences **192**, 120–142 (2012)
3. Auger, A., Hansen, N.: A restart cma evolution strategy with increasing population size. In: 2005 IEEE Congress on Evolutionary Computation, vol. 2, pp. 1769–1776. IEEE
4. Aydilek, I.B.: A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems. Applied Soft Computing **66**, 232–249 (2018)
5. Van den Bergh, F., Engelbrecht, A.P.: A cooperative approach to particle swarm optimization. IEEE Transactions on Evolutionary Computation **8**(3), 225–239 (2004)
6. Chen, X., Tianfield, H., Du, W.: Bee-foraging learning particle swarm optimization. Applied Soft Computing **102**, 107134 (2021)
7. Chen, X., Xu, B.: Teaching-learning-based artificial bee colony. In: International Conference on Swarm Intelligence, pp. 166–178. Springer
8. Cheng, M.Y., Prayogo, D.: Symbiotic organisms search: A new metaheuristic optimization algorithm. Computers and Structures **139**, 98 – 112 (2014)
9. Cui, L., Li, G., Lin, Q., Du, Z., Gao, W., Chen, J., Lu, N.: A novel artificial bee colony algorithm with depth-first search framework and elite-guided search equation. Information Sciences **367**, 1012–1044 (2016)
10. Cui, L., Li, G., Lin, Q., Du, Z., Gao, W., Chen, J., Lu, N.: A novel artificial bee colony algorithm with depth-first search framework and elite-guided search equation. Information Sciences **367**, 1012–1044 (2016)
11. Cui, L., Li, G., Lin, Q., Du, Z., Gao, W., Chen, J., Lu, N.: A novel artificial bee colony algorithm with depth-first search framework and elite-guided search equation. Information Sciences **367-368**, 1012–1044 (2016)
12. Cui, L., Li, G., Luo, Y., Chen, F., Ming, Z., Lu, N., Lu, J.: An enhanced artificial bee colony algorithm with dual-population framework. Swarm and Evolutionary Computation **43**, 184–206 (2018)
13. Farshi, T.R., Ardabili, A.K.: A hybrid firefly and particle swarm optimization algorithm applied to multilevel image thresholding. Multimedia Systems **27**(1), 125–142 (2021)
14. Gao, W., Chan, F.T., Huang, L., Liu, S.: Bare bones artificial bee colony algorithm with parameter adaptation and fitness-based neighborhood. Information Sciences **316**, 180–200 (2015)
15. Gao, W., Liu, S.: Improved artificial bee colony algorithm for global optimization. Information Processing Letters **111**(17), 871–882 (2011)
16. Gao, W., Liu, S., Huang, L.: A global best artificial bee colony algorithm for global optimization. Journal of Computational and Applied Mathematics **236**(11), 2741–2753 (2012)
17. Gao, W.f., Liu, S.y., Huang, L.l.: A novel artificial bee colony algorithm based on modified search equation and orthogonal learning. IEEE Transactions on Cybernetics **43**(3), 1011–1024 (2013)
18. Gao, W.f., Liu, S.y., Huang, L.l.: A novel artificial bee colony algorithm based on modified search equation and orthogonal learning. IEEE Transactions on Cybernetics **43**(3), 1011–1024 (2013)
19. Gao, W.F., Liu, S.Y., Huang, L.L.: A novel artificial bee colony algorithm with powell's method. Applied Soft Computing **13**(9), 3763–3775 (2013)
20. Garg, V., Deep, K.: Performance of laplacian biogeography-based optimization algorithm on cec

2014 continuous optimization benchmarks and camera calibration problem. Swarm and Evolutionary Computation **27**, 132 – 144 (2016)
21. GhaemiDizaji, M., Dadkhah, C., Leung, H.: Ohda: An opposition based high dimensional optimization algorithm. Applied Soft Computing **91**, 106185 (2020)
22. Gupta, S., Deep, K.: A novel random walk grey wolf optimizer. Swarm and Evolutionary Computation **44**, 101–112 (2019)
23. Heidari, A.A., Pahlavani, P.: An efficient modified grey wolf optimizer with lévy flight for optimization tasks. Applied Soft Computing **60**, 115 – 134 (2017)
24. Jadon, S.S., Bansal, J.C., Tiwari, R., Sharma, H.: Accelerating artificial bee colony algorithm with adaptive local search. Memetic Computing **7**(3), 215–230 (2015)
25. Jadon, S.S., Tiwari, R., Sharma, H., Bansal, J.C.: Hybrid artificial bee colony algorithm with differential evolution. Applied Soft Computing **58**, 11–24 (2017)
26. Jiang, J., Meng, X., Chen, Y., Qiu, C., Liu, Y., Li, K.: Enhancing tree-seed algorithm via feed-back mechanism for optimizing continuous problems. Applied Soft Computing **92**, 106314 (2020)
27. Kang, F., Li, J., Li, H.: Artificial bee colony algorithm and pattern search hybridized for global optimization. Applied Soft Computing **13**(4), 1781–1791 (2013)
28. Kannan, B., Kramer, S.N.: An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design (1994)
29. Karaboga, D.: An idea based on honey bee swarm for numerical optimization
30. Karaboga, D., Akay, B.: A comparative study of artificial bee colony algorithm. Applied mathematics and computation **214**(1), 108–132 (2009)
31. Karaboga, D., Basturk, B.: On the performance of artificial bee colony (abc) algorithm. Applied Soft Computing **8**(1), 687–697 (2008)
32. Karaboga, D., Gorkemli, B.: A quick artificial bee colony (qabc) algorithm and its performance on optimization problems. Applied Soft Computing **23**, 227–238 (2014)
33. Karaboga, D., Gorkemli, B., Ozturk, C., Karaboga, N.: A comprehensive survey: artificial bee colony (abc) algorithm and applications. Artificial Intelligence Review pp. 21–57 (2014)
34. Kiran, M.S., Hakli, H., Gunduz, M., Uguz, H.: Artificial bee colony algorithm with variable search strategy for continuous optimization. Information Sciences **300**, 140–157 (2015)
35. Kıran, M.S., Fındık, O.: A directed artificial bee colony algorithm. Applied Soft Computing **26**, 454–462 (2015)
36. Li, C., Yang, S., Nguyen, T.T.: A self-learning particle swarm optimizer for global optimization problems. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **42**(3), 627–646 (2011)
37. Liang, J., Qu, B., Suganthan, P., Chen, Q.: Problem definitions and evaluation criteria for the cec 2015 competition on learning-based real-parameter single objective optimization. Technical Report201411A, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore **29**, 625–640 (2014)
38. Liang, J., Qu, B., Suganthan, P., Hernandez-Diaz, A.: Problem definitions and evaluation criteria for the cec 2013 special session on real-parameter optimization. Technical Report, Nanyang Technological University, Singapore (2013)
39. Liang, J.J., Qin, A.K., Suganthan, P.N., Baskar, S.: Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Transactions on Evolutionary Computation **10**(3), 281–295 (2006)
40. Liang, J.J., Qu, B.Y., Suganthan, P.N.: Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore **635** (2013)
41. Luo, J., Wang, Q., Xiao, X.: A modified artificial bee colony algorithm based on converge-onlookers approach for global optimization. Applied Mathematics and Computation **219**(20), 10253–10262 (2013)
42. Miao, K., Feng, Q., Kuang, W.: Particle swarm optimization combined with inertia-free velocity and direction search. Electronics **10**(5) (2021)
43. Mirjalili, S., Mirjalili, S.M., Lewis, A.: Grey wolf optimizer. Advances in Engineering Software **69**, 46–61 (2014)
44. Rashedi, E., Nezamabadi-pour, H., Saryazdi, S.: Gsa: A gravitational search algorithm. Information Sciences **179**(13), 2232 – 2248 (2009). Special Section on High Order Fuzzy Sets
45. Sadollah, A., Bahreininejad, A., Eskandar, H., Hamdi, M.: Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. Applied Soft Computing **13**(5), 2592–2612 (2013)
46. Saleh, R.A.A., Akay, R.: Indoor localization using artificial bee colony with levy flight. Avrupa Bilim ve Teknoloji Dergisi pp. 152–156 (2019)
47. dos Santos Coelho, L., Alotto, P.: Gaussian artificial bee colony algorithm approach applied to loney's solenoid benchmark problem. IEEE Transactions on Magnetics **47**(5), 1326–1329 (2011)
48. Sharma, H., Bansal, J.C., Arya, K., Yang, X.S.: Levy flight artificial bee colony algorithm. International Journal of Systems Science **47**(11), 2652–2670 (2016)
49. Sharma, H., Bansal, J.C., Arya, K.V.: Opposition based lévy flight artificial bee colony. Memetic Computing **5**(3), 213–227 (2013)
50. Shi, X., Li, Y., Li, H., Guan, R., Wang, L., Liang, Y.: An integrated algorithm based on artificial bee colony and particle swarm optimization. In: 2010 Sixth international conference on natural computation, vol. 5, pp. 2586–2590. IEEE
51. TSai, P.W., Pan, J.S., Liao, B.Y., Chu, S.C.: Enhanced artificial bee colony optimization. International Journal of Innovative Computing, Information and Control **5**(12), 5081–5092 (2009)
52. Wang, H., Wang, W., Xiao, S., Cui, Z., Xu, M., Zhou, X.: Improving artificial bee colony algorithm using a new neighborhood selection mechanism. Information Sciences **527**, 227 – 240 (2020)
53. Wang, H., Wu, Z., Rahnamayan, S., Sun, H., Liu, Y., Pan, J.s.: Multi-strategy ensemble artificial bee colony algorithm. Information Sciences **279**, 587–603 (2014)
54. Westera, W.: How people learn while playing serious games: A computational modelling approach. Journal of Computational Science **18**, 32–45 (2017)
55. Wolpert, D.H., Macready, W.G.: No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation **1**(1), 67–82 (1997)

56. Xiang, W.L., An, M.Q.: An efficient and robust artificial bee colony algorithm for numerical optimization. Computers and Operations Research **40**(5), 1256–1265 (2013)
57. Xiang, Y., Peng, Y., Zhong, Y., Chen, Z., Lu, X., Zhong, X.: A particle swarm inspired multi-elitist artificial bee colony algorithm for real-parameter optimization. Computational Optimization and Applications **57**(2), 493–516 (2014)
58. Yang, X., Suash Deb: Cuckoo search via lévy flights. In: 2009 World Congress on Nature Biologically Inspired Computing (NaBIC), pp. 210–214 (2009)
59. Zhao, H., Pei, Z., Jiang, J., Guan, R., Wang, C., Shi, X.: A hybrid swarm intelligent method based on genetic algorithm and artificial bee colony. In: International Conference in Swarm Intelligence, pp. 558–565. Springer
60. Zhou, X., Lu, J., Huang, J., Zhong, M., Wang, M.: Enhancing artificial bee colony algorithm with multi-elite guidance. Information Sciences **543**, 242 – 258 (2021)
61. Zhou, X., Wang, H., Wang, M., Wan, J.: Enhancing the modified artificial bee colony algorithm with neighborhood search. Soft Computing **21**(10), 2733–2743 (2017)
62. Zhu, G., Kwong, S.: Gbest-guided artificial bee colony algorithm for numerical function optimization. Applied Mathematics and Computation **217**(7), 3166–3173 (2010)