Applied Intelligence (2021) 51:1992–2017
https://doi.org/10.1007/s10489-020-01898-8

A novel bat algorithm with dynamic membrane structure for optimization problems

Bisan Alsalibi[1] · **Laith Abualigah**[2] · **Ahamad Tajudin Khader**[1]

Abstract

To improve the optimization efficiency for different optimization problems and take advantage of the dynamic membrane computing framework, this paper proposes an improved bat algorithm, namely, Dynamic Membrane-driven Bat Algorithm (DMBA). The dynamic construction of the DMBA algorithm aims at enhancing population diversity by balancing the exploration-exploitation tradeoff. Unlike the static membrane algorithms, the membranes in DMBA will be dynamically evolved by using merging and separation rules which help in maintaining the diversity of the population. The experimental results on a set of well-known benchmark functions including CEC 2005, CEC 2011, and CEC 2017 clearly prove the effectiveness of the proposed DMBA algorithm in terms of maintaining the diversity and exploitation capabilities compared to that of the others. It is shown that the proposed DMBA algorithm is superior to recent variants of the bat algorithm and other well-known algorithms in terms of solution accuracy and convergence speed.

Keywords Dynamic membrane structure · Parallel membrane framework · Bat algorithm · Optimization problems

1 Introduction

Recently, a myriad of meta-heuristic optimization algorithms has been introduced in the literature to solve a variety of optimization problems [1–4]. Among these, Bat Algorithm (BA) is motivated by the typical echolocation behavior of bats [5]. Despite its successful applications in several areas, the classical BA has two drawbacks [6, 7]. Firstly, it cannot keep track of the current best positions which causes a rapid loss of population diversity and premature convergence. Secondly, it suffers from a slow convergence speed [8–10].

To overcome these drawbacks, BA algorithm has been enhanced by various researchers to improve its efficiency. To that aim, multiple strategies have been employed such as diversifying the initial bat population and enhancing the search characteristics. For instance, chaos-based bat algorithm (CBA) has been proposed by [11] where different chaotic maps have been incorporated into the standard BA to improve its global search behavior. Similarly, chaotic maps have been used for the same purposes by [8, 12]. In their work, a modified version of BA is proposed by employing the concepts of multi-population strategies [13]. A discrete BA is proposed to solve feature selection and classification problems [14], this version demonstrated the superiority of BA over other swarm-based approaches. Along similar lines, another discrete version of BA has been proposed by [15] to solve the traveling salesman problem (TSP) problems. A dynamic virtual BA (DVBA) has been proposed by [16] where only two bats are required to reach the optimal solution. These bats are called the explorer, which explores the search space and the other bat is called the exploiter, which concentrates the search around the best solutions. A double-subpopulation version of the BA has been proposed in which the external subpopulation uses a dynamic weight mechanism to update the bat positions [6]. The interior subpopulation uses Levy flight to enhance the local exploiting searchability. An enhanced version of

---

✉ Laith Abualigah
Aligah.2020@gmial.com

Bisan Alsalibi
bisan.salibi@usm.my

Ahamad Tajudin Khader
tajudin@usm.my

[1] School of Computer Sciences, Universiti Sains Malaysia, Penang, USM 11800, Malaysia

[2] Faculty of Computer Sciences and Informatics, Amman Arab University, Amman 11953, Jordan

the classical BA, called (EBA), has been proposed [17]. In this work, an Inertia weight has been introduced to the velocity update equation in order to balance the global and local search capabilities of the standard BA. Directional BA (dBA) has been proposed [18], where the directional echolocation has been introduced to the classical BA.

Ten chaotic maps have been embedded into the Gravitational Search Algorithm (GSA) [19]. In [20], a hybrid bat algorithm (HBA) has been introduced by hybridizing extremal optimization (EO) algorithm. In another work, an enhanced variant of the bat algorithm called ILSSIWBA has been proposed which combined the iterative local search with inertial weight. Most recently, a new version of the bat algorithm called MBADE has been proposed by [21] where the standard bat algorithm has been combined with differential evolution (DE) algorithm.

The structured population mechanism and the multi-population concepts are considered as efficient models for evolutionary algorithms (EAs) [22–24]. The main motivation behind the membrane evolutionary mechanisms is to enhance the diversity behavior of EAs as it enables its execution in a parallel manner using multi-core machines. A parallel variant of harmony search algorithm is proposed in [25]. In their work, a static membrane structure and communication rules have been used to enable the communication between the membranes on different cores.

Membrane models with active membranes have been used to deal with many complex and large problems due to membrane division, separation, merging and creation [22]. Hence, the idea of dynamic membrane structure is used in this work. In this paper, an enhanced bat algorithm (BA) is proposed to solve benchmark optimization problems, namely, Dynamic Membrane-driven Bat Algorithm (DMBA). A dynamic membrane structure has been incorporated into DMBA to maintain the diversity of population. Merging and separation rules are used to build the dynamic membrane structure. Furthermore, evolution and communication rules of membrane systems have been used to enhance the trajectories of bats during the search process. Finally, to increase the speed of the search process, a parallel implementation has been considered to the proposed method DMBA as well. To validate the performance of the proposed method (DMBA), mathematical benchmark functions from CEC2005, CEC2011 and CEC2017 are utilized. Moreover, the sensitivity analysis and performance of the proposed algorithm are compared with several other algorithms published in the literature and its parameter setting is studied using the same functions.

The rest of this paper is organized as follows: The related works are presented in Section 2. The background part, which describes the classical BA with its basic design and a brief overview of membrane computing is presented in Section 3. The description of the proposed Dynamic Membrane-driven Bat Algorithm (DMBA) is described in Section 5. The experiments and results are provided in Section 6. Finally, this paper is concluded and the potential future works are given in Section 7.

2 Membrane-inspired evolutionary algorithms

2.1 Background

Paun investigated the chemical reactions that take place within living cells and introduced membrane systems in 1998 [26]. Since its introduction, various membrane models have been proposed which are known as P systems in honor of their initiator G. Paun. The three popular models are cell-like P system [27], tissue-like P system [28] and spiking neural-like P system [29]. In the past years, a novel research line of MC, that aims to combine evolutionary algorithms and membrane computing, has been introduced [30]. The obtained algorithms are called membrane-inspired evolutionary algorithms [22]. The integration of membrane systems and evolutionary approaches have been firstly introduced by Nishida [31]. Where the first version of MC has been proposed by combining a Nested Membrane Structure (NMS) with tabu search algorithm to solve the TSP. Since then, various kinds of MIEA have been proposed as demonstrated in [22]. Liu and Fan [32] proposed a tissue-like P system with the Covariance Matrix Adaptation-Evolution Strategy algorithm to solve numerical optimization problems.

2.2 P system with active membranes

P systems with active membranes can be defined as follows [33]:

$$\Pi = \left(O, \Sigma, \varepsilon, H, \mu, \left(M_1, \ldots, M_q\right), \left(R_1, \ldots, R_q\right), i_o\right)$$

where:

1. Π corresponds to the P system name;
2. O represents a finite alphabet of objects;
3. $\Sigma \subset O$ is the input;
4. $\varepsilon \subseteq O$ represents the objects in the environment;
5. μ is the type of membrane structure;
6. M_i represents a set of strings over O;
7. R_i is a finite set of rules.
8. $i_o \in \{0, 1, 2, \ldots, q\}$ indicates the output membrane where the result can be located.

2.2.1 An illustrative example of static membrane systems

In order to clarify the concepts of the membrane system and the mechanism of applying the aforementioned rules, an illustrative example is given to find the maximum of the function $f(X) = 15X - X^2$. Here, each solution will be represented by a bat position. Each bat position is encoded as $X_{i,k}^{j}$, where i is the bat number, j is the iteration number and k is the membrane number. Each bat velocity is represented as $V_{i,k}^{j}$ and each bat frequency is represented as $f_{i,k}^{j}$. In this case, the objective function is defined as $(-1)(\mathrm{f(X)}) = 0$. To solve the maximization problem, let the number of bats in each membrane is 5 and the number of elementary membranes is 4. The maximum number of iterations (T) is 10 and the max iteration in each membrane is set to 3. The P system is defined as follows:

$\Pi = (O, \Sigma, \varepsilon, M_0, M_1, M_2, M_3, M_4, M_5, R, i_o)$ where:

1. $\left(O = X_{i,k}^{j}, V_{i,k}^{j}, f_{i,k}^{j}, i \in \{1,2,3,4,5\}, j \in \{1,2,3, \ldots, 10\}, k \in \{2,3,4\}\right)$ (the set of objects);
2. $\Sigma = \{\phi\}$;
3. $\mu = \left[\,[\,]_2\,[\,]_3\,[\,]_4\,[\,]_5\right]_1$;
4. $M_1 = \{\phi\}$ (initial objects in the skin);
5. $M_2 = \left\{X_{i,2}^{j}, V_{i,2}^{j}, f_{i,2}^{j} : i \in \{1,2,3,4,5\}, j \in \{1,2,3, \ldots, 10\}\right\}$ (initial objects in membrane 2);
6. $M_3 = \left\{X_{i,3}^{j}, V_{i,3}^{j}, f_{i,3}^{j} : i \in \{1,2,3,4,5\}, j \in \{1,2,3, \ldots, 10\}\right\}$ (initial objects in membrane 3);
7. $M_4 = \left\{X_{i,4}^{j}, V_{i,4}^{j}, f_{i,4}^{j} : i \in \{1,2,3,4,5\}, j \in \{1,2,3, \ldots, 10\}\right\}$ (initial objects in membrane 4);
8. $M_5 = \{\phi\}$
9. R_i is the set of the following evolution and in/out communication rules:
    - $R1 : X_3^{pbest} \rightarrow Max(15X_{i,3}^{j} - \left(X_{i,3}^{j}\right)^2)$
    - $R2 : X^{gbest} \rightarrow Max(15X_k^{pbest} - \left(X_k^{pbest}\right)^2)$
    - $R3 : [a^n, X_3^{pbest}]_3 \rightarrow [\;\;]_3 X_3^{pbest}$ $R4 : X^{gbest}[\;\;]_k \rightarrow [X^{gbest}\;]_k$
    - $R5 : [T, X^{gbest}]_1 \rightarrow [\;\;]_1 X^{gbest}$ $R6 : [b^n, X_3^{Worst}]_3 \rightarrow [X_3^{Worst}]_5$
    - $R7 : X_{i,3}^{j} \rightarrow X_{i,3}^{j-1} + V_{i,3}^{j}, aX^{gbest}$
    - $R8 : V_{i,3}^{j} \rightarrow V_{i,3}^{j-1} + \left(X_3^{pbest} - V_{i,3}^{j}\right) f_{i,3}^{j}, b$
10. $i_0 = 1$ indicates the output membrane;

The first ten membrane configurations are shown in Fig. 1 which illustrates how the objects are evolved and communicated between membranes. As can be seen from Fig. 1, the cell-like P system consists of five membranes including the skin. The skin membrane (corresponds to the plasma membrane) is labeled with {1}, whereas the rest of membranes are labeled from $\{2-5\}$. The initial configuration of the P system is shown in Fig. 1a. Figure 2 shows the distribution of 20 bats in the search space for the first 10 iterations. In the first iteration, bats dispersed at randomized locations of the search space. It can be clearly seen in the figure that distances of the bats to the global optimum solution reduce as the iteration number increases. Note that the global optimum for this maximization problem is at bat position 7.5 and the fitness value is 56.25.

3 Binary bat algorithm

The remote sensing ability allows bats to navigate, detect and track small preys, which are several meters away even in complete darkness. Most bat species, mainly micro-bats, use echolocation for communication, recognition of various insect types, sensing and estimating the distance to the prey while moving over obstacles noiselessly. Echolocation works as a type of sonar where micro-bats send out a loud and short sound wave. Upon hitting an object, an echo of the signal will return back to bats ears after a certain time interval. Hence, bats are able to calculate the distance to the object producing the echo [34].

A transfer function has been used to map the bat algorithm into a binary search space as illustrated in (1).

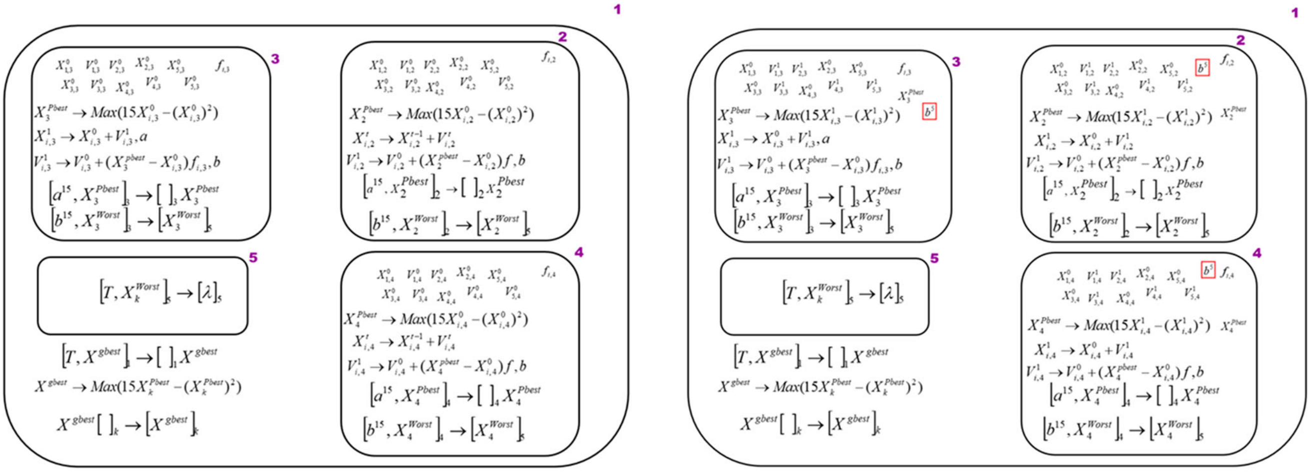
$$T(v_i^t) = erf\left(\frac{\sqrt{\pi}}{2} v_i^t\right) \tag{1}$$

the following equation is used to update the bat positions.

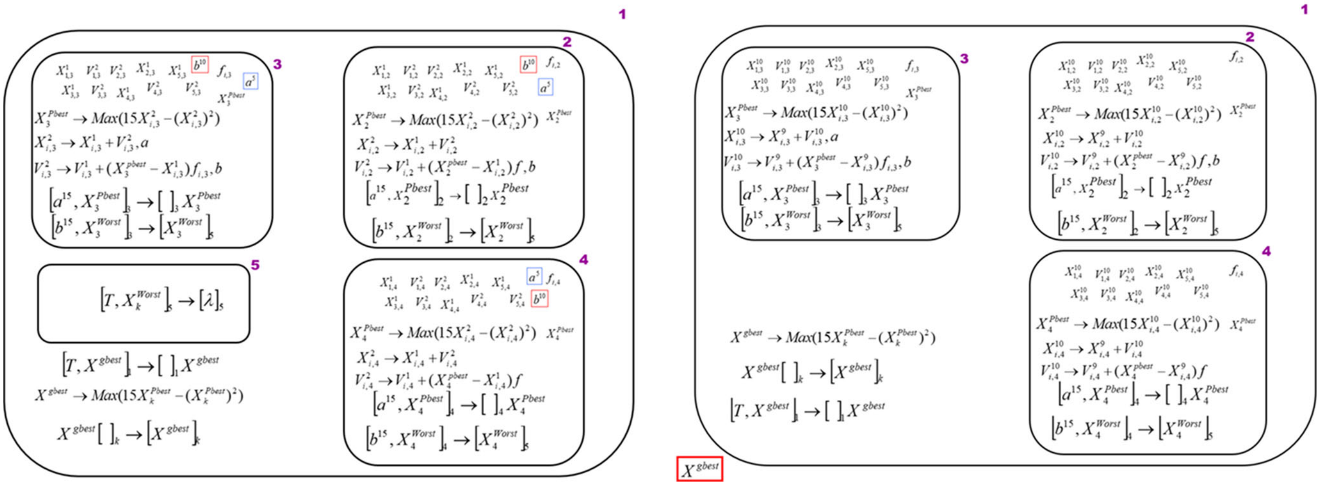
$$x_i^t = \begin{cases} 1 & \text{if } T(v_i^t) \geq rand \\ 0 & otherwise \end{cases} \tag{2}$$

4 The static membrane-inspired bat algorithm (MIBBA)

In our previous work, a static membrane-inspired bat algorithm called (MIBBA) was proposed for optimizing the facial features to recognize faces under unconstrained conditions. MIBBA starts by initializing the configuration of the system including the number of elementary membranes (which is fixed during the entire procedure) and the number of bats. In the second step of the MIBBA algorithm, the membrane structure is defined, where a specific number of elementary membranes are placed inside the skin membrane. Next, the bat population is randomly distributed. In each membrane, Binary Bat Algorithm (BBA) is running in a parallel manner to select the optimal subset of facial features. After a specific number of generations, information about the current local bats from each membrane is transferred to the skin. In the skin, a local search is performed using the Great Deluge Algorithm to find a better solution around the local best bats. Then, the local best solutions are



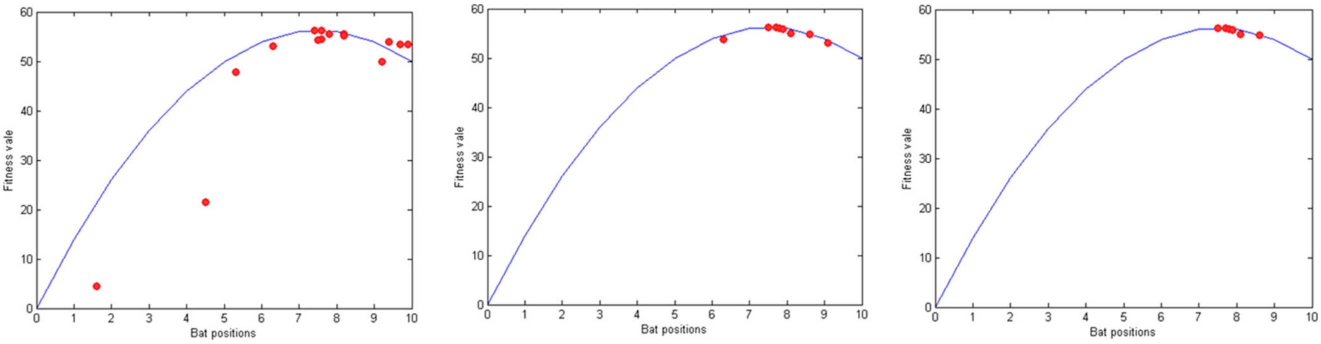
(a) Configuration 1 (b) Configuration 2



(c) Configuration 3 (d) Configuration 10

Fig. 1 The membrane configuration during the first 10 iterations

compared, and the global best bat is transferred from the skin to other membranes by applying the global communication Rule. The process is iterated until the termination condition is met. More details about the static MIBBA can be found in [30]. Note that in the static MIBBA the number of membranes did not change during the iterations.



(a) Iteration 1 (b) Iteration 5 (c) Iteration 10

Fig. 2 Distribution of bat population in the first 10 iterations

However, In this current work, DMBA has a dynamic membrane structure, namely its membrane structure, and the number of membranes is changed during computation due to the use of merging rules.

5 The proposed dynamic membrane-driven bat algorithm (DMBA)

In this section, we combine the strong theoretical foundations of active membranes with the adaptive nature of BA to propose a novel method (DMBA). Technically, the membrane-inspired framework consists of a dynamic membrane structure. Which initially has $m - 1$ elementary membranes, labeled as $2, 3, \ldots, m$ placed inside the skin membrane, denoted by 1. There are four types of rules: (i) Evolution rules in each of the elementary membranes, from 2 to $m - 1$, which are used to update an individual according to the evolutionary mechanism of the bat algorithm. (ii) Communication rules for sending the fittest individual from each of the elementary membranes to the skin membrane and then the overall global best individual from the skin back to each corresponding membrane. (iii) Separation rules and finally (iv) Merging rules to merge elementary membranes into a single membrane [35]. The flowchart of DMBA algorithm is given in Fig. 3. The steps of DMBA method can be described as follows:

1. **Initialization Step:** In this step, the membrane algorithm configurations need to be initialized. This includes the structure of the membrane system, the number of primary membranes, symbol-objects number, the communication frequency, and the maximum number of iterations.
2. **Membrane structure construction Step:** Initially, $m-1$ elementary membranes are placed in the skin which can be described as $\left[\,[\,]_1\,[\,]_2\,[\,]_3\,,\ldots[\,]_m\right]_1$.

   The proposed membrane algorithm can be represented by the following construct:

   $$\Pi = \left(O, \mu, W_1, \ldots, W_q, R, o_{out}\right), where,$$

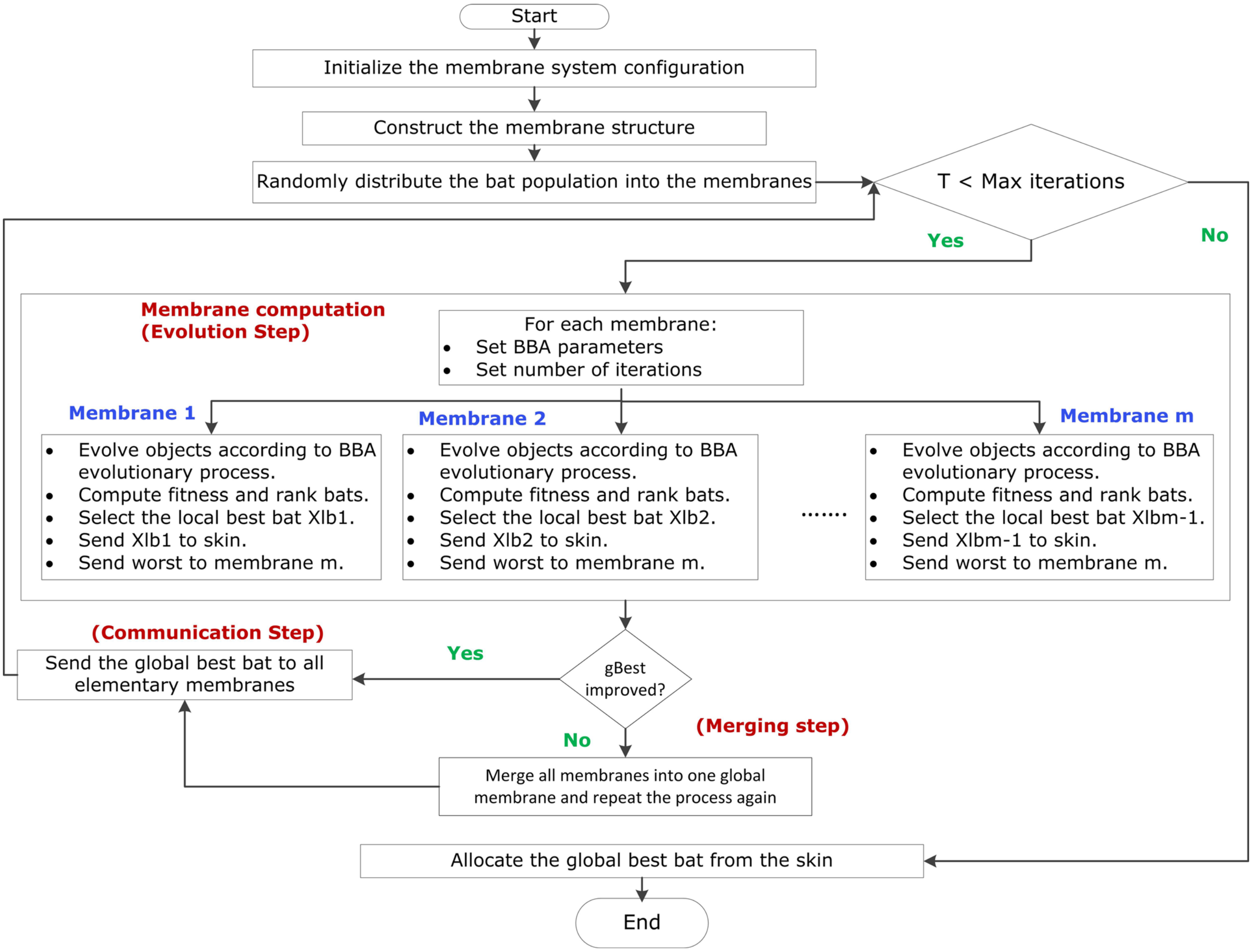


Fig. 3 The flowchart of the proposed DMBA algorithm

- $O = \{X_1, X_2, X_3, \ldots, X_N\}$ corresponds to all the objects involved in the system such as the bat solutions, where N is the total number of bats.
- $\mu = \big[\, [\;]_2 [\;]_3 [\;]_4, \ldots [\;]_m \big]_1$ is the one level membrane structure;
- $W_1, \ldots, W_m$ corresponds to the initial multisets as represented in the following:

  $W_1 = \{\lambda\}$ is the initial multiset of the skin;
  $W_2 = \{X_1, X_2, X_3, ..., X_n, a, b\}$ is the multiset of objects in membrane 1;
  $W_3 = \{X_{n+1}, X_2, X_3, ..., X_r, a, b\}$;
  $\vdots$
  $W_m = \{X_{k+1}, X_2, X_3, ..., X_N, a, b\}$;
  R is the set of required rules , which are:

  - Evolution rules: In this rule, the bat algorithm is executed in each of the membranes, from 1 to m.
  - Send in and Send out Communication rules: which are responsible of sending the local best solution X_{lbm} from each of the m membranes into the skin cell and then the global best solution X_{gb} from the skin cell to all m membranes.

    The formulation of the send-in and send-out communication rules is as follows:

    $$[a^n, X_{lbm}]_i \rightarrow [a^0]_i X_{lbm}, i \in 2, 3, \ldots m \tag{3a}$$

    $$[a^j \rightarrow a^{j+1}]_i, i \in 2, 3, \ldots m, j \in 0, 1, \ldots n \tag{3b}$$

    In (3a), the local best solution is sent from each of the m membranes to the skin. Here, variable a is utilized to execute the local communication rule. After applying rule (3a), variable a will be initialized again.

    $$X_{gb}[\;]_i \rightarrow [X_{gb}]_i, i \in 2, 3, \ldots m \tag{4}$$

    This rule is invoked to send the global best solution from the skin to all m membranes.
  - Merging rule: If the global best solution has not enhanced over a prescribed number of consecutive iterations (set to 10 throughout the paper), the merging rule (Rule 5a) will be applied.

    $$c^k, [\;]_2[\;]_3[\;]_4, \ldots [\;]_m \rightarrow [\;]_2 \tag{5a}$$

    $$[c^j \rightarrow c^{j+1}]_m, j \in 0, 1, \ldots k \tag{5b}$$

    Rule (5a) is invoked to merge all elementary membranes into a single membrane, where all of their objects will be placed inside the merged membrane. Rule (5b) is used to update counter c in each iteration.
- o_{out} is the output membrane where. This membrane has a set of terminal symbols (X_{gb}), where X_{gb} is the global best solution.

3. **Bat distribution Step:** N individuals of the whole bat population will be randomly placed into m membranes.
4. **Evolution Step:** Firstly, set the number of iterations for the binary bat algorithm inside each of the membranes. In this step BBA algorithm will be executed in parallel inside each of the membranes. The pseudo-code of the classical BA is shown in Algorithm 1. The first step in BA is the initialization phase where the fitness function, bat population (the initial bat position x_i and velocity v_i), its related parameters (frequency f_i, loudness A_i and pulse rate r_i) need to be defined and initialized (lines 1–7). Afterwards, for every time step t (iteration), having M the maximum number of iterations, the bat's movement is given by updating other parameters such as velocity, frequency and position using (6), (7), and (8), respectively, as follows:

Algorithm 1 Algorithm of the classical Bat Algorithm (BA) [5].

```
1  begin
2      $N \longleftarrow$ the number of individuals (bats) in the population ;
3      $M \longleftarrow$ maximum number of iterations.
4      $f_{min} \longleftarrow$ minimum frequency, $f_{max} \longleftarrow$ maximum frequency;
5      $A_i, r_i \longleftarrow$ loudness and pulse rate parameters;
6      $t \longleftarrow$ iteration;
7      Initlaize the bat population $x_i, v_i$. t=0
9      while $t \leq M$ do
10         for each bat $b_i \in population$ do
11             Generate new solutions using (8);
12         end
13         if rand (0,1) $> r_i$ then
14             Select a solution among the best solutions.
15             Generate a local solution using (9).
16         end
17         if rand (0,1) $< A_i$ and $f(x_i) < f(x^{best})$ then
18             Accept the new solutions
19             Increase $r_i$ using (10)
20             Decrease $A_i$ using (11)
21         end
22         Rank the bats and find the current best $x^{best}$.
23         $t \longleftarrow$ t + 1
24     end
25     RETURN $x^{best}$;
26 end
```

$$f_i = f_{min} + (f_{min} - f_{max}) \times \beta, \tag{6}$$

$$v_i^j(t) = v_i^j(t-1) + [\hat{x^j} - x^j(t-1)]f_i, \tag{7}$$

$$x_i^j(t) = x_i^j(t-1) + v_i^j(t), \tag{8}$$

where β is a random number of a uniform distribution in the interval [0,1] and $\hat{x^j}$ is the current best solution achieved so far. Note that, the frequency f_i is used to control the range and pace of bats movement in the search space by adjusting the velocity change. Yang [5] has adapted the random walk as in (9).

$$x_{new} = x_{best} + \epsilon \hat{A}(t) \tag{9}$$

where ϵ is a random number and $\hat{A}(t)$ is the average loudness at the time t. For each iteration, if the bats are approaching their preys, the loudness A_i and the emission pulse rate r_i must be updated using (10) and (11), respectively, as follows:

$$A_i(t+1) = \alpha A_i(t) \tag{10}$$

$$r_i(t+1) = r_i(0)[1 - exp(-\gamma t)] \tag{11}$$

where α and γ are constants.

5. **Communication Step:** As soon as the m membranes finish their required number of iterations, the local best solution is sent to the skin by executing Equation (3a) and Equation (3b). Next, the second communication rule is applied, as in rule (10), for sending the global best solution from the skin to all the m membranes. The BBA algorithm is performed in parallel inside all of the four elementary membranes. After a predetermined number of iterations, each elementary membrane sends its local best bat solution to the skin membrane.
6. **Merging step:** If the global best solution has not changed over a prescribed number of consecutive iterations (set to 10 throughout the paper), the merging rule (Rule 5a) will be applied.
7. **Termination step:** The algorithm will continue to execute until the termination criteria is met, which is set to be reaching the maximum number of iterations.
8. **Output stage:** The result will be the final best solution that is collected from the skin.

6 Experiments and results

To validate the efficiency of the proposed DMBA in terms of convergence behavior and solution quality, four numerical experiments have been conducted. Firstly, a set of 20 unconstrained benchmark functions with several dimensions have been employed including unimodal and multimodal functions [36]. The list of 11 unimodal test functions has been considered as described in Table 1. The list of multimodal benchmark functions is presented in Table 2. The values of f (x^*) are shown in Tables 1 and 2, while f (x_{best}) is the average of the best solutions which are experimentally obtained. Secondly, CEC 2005 benchmark which contains 25 test functions have been used [37]. Thirdly, cec 2011 real-world problems with selected 5 problems have been used. Lastly, IEEE CEC 2017 benchmark problems which contain 30 test functions have been used [38].

6.1 Optimization of unconstrained test functions

To evaluate the efficiency of DMBA on unconstrained problems, it has been statistically compared with the results of classical BA [5], Enhanced BA (EBA) [17] and the static MIBBA [30]. The size of the bat population is set

Table 1 Unimodal benchmark functions and their characteristics

Function	Mathematical formulation	Range	f_{min}
Sphere	$F_1(x) = \sum_{i=1}^{n} x_i^2$	[−100,100]	0
Schwefel 2.22	$F_2(x) = \sum_{i=0}^{n} \lvert x_i \rvert + \prod_{i=0}^{n} \lvert x_i \rvert$	[−10,10]	0
Schwefel 1.2	$F_3(x) = \sum_{i=1}^{d} (\sum_{j=1}^{i} x_j)^2$	[−100,100]	0
Rosenbrock	$F_4(x) = \sum_{i=1}^{n-1} (100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2)$	[−30,30]	0
Step	$F_5(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$	[−100,100]	0
Quartic Noise	$F_6(x) = \sum_{i=0}^{n} i x_i^4 + \text{random}[0, 1)$	[−1.28,1.28]	0
Zakharov	$F_7(x) = \sum\limits_{j=1}^{n} x_j^2 + (\sum\limits_{j=1}^{n} 0.5 j x_j)^2 + (\sum\limits_{j=1}^{n} 0.5 j x_j)^4$	[−5,10]	0
Dixon-price	$F_8(x) = (x_i - 1)^2 + \sum\limits_{i=1}^{n} i(2x_i^2 - x_{i-1})^2$	[−10,10]	0
Stepint	$F_9(x) = 25 + \sum_{i=1}^{5} \lfloor x_i \rfloor$	[−5.12,5.12]	0
SumSquares	$F_{10}(x) = \sum_{i=1}^{n} i x_i^2$	[−10,10]	0
Easom	$F_{11}(x) = -\cos(x_1) cos(x_2) e^{-(x_1-\pi)^2-(x_2-\pi)^2}$	[−100,100]	−1

Table 2 Multimodal benchmark functions

Function	Mathematical formulation	Range	f_{min}
Schwefel 2.26	$F_{12}(x) = \sum_{i=1}^{n}(-x_i sin(\sqrt{\lvert x_i \rvert}))$	[−500,500]	$-418.98 \times n$
Rastrigin	$F_{13}(x) = \sum_{i=1}^{n}[x_i^2 - 10cos(2\pi x_i) + 10]$	[−5.12,5.12]	0
Ackley	$F_{14}(x) = -20exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2})$ $-exp(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)) + 20 + e$	[−32,32]	0
Griewank	$F_{15}(x) = 1 + \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}})$	[−600,600]	0
Penalized 1	$F_{16}(x) = \frac{\pi}{n}\{10sin(\pi y_1)\} + \sum_{i=1}^{n-1}(y_i - 1)^2$ $[1 + 10sin^2(\pi y_{i+1}) + \sum_{i=1}^{n} u(x_i, 10, 100, 4)],$ where $y_i = 1 + \dfrac{x_i + 1}{4}, u(x_i, a, k, m) \begin{cases} K(x_i - a)^m & \text{if } x_i > a \\ 0 & -a \leq x_i \geq a \\ K(-x_i - a)^m & -a \leq x_i \end{cases}$	[−50,50]	0
Penalized 2	$F_{17}(x) = 0.1(sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1+$ $sin^2(3\pi x_i + 1)] + (x_n - 1)^2 1 + sin^2(2\pi x_n)) + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	[−50,50]	0
Langerman2	$F_{18}(x) = \sum_{i=1}^{m} c_i (e^{-\frac{1}{\pi}\sum_{j=1}^{n}(x_j - a_{ij})^2 cos(\pi \sum_{j=1}^{n}(x_j - a_{ij})^2)})$	[0,10]	−1.08
Langerman5	$F_{19}(x) = \sum_{i=1}^{m} c_i (e^{-\frac{1}{\pi}\sum_{j=1}^{n}(x_j - a_{ij})^2 cos(\pi \sum_{j=1}^{n}(x_j - a_{ij})^2)})$	[0,10]	−1.5
Langerman10	$F_{20}(x) = \sum_{i=1}^{m} c_i (e^{-\frac{1}{\pi}\sum_{j=1}^{n}(x_j - a_{ij})^2 cos(\pi \sum_{j=1}^{n}(x_j - a_{ij})^2)})$	[0,10]	NA

Table 3 The comparative results on unimodal test functions in terms of mean and standard deviation

F.	D	BA		EBA		MIBBA		DMBA	
		Mean	Std. dev.	Mean	Std. dev.	Mean	Std. dev.	Mean	Std. dev
1	10	4.3E+02	2.8E+02	1.1E-31	3.9E-31	1.2E-22	1.4E-21	7.8E-27	2.8E-26
	30	4.6E+03	2.04E+03	2.06E-05	6.25E-06	3.2E-6	1.2E-5	5.7E-08	1.8E-07
	50	1.06E+04	3.24E+03	2.08E-05	4.3E-06	1.03E-5	1.3E-5	2.15E-06	1.1E-06
2	10	3.01E-01	7.48E-02	1.04E-02	3.23E-02	1.4E-7	1.22E-7	5.27E-19	1.2E-18
	30	4.09E+00	3.42E+00	3.1E-01	3.9E-01	2.1E-01	1.1E-01	2.92E-05	5.16E-05
	50	2.07E+01	7.14E+00	1.34E+00	7.17E-01	1.4E0	1.4E0	0.00103	0.00123
3	10	1.05E+03	5.01E+02	1.84E-13	1.01E-12	2.4E-15	1.2E-16	9.94E-26	5.37E-25
	30	1.26E+04	5.6E+03	1.18E-04	3.96E-05	8.4E-08	0.0264	8.4E-08	0.0264
	50	3.95E+04	2.37E+04	1.07E-02	6.06E-03	1.07E-2	6.6E-3	0.00011	6.34E-05
4	10	7.46E+02	1.93E+03	1.32E-01	7.27E-01	1.2E-2	1.1E-1	0.0344	0.1109
	30	1.98E+05	4.29E+05	2.1E+01	5.96E+00	1.1E01	2.1E00	0.3475	0.5042
	50	1.04E+06	6.12E+05	5.45E+01	2.16E+01	1.4E01	1.1E01	21.535	0.564
5	10	5.2E+02	3.48E+02	0	0	5.95E-13	4.7E-13	5.95E-13	4.8E-12
	30	5.4E+03	1.3E+03	9.66E-01	1.06E+00	3.4E-2	1.1E-2	4.7E-05	9.6E-06
	50	1.08E+04	2.61E+03	7.1E+00	3.60E+00	1.4E00	1.1E00	0.3475	0.50427
6	10	9.1E-02	3.51E-02	1.6E-03	1.06E-03	1.4E-2	1.2E-2	0.00012	0.00013
	30	1.48E+00	5.79E-01	3.77E-02	1.6E-02	1.2E-2	1.01E-2	0.0025	0.00276
	50	4.85E+00	2.9E+00	4.8E-02	2.6E-02	1.2E-2	1.2E-2	0.01519	0.01536
7	10	3.46E-02	1.59E-02	3.49E-11	1.91E-10	3.4E-2	3.4E-2	3.79E-30	1.56E-29
	30	1.71E+02	1.0E+02	4.4E-05	1.0E-05	4.4E-05	1.0E-05	5.74E-18	2.26E-17
	50	8.6E+02	2.1E+02	2.06E-04	3.01E-04	1.06E-4	2.1E-5	1.06	2.0783
8	10	7.56E-01	3.5E-02	6.66E-01	1.79E-09	6.5E-1	2.2E-1	0.66671	0.00027
	30	1.12E+01	1.4E+01	6.67E-01	6.48E-04	2.3E-1	1.1E-1	0.670765	0.01025
	50	4.92E+02	6.9E+02	6.9E-01	1.56E-01	6.3E-1	1.5E-1	0.66874	0.0027
9	5	0	0	0	0	0	0	0	0
10	10	9.78E-02	3.61E-02	6.4E-29	3.4E-28	3.4E-2	3.4E-2	2.7E-05	7.0E-04
	30	1.1E+01	1.3E+01	5.4E-04	1.7E-04	3.4E-3	1.2E-3	1.02E-05	2.4E-06
	50	1.5E+02	9.0E+01	1.1E-03	6.4E-04	1.1E-2	1.1E-1	1.3E-04	1.4E-04
11	2	-9.6E-01	1.83E-01	−1.0E-00	0.0E+00	−1.0E-00	0.0E+00	−1.0E-00	0.0E+0

to 50, the number of runs is set to 30 and the maximum number of generations is 2000. Experimentally the pulse rate (r), the factors updating r and A (α and γ) are set to 0.9. For each optimization process, typical statistical results (mean, standard deviation, t-value ,and p-value) have been documented as presented in Tables 3 and 4 for unimodal functions and Tables 5 and 6 for multimodal functions. Note that for all the benchmark functions described in Tables 1 and 2, the algorithm with lower values of 'Mean', and 'std.' is considered the best. The value of t statistics can be positive or negative. Positive t value ($t > 1.96$, $p < 0.05$) indicates that DMBA is significantly outperformed BA or EBA. However, negative t value ($t < -1.96$, $p < 0.05$) means that BA or (EBA) is significantly better than DMBA. The significance (Sig.) column in Tables 4 and 6 indicates which algorithm is significantly better.

As can be seen in Tables 4 and 6, there are 46 optimization scenarios with different dimensions for 20 test problems. t and p-values indicate that DMBA has exhibited better performance than the classical BA on 41 out of 46 optimization processes, while both approaches have similar behavior on five processes (e.g. Penalized 1 (F_{16}), Stepint (F_9), Easom (F_{11})). In the case of EBA, experimental results in Tables 3, 4, 5 and 6 indicate that DMBA has significantly outperformed EBA and MIBBA in (28 out of 46) situations. In 17 optimization processes, there was no significant difference between the performance of EBA and DMBA where both approaches exhibited similar performance. It can be seen from the results that when the dimensionality of the problem is low ($D = 10$), both EBA and DMBA have similar performance. However, for the step function with 10 dimensions, EBA has significantly outperformed DMBA. On the other hand, with increasing the problem dimension, DMBA has outperformed BA and EBA in the majority of cases ($D = 30$, $D = 50$). This confirms that the strength of DMBA is in its superiority over compared approaches when the dimensionality of the testing function increases.

Table 4 The comparative results on unimodal test functions in terms of statistical t-value, p-value and significance

F.	D	BA vs DMBA			EBA vs DMBA		
		t	P	Sig.	t	P	Sig.
1	10	8.4	0.000	DMBA	−1.5	0.13	N.S.
	30	12.48	0.000	DMBA	18.5	0.000	DMBA
	50	17.9	0.000	DMBA	31.2	0.000	DMBA
2	10	22.04	0.000	DMBA	1.76	0.088	N.S.
	30	6.5	0.000	DMBA	4.2	0.000	DMBA
	50	11.79	0.000	DMBA	11.25	0.000	DMBA
3	10	11.48	0.000	DMBA	1.00	0.327	N.S.
	30	12.19	0.000	DMBA	0.98	0.335	N.S.
	50	9.13	0.000	DMBA	9.67	0.000	DMBA
4	10	2.1	0.043	DMBA	0.87	0.393	N.S.
	30	2.52	0.017	DMBA	20.83	0.000	DMBA
	50	9.3	0.000	DMBA	8.58	0.000	DMBA
5	10	8.2	0.000	DMBA	−6.78	0.000	EBA
	30	22.24	0.000	DMBA	4.9	0.000	DMBA
	50	22.67	0.000	DMBA	12	0.000	DMBA
6	10	14.26	0.000	DMBA	9.25	0.000	DMBA
	30	14.04	0.000	DMBA	13.62	0.000	DMBA
	50	9.14	0.000	DMBA	16.1	0.000	DMBA
7	10	11.91	0.000	DMBA	1.0	0.325	N.S.
	30	9.36	0.000	DMBA	24.1	0.000	DMBA
	50	22.24	0.000	DMBA	2.81	0.009	DMBA
8	10	14.08	0.000	DMBA	14.54	0.000	DMBA
	30	4.007	0.000	DMBA	2.15	0.04	DMBA
	50	3.8	0.001	DMBA	1.08	0.288	N.S.
10	10	15.1	2.6E-15	DMBA	0.2	0.83	N.S.
	30	8.4	7.6E-05	DMBA	16.7	1.9E-16	DMBA
	50	9.3	3.2E-10	DMBA	8.2	1.2E-05	DMBA

Table 5 The comparative results on multimodal test functions in terms of mean and standard deviation. (Best results are highlighted in bold)

F.	D	BA		EBA		MIBBA		DMBA	
		Mean	Std. dev.	Mean	Std. dev.	Mean	Std. dev	Mean	Std. dev
12	10	−2.03E+03	−1.5E+03	−2.5E+03	−2.5E+03	−2.03E+03	1.5E+03	**−2.9E+03**	**1.7E+02**
	30	−3.5E+03	4.8E+03	−5.9E+03	9.6E+02	−5.9E+03	1.3E+03	**−7.9E+03**	**1.3E+03**
	50	−4.84E+03	8.8E+02	−1.1E+04	1.6E+03	−1.1E+04	1.5E+03	**−1.5E+04**	**3.6E+02**
13	10	3.1E+01	7.6E+00	1.0E+01	4.1E+00	3.1E+01	7.6E+00	**0.0E+00**	**0.0E+00**
	30	1.4E+02	2.7E+01	3.4E+01	1.3E+01	1.1E01	1.1E01	**4.5E-01**	**1.2E+00**
	50	2.3E+02	8.4E+01	5.2E+01	1.4E+01	**1.6E+00**	**3.1E+00**	1.6E+00	**3.1E+00**
14	10	1.6E+00	2.3E+00	4.2E-09	2.3E-08	1.1E-7	2.3E-9	**2.7E-15**	**1.8E-15**
	30	9.2E+00	2.1E+00	4.6E-01	5.8E-01	5.6E-1	5.2E-1	**3.7E-01**	**4.5E-01**
	50	1.1E+01	1.3E+00	1.8E+00	4.1E-01	1.9E00	2.2E-1	**1.7E+01**	**1.8E-1**
15	10	1.2E+01	5.4E+00	9.0E-01	6.6E-01	1.1E01	1.2E-1	**1.2E-03**	**6.7E-03**
	30	6.5E+01	1.8E+01	7.3E-03	9.5E-03	5.3E-3	1.2E-2	**3.3E-06**	**1.6E-05**
	50	1.2E+02	2.7E+01	**4.7E-03**	**1.1E-01**	1.3E02	1.1E01	3.1E-02	1.1E-01
16	30	2.1E+03	1.1E+04	1.0E+00	1.8E+00	2.1E03	1.1E02	**2.6E-01**	**1.1E-01**
17	30	1.7E+02	6.41E+01	7.74E-03	9.64E-03	8.3E-3	1.1E-3	**6.8E-03**	**3.49E-03**
18	2	−1.08E+0	2.29E-04	−1.08E+0	6.51E-16	−1.4E0	6.51E-16	**−1.7E+0**	**7.30E-17**
19	5	−9.53E-01	3.8E-01	−1.1E+0	2.8E+00	−1.01E0	1.8E0	**−1.49E+0**	**0.0E+0**
20	10	−9.4E-01	1.8E-01	−6.4E-01	3.0E+0	−4.6E-1	3.0E+0	**−8.6E-01**	**2E+00**

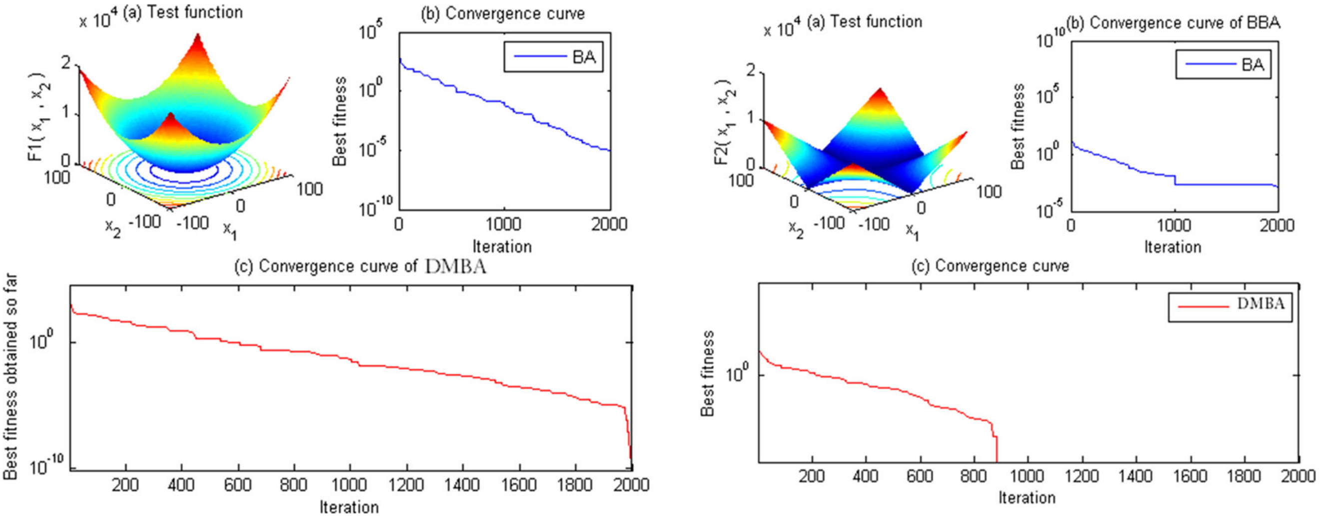
6.1.1 Convergence behaviour

The convergence behavior of the standard bat algorithm and the membrane-inspired bat algorithm is shown in Fig. 4 for unimodal test functions ($F_1 - F_4$). In all convergence curves, the X-axis shows the generation (iteration) number, and the Y-axis shows the value of the best fitness obtained so far (Fig. 5).

From these convergence curves, it can be seen that the membrane-inspired bat algorithm has a better convergence rate than the classical bat algorithm. This good convergence trend can be attributed to the fact that the incorporation

Table 6 The comparative results on multimodal test functions in terms of t-value, p-value and significance (N.S. is Not Significant)

F.	D	BA vs DMBA			EBA vs DMBA		
		t	P	Sig.	t	P	Sig.
12	10	50	0.0000	DMBA	8.1	0.0000	DMBA
	30	3.6	0.0010	DMBA	16.9	0.0000	DMBA
	50	67	0.0000	DMBA	0.03	0.9971	N.S.
13	10	22.11	0.0000	DMBA	13.3	0.0000	DMBA
	30	30.3	0.0000	DMBA	16.2	0.0000	DMBA
	50	15	0.0000	DMBA	25	0.0000	DMBA
14	10	3.8	0.0006	DMBA	1	0.3244	N.S.
	30	29.7	0.0000	DMBA	3.7	0.0009	DMBA
	50	5.2	0.0000	DMBA	11.6	0.0000	DMBA
15	10	11.8	0.0000	DMBA	7.6	0.0000	DMBA
	30	19.4	0.0000	DMBA	4.2	0.0002	DMBA
	50	23.2	0.0000	DMBA	−1.3	0.1839	N.S.
16	30	1.04	0.3082	N.S.	2.3	0.0263	DMBA
17	30	14.6	0.0000	DMBA	0.8	0.4162	N.S.
18	2	0	1	N.S.	0	1	N.S.
19	5	7.5	0.000	DMBA	6E-01	5.4E-01	N.S.
20	10	1.4	0.16	N.S.	−0.9	1.2E-01	N.S.



(a) (Sphere (F_1))

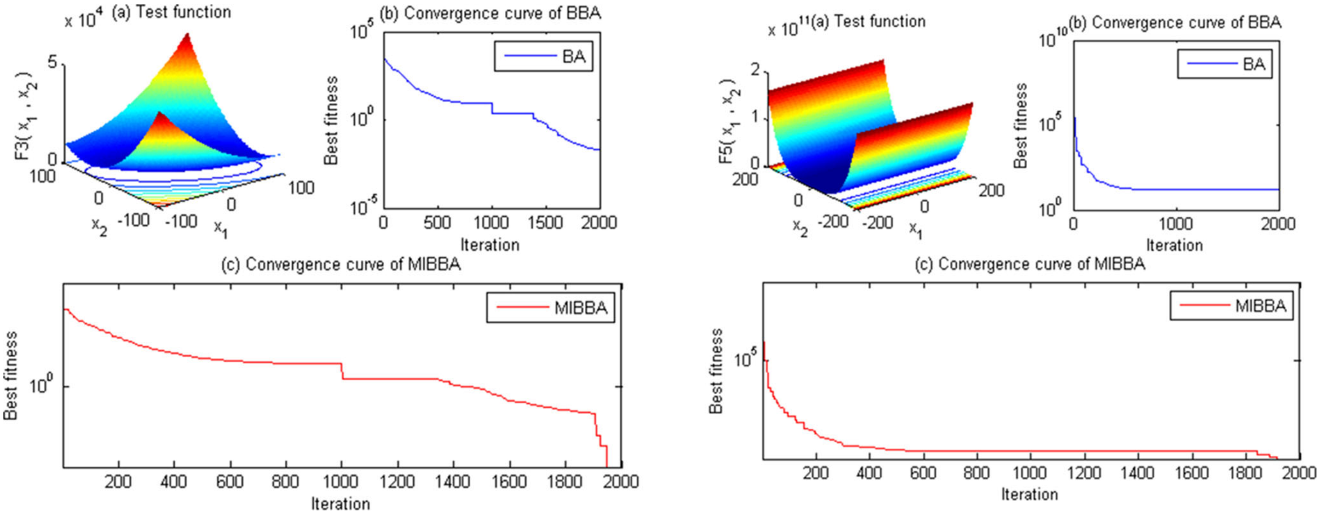
(b) (Schwefel's 2.22 (F_2))

Fig. 4 Convergence of BA and DMBA for the test function F_1 (**a**), F_2 (**b**)

of the membrane computing structure and rules helps in maintaining the diversity of the population during the generation and evaluation process. As a consequence, the fitness value of the best bat solution kept changing and thus prevented the algorithm from being stuck in local optima.

Figure 6 shows the distribution of 100 bats in the search space using DMBA for 2D step function with range [−100, 100]. In the first iteration, bats dispersed at randomized locations of the search space. It can be clearly seen in the figure that distances of the bats to the global optimum solution (represented in green) reduces as the iteration increased. Figure 6d shows that all bats have converged to the optimum values of ($x_1 = x_2 = 0$) at iteration 100.

Furthermore, to show the ability of DMBA in avoiding local optima, the multimodal Schwefel function with range [−500, 500] has been used as shown in Fig. 7. At the first iteration, the current best bat solution was located at a local optimal point. As iterations proceed, bats started to converge to the global optimum solution.



(a) (Schwefel 1.2 (F_3))

(b) (Rosenbrock (F_4))

Fig. 5 Convergence of BA and DMBA for the test function F_3 (**a**), F_4 (**b**)

6.1.2 Comparisons with enhanced versions of BA

The results of the proposed DMBA algorithm have been also compared with improved variants of the conventional bat algorithm including Hybrid Bat Algorithm (HBA) [39], Hybrid BA with Random Forest (HBARF) [40], Modified BA (MoBA) [41], Chaotic Bat Swarm Optimization (CBSO) [12], Binary Bat Algorithm (BBA) [42], Simulated Annealing Gaussian Bat Algorithm (SAGBA) [43], directional Bat Algorithm (dBA) [18], static membrane-inspired bat algorithm (MIBBA) [30], and Enhanced Bat Algorithm (EBA) [17].

In the first experiment, five benchmark functions with 10 dimensions each have been used. The results are summarized in Table 7. The best results are highlighted in bold font. From the results presented in Table 7, it can be seen that DMBA has outperformed state-of-the-art BA variants on Rastrigin, Rosenbrock, and Ackley functions. DMBA and EBA have exhibited similar performance on the Sphere function, whereas HBA has obtained better results on Griewank function.

In the second experiment, six test functions are commonly used to validate the performance of the bat algorithm variants. Those functions are the sphere, Rastrigin, Griewank, Ackley, Rosenbrock, and Zakharov. Different optimization scenarios have been involved using multiple ranges, dimensions (D), population size (N), the maximum number of iterations (t_{max}) and various numbers of trials. The results of this experiments are tabulated in Table 8. Experimental results show that DMBA outperforms the other compared methods in a total of 23 optimization processes out of 32 benchmarking situations, which represents 71.875% of testing scenarios. The proposed DMBA has been outperformed by CBSO variant in two out of four optimization processes. However, a considerable number of bats populations has been used in CBSO to cope with its diversification deficiency. The most recently proposed BA variant, dBA [18], has been outperformed by DMBA in 3 out of 6 optimization processes.

6.1.3 Diversity behaviour of DMBA

To evaluate the diversity behaviour of the proposed DMBA, the pair-wise Hamming distance measurement has been used. The average collective results of 50 independent runs on test function (F_1) has been used to test the diversity of DMBA on 8 different membranes (Fig. 8). The results are shown in Fig. 9. The results clearly show that DMBA has the ability to sustain higher degree of population diversity for a longer period of time.

6.2 Optimization of CEC 2017 benchmark functions

In this experiment, the proposed algorithm (DMBA) has been evaluated using CEC 2017 benchmark functions. The specifications of the CEC 2017 functions are given in Table 9.

These benchmark functions are classified into four groups; the first group is the Unimodal functions (from F1

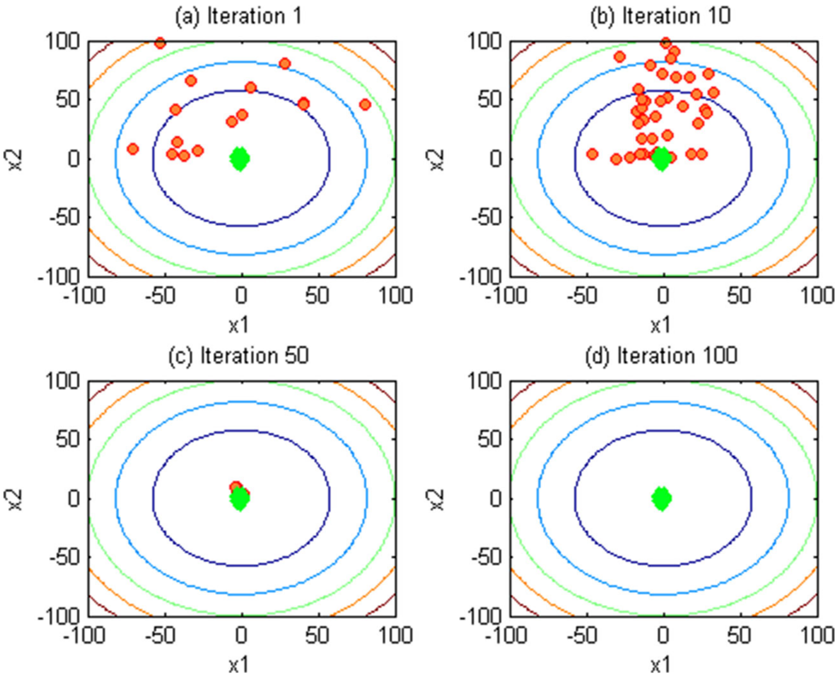
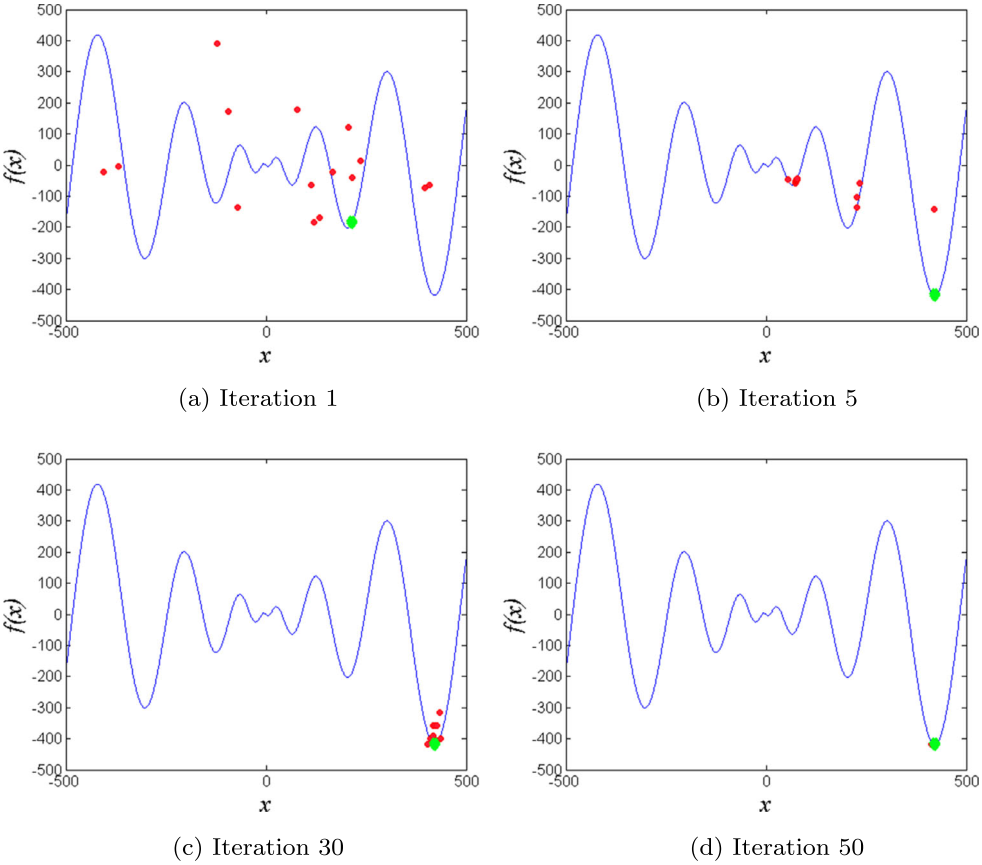


Fig. 6 Locations of 100 bats using DMBA for 2D Step function. **a** Iteration 1. **b** Iteration 10. **c** Iteration 50. **d** Iteration 100 (green indicates global best)

Fig. 7 Distribution of bat population in the first 50 iterations on Schwefel function (green indicates current best solution)



(a) Iteration 1

(b) Iteration 5

(c) Iteration 30

(d) Iteration 50

to F3 as presented in Table 9), which is utilized to assess the optimization accuracy and the convergence speed of the proposed algorithms. The second group is called the Simple Multimodal Functions (From F4 to F10 as shown in Table 9), and they are utilized to evaluate the performance of the proposed algorithm in terms of the exploitation ability. The third group is the Hybrid Functions (from F11 to F20 as shown in Table 9), which is used to assess the optimization efficiency and the convergence acceleration of the proposed algorithm. Finally, the fourth group is the Composition Functions (from F13 to F15 in Table 9).

The BA, modified BA (MBA) [21], MBADE [21] and the proposed DMBA algorithms were compared. Table 10 shows the performance of the proposed algorithms on CEC 2017 benchmark functions with dimension space set to 10. The results presented in Table 10 confirmed that the proposed algorithm (DMBA) achieved better performance in the majority of cases compared to other methods (MBA, MBADE) and the basic BA. The proposed algorithm (DMBA) achieved the optimum results in the first group (F1 and F3). Furthermore, DMBA got the best results in the second group for all cases (seven out of seven functions (F4 to F10). With regard to the third group, the proposed algorithm (DMBA) got better results in three out of three cases (all given functions from F11 to F20. Finally, DMBA obtained the optimum results in the fourth group (F21 to F30) for all situations. These test problems have been applied to evaluate exploration and exploitation ability of the proposed algorithm.

Table 11 shows the performance of the proposed algorithm and the compared methods (i.e., BA, MBA, MBADE, and DMBA) also on CEC 2017 benchmark function problems when the dimension space is fixed to 30. The results presented in Table 12 are committed to the results showed early, according to the Friedman ranking test. The results confirmed that the proposed algorithm (DMBA) got better performance in all cases compared to other proposed methods (MBA, MBADE) and the basic BA when the dimensions are equal to 10 and 30. It can be concluded from the results that the modification of BA reached its main goals in improving its performance when membrane computing framework is

Table 7 Comparative results of DMBA and recent bat algorithm variants using five unconstrained benchmark functions

Function	Method	Best	Worst	Mean	Std. dev.
Sphere	MOBA	3.73E-03	1.60E-02	8.80E-03	3.34E-03
	HBA	4.83E-09	2.89E-03	1.26E-04	1.66E-07
	HBARF	2.36E-06	5.90E-02	5.92E-03	1.22E-02
	EBA	**1.64E-35**	**1.7E-30**	**1.13E-31**	**3.91E-31**
	MIBBA	2.22E-32	1.41E-20	5.51E021	2.12E-22
	DMBA	5.14E-34	1.29E-25	7.87E-27	2.83E-26
Rosenbrock	MOBA	7.44E+00	1.64E+01	1.03E+01	1.94E+00
	HBA	6.34E-02	5.10E+02	6.22E+01	7.73E+00
	HBARF	5.00E-05	1.99E+00	2.64E-01	5.44E-01
	EBA	**4.19E-12**	3.98	0.132	0.727
	MIBBA	3.3E-03	1.4E00	1.2E-2	7.1E-1
	DMBA	4.29E-04	**4.84E-01**	**3.44E-02**	**1.11E-01**
Rastrigin	MOBA	1.46E+01	3.48E+01	2.49E+01	4.35E+00
	HBA	5.12E+00	2.38E+01	1.55E+01	1.69E+01
	HBARF	3.09E-05	1.02E+01	5.92E-01	2.00E+00
	EBA	3.97E+00	1.98E+01	1.01E+01	4.14E+00
	MIBBA	2.2E-4	1.4E-3	3.3E-2	1.2E-2
	DMBA	**0**	**0**	**0**	**0**
Griewank	MOBA	2.05	20.6	8.12	5.39
	HBA	2.25E-09	**3.97E-05**	**3.18E-06**	**1.14E-07**
	HBARF	1.44E-11	6.35E-04	3.92E-05	1.25E-04
	EBA	0.0566	2.67E+00	9.04E-01	6.57E-01
	MIBBA	**0**	4.4E-2	1.5E-3	7.7E-3
	DMBA	**0**	3.65E-02	1.22E-03	6.67E-03
Ackley	MOBA	3.61E-02	1.79E+00	1.67E-01	3.60E-01
	HBA	6.31E-04	2.00E+01	1.16E+01	1.78E+01
	HBARF	7.21E-04	3.53E-01	3.14E-02	6.87E-02
	EBA	4.44E-15	1.26E-07	4.21E-09	2.30E-08
	MIBBA	4.4E-4	5.2E-2	3.2E-3	1.1E-3
	DMBA	**8.88E-16**	**4.44E-15**	**2.66E-15**	**1.8E-15**

used with BA. The distribution results of the proposed and compared methods are shown in Fig. 10. The distribution results are presented over several functions from all groups. All sub-figures clearly show that the distributions of the obtained results by the proposed method (DMBA) are much better than the other comparative methods. All the results achieved by DMBA have a high density close to the optimal solutions, which proved that the performance of DMBA using these modifications is superior.

Table 13 shows the comparisons of the proposed algorithm (DMBA) with other similar methods on CEC 2017 test functions using 30 Dim. Eight well-known published algorithms are chosen to validate the efficiency of the proposed algorithm (DMBA). These methods are FA [44], PSO [45], GSA [46], GSO [47] , FFPSO [44], DVBA [48], HPSOFF [44], BA [49], and DMBA. The proposed DMBA obtained better results in comparison with other similar methods in the majority of cases (14 out of 30 test functions). Generally, the results confirmed the superiority of the proposed algorithm by reaching the best global optimum solution. Table 13 also shows the statistic rank test, Friedman rank test, for the comparative algorithms on CEC 2017 test functions using 30 dimension space value (Dim). Moreover, the proposed DMBA got the best Mean rank value (2.13), followed by DVBA(3.10), GSA (3.46), GSO (4.23), PSO (4.63), HPSOFF (4.86), FA (6.46), FFPSO (8.00), and BA (8.03). The proposed DMBA got

Table 8 Comparisons between DMBA and recent variants of BA using six test functions

F	Bounds	D	N	t_{max}	Trials	DMBA		Literature		BA variant
						Mean	Std. dev.	Mean	Std. dev.	
Sphere	[−5.12, 5.12]	30	10000	100	30	6.46E-07	8.67E-07	7.92E-06	8.06E-07	CBSO
	[−100, 100]	5	30	500	30	3.51E-33	1.4E-32	1.8E+00	2.5E+00	BBA
	[−100, 100]	30	40	2000	50	8.33E-19	3.04E-18	4.61E-05	N.A.	SAGBA
	[−5.12, 5.12]	10	50	2000	30	1.5E-233	0	1.31E-31	3.9E-31	EBA
	[−5.12, 5.12]	60	50	6000	30	1.83E-12	6.35E-12	1.08E+01	3.7E+00	MoBA
	[−100, 100]	30	40	2000	50	8.33E-19	3.04E-18	7.32E-26	8.4E-26	dBA
	[−100, 100]	30	40	2000	50	3.33E-13	3.04E-12	7.32E-12	2.4E-12	MIBBA
Griewank	[−600,600]	30	10000	100	30	1.13E-02	3.11E-01	1.04E-02	2.10E-03	CBSO
	[−600,600]	5	30	500	30	9.88E-02	1.72E-01	2.46E-01	8.39E-02	BBA
	[−600,600]	2	40	2000	50	0.00E+00	0.00E+00	3.64E-08	N.A.	SAGBA
	[−600,600]	10	50	2000	30	0.00E+00	0.00E+00	9.01E-01	6.57E-01	EBA
	[−600,600]	60	50	6000	30	0.00E+00	0.00E+00	3.19E+02	5.64E+01	MoBA
	[−600,600]	30	10000	100	30	1.13E-02	3.11E-01	1.97E-02	1.37E-02	dBA
	[−600,600]	30	10000	100	30	2.13E-01	3.11E+00	5.47E-01	0.37E-01	MIBBA
Rastrigin	[−5.12, 5.12]	5	30	500	30	4.78E-10	2.62E-10	1.585	1.3352	BBA
	[−5.12, 5.12]	2	40	2000	50	0.00E+00	0.00E+00	1.58E-07	N.A.	SAGBA
	[−5.12, 5.12]	10	50	2000	30	0.00E+00	0.00E+00	1.01E+01	4.14E+00	EBA
	[−5.12, 5.12]	60	50	6000	30	0.00E+00	0.00E+00	3.84E+02	1.21E +02	MoBA
	[−5.12, 5.12]	60	50	6000	30	0.00E+00	0.00E+00	3.70E+02	3.91E+01	dBA
	[−5.12, 5.12]	60	50	6000	30	0.00E+00	0.00E+00	1.70E+03	1.91E+02	MIBBA
Ackley	[−32, 32]	5	30	500	30	2.90E-15	1.79E-15	1.16E+00	7.28E-01	BBA
	[−32, 32]	2	40	2000	50	8.88E-16	1.0023E-31	6.19E-04	N.A.	SAGBA
	[−32, 32]	10	50	2000	30	3.25E-15	1.703E-15	4.21E-09	2.30E-08	EBA
	[−32, 32]	60	50	6000	30	17.6	6.987	1.45E+01	8.07E-01	MoBA
	[−32, 32]	60	50	6000	30	17.6	6.98	1.09E+01	5.28E+00	dBA
	[−32, 32]	60	50	6000	30	15.2	5.98	2.19E+07	1.23E+04	dBA
Rosenbrock	[−5, 10]	30	10000	100	30	3.11E+01	3.00E+00	0.2194	0.018	CBSO
	[−30, 30]	5	30	500	30	2.07E+00	3.69E-01	25.07	28.44	BBA
	[−30, 30]	2	40	2000	50	2.14E-04	1.57E-04	4.68E-06	N.A.	SAGBA
	[−30, 30]	10	50	2000	30	6.63E+00	4.12E-01	1.32E-01	7.27E-01	EBA
	[−2.408, 2.408]	60	50	6000	30	5.89E+01	1.52E+00	2.57E+02	6.19E+01	MoBA
	[−5, 10]	30	10000	100	30	3.11E+01	3.0E+00	5.76E+01	8.45E+01	dBA
	[−5, 10]	30	10000	100	30	1.11E+02	1.0E+01	4.76E+02	1.45E+01	MIBBA
Zakharov	[−5, 10]	30	10000	100	30	5.85E-07	6.84E-07	1.07E-05	4.52E-07	CBSO
	[−5, 10]	2	40	2000	50	1.09E+01	1.12E+01	6.12E-10	N.A	SAGBA
	[−5, 10]	10	100	100	25	8.38E-09	2.80E-08	3.49E-11	1.91E-10	EBA
	[−5, 10]	30	10000	100	30	5.85E-07	6.84E-07	3.25E+01	6.95E+ 00	dBA
	[−5, 10]	30	10000	100	30	5.85E-07	1.84E-07	2.55E+01	1.65E+ 00	MIBBA

the best final ranking value (ranked the first, followed by DVBA(ranked the second), GSA (ranked the third), GSO (ranked the fourth), PSO (ranked the fifth), HPSOFF (ranked the sixth), FA (ranked the seventh), FFPSO (ranked the eighth), and BA (ranked the ninth). It can be concluded from these results that DMBA got better results compared with other published methods in the literature using the cec 2017 benchmark functions.

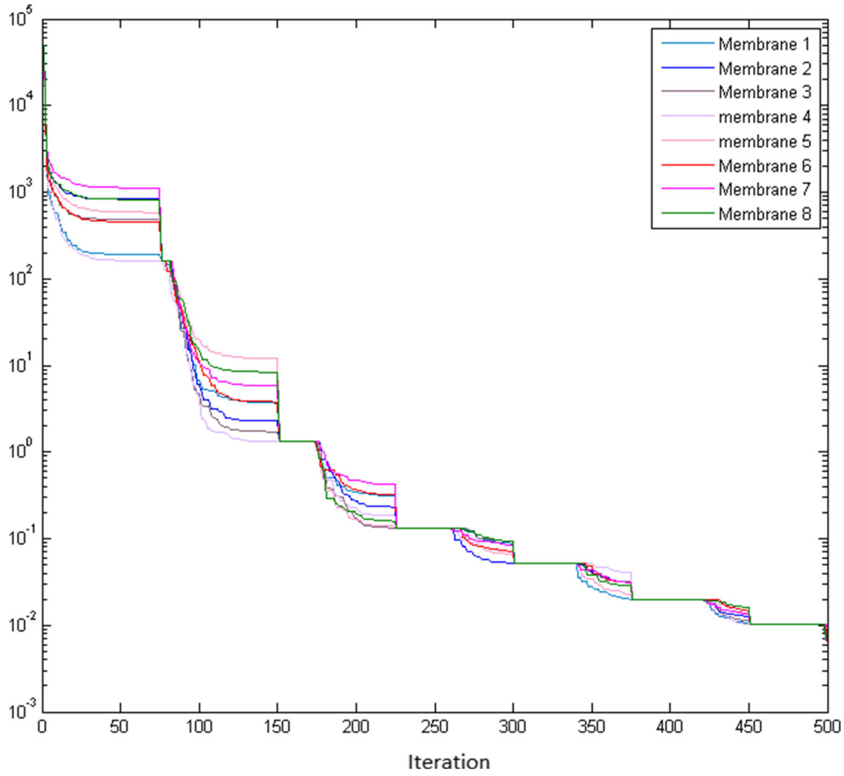


Fig. 8 Diversity behaviour of the DMBA using test function (F_1) against 500 iterations

6.3 Optimization of IEEE CEC 2011 real world problems

The performance of DMBA has been evaluated on the IEEE CEC 2011 real-world benchmark problems [51]. These problems include:

- P_1: Parameter Estimation of FM Sound Waves
- P_2: Optimal control of a non-linear stirred tank reactor
- P_3: Tersoff Potential Function Min.Problem-Si(B)
- P_4: Spread Spectrum Radar Polly phase Code Design
- P_5: Large scale pricing problem

A brief summary of the previous five problems is reported in Table 14. For a more comprehensive details about these problems, interested readers are referred to [51]. Follwoing the CEC 2011 rules, 25 repeated runs and 150000 function evaluations were used. The BA, modified BA (MBA) [21], MBADE [21] and the proposed DMBA algorithms were compared.

Table 15 shows that the proposed DMBA algorithm produced the best mean value for four of the five problems. For the P_2 problem, all four algorithms produced the same results.

Furthermore, almost no algorithm achieved satisfactory results for the P_5 problem, which indicates the difficulty to converge to the unique theoretical optimum of this test problem. Nevertheless, due to the great search capability of DMBA, DMBA is superior to all the comparative algorithms for most of these test problems.

6.4 Comparison of DMBA wth other multi-population algorithms on numerical optimization functions

In this section, the proposed DMBA has been compared with other popular multi-populations algorithms. These

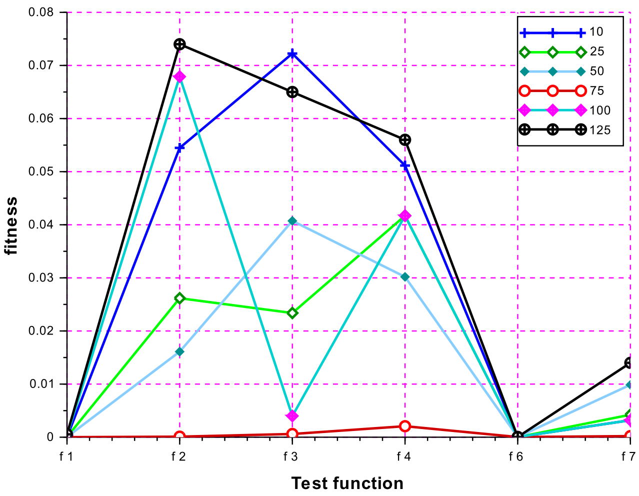


Fig. 9 Effect of communication frequency on the DMBA performance using test functions (F_1, F_2, F_3, F_4, F_6, F_7)

Table 9 Description of CEC 2017 benchmark functions

No.	Group	Description	Fi*
1	Unimodal functions	Shifted and Rotated Bent Cigar Function	100
2		Shifted and Rotated Sum of Different Power Function	200
3		Shifted and Rotated Zakharov Function	300
4	Simple Multimodal Functions	Shifted and Rotated Rosenbrock's Function	400
5		Shifted and Rotated Rastrigin's Function	500
6		Shifted and Rotated Expanded Scaffer's F6 Function	600
7		Shifted and Rotated Lunacek Bi-Rastrigin Function	700
8		Shifted and Rotated Non-Continuous Rastrigin's Function	800
9		Shifted and Rotated Levy Function	900
10		Shifted and Rotated Schwefel's Function	1000
11	Hybrid functions	Hybrid Function 1 (N = 3)	1100
12		Hybrid Function 2 (N = 3)	1200
13		Hybrid Function 3 (N = 3)	1300
14		Hybrid Function 4 (N = 4)	1400
15		Hybrid Function 5 (N = 4)	1500
16		Hybrid Function 6 (N = 4)	1600
17		Hybrid Function 6 (N = 5)	1700
18		Hybrid Function 6 (N = 5)	1800
19		Hybrid Function 6 (N = 5)	1900
20		Hybrid Function 6 (N = 6)	2000
21	Composition Functions	Composition Function 1 (N = 3)	2100
22		Composition Function 2 (N = 3)	2200
23		Composition Function 3 (N = 4)	2300
24		Composition Function 4 (N = 4)	2400
25		Composition Function 5 (N = 5)	2500
26		Composition Function 6 (N = 5)	2600
27		Composition Function 7 (N = 6)	2700
28		Composition Function 8 (N = 6)	2800
29		Composition Function 9 (N = 3)	2900
30		Composition Function 10 (N = 3)	3000

algorithms are: the island bat algorithm (iBA) [52], Directional bat algorithm (dBA) [18], Novel bat algorithm (NBA) [53], global-best bat-inspired algorithm (GBA) [54], membrane-inspired harmony search algorithm (MIBHS) [25] and membrane-inspired bat algorithm (MIBBA) [30]. In this experiment, 10 (F1–F10) numerical optimization functions published in [37]. These functions are:

- F_1: Shifted Sphere.
- F_2: Shifted Schwefel 1.2
- F_3: Shifted Rotated Elliptic
- F_4: Shifted Schwefel 1.2 with Noise
- F_5: Schwefel 2.6
- F_6: Shifted Rosenbrock
- F_7: Shifted Rotated Griewank
- F_8: Shifted Rotated Ackley
- F_9: Shifted Rastrigin
- F_{10}: Shifted Rotated Rastrigin

The results are shown in Tables 16 and 17. Best results with lowest error rate are shown in bold font. The superior performance of the proposed algorithm can be clearly seen from the results. DMBA has outperformed all the other comparative algorithms in four out of 10 test functions. It has similar performance with the other algorithms in F_1 and F_2 test functions. These two functions are simple so that all the comparative algorithms can perform well. iBA algorithm has slightly outperformed DMBA in the F_3 test function.

Table 10 The comparative results on CEC 2017 benchmark functions, Dim=10 in terms of mean and standard deviation

10 Dim Function	BA		MBA		MBADE		DMBA	
No.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
F1	1,1542E+10	6,3627E+09	2,3656E+09	5,6525E+09	1,5691E+08	2,4625E+08	2,2589E+07	2,3059E+08
F2	2,3529E+11	1,4256E+12	1,3690E+09	3,2566E+10	1,3563E+08	3,2338E+08	1,1250E+08	2,3016E+08
F3	3,5692E+04	2,9560E+04	3,5973E+04	1,5297E+04	2,2236E+04	1,6668E+04	1,1060E+04	1,9658E+03
F4	6,8654E+02	5,4167E+03	2,4123E+03	3,3624E+01	3,3698E+02	4,9633E+01	2,1525E+02	4,9865E+01
F5	5,9447E+02	2,6542E+01	2,4556E+02	2,4864E+01	4,3354E+02	1,5699E+01	2,2514E+02	1,3622E+01
F6	6,2376E+02	5,7474E+01	4,8345E+02	1,4856E+01	3,2443E+02	5,5352E+00	2,3453E+02	1,3523E+01
F7	7,4935E+02	2,2487E+01	8,1546E+02	6,3491E+01	6,8694E+02	1,5758E+01	4,2975E+02	1,6245E+01
F8	8,4397E+02	1,6496E+01	4,5964E+02	2,9654E+01	5,3681E+02	1,3579E+01	3,1548E+02	1,0214E+01
F9	4,7410E+03	1,7545E+02	1,6252E+03	3,9524E+02	1,3659E+03	2,7546E+02	1,1445E+03	2,2547E+02
F10	3,6634E+03	3,6975E+02	2,5654E+03	2,3654E+02	2,6543E+03	3,2589E+02	1,4564E+03	2,3633E+02
F11	3,2145E+04	4,2742E+03	2,5652E+04	1,6556E+05	1,3605E+03	1,2379E+02	1,1179E+03	3,4560E+01
F12	7,7778E+07	6,2686E+07	3,3654E+06	6,5658E+07	2,3557E+05	4,7352E+06	2,1258E+04	6,6547E+03
F13	4,3983E+04	6,5652E+05	2,6535E+04	5,5580E+08	1,2585E+04	6,3256E+03	1,1123E+03	3,6985E+03
F14	2,2112E+04	9,4564E+03	6,0251E+03	1,5453E+06	4,1565E+03	3,4549E+03	2,2396E+02	4,2355E+03
F15	4,2563E+04	1,4673E+04	6,3652E+03	9,5632E+07	2,2525E+03	1,2352E+04	2,3235E+02	2,7452E+04
F16	6,3567E+03	1,5854E+02	2,5757E+03	3,1814E+02	2,5642E+03	1,5245E+02	1,4658E+02	1,5676E+02
F17	4,8855E+03	5,6348E+01	2,3652E+03	2,9856E+02	1,4567E+03	4,5820E+01	1,3601E+02	3,5678E+01
F18	6,3824E+06	1,3643E+04	4,3497E+04	1,5697E+06	3,2656E+04	1,5765E+05	1,5868E+03	1,9995E+04
F19	1,4652E+06	1,8764E+04	2,3695E+04	4,5479E+07	1,8365E+04	4,7450E+03	2,1477E+03	3,8570E+04
F20	3,9638E+04	4,3621E+02	2,1458E+03	1,3654E+02	1,1421E+03	3,2452E+02	1,1236E+02	1,0587E+02
F21	2,7412E+04	5,7854E+02	2,6568E+03	3,5963E+01	2,3535E+03	3,8342E+01	1,2487E+02	2,2450E+01
F22	2,1345E+04	1,1225E+02	4,5540E+03	6,4335E+02	2,3652E+03	2,8563E+02	2,1453E+02	2,0258E+02
F23	5,6364E+04	1,4653E+01	3,9452E+03	2,4616E+02	2,1256E+03	2,4617E+01	2,1247E+02	2,1652E+01
F24	2,9681E+05	6,8753E+01	3,2648E+04	1,1265E+02	2,3179E+03	1,0147E+02	1,1420E+03	1,3478E+02
F25	9,3659E+04	6,5958E+01	4,2915E+03	5,4594E+02	2,1216E+03	3,8374E+01	1,4552E+03	2,2874E+01
F26	7,1743E+04	4,3977E+02	5,3485E+03	4,5548E+02	3,3304E+03	5,8464E+02	2,2573E+03	1,4357E+02
F27	4,1974E+03	7,6955E+01	3,8988E+03	4,3976E+02	3,1263E+03	4,3485E+01	2,0254E+02	3,4752E+01
F28	5,2682E+04	4,5186E+02	4,7589E+03	3,5468E+02	4,5858E+03	5,6378E+01	2,2549E+02	1,5424E+02
F29	4,6323E+03	7,9721E+01	3,1147E+03	2,6510E+02	2,4562E+03	1,1437E+02	1,2341E+03	5,5478E+01
F30	7,4214E+06	2,8851E+06	2,5685E+06	2,7134E+05	4,8780E+05	2,4125E+05	5,5464E+03	3,8365E+04

In the second part of the experiment, the proposed DMBA algorithm has been compared with other popular membrane-inspired evolutionary algorithms including membrane-inspired harmony search (MIHS) and membrane-inspired binary bat algorithm (MIBBA). To make a fair comparison, four membrane structure has been used. The results of this experiment are shown in Table 18. As can be seen from the results, the proposed DMBA algorithm outperformed the other algorithms in four out of 10 functions. It obtained same results as MIHS in four out of ten functions. This indicates the efficiency of the proposed algorithm for solving benchmark functions.

6.5 Computational complexity analysis

In randomized search meta-heuristics algorithms, measuring the computational time is not always a reliable criterion as it depends on the specification of the runtime environment and the coding style. Hence, it is preferred to compute

Table 11 The comparative results on CEC 2017 benchmark functions, Dim=30 in terms of mean and standard deviation

30 Dim								
Function	BA		MBA		MBADE		DMBA	
No.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
F1	3,3661E+11	3,4784E+09	1,8614E+10	1,8718E+09	6,6416E+08	2,6544E+08	1,5448E+07	8,4895E+05
F2	2,4874E+49	3,8750E+39	3,6178E+46	4,6964E+57	3,5875E+39	1,5354E+30	3,5418E+27	1,1647E+25
F3	2,8455E+09	4,6224E+08	2,4768E+05	1,8774E+09	1,6412E+05	3,9763E+04	1,4688E+04	2,4861E+04
F4	3,5378E+04	1,3586E+03	4,3575E+03	1,5754E+04	1,2753E+03	3,3546E+02	1,5499E+02	3,4546E+02
F5	5,2582E+03	2,1258E+01	1,7857E+02	6,7847E+01	3,9735E+02	3,8643E+01	7,5688E+01	2,5548E+01
F6	6,8453E+02	6,5455E+00	3,6481E+02	1,8746E+01	4,4586E+02	8,7985E+00	2,0256E+02	1,6847E+01
F7	1,7985E+03	1,6329E+02	2,9927E+03	2,0330E+02	1,1241E+03	5,3149E+01	1,0634E+03	3,8217E+01
F8	1,6548E+04	3,9765E+01	1,6873E+03	3,5455E+01	1,5428E+03	3,3185E+01	1,2425E+02	3,5354E+01
F9	1,0143E+04	1,6887E+03	3,4572E+03	3,5724E+03	5,5724E+03	2,2475E+03	2,4584E+02	2,9765E+03
F10	9,8757E+04	7,6483E+03	1,1642E+04	6,1254E+02	4,1394E+03	5,2931E+02	6,2946E+02	5,6481E+02
F11	3,4589E+04	3,3698E+03	3,9485E+03	1,3659E+04	3,2697E+03	1,2456E+03	3,3465E+02	4,6481E+02
F12	3,9450E+09	9,4541E+07	2,7222E+06	5,3696E+09	3,3120E+06	2,2237E+05	7,9259E+05	3,7870E+04
F13	4,1745E+09	6,1693E+08	2,7642E+06	1,2479E+04	2,4750E+05	1,6471E+06	2,0185E+03	1,6485E+04
F14	1,3452E+07	1,6485E+06	5,3485E+07	3,6485E+07	1,7680E+05	1,4258E+06	4,4369E+04	5,3168E+05
F15	1,7255E+08	6,1141E+07	7,1425E+04	3,3501E+04	1,2476E+04	1,1726E+06	2,3512E+03	2,3451E+04
F16	4,4645E+04	2,9646E+02	3,6364E+03	2,6620E+03	2,6633E+03	4,0749E+02	1,8761E+02	4,5412E+02
F17	3,0694E+04	1,8157E+02	2,3587E+04	4,9353E+04	1,5584E+03	2,5871E+02	1,9954E+02	2,7854E+02
F18	2,2581E+09	1,9753E+07	1,2445E+08	5,7822E+08	7,2789E+06	9,2564E+06	2,5574E+04	2,1144E+05
F19	1,6632E+09	6,2434E+07	1,2654E+08	3,5465E+09	8,4651E+06	9,5542E+06	5,9657E+03	5,7686E+04
F20	3,9641E+03	1,3552E+02	3,7505E+03	2,1617E+02	2,9243E+03	2,5124E+02	1,1485E+03	1,8240E+02
F21	2,8705E+03	2,5474E+01	2,3580E+03	6,5385E+01	2,2582E+03	5,7252E+01	1,3257E+02	2,9584E+01
F22	5,3253E+04	1,6975E+03	2,2747E+04	3,2465E+02	4,7653E+03	3,1479E+03	3,7366E+02	3,4553E+03
F23	3,3652E+04	4,9488E+02	3,3659E+03	2,8141E+02	3,4669E+03	9,0778E+02	2,0124E+03	4,6524E+02
F24	5,7656E+04	5,1865E+05	4,5439E+03	2,5680E+05	3,2579E+03	9,8243E+05	1,1582E+03	1,4577E+04
F25	4,9540E+05	4,8632E+07	1,7658E+04	3,5669E+05	3,3520E+03	1,4526E+04	1,0623E+03	7,5721E+03
F26	7,8541E+04	6,7612E+02	1,6653E+04	1,5541E+03	5,8686E+03	1,6655E+03	3,6540E+02	1,5275E+03
F27	3,6456E+04	6,7456E+01	5,5649E+03	7,3547E+02	3,4656E+03	9,7626E+03	2,6543E+03	6,6540E+03
F28	5,6356E+04	4,3252E+02	1,3425E+04	1,5514E+03	3,7438E+03	2,4540E+02	3,3855E+02	5,6645E+02
F29	5,4565E+04	3,5675E+02	5,2454E+04	1,6535E+05	4,8286E+03	4,8654E+02	1,6749E+03	2,7644E+02
F30	1,6452E+09	6,3125E+07	4,1452E+09	1,6496E+09	2,9754E+07	1,3647E+07	4,1495E+04	2,4585E+06

the computational complexity of the proposed algorithm as it plays a significant role in evaluating the efficiency of algorithms. Let N be the number of bats, D denotes the dimensionality of the features, I is the maximum number of iterations and m is the number of membranes. The computational complexity of BA can be approximated by aggregating the complexity of the consecutive phases of BA. The computational complexities of all BA phases in terms of $\mathcal{O}$ notation are as follows:

- Initialization stage: $\mathcal{O}(ND)$
- Evaluation and selecting current best solution: $\mathcal{O}(ND+N)$
- Iterative stage: $\mathcal{O}(IND)$
- Evaluation and selecting global best solution: $\mathcal{O}(ND+N)$

Hence, the total complexity of BA is $\mathcal{O}(3ND + 2N + IND)$. Without loss of generality, since $NID > ND$, the computational complexity of BA is $\mathcal{O}(IND)$. In the case of *DMBA*, considering a parallel implementation on a multi core machine, since the bats are scattered across all the elementary membranes, the complexity of the algorithm is $\mathcal{O}\left(\frac{IND}{m}\right)$. Furthermore, as the convergence of *DMBA* is faster than that of BA, *DMBA* requires less number of

Table 12 Ranks achieved by statistical Friedman rank test on CEC 2017 test functions using 10 and 30 Dim

Function No.	Dim	Compared algorithms BA	MBA	MBADE	DMBA
F1	10	4	3	2	1
F1	30	4	3	2	1
F2	10	4	3	2	1
F2	30	4	3	2	1
F3	10	3	4	2	1
F3	30	4	3	2	1
F4	10	3	4	2	1
F4	30	4	3	2	1
F5	10	4	3	2	1
F5	30	4	2	3	1
F6	10	4	3	2	1
F6	30	4	2	3	1
F7	10	4	3	2	1
F7	30	4	3	2	1
F8	10	4	2	3	1
F8	30	4	3	2	1
F9	10	4	3	2	1
F9	30	4	2	3	1
F10	10	4	2	3	1
F10	30	4	3	2	1
F11	10	4	3	2	1
F11	30	4	3	2	1
F12	10	4	3	2	1
F12	30	4	3	2	1
F13	10	4	3	2	1
F13	30	4	3	2	1
F14	10	4	3	2	1
F14	30	4	3	2	1
F15	10	4	3	2	1
F15	30	4	3	2	1
F16	10	4	3	2	1
F16	30	4	3	2	1
F17	10	4	3	2	1
F17	30	4	3	2	1
F18	10	4	3	2	1
F18	30	4	3	2	1
F19	10	4	3	2	1
F19	30	4	3	2	1
F20	10	4	3	2	1
F20	30	4	3	2	1
F21	10	4	3	2	1
F21	30	4	3	2	1
F22	10	4	3	2	1
F22	30	4	3	2	1
F23	10	4	3	2	1
F23	30	4	2	2	1
F24	10	4	3	2	1
F24	30	4	3	2	1
F25	10	4	3	2	1
F25	30	4	3	2	1
F26	10	4	3	2	1
F26	30	4	3	2	1
F27	10	4	3	2	1
F27	30	4	3	2	1
F28	10	4	3	2	1
F28	30	4	3	2	1
F29	10	4	3	2	1
F29	30	4	3	2	1
F30	10	4	3	2	1
F30	30	4	3	2	1
Mean rank	10	3.93	3.00	2.06	1.00
	30	4.00	2.83	2.16	1.00
Final ranking	10	4	3	2	1
	30	4	3	2	1

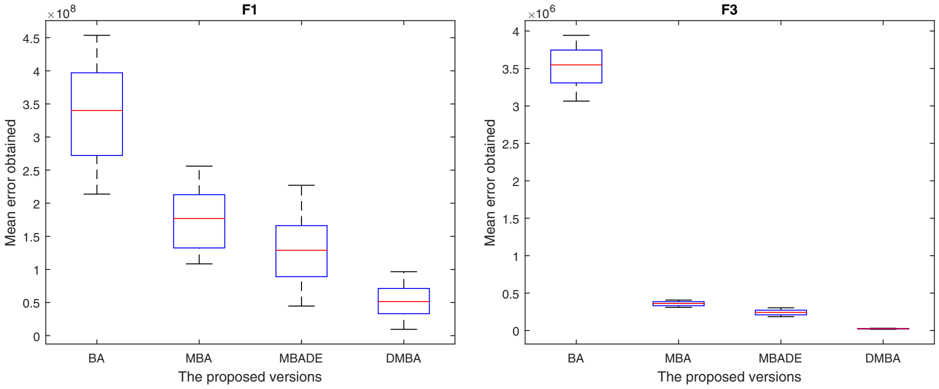
iterations to reach the optimal solution; thus, the complexity of *DMBA* can be reduced to $\mathcal{O}\left(\frac{IND}{m \log I}\right)$.

7 Conclusions and future work

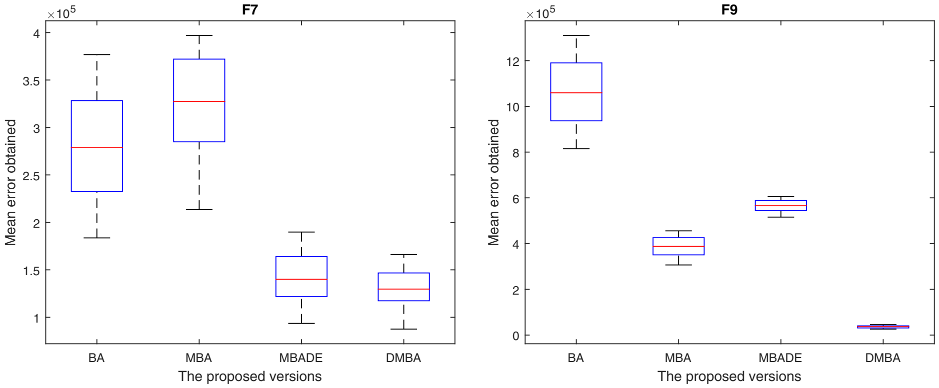
In this work, a new membrane algorithm called Dynamic Membrane-driven Bat Algorithm (DMBA) which is inspired by the structure and functioning of living cells has been introduced. DMBA is different from the static membrane algorithms in which the membrane structure is dynamically changing during the iteration process.

The dynamic construction of the proposed (DMBA) structure during the evaluation process aims at enhancing the population diversity as well as the exploration-exploitation tradeoff. A parallel framework is proposed to enhance the convergence and diversity behavior of the classical bat algorithm. The main weakness of the classical bat algorithm, its tendency to lose diversity during the optimization process so that it can fall into premature, which is solved in this paper. The performance of DMBA

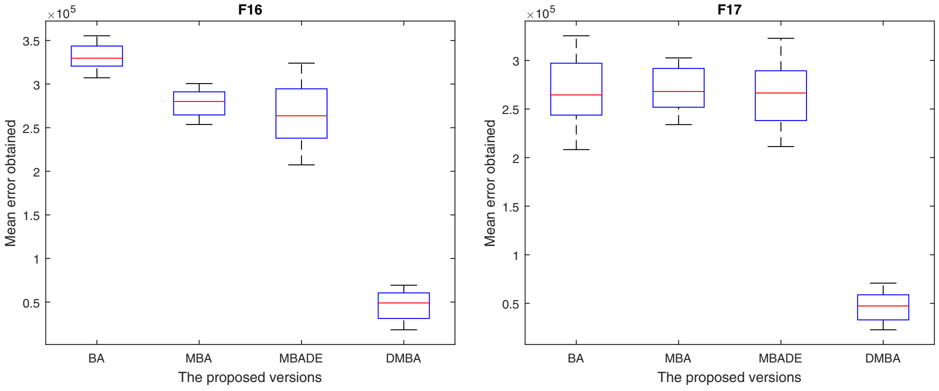
Fig. 10 The distribution of the compared algorithms on CEC 2017 test functions



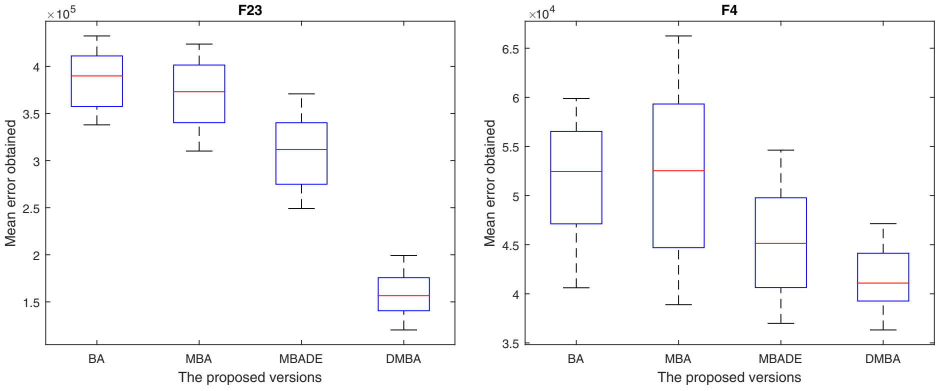
(a) F1 (b) F3



(c) F7 (d) F9



(e) F16 (f) F17



(g) F23 (h) F24

Table 13 Comparative results of DMBA with other similar methods on CEC 2017 test functions using 30 Dim

30 Dim Function	Comparative algorithms								
No. Ranking	FA [44]	PSO [45]	GSA [46]	GSO [47]	FFPSO [44]	DVBA [48]	HPSOFF [49]	BA [50]	DMBA Our proposed
F1	3,3061E+10	5,5573E+09	3,1400E+08	2,2000E+08	1,1348E+11	1,0500E+05	6,0544E+09	3,3661E+11	1,5448E+07
Rank	7	5	4	3	8	1	6	9	2
F2	1,8379E+39	3,9099E+32	1,5500E+10	1,4400E+10	1,0418E+57	2,9100E+04	3,1775E+34	2,4874E+49	3,5418E+27
Rank	7	5	3	2	9	1	6	8	4
F3	2,1556E+05	1,3257E+05	8,6400E+04	9,9600E+04	6,7723E+09	1,5900E+04	1,3635E+05	2,8455E+09	1,4688E+04
Rank	7	5	3	4	9	2	6	8	1
F4	6,8240E+03	9,0860E+02	1,3900E+03	1,8400E+03	4,4924E+04	1,8800E+02	1,2583E+03	3,5378E+04	1,5499E+02
Rank	7	3	5	6	9	2	4	8	1
F5	8,7976E+02	8,0387E+02	2,0000E+01	2,0400E+01	1,1284E+03	2,0100E+01	8,1763E+02	5,2582E+03	7,5688E+01
Rank	7	5	1	3	8	2	6	9	4
F6	6,7297E+02	6,5628E+02	3,0400E+01	1,2600E+02	7,2989E+02	2,3400E+02	6,7252E+02	6,8453E+02	2,0256E+02
Rank	7	5	1	2	9	4	6	8	3
F7	1,7985E+03	1,1273E+03	1,6300E+03	9,6200E+01	2,9927E+03	1.0700E+03	1,1241E+03	1,7985E+03	1,0634E+03
Rank	7	5	6	1	9	3	4	7	2
F8	1,1591E+03	1,0857E+03	1,4600E+03	4,1700E+02	1,3608E+03	3,0000E+01	1,1756E+03	1,6548E+04	1,2425E+02
Rank	5	4	8	3	7	1	6	9	2
F9	1,3053E+04	6,6038E+03	1,6400E+03	8,0900E+02	3,0274E+04	2,1400E+03	9,0601E+03	1,0143E+04	2,4584E+02
Rank	8	5	3	2	9	4	6	7	1
F10	9,4664E+03	9,2920E+03	3,9100E+03	7,6600E+03	1,0715E+04	7,9000E+03	8,9099E+03	9,8757E+04	6,2946E+02
Rank	7	6	2	3	8	4	5	9	1
F11	1,1963E+04	3,6404E+03	4,3800E+03	1,7600E+04	3,4403E+04	3,9300E+02	4,5469E+03	3,4589E+04	3,3465E+02
Rank	6	3	4	7	8	2	5	9	1
F12	3,0340E+09	6,4666E+08	2,0300E+02	1,1400E+00	2,7727E+10	9,8000E+01	3,3040E+08	3,9450E+09	7,9259E+05
Rank	7	6	3	1	9	2	5	8	4
F13	1,1545E+09	1,9336E+08	3,7300E+04	5,5400E+04	2,7922E+10	5,1000E+04	2,7671E+07	4,1745E+09	2,0185E+03
Rank	7	6	2	4	9	3	5	8	1
F14	1,8933E+06	1,1117E+06	7,2600E+05	5,5900E+05	5,6035E+07	2,0000E+04	1,1120E+06	1,3452E+07	4,4369E+04
Rank	7	5	4	3	9	1	6	8	2
F15	1,0725E+08	3,9118E+07	1,3700E+03	2,9800E+03	7,1411E+09	2,4900E+03	1,5676E+06	1,7255E+08	2,3512E+03
Rank	7	6	1	2	9	4	5	8	3
F16	4,4164E+03	3,9514E+03	1,3700E+01	9,4800E+05	9,6352E+03	1,2700E+02	3,6383E+03	4,4645E+04	1,8761E+02
Rank	6	5	1	9	7	2	4	8	3
F17	3,0044E+03	2,6556E+03	2,3000E+07	4,5000E+07	3,3347E+04	8,0900E+04	2,5254E+03	3,0694E+04	1,9954E+02
Rank	4	3	8	9	6	7	2	5	1
F18	2,9281E+07	7,9043E+06	5,3600E+04	8,1600E+06	1,0554E+09	7,9000E+04	8,5589E+06	2,2581E+09	2,5574E+04
Rank	7	4	2	5	8	3	6	9	1
F19	1,6603E+08	5,1700E+07	1,7300E+02	2,3200E+02	8,2628E+09	1,8100E+01	8,4341E+06	1,6632E+09	5,9657E+03
Rank	7	6	2	3	9	1	5	8	4
F20	3,1162E+03	3,0177E+03	2,5400E+05	1,2600E+05	3,7975E+03	8,2900E+03	2,9083E+03	3,9641E+03	1,1485E+03
Rank	4	3	9	8	5	7	2	6	1
F21	2,6405E+03	2,6009E+03	1,0100E+07	1,8800E+07	2,8880E+03	8,4500E+04	2,5982E+03	2,8705E+03	1,3257E+02
Rank	4	3	8	9	6	7	2	5	1
F22	8,0003E+03	6,8425E+03	1,2700E+03	3,0500E+03	1,2117E+04	4,7000E+02	5,7433E+03	5,3253E+04	3,7366E+02
Rank	7	6	3	4	8	2	5	9	1

Table 13 (continued)

30 Dim Function	Comparative algorithms								
No.	FA	PSO	GSA	GSO	FFPSO	DVBA	HPSOFF	BA	DMBA
Ranking	[44]	[45]	[46]	[47]	[44]	[48]	[49]	[50]	Our proposed
F23	3,1542E+03	3,0819E+03	2,6600E+02	4,2600E+02	4,0749E+03	3,1600E+03	3,0969E+03	3,3652E+04	2,0124E+03
Rank	6	4	1	2	8	7	5	9	3
F24	3,3056E+03	3,2139E+03	2,1500E+02	2,0300E+02	4,4529E+03	2,4900E+02	3,2719E+03	5,7656E+04	1,1582E+03
Rank	7	5	1	2	8	3	6	9	4
F25	4,9480E+03	3,2145E+03	2,0500E+02	3,3500E+02	1,5958E+04	2,1900E+02	3,2320E+03	4,9540E+05	1,0623E+03
Rank	7	5	1	3	8	2	6	9	4
F26	8,8041E+03	6,6461E+03	1,9000E+02	2,0500E+02	1,6163E+04	1,9000E+02	7,2386E+03	7,8541E+04	2,6540E+02
Rank	7	5	1	3	8	1	6	9	4
F27	6,6059E+03	3,3573E+03	1,7400E+03	3,5500E+03	5,8099E+03	5,7000E+03	3,4360E+03	3,6456E+04	2,6543E+03
Rank	8	3	1	5	7	6	4	9	2
F28	5,6940E+03	3,6388E+03	2,3100E+03	1,9900E+04	1,3535E+04	1,5600E+03	3,7938E+03	5,6356E+04	3,3855E+02
Rank	6	4	3	8	7	2	5	9	1
F29	5,7051E+03	4,9240E+03	3,3000E+07	8,1300E+05	9,3924E+04	1,8400E+04	4,8406E+03	5,4565E+04	1,6749E+03
Rank	4	3	9	8	7	5	2	6	1
F30	1,7163E+08	4,8514E+07	1,7900E+06	7,7300E+05	4,1425E+09	1,1900E+05	2,5749E+07	1,6452E+09	4,1495E+04
Rank	7	6	4	3	9	2	5	8	1
Mean rank	6.46	4.63	3.46	4.23	8.00	3.10	4.86	8.03	2.13
Final ranking	7	5	3	4	8	2	6	9	1

algorithm is evaluated and assessed using different well-known benchmark functions to study the characteristics of the proposed method (DMBA). Also, The DMBA is compared with the other well-known published algorithms in the literature using CEC 2005, CEC 2011, and CEC 2017 optimization functions. Experimental results show that the proposed algorithm (DMBA) outperformed most of the comparative algorithms.

In the future work, to confirm further the optimization capability of the DMBA algorithm, it will also be applied to some real-life optimization problems such as neural network optimization [8], feature selection [55] and economic load dispatch problem. Moreover, local search can be hybridized with BA to further enhancement in the exploitation capability of BA. Other recently produced and successful meta-heuristic methods can be employed with the proposed method. More work is necessary to further investigate the strength and limitations of the proposed DMBA. In this regard, several variant versions of DMBA can be proposed by redefining new membrane rules or introducing new membrane structure [56].

We also intend to delve deeper into the possible parallelization of DMBA for large-scale optimization problems and exploring the use of a different number of membranes in different cores simultaneously [25, 57].

Table 14 Summary of the five IEEE CEC 2011 problems

Problem	Dimensions	Constraints
P_1	6	Bound constrained
P_2	1	Unconstrained
P_3	30	Bound constrained
P_4	20	Bound constrained
P_5	126	Linear equality constraints

Table 15 The comparative results on the first 5 problems of cec 2011 test functions in terms of Best, Mean and Standard deviation

	Metric	Test problems				
		P_1	P_2	P_3	P_4	P_5
BA	Best	9.942E-01	0.0000E+00	−3.145E+01	8.329E-01	2.932E+06
	Mean	1.9740E+01	0.0000E+00	−2,0839E+01	1.2503E+00	3,3873E+06
	Std	5.9418E+00	0.0000E+00	9.3527E +00	1,8169E-0	1,8603E+ 05
MBA	Best	1.0563E-06	0.0000E+00	−3.6845E+01	6,3208E-01	2.0566E+04
	Mean	5.4223E+00	0.0000E+00	−3.3316E+01	9.7435E-01	1.4889E+ 05
	Std	5.5017E+00	0.0000E +00	1.9022E+00	1.2582E-01	2.3651E+ 05
MBADE	Best	0.0000E+00	0.0000E+00	−3.6845E+01	5.5024E-01	5.5695E +02
	Mean	4.0220E+00	0,0000E+00	−3,4281E+01	8,5138E-01	1.6259E+03
	Std	5.4949E+00	0.0000E+00	1.7565E+00	1.5846E-01	6.2465E+02
MIBBA	Best	0.0000E+00	1.22E+00	−3.145E+01	8.122E-01	1.22E+02
	Mean	5.8914E+00	0.52E+00	−3.22E+01	7.9E-01	1.22E+01
	Std	5.22E+00	0.33E+00	−3.55E+01	1.1E-01	1.05E+01
DMBA	Best	0.0000E+00	0.0000E+00	−3.6845E+01	7.0000E-01	7.9995E+01
	Mean	3.9914E+00	0.0000E+00	−3.5038E+01	7.4840E-01	2.1310E+02
	Std	5.1123E+00	0.0000E+00	8.3292E-01	1.2491E-01	6.7234E+01

Table 16 Average error rate obtained in CEC 2005 (dimension 10)

Algo.	F_1	F_2	F_3	F_4	F_5	F_6	F_7
BA	**1.00E-09**	**1.00E-09**	2.41E+02	3.81E+02	2.92E+01	4.96E-02	7.04E+01
iBA	**1.00E-09**	**1.00E-09**	**2.48E+01**	2.99E+03	1.47E-01	1.69E-03	4.59E+01
dBA	**1.00E-09**	**1.00E-09**	2.36E+05	1.22E03	**1.00E-09**	3.54E+01	**4.31E-01**
NBA	**1.00E-09**	4.38E-07	3.46E+03	2.54E+03	8.81E-02	7.97E-01	9.32E+00
GBA	**1.00E-09**	**1.00E-09**	2.41E+02	3.81E+02	2.92E+01	4.96E-02	7.04E+01
MIBBA	**1.00E-09**	**1.00E-09**	2.93E+01	4.59E+02	2.00E+01	2.33E+02	4.0E+01
DMBA	**1.00E-09**	**1.00E-09**	2.52E+01	**3.22E+02**	1.43E-01	**1.22E-03**	**4.31E-01**

Table 17 Average error rate obtained in CEC 2005 (dimension 10)

Algo.	F_8	F_9	F_{10}
BA	2.01E+01	3.94E+01	5.5E+01
iBA	**2.00E+01**	2.48E+01	4.0E+01
dBA	2.04E+01	8.22E+00	**1.0E+01**
NBA	2.01E+01	3.82E+01	6.6E+01
GBA	2.01E+01	3.94E+01	5.5E+01
MIBBA	**2.00E+01**	2.33E+02	4.0E+01
DMBA	**2.00E+01**	**7.43E+00**	3.55E+01

Table 18 The comparison of HS, MIHS, MIBBA and DMBA for the benchmark functions from CEC2005, dimension 30 in terms of Mean values

Algo.	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
HS	8.7e-07	435	8.7e6	1857	3575	133.5	2713	20.8	1.4e-4	256
MIHS	6.3e-08	2.6e-07	1.5e6	371	3167	68.1	2713	20.8	1.2e-05	149
MIBBA	8.1e-07	8.5e-07	5.5e6	570	3107	69	2713	20.8	1.5e-4	149
DMBA	6.1e-08	2.6e-07	1.9e6	371	3107	68	2713	20.8	5.9e-5	141

References

1. Abualigah L (2020) Group search optimizer: a nature-inspired meta-heuristic optimization algorithm with its results, variants, and applications. Neural Comput Appl 1–24
2. Luo J, Shi B (2019) A hybrid whale optimization algorithm based on modified differential evolution for global optimization problems. Appl Intell 49(5):1982–2000. https://doi.org/10.1007/s10489-018-1362-4
3. Laskar NM, Guha K, Chatterjee I, Chanda S, Baishnab KL, Paul PK (2019) Hwpso: a new hybrid whale-particle swarm optimization algorithm and its application in electronic design optimization problems. Appl Intell 49(1):265–291. https://doi.org/10.1007/s10489-018-1247-6
4. Luo J, Liu Z (2019) Novel grey wolf optimization based on modified differential evolution for numerical function optimization. Appl Intell 50:468–486
5. Yang X-S (2010) A new metaheuristic bat-inspired algorithm. In: González J, Pelta D, Cruz C, Terrazas G, Krasnogor N (eds) Nature inspired cooperative strategies for optimization (NICSO 2010), vol 284 of studies in computational intelligence. Springer, Berlin, pp 65–74
6. Jun L, Liheng L, Xianyi W (2015) A double-subpopulation variant of the bat algorithm. Appl Math Comput 263:361–377
7. Banati H, Chaudhary R Multi-modal bat algorithm with improved search (mmbais). J Comput Sci, 130–144
8. Jaddi NS, Abdullah S, Hamdan AR (2015) Optimization of neural network model using modified bat-inspired algorithm. Appl Soft Comput 37:71–86
9. Abualigah LMQ (2019) Feature selection and enhanced krill herd algorithm for text document clustering. Springer, Berlin
10. Abualigah L, Diabat A (2020) A comprehensive survey of the grasshopper optimization algorithm: results, variants, and applications. Neural Comput Appl 1–24
11. Gandomi AH, Yang X-S (2014) Chaotic bat algorithm. J Comput Sci 5(2):224–232. empowering Science through Computing + BioInspired Computing
12. Jordehi AR (2015) Chaotic bat swarm optimisation (cbso). Appl Soft Comput 26:523–530
13. Jaddi NS, Abdullah S, Hamdan AR (2015) Multi-population cooperative bat algorithm-based optimization of artificial neural network model. Inf Sci 294:628–644. innovative Applications of Artificial Neural Networks in Engineering
14. Nakamura R, Pereira L, Costa K, Rodrigues D, Papa J, Yang X-S (2012) Bba: a binary bat algorithm for feature selection. In: 2012 25th SIBGRAPI conference on graphics, patterns and images (SIBGRAPI), pp 291–297. https://doi.org/10.1109/SIBGRAPI.2012.47
15. Osaba E, Yang X-S, Diaz F, Lopez-Garcia P, Carballedo R (2016) An improved discrete bat algorithm for symmetric and asymmetric traveling salesman problems. Eng Appl Artif Intell 48:59–71
16. Topal AO, Altun O (2016) A novel meta-heuristic algorithm: dynamic virtual bats algorithm. Inf Sci 354:222–235
17. Yilmaz S, Küçüksille EU (2015) A new modification approach on bat algorithm for solving optimization problems. Appl Soft Comput 28(1):259–275. https://doi.org/10.1016/j.asoc.2014.11.029
18. Chakri A, Khelif R, Benouaret M, Yang X-S (2017) New directional bat algorithm for continuous optimization problems. Expert Syst Appl 69:159–175
19. Mirjalili S, Gandomi AH (2017) Chaotic gravitational constants for the gravitational search algorithm. Appl Soft Comput 53:407–419
20. Liu Q, Wu L, Xiao W, Wang F, Zhang L (2018) A novel hybrid bat algorithm for solving continuous optimization problems. Appl Soft Comput 73:67–82. https://doi.org/10.1016/j.asoc.2018.08.012
21. Yildizdan G, Ömer Kaan Baykan A (2020) novel modified bat algorithm hybridizing by differential evolution algorithm. Expert Systems Appl 141:112–118. https://doi.org/10.1016/j.eswa.2019.112949
22. Gheorghe M, Zhang G, Pan L, Perez-Jimenez M (2014) Evolutionary membrane computing: a comprehensive survey and new results. Inf Sci 279(0):528–551
23. Heraguemi KE, Kamel N, Drias H (2016) Multi-swarm bat algorithm for association rule mining using multiple cooperative strategies. Appl Intell 45:1021–1033
24. Abualigah L (2020) Multi-verse optimizer algorithm: a comprehensive survey of its results, variants, and applications. Neural Comput Appl 1–21
25. Maroosi A, Muniyandi RC, Sundararajan E, Zin AM (2016) A parallel membrane inspired harmony search for optimization problems: a case study based on a flexible job shop scheduling problem. Appl Soft Comput 49:120–136
26. Paun G (2000) Computing with membranes. J Comput Syst Sci 61:108–143
27. Yahya RI, Hasan S, George LE, Alsalibi B (2015) Membrane computing for 2d image segmentation. Int J Adv Soft Comput Appl 7(1):35–50
28. Martin-Vide C, Păun G, Pazos J, Rodríguez-Patón A (2003) Tissue p systems. Theor Comput Sci 296(2):295–326
29. Song T, Liu X, Zeng X (2015) Asynchronous spiking neural p systems with anti-spikes. Neural Process Lett 42(3):633–647. https://doi.org/10.1007/s11063-014-9378-1
30. Alsalibi B, Venkat I, Al-Betar MA (2017) A membrane-inspired bat algorithm to recognize faces in unconstrained scenarios. Eng Appl Artif Intell 64:242–260
31. Nishida TY (2006) Membrane algorithms: approximate algorithms for NP-complete optimization problems. Springer, Berlin, pp 303–314
32. Liu C, Fan L (2016) A hybrid evolutionary algorithm based on tissue membrane systems and cma-es for solving numerical optimization problems. Knowl-Based Syst 105:38–47
33. Pan L, Alhazov A, Isdorj T-O (2005) Further remarks on p systems with active membranes, separation, merging, and release rules. Soft Comput 9(9):686–690
34. Alomari OA, Khader AT, Al-Betar MA, Abualigah LM (2017) Gene selection for cancer classification by combining minimum redundancy maximum relevancy and bat-inspired algorithm. Int J Data Min Bioinform 19(1):32–51
35. Yahya RI, Shamsuddin SM, Yahya SI, Hasan S, Al-Salibi B, Al-Khafaji G (2016) Image segmentation using membrane computing: a literature survey. In: Bio-inspired computing-theories and applications. Springer, pp 314–335
36. Karaboga D, Akay B (2009) A comparative study of artificial bee colony algorithm. Appl Math Comput 214(1):108–132
37. Suganthan PN, Hansen N, Liang JJ, Deb K, Chen YP, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization, Tech. rep., Nanyang Technological University, Singapore
38. Liang J, Qu B-Y, Suganthan PN (2017) Problem definitions and evaluation criteria for the cec 2017 special session and competition on single objective real-parameter numerical optimization
39. Fister I, Yang XS A hybrid bat algorithm. Elektrotehniski Vestnik/Electrotech Rev 80(1):34–68
40. Fister IJr, Fister D, Fister I (2013) Differential evolution strategies with random forest regression in the bat algorithm. In: Proceedings of the 15th annual conference companion on genetic and evolutionary computation, GECCO ’13 companion. ACM, New York, pp 1703–1706

41. Yilmaz S, Kucuksille EU, Cengiz Y (2014) Modified bat algorithm. Electron Electr Eng 20(2):71–78
42. Mirjalili S, Mirjalili SM, Yang X-S (2014) Binary bat algorithm. Neural Comput Appl 25(3):663–681
43. He X-S, Ding W-J, Yang X-S (2014) Bat algorithm based on simulated annealing and gaussian perturbations. Neural Comput Appl 25(2):459–468
44. Aydilek IB (2018) A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems. Appl Soft Comput 66:232–249
45. Ngo TT, Sadollah A, Kim JH (2016) A cooperative particle swarm optimizer with stochastic movements for computationally expensive numerical optimization problems. J Comput Sci 13:68–82
46. Yazdani S, Nezamabadi-pour H, Kamyab S (2014) A gravitational search algorithm for multimodal optimization. Swarm Evol Comput 14:1–14
47. Abedinpourshotorban H, Shamsuddin SM, Beheshti Z, Jawawi DN (2016) Electromagnetic field optimization: a physics-inspired metaheuristic optimization algorithm. Swarm Evol Comput 26:8–22
48. Topal AO, Altun O (2016) A novel meta-heuristic algorithm: dynamic virtual bats algorithm. Inf Sci 354:222–235
49. Yang X-S, Gandomi AH Bat algorithm: a novel approach for global engineering optimization. Eng Comput, 464–483
50. Mirjalili S, Mirjalili SM, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27(2):495–513
51. Das S, Suganthan PN (2011) Problem definitions and evaluation criteria for cec 2011 competition on testing evolutionary algorithms on real world optimization problems
52. Guo S-S, Wang J-S, Ma X-X (2019) Improved bat algorithm based on multipopulation strategy of island model for solving global function optimization problem. Comput Intell Neurosci, 12 pages
53. Meng X-B, Gao X, Liu Y, Zhang H (2015) A novel bat algorithm with habitat selection and doppler effect in echoes for optimization. Expert Syst Appl 42(17):6350–6364. https://doi.org/10.1016/j.eswa.2015.04.026. http://www.sciencedirect.com/science/article/pii/S0957417415002560
54. Alomari OA, Khader AT, Al-Betar MA, Abualigah LM (2017) Mrmr ba: a hybrid gene selection algorithm for cancer classification. J Theor Appl Inf Technol 95(12):2610–2618
55. Alsalibi B, Venkat I, Subramanian KG, Lutfi S, Wilde PD (2015) The impact of bio-inspired approaches towards the advancement of face recognition. ACM Comput Surv 48(1):1–33
56. Song B, Li K, Orellana-Martín D, Valencia-Cabrera L, Pérez-Jiménez MJ (2020) Cell-like p systems with evolutional symport/antiport rules and membrane creation. Inf Comput 104542. https://doi.org/10.1016/j.ic.2020.104542. http://www.sciencedirect.com/science/article/pii/S0890540120300304
57. Kechid A, Drias H (2020) Cultural coalitions detection approach using gpu based on hybrid bat and cultural algorithms. Appl Soft Comput 106368. https://doi.org/10.1016/j.asoc.2020.106368. http://www.sciencedirect.com/science/article/pii/S1568494620303082