Interdisciplinary Sciences: Computational Life Sciences (2019) 11:496–507
https://doi.org/10.1007/s12539-018-0295-2

ORIGINAL RESEARCH ARTICLE

Modified Global Flower Pollination Algorithm and its Application for Optimization Problems

Moh'd Khaled Yousef Shambour[1] · Ahmed A. Abusnaina[2] · Ahmed I. Alsalibi[3]

Received: 29 January 2018 / Revised: 11 March 2018 / Accepted: 22 March 2018 / Published online: 28 March 2018

Abstract

Flower Pollination Algorithm (FPA) has increasingly attracted researchers' attention in the computational intelligence field. This is due to its simplicity and efficiency in searching for global optimality of many optimization problems. However, there is a possibility to enhance its search performance further. This paper aspires to develop a new FPA variant that aims to improve the convergence rate and solution quality, which will be called modified global FPA (mgFPA). The mgFPA is designed to better utilize features of existing solutions through extracting its characteristics, and direct the exploration process towards specific search areas. Several continuous optimization problems were used to investigate the positive impact of the proposed algorithm. The eligibility of mgFPA was also validated on real optimization problems, where it trains artificial neural networks to perform pattern classification. Computational results show that the proposed algorithm provides satisfactory performance in terms of finding better solutions compared to six state-of-the-art optimization algorithms that had been used for benchmarking.

Keywords Flower Pollination Algorithm · Computational intelligent · Optimization problems · Exploration · Artificial neural networks

1 Introduction

Algorithms inspired from nature are widely used in the optimization field including, Genetic Algorithm (GA) [1], Particle Swarm Optimization (PSO) [2], Simulated Annealing (SA) [3], Artificial Bee Colony (ABC) [4], Cuckoo Search (CS) [5], Harmony Search Algorithm (HSA) [6], Firefly Algorithm (FA) [7], and Bat Algorithm (BA) [8].

Flower pollination is an intriguing process in the natural world. The evolutionary process characteristics inspired Yang [9] to propose a new optimization algorithms called Flower Pollination Algorithm (FPA). FPA belongs to bio-inspired algorithms that simulate flower pollination behavior in nature, which is characterized by simple implementation, adaptability, flexibility, use of less control parameters, and an overall good search performance [9].

Versions of FPA Algorithm have been successfully applied in several real-world applications and optimization problems, such as image segmentation [10, 11], feature selection [12, 13], benchmark optimization functions [9, 14], economic dispatch problems [15–17], constrained engineering optimization problems [18, 19], wireless sensor network [20, 21], and many others [22–24].

The success of the search behavior of FPA mainly relies on finding a suitable balance between exploitation and exploration forces. Exploitation involves an intensive search of existing solutions that have been explored previously, whereas exploration is the ability to search and discover new regions of the search space for further possibilities.

In this paper, a new FPA variant called modified global FPA (mgFPA) is proposed aiming to improve the exploration ability of FPA. The exploration part of mgFPA is adjusted to target specific search regions through utilizing the information gathered from two randomly selected solutions during evolution process. This allows opportunities to the algorithm search process to investigate good search regions and converge the search to global or near-global optima.

✉ Moh'd Khaled Yousef Shambour
myshambour@uqu.edu.sa

[1] The Custodian of the Two Holy Mosques Institute for Hajj and Umrah Research, Umm Al-Qura University, Makkah, Saudi Arabia

[2] Department Of Computer Science, Birzeit University, Ramallah, Palestine

[3] Israa University, Gaza, Palestine

To validate and assure the efficiency of mgFPA, we applied the algorithm on two different optimization problems, a set of 23 numerical benchmark functions, and in training the artificial neural network (ANN) for pattern classification problem. The results illustrate that the proposed mgFPA is approximate to or better than the results of six state-of-the-art optimization algorithms.

The paper is organized as follows: Sect. 2 presents the FPA and explains how it works. An overview of recent work in FPA is presented in Sect. 3. Section 4 presents the proposed FPA. Experimental results and discussion are presented in Sect. 5. Thereafter, Sect. 6 provides the training artificial neural networks by mgFPA. Finally, Sect. 7 contains some concluding remarks and suggests further work.

2 Flower Pollination Algorithm (FPA)

FPA, an optimization algorithm, was developed according to the natural process of flower pollination [9]. This algorithm has become popular due to its efficiency in solving a wide range of optimization problems from various disciplines and applications [22, 23].

The reproduction process is based on two basic forms of pollination: (1) biotic\cross pollination, where the pollen is transferred to long distances via nature pollinators, such as bees, butteries, beetles, and bats and (2) abiotic\self-pollination, where the pollen is usually carried for short distances through wind and water diffusion.

The natural concepts of FPA are presented as follows [25]:

- Flowers represent the stored solutions in a population (pop), as shown in Eq. (1).

$$\text{pop} = \begin{bmatrix} \overrightarrow{X^1} = [x_1^1 & x_2^1 & \dots & x_{ND}^1] \\ \overrightarrow{X^2} = [x_1^2 & x_2^2 & \dots & x_{ND}^2] \\ \vdots \quad \vdots & \vdots & \ddots & \vdots \\ \overrightarrow{X^n} = [x_1^n & x_2^n & \dots & x_{ND}^n] \end{bmatrix} = \begin{bmatrix} f(\overrightarrow{X^1}) \\ f(\overrightarrow{X^2}) \\ \vdots \\ f(\overrightarrow{X^n}) \end{bmatrix} \tag{1}$$

  where the population includes n number of flowers\solution (i.e., $\overrightarrow{X^1}, \overrightarrow{X^2}, \overrightarrow{\dots}, \overrightarrow{X^n}$). ND is the number of decision variables, while $f(\overrightarrow{X^k})$ represents the fitness function value for solution $k = \{1, 2, \dots, n\}$.
- Biotic\cross pollination represents the global search with the potential Lèvy flights properties which is used to update the solution positions, as shown in Eq. (2).

$$x_{t+1}^i = x_t^i + \gamma L(x^* - x_t^i), \tag{2}$$

  where x_t^i represents the solution at t iteration, x^* is the best observed solution, γ is a scaling factor, and L is the Lèvy distribution that corresponds to the strength of the pollination, as given in Eq. (3):

$$L \sim \frac{\lambda\Gamma(\lambda)\sin(\Pi\lambda/2)}{\Pi}\frac{1}{s^{1+\lambda}}, \tag{3}$$

  where Γ is the standard gamma function, and this distribution is valid for large steps s>0.
- Abiotic\self-pollination represents the local search procedure that is based on modifying the solutions according to two randomly selected solutions. The FPA local pollination is presented in Eq. (4):

$$x_{t+1}^i = x_t^i + r_1 \times (x_t^a - x_t^b), \tag{4}$$

  where x_t^a and x_t^b are two solutions selected from two different flowers of the same species, and r_1 represents the uniform random distribution between [0,1].

2.1 Fundamental Steps of FPA

Figure 1 shows the basic flowchart of FPA including the basic search steps which are described as follows:

- **Step 1: Initialize FPA control parameters** The control parameters are population size (n), maximum number of iterations ($MaxItr$), and switch probability (P).
- **Step 2: Initialize the population** The initial solutions $X^1, X^2, \dots, X^n$ are generated randomly according to the given bounded upper and lower bounds [LB, UB], and the fitness value for each solution is evaluated.
- **Step 3: Obtain the best solution from initial population** The solution with the best fitness value (X^*) is defined (according to minimizing or maximizing problem being optimized).
- **Step 4: Generate new population** The local or global search strategy is determined according to the value of switching probability $P \in [0, 1]$, such that

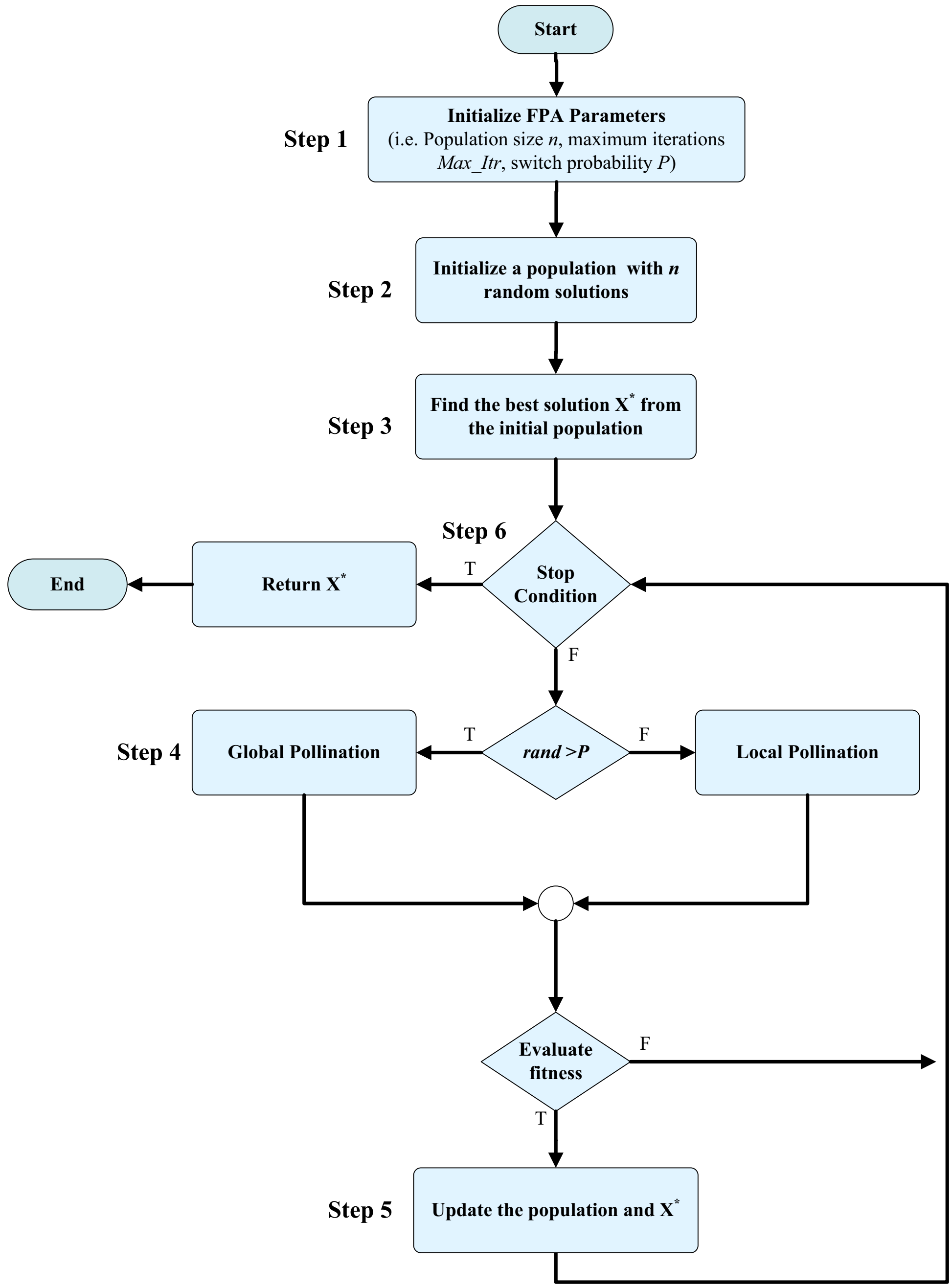
$$x_{t+1}^i = \begin{cases} x_t^i + \gamma L(x^* - x_t^i), & \text{rand}_1 > P \\ x_t^i + \text{rand}_2 \times (x_t^a - x_t^b), & \text{rand}_1 \le P. \end{cases} \tag{5}$$

- **Step 5: Update the population and the best solution** Replace the new solutions that have better quality than the stored solution and obtain the best solution X^*, in terms of the fitness value.
- **Step 6: Check the stopping criteria** Steps 4 and 5 are repeated until the maximum iteration number is achieved.

3 Literature Review

FPA and its variants have been successfully adapted and applied to solve a wide range of optimization problems. This section presents an overview of recent research activities and findings.

Fig. 1 Flowchart of the FPA



Kaur and Arora [21] studied the performance of four nature-inspired algorithms, namely, FPA, FA, Grey Wolf Optimization (GWO), and PSO to find the best location estimation of wireless sensor nodes. Results showed that FPA exhibits higher localization accuracy in estimating the position of nodes compared with other algorithms.

Abdel-Basset et al. [26] performed a comparative study between CS and FPA on ten numerical functions selected from CEC-2017 [27]. FPA exhibits better performance than CS in terms of computation speed, whereas the latter is better in obtaining best solutions. The authors recommend selecting an appropriate algorithm according to the requirements of a given problem (i.e., time vs. accuracy).

Zhou et al. [28] applied greedy search procedure and dynamic switching parameter to improve the local and global pollination of FPA by finding a good balance between exploration and exploitation abilities. The proposed method was tested on 18 numerical functions and two structural engineering optimization problems from CEC-2018 [29]. The observed results show that the proposed method can find accurate solutions and display fast convergence with a high level of stability.

Abdelaziz et al. [17] adapted FPA to solve economic load and emission dispatch problems in a power system, where the cost function is limited by the output limits of generation units and transmission losses. FPA shows improved performance compared with 14 stochastic algorithms, such as PSO and differential HSA, in terms of solution cost and computational time.

The switch probability (P) manages the exploitation and exploration processes of FPA, such that, if the value of P is greater than 0.5, then the likelihood of using local pollination procedure is higher than that of global pollination procedure. Salgotra and Singh [30] proposed a new FPA version that incorporated the use of dynamic switching of P value, new mutation operators, and adapting local search. The probability of the proposed switch is linearly decreased by algorithm iterations, starting from an initial P value equal to 0.8. The dynamic switching probability exhibits reliable performance in exploring and exploiting the search space. The statistical results on 17 standard benchmark functions show that the adaptive Lèvy FPA has superior performance compared to well-known methods including differential evolution, ABC, BA, FA, and GWO.

Xu et al. [31] proposed an improved variant of FPA to minimize the cost of machine production by optimizing the parameter settings of multi-pass turning process. The proposed algorithm replaces the randomized process of generating initial solutions utilizing good point set theory while using Deb's heuristic rules [32]. The simulation experiments show that the results of proposed approach are comparative with other previously published results.

Alyasseri et al. [23] reported that hybridizing FPA with other algorithms has the most modifications that added to the original FPA. Wang et al. [10] proposed a hybrid FPA based on a modified randomized location for multi-threshold image segmentation. The proposed method was used to explore the optimal threshold values for maximizing Otsus objective functions [33] on eight medical images. The method shows high efficiency on the basis of stability, computation cost, and solution quality.

Abdel-Baset and Hezam [34] also modified the standard FPA, where it was hybridized with GA to improve the search accuracy. The experimental results over a set of constrained optimization problems show that incorporating GA to FPA improves the quality of produced solutions.

Another hybrid version of FPA was proposed to improve the exploitation capabilities of the basic FPA. In this version, the clonal selection algorithm was injected inside FPA and used to solve two different optimization problems, namely, numerical benchmark functions [14] and feature selection problem [13]. Further research can be found in [22–24, 35].

Several studies proof that the computational performance of such algorithm could be improved through utilizing the experience of previous search generation process, such as in [36–39]. This paper provides a new improvement to the search performance of FPA by allowing the current population to guide the search process toward promising regions of search space and thus increase the chances of finding the optimal solution.

4 Proposed Method

The search efficiency of an algorithm can be ameliorated through the efficient use of exploration and exploitation search strategies. In this paper, the exploration part of FPA is modified, such that the good experience of current solutions can be effectively utilized in generating new solutions. This modification can be achieved by guiding the exploration search procedure into more promising areas that are previously explored by current solutions, thereby increasing the convergence rate.

The proposed modified global Flower Pollination Algorithm (mgFPA) explores the search space of the problem domain by either selecting the basic global pollination (BGP) or heuristic bounded search space (HBSS) mechanism. Both mechanisms present equally probability of selection during evolution. The proposed HBSS procedure narrows the search process to a certain area of search space using the information of two randomly selected parents, as shown in Eq. 6:

$$x_{t+1}^{i_j} = \left(\max\left(x_t^{a_j}, x_t^{b_j}\right) - \min\left(x_t^{a_j}, x_t^{b_j}\right)\right) \cdot r_2 + \min\left(x_t^{a_j}, x_t^{b_j}\right), \quad (6)$$

where $x_t^{i_j}$ represents the jth variable of ith solution vector at t iteration, x_t^a and x_t^b are two randomly selected solutions, and r_1, r_2 represent the uniform random distribution between [0,1].

The search procedure of HBSS is focused on the most promising regions of the search space according to the experience of the current population. The BGP procedure is required to keep the algorithm from being trapped into local minima by exploring the entire search space. The steps of the proposed mgFPA can be summarized as the pseudocode shown in Algorithm 1.

Algorithm 1 Pseudo code of mgFPA.

```
Begin
Initialize FPA control parameters
  [number of flowers (n),max iteration (MaxItr), switch probability (P),
   problem dimensions (ND)]
Initialize a population (pop) of n flowers
  [pop= x^1, x^2, ..., x^n ]
Observe the best solution x* ∈ pop
t ← 1
do
  for each i ∈ (1, n) do
    if U(0,1) < P
      for each j ∈ (1, ND) do        \* Global Pollination
        if U(0,1) < 0.5
          x_{t+1}^{ij} = x_t^{ij} + γL(x* − x_t^{ij})
        else
          x_{t+1}^{ij} = (max(x_t^{aj}, x_t^{bj}) − min(x_t^{aj}, x_t^{bj})) × U(0,1) + min(x_t^{aj}, x_t^{bj})
            \\ where a, b ∈ (1, n), a ≠ b
        end if
      end for
    else
    x_{t+1}^i = x_t^i + U(0,1) × (x_t^c − x_t^d)        \* Local Pollination
      \\where c, d ∈ (1, n), c ≠ d
    end if
    if (f(x_{t+1}^i) < f(x_t^i)) then
      update x_t^i by x_{t+1}^i
    end if
    if (fx_{t+1}^i) < f(x*)) then
      update x* by x_{t+1}^i
    end if
  end for
  t ← t + 1
Until t > MaxItr
End
```

5 Experiments and Results

Several types of optimization functions with different attributes were used to release the improvement effect of our proposed method. The best mean and standard deviation and number of optimal solutions obtained for every function are reported over 30 runs.

5.1 Benchmark Functions

A set of 23 test functions, which were frequently used in the literature as benchmark, were used to validate the performance of the proposed method [40, 41]. Table 1 presents the function expression, domain range, global minima, and properties for each test optimization function.

5.2 Experimental Setup

The control parameters of the proposed and compared algorithms present the same values, which are set according to [14], where number of flowers (i.e., population size) $n = 50$, number of iterations $MaxIter = 1500$,dimensionality $d = 2$, switch probability $P = 0.8$, and $\gamma = 0.01$.

The proposed algorithm is coded in Matlab 2014 and the experiments are executed under Windows 7 operating system, Intel core i5 CPU @ 3.4 GHz with a memory of 16.00 GB.

5.3 Results and Discussion

This section presents the results of applying mgFPA on various sets of test functions and provides a comparison between mgFPA and other rival optimization methods. mgFPA is compared against the basic FPA, Modified Flower Pollination Algorithm (MFPA), GA, BAT, FF, and SA algorithms [14].

As shown in Table 2, the mean values of mgFPA are higher than or equal to those of the original FPA and MFPA in 82.6 and 78.2% of the total experiment cases, respectively. mgFPA also achieves the best mean values in 95.6% of the cases compared with other competitive algorithms. Moreover, mgFPA achieves the best, or equal to the best, minimum values in 94.4% of the total experiment cases. Furthermore, the standard deviation values of mgFPA are typically lower than those of the other algorithms in most cases, indicating the higher stability of the proposed algorithm.

Average error rate (AER) was used to evaluate the performance and stability of the compared methods (Table 3). AER is computed using Eq. 7:

$$\mathrm{AER} = |f_i(x^*) - f_i(x^{\mathrm{best}})| \tag{7}$$

where $f_i(x^*)$ is the optimal solution for a given function $f_i(x)$, and $f_i(x^{\mathrm{best}})$ is the average of the best solution in 30 runs for a given optimization method. The best results reported have been highlighted in bold.

It can be observed from the table above that mgFPA exhibits the lowest AER values in 78.2% of the test functions compared with the other algorithms. Meanwhile, FPA and MFPA present the lowest values in 39.1 and 65.2% of the cases, respectively. This finding demonstrates that the proposed mgFPA exhibits satisfactory search ability over other algorithms.

6 Training Artificial Neural Networks by mgFPA

Artificial neural network (ANN) is a simplified mathematical approximation of biological neural system in terms of structure and function. The most important part of ANN is the learning process (i.e., training algorithm), which focuses on adjusting the neuron weight values to minimize the error between the actual ANN output and the desired output[42].

Feed-forward ANN weights (including biases) are adjusted using mgFPA to solve a given classification problem. mgFPA is applied regularly until the training termination condition is met. Each ANN is represented by a vector, which forms the complete set of ANN structure with their corresponding weights and biases.

Each individual in the population (i.e., flower) represents an ANN network. The sum squared errors (SSE) is an

Table 1 Numerical benchmark functions and their properties

Function name	Expression	Domain	Global minimum	Category
Ackley's function	$f_1(x) = -20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{N} x_i^2}\right) - \exp\left(\frac{1}{30}\sum_{i=1}^{N}\cos(2\pi x_i)\right) + 20 + e$	[−32,32]	$f(x^*) = 0, x^* = (0, \dots 0)$	Multimodal
Sphere function	$f_2(x) = \sum_{i=1}^{N} x_i^2$	[−100,100]	$f(x^*) = 0, x^* = (0, \dots 0)$	Unimodal
Easoms function	$f_3(x) = (-1^{d+1})\cos(x_1)\cos(x_2)exp(-(x-\pi)^2 - (y-\pi)^2)$	[-100,100]	$f(x^*) = -1, x^* = (\pi, \pi)$	Multimodal
Griewank function	$f_4(x) = \frac{1}{4000}\sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	[−600,600]	$f(x^*) = 0, x^* = (0, \dots 0)$	Multimodal
Rastrigin function	$f_5(x) = \sum_{i=1}^{N}(x_i^2 - 10\cos(2\pi x_i) + 10)$	[−5.12,5.12]	$f(x^*) = 0, x^* = (0, \dots 0)$	Multimodal
Rosenbrock function	$f_6(x) = \sum_{i=1}^{N-1}(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	[−30,30]	$f(x^*) = 0, x^* = (1, \dots 1)$	Multimodal
Zakharovs function	$f_7(x) = f(x_1, \dots, x_n) = \sum_{i=1}^{n} x_i^2 + (\sum_{i=1}^{n} 0.5ix_i)^2 + (\sum_{i=1}^{n} 0.5ix_i)^4$	[−5,10]	$f(x^*) = 0, x^* = (0, \dots 0)$	Unimodal
Michalewicz's function	$f_8(x) = \sum_{i=1}^{n-1}(\sin(x_{i+1})\sin^{20}(\frac{2x_{i+1}^2}{\pi}) + \sin(x_i)\sin^{20}(\frac{x_{i+1}^2}{\pi}))$	[0, π]	$f(x^*) = -1.8013, x^* = (0, 0)$	Multimodal
Dixon-price Function	$f_9(x) = (x_1 - 1)^2 + \sum_{i=2}^{n} i(2x_i^2 - x_{i-1})^2$	[−10, 10]	$f(x^*) = 0, x^* = 2^{-(\frac{2i-2}{2i})}$	Unimodal
Levys function	$f_{10}(x) = \sin^2(\pi y_i) + \sum_{i=1}^{d-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_i + 1)] + (y-1)^2(1 + \sin^2(2\pi y))$	[−10, 10]	$f(x^*) = 0, x^* = (1, \dots, 1)$	Multimodal
Cross-in-Tray function	$f_{11}(x) = -0.0001(\|\sin(x)\sin(y)exp(\|100 - \frac{\sqrt{x^2+y^2}}{\pi}\|)\| + 1)^{0.1}$	[−10, 10]	$f(x^*) = -2.06261, x^* = (\pm1.3491, \pm1.3491)$	Multimodal
Drop-wave function	$f_{12}(x) = -\frac{1 + \cos(12\sqrt{x^2+y^2})}{(0.5(x^2+y^2) + 2)}$	[−5.2,5.2]	$f(x^*) = -1, x^* = (0, 0)$	Multimodal
Eggholder function	$f_{13}(x) = \left(-(x_2 + 47)\sin\left(\sqrt{\|x_2 + \frac{x_1}{2} + 47\|}\right) + \sin\left(\sqrt{\|x_1 - (x_2 + 47)\|}\right)\right)$	[−5.12,5.12]	$f(x^*) = -959.6407, x^* = (512, 404.2319)$	Multimodal
Holder table function	$f_{14}(x) = -\|\sin(x)\cos(y)exp(\|1 - \frac{\sqrt{x^2+y^2}}{\pi}\|)\|$	[−10,10]	$f(x^*) = -19.2085, x^* = (\pm8.05502, \pm9.66459)$	Multimodal
Schaffer function N.2	$f_{15}(x) = 0.5 + \frac{\sin^2(x^2 - y^2) - 0.5}{(1 + 0.001(x^2+y^2))^2}$	[−100,100]	$f(x^*) = 0, x^* = (0, 0)$	Multimodal
Shubert function	$f_{16}(x) = \left(\sum_{i=1}^{5}\cos((i+1)x_1)\right)\left(\sum_{i=1}^{5}\cos((i+1)x_2)\right)$	[−5.12,5.12]	$f(x^*) = -186.7309$	Multimodal
Schwefel function	$f_{17}(x) = 418.9829d - \sum_{i=1}^{n} x_i\sin(\sqrt{\|x_i\|})$	[−500,500]	$f(x^*) = 0, x^* = (420.9687, \dots, 420.9687)$	Multimodal
Schaffer function N.4	$f_{18}(x) = 0.5 + \frac{\cos^2(\sin(\|x^2 - y^2\|)) - 0.5}{(1 + 0.001(x^2+y^2))^2}$	[−100,100]	$f(x^*) = 0.2925, x^* = (0, 1.25313)$	Multimodal
Beale function	$f_{19}(x) = (1.5 - x + xy)^2 + (2.25 - x + xy^2)^2 + (2.625 - x + xy^3)^2$	[−4.5,4.5]	$f(x^*) = 0, x^* = (3, 0.5)$	Multimodal
Rotated hyper-ellipsoid function	$f_{20}(x) = \sum_{i=1}^{N}\left(\sum_{j=1}^{i} x_j\right)^2$	[−65.536,65.536]	$f(x^*) = 0, x^* = (0, 0)$	Unimodal
Matyas function	$f_{21}(x) = 0.26(x^2 + y^2) - 0.48xy$	[−10,10]	$f(x^*) = 0, x^* = (0, 0)$	Unimodal

Table 1 (continued)

Function name	Expression	Domain	Global minimum	Category
StyblinskiTang function	$f_{22}(x) = \frac{1}{2}\sum_{i=1}^{n}(x_i^4 - 16x_i^2 + 5x_i)$	[−5,5]	$f(x^*) = -39.16599d, x^* = (-2.903534)$	Multimodal
De Jong function N.5	$f_{23}(x) = \left(0.002 + \sum_{i=1}^{25} \frac{1}{i + (x_1 - a_{1i})^6 + (x_2 - a_{2i})^6}\right)^{-1}$	[−65.536, 65.536]	$f(x^*) = 0$	Multimodal

objective function that will be evaluated. The SSE should be minimized in accordance with Eq. 8. The bipolar sigmoid is considered the neuron activation function:

$$f(f_i) = \text{SSE} = \sum_{p=1}^{P} \sum_{x=1}^{\text{Noutputs}} \left(D_p^x - A_p^x\right)^2 \tag{8}$$

where: $f(f_i)$: fitness value of an individual flower f_i. P: number of patterns in the classification problem. *Noutputs*: number of output neurons at the output layer. D_p^x: the desired x^{th} output of pth pattern. A_p^x: the actual x^{th} output of p^{th} pattern.

The merits of mgFPA are validated using five widely used classification problems. The classification problems obtained from UCI Machine Learning Repository [43], namely, Haberman, Iris, Glass, Wisconsin Breast Cancer, and Diabetes.

The ANN structure was designed based on three-layer architecture (input-hidden-output) [44, 45] as follows Haberman: 3-4-2, Iris:4-5-3, Cancer: 9-8-2, Diabetes: 8-7-2, Glass: 9-12-6. Each problem is divided into two parts (80:20); that is, 80% of the data set is used for training ANN, and the remaining of 20% is used for testing the accuracy of ANN.

The performance of mgFPA is assessed using two metrics, namely, classification accuracy of ANN and the obtained SSE. The performance of mgFPA is compared against six training algorithms, namely, FPA, Mussels Wandering Optimization (MWO), GA, E-MWO, Harmony Search Best-to-Worst (HS-BtW), and Back Propagation (BP). The results of Enhanced-Mussels Wondering Optimization (E-MWO), MWO, HS-BtW, GA, and BP are taken from a previous work [44]. Both the mean and the best out of 20 run values are reported.

Table 4 shows the classification accuracy results of ANN trained by different optimization algorithms. Results show no superiority of one algorithm over others that can efficiently train the ANN to obtain the highest accuracy for all problems. This phenomenon can be explained by the help of No Free Lunch (NFL) theorem; that is, no such algorithm will perform equally and efficiently on all problems [46]. However, the comparison demonstrates the mgFPA is competitive to state-of-the-art algorithms. In addition, mgFPA achieves the highest accuracy in the Glass problem, which is a multi-class problem that is difficult to classify because of its complex and large dimensional space. Based on the observed mean results, mgFPA exhibits better or equal performance compared with the original FPA in 80% of the total experiment cases.

7 Conclusion and Further Work

FPA is a swarm-based algorithm that was introduced to solve various types of optimization problems. This work improves the exploration side of FPA by contributing

Table 2 Statistical results of mgFPA and other rival optimization algorithms

	FPA	mgFPA	MFPA	BAT	FF	GA	SA
f1: Ackleys function							
Min	3.64153E−14	8.88178E−16	8.88178E−16	9.96441E−06	9.27835E−06	2.22971E−07	0.00000106
Mean	2.07259E−12	**8.88178E−16**	**8.88178E−16**	2.36293	8.17798E−05	1.24595E−06	0.086127
Std.	2.47204E−12	0	0	2.02684	3.15732E−05	1.2319E−06	0.471006
f2: Sphere function							
Min	9.52325E−33	8.8621E−116	2.98961E−70	7.14891E−13	1.10737E−12	7.33358E−16	0.000000706
Mean	6.19243E−27	**1.8262E−110**	2.00588E−61	5.21099E−11	4.72459E−11	1.53934E−13	0.0000265
Std.	2.81558E−26	6.9001E−110	1.09744E−60	7.47075E−11	3.72695E−11	1.6844E−13	0.000042
f3: Easoms function							
Min	−1	−1	−1	−1	−1	−1	−0.99596
Mean	**−1**	**−1**	**−1**	−0.166696	−0.866667	**−1**	−0.08371
Std.	0	0	0	0.379035	0.345746	2.7768E−13	0.240978
f4: Griewanks function							
Min	2.65431E−08	0	0	1.8846E−12	3.57752E−08	2.22045E−16	0.013878
Mean	5.58634E−06	**0**	**0**	0.307127	0.000619655	1.16063E−13	0.182588
Std.	1.40067E−05	0	0	0.305973	0.00188366	1.7023E−13	0.134813
f5: Rastrigins function							
Min	0	0	0	5.98135E−10	3.6469E−10	4.0145E−13	3.82E−08
Mean	1.6982E−13	**0**	**0**	2.02308	9.2122E−09	0.0663306	0.657481
Std.	5.8989E−13	0	0	2.22454	7.45866E−09	0.252429	0.749738
f6: Rosenbrock’s function							
Min	4.59E−10	8.93E−05	0	5.72513E−08	1.24204E−09	1.58342E−06	0.006559
Mean	7.70E−07	5.74E−04	**3.30336E−31**	0.260185	1.93476E−07	2.40E−03	1.53167
Std.	2.02E−06	5.50E−04	1.76315E−30	0.809771	1.75E−07	7.97E−04	2.3835
f7: Zakharov’s function							
Min	3.50262E−31	1.4817E−109	9.97878E−71	3.66828E−12	6.40746E−13	4.80241E−16	8.1E−09
Mean	1.52229E−26	**3.0781E−104**	4.49272E−44	9.82549E−11	1.3912E−10	5.49697E−13	0.0000589
Std.	4.79979E−26	1.5685E−103	2.46077E−43	7.86452E−11	1.34631E−10	7.621E−13	0.000249
f8: Michalewicz’s function							
Min	−1.8013	−1.8013	−1.8013	−1.98795	−1.8013	−1.8013	−1.80128
Mean	**−1.8013**	**−1.8013**	**−1.8013**	−1.81228	**−1.8013**	**−1.8013**	−1.78277
Std.	9.03362E−16	9.03362E−16	9.03362E−16	0.0519564	7.40769E−11	1.57E−11	0.075241
f9: Dixon and price’s function							
Min	7.26608E−15	3.69779E−32	3.69779E−32	1.40853E−11	2.63813E−11	2.35428E−14	2.84E−08
Mean	1.99551E−11	**3.69779E−32**	**3.69779E−32**	2.88258E−10	6.09103E−10	3.52434E−11	0.000181
Std.	6.00129E−11	0	0	2.84653E−10	5.42748E−10	5.5036E−11	0.000379
f10: Levy’s function							
Min	4.00861E−30	1.49976E−32	1.49976E−32	1.51025E−14	2.24749E−12	4.54259E−16	0.000011
Mean	1.00577E−25	**1.49976E−32**	**1.49976E−32**	1.33682E−11	5.03791E−11	2.18392E−13	0.001214
Std.	2.70703E−25	1.11348E−47	1.11348E−47	1.191E−11	5.03387E−11	3.9346E−13	0.002051
f11: Cross-in-tray function							
Min	-2.06261	-2.06261	-2.06261	-2.06261	-2.06261	-2.05408	-2.06261
Mean	**−2.06261**	**−2.06261**	**−2.06261**	**−2.06261**	**−2.06261**	−2.0376	**−2.06261**
Std.	8.58E−11	9.03E−16	9.03E−16	1.23E−11	1.4E−11	0.00567	0.00239
f12: Drop-wave function							
Min	−1	−1	−1	−1	−1	−0.99996	−0.99998
Mean	**−1**	**−1**	**−1**	−0.93335	**−1**	−0.95722	−0.92983
Std.	4.95E−10	0	0	0.030208	1.61E−09	0.030176	0.034068
f13: Eggholder function							
Min	−959.641	−959.641	−959.641	−959.641	−959.641	−32.806	−956.366

Table 2 (continued)

	FPA	mgFPA	MFPA	BAT	FF	GA	SA
Mean	**−959.641**	**−959.641**	**−959.641**	−831.837	−816.391	−30.9245	−542.529
Std.	5.78E−13	5.78E−13	5.78E−13	105.893	105.934	0.474535	166.754
f14: Holder table function							
Min	−19.2085	−19.2085	−19.2085	−19.2085	−19.2085	−1.72536	−19.2085
Mean	**−19.2085**	**−19.2085**	**−19.2085**	−18.7184	**−19.2085**	−1.66777	**−19.2085**
Std.	0.000000311	5.71E−15	7.81E−15	1.11466	1.44E−09	0.014743	0.000044
f15: Schaffer function N. 2							
Min	0	0	0	1.82E−14	1.1E−12	1.89E−11	0.00961
Mean	**0**	**0**	**0**	0.034009	1.71E−11	0.0000114	0.082308
Std.	0	0	0	0.037992	1.49E−11	0.0000313	0.060543
f16: Shubert function							
Min	−186.731	−186.731	−186.731	−186.731	−186.731	−69.9381	−186.731
Mean	**−186.731**	**−186.731**	**−186.731**	−173.261	**−186.731**	−24.3187	**−186.731**
Std.	0.00000734	2.30E−14	5.28E−15	31.9251	8.55E−08	13.1135	0.000000925
f17: Schwefel function							
Min	2.55E−05	2.54E−05	2.55E−05	2.55E−05	0.2.55E−05	830	2.62E−05
Mean	2.55E−05	**2.54E−05**	2.55E−05	113.897	86.1969	830.075	385.901
Std.	1.5E−09	0	0	86.0515	80.7233	3.93E−13	161.14
f18: Schaffer function N. 4							
Min	0.500091	0.500091	0.500091	0.5	0.500091	0.539118	0.500096
Mean	**0.500091**	**0.500091**	**0.500091**	0.5	**0.500091**	0.539745	0.50011
Std.	3.66E−09	1.17E−10	1.94E−08	0	2.09E−09	0.000243	0.0000121
f19: Beale function							
Min	7.11E−27	0	0	3.28E−12	4.5E−13	5.23E−14	0.00000311
Mean	1.98E−21	**0**	**0**	0.117083	6.15E−11	1.6E−11	0.003245
Std.	5.85E−21	0	0	0.239276	5.94E−11	2E−11	0.008865
f20: Rotated hyper-ellipsoid function							
Min	2.43E−29	1.18E−115	2.69E−67	6.37E−13	2.15E−10	4.41E−15	1.34E−09
Mean	9.74E−26	**5.18E−109**	3.31E−62	7.86E−11	1.18E−08	2.78E−13	0.00000549
Std.	2.79E−25	2.18E−108	1.07E−61	9.8E−11	1.18E−08	4.12E−13	0.0000105
f21: Matyas function							
Min	5.06E−34	3.13E−58	1.14E−68	3.63E−13	7.5E−13	1.63E−14	0.000000161
Mean	6.92E−28	**8.19E−55**	2.08E−51	7.44E−12	1.56E−11	1.53E−12	0.00184
Std.	3.32E−27	1.08E−54	1.02E−50	6.7E−12	1.35E−11	2.37E−12	0.00248
f22: styblinski-tang function							
Min	−78.3323	−78.3323	−78.3323	−78.3323	−78.3323	−78.3323	−78.3323
Mean	**−78.3323**	**−78.3323**	**−78.3323**	−75.505	−78.3219	−58.5409	−78.2412
Std.	1.45E−14	1.44E−14	1.45E−14	5.75136	0.056953	9.53754	0.499163
f23: De Jong function N05							
Min	0.998004	0.998004	0.998004	0.998004	0.998004	12.6705	0.998004
Mean	**0.998004**	**0.998004**	0.**0.998004**	7.5505	1.48794	12.6705	5.67102
Std.	1.3E−14	1.12E−16	1.8E−16	5.72583	0.609329	1.44E−13	5.5792

existing solutions in guiding the search process toward the good promising regions of search space. The performance of the proposed algorithm is evaluated on two optimization problems, numerical benchmark functions, and ANN weight adjustment.

For each optimization problem, the proposed algorithm is judged against six state-of-the-art optimization algorithms. The results of the benchmark problems show that the proposed mgFPA algorithm exhibits better than or equal performance to the original FPA and MFPA in 82.6 and 78.2% of

Table 3 AER obtained for the 23 test functions

Algorithm	FPA	mgFPA	MFPA	BAT	FF	GA	SA
f1	2.07259E−12	**8.88178E−16**	**8.88178E−16**	2.36293	8.17798E−05	1.24595E−06	0.086127
f2	6.19243E−27	**1.8262E−110**	2.00588E−61	5.21099E−11	4.72459E−11	1.53934E−13	0.0000265
f3	**0**	**0**	**0**	0.833304	0.133333	0	0.91629
f4	5.58634E−06	**0**	**0**	0.307127	0.000619655	1.16063E−13	0.182588
f5	1.6982E−13	**0**	**0**	2.02308	9.21223E−09	0.0663306	0.657481
f6	7.70467E−07	5.74E−04	**3.30336E−31**	0.260185	1.93476E−07	0.00240795	1.53167
f7	1.52229E−26	**3.0781E−104**	4.492E−44	9.825E−11	1.391E−10	5.4969E−13	0.0000589
f8	**0**	3.4101E−06	**0**	0.01098	**0**	**0**	0.01853
f9	1.99551E−11	**3.697E−32**	**3.697E−32**	2.88258E−10	6.09103E−10	3.52434E−11	0.000181
f10	1.0057E−25	**1.49976E−32**	**1.49976E−32**	1.33682E−11	5.03791E−11	2.18392E−13	0.001214
f11	**0**	1.87082E−06	**0**	**0**	**0**	0.02501	0.00159
f12	**0**	**0**	**0**	0.06665	**0**	0.04278	0.07017
f13	0.0003	**3.727E−05**	0.0003	127.8037	143.2497	928.7162	417.1117
f14	**0**	2.567E−06	**0**	0.4901	**0**	17.54073	**0**
f15	**0**	**0**	**0**	0.034009	1.71E−11	0.0000114	0.082308
f16	0.0001	**8.831E−06**	0.0001	13.4699	0.0001	162.4122	0.0001
f17	**2.55E−05**	**2.545E−05**	**2.55E−05**	113.897	86.1969	830.075	385.901
f18	**0.207512**	**0.207512**	**0.207512**	0.207421	0.207512	0.247166	0.207531
f19	1.98E−21	**0**	**0**	0.117083	6.15E−11	1.6E−11	0.003245
f20	9.74E−26	**5.187E−109**	3.31E−62	7.86E−11	1.18E−08	2.78E−13	0.00000549
f21	6.92E−28	**8.191E−55**	2.08E−51	7.44E−12	1.56E−11	1.53E−12	0.00184
f22	39.16631	39.16634	39.16631	**36.33901**	39.15591	39.37491	39.07521
f23	**0.998004**	**0.998004**	**0.998004**	7.5505	1.48794	12.6705	5.67102

Table 4 Classification Accuracy of ANN trained by mgFPA and another six algorithms

Dataset	mgFPA	FPA	E-MWO	MWO	HS-BtW	GA	BP
Haberman							
Best	79.0	77.4	83.8	79.0	75.8	77.4	74.1
Mean	73.9	71.7	**78.0**	77.1	69.3	74.1	71.8
Iris							
Best	96.6	96.6	100.0	100.0	96.6	96.6	96.6
Mean	91.1	93.1	91.0	89.6	86.8	84.6	**96.6**
Cancer							
Best	98.5	98.5	98.5	98.5	99.2	99.2	97.8
Mean	97.1	96.9	97.1	97.3	**98.2**	97.4	96.1
Diabetes							
Best	77.9	81.1	92.8	79	77.9	79.2	79.2
Mean	75.5	75.5	**78.0**	74.5	75.3	73.8	75.4
Glass							
Best	**100.0**	66.6	95.3	60.4	72.0	62.7	72.0
Mean	**61.1**	59.2	58.7	49.1	58.8	45.2	60.1

the total experiment cases, respectively, and in 95.6% of the compared cases of other competitive algorithms. Moreover, the results of ANN weight adjustment show that mgFPA yields better or equal performance to FPA in 80% of the total experiment cases. Hence, the proposed mgFPA is very competitive to the other state-of-the-art algorithms.

Future work will be focused to verify the performance of the proposed mgFPA on other real-world optimization problems, such as timetabling problems, clustering and image processing.

References

1. Holland JH (1992) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press, Cambridge
2. Kennedy J (2011) Particle swarm optimization. In: Encyclopedia of machine learning. Springer, New York, pp 760–766
3. Kirkpatrick S, Gelatt CD, Vecchi MP et al (1983) Optimization by simulated annealing. Science 220(4598):671–680
4. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm. J Global Optim 39(3):459–471
5. Yang X-S, Deb S (2009) Cuckoo search via lévy flights. In: Nature & Biologically Inspired Computing. NaBIC 2009. World Congress on, IEEE, pp 210–214
6. Geem ZW, Kim JH, Loganathan G (2001) A new heuristic optimization algorithm: harmony search. Simulation 76(2):60–68
7. Yang X-S (2010) Firefly algorithm, stochastic test functions and design optimisation. Int J Bio Inspir Comput 2(2):78–84
8. Yang X-S (2010) A new metaheuristic bat-inspired algorithm, Nature inspired cooperative strategies for optimization (NICSO 2010), pp 65–74
9. Yang X-S (2012) Flower pollination algorithm for global optimization. In: UCNC. Springer, New York, pp 240–249
10. Wang R, Zhou Y, Zhao C, Wu H (2015) A hybrid flower pollination algorithm based modified randomized location for multi-threshold medical image segmentation. Bio Med Mater Eng 26(s1):S1345–S1351
11. Ouadfel S, Taleb-Ahmed A (2016) Social spiders optimization and flower pollination algorithm for multilevel image thresholding: a performance study. Expert Syst Appl 55:566–584
12. Rodrigues D, Yang X-S, De Souza AN, Papa JP (2015) Binary flower pollination algorithm and its application to feature selection. In: Recent advances in swarm intelligence and evolutionary computation. Springer, New York, pp 85–100
13. Sayed SA-F, Nabil E, Badr A (2016) A binary clonal flower pollination algorithm for feature selection. Pattern Recogn Lett 77:21–27
14. Nabil E (2016) A modified flower pollination algorithm for global optimization. Expert Syst Appl 57:192–203
15. Souza ROG, Oliveira ES, Junior ICS, Marcato ALM, de Olveira MT (2017) Flower pollination algorithm applied to the economic dispatch problem with multiple fuels and valve point effect. In: Portuguese Conference on Artificial Intelligence. Springer, New York, pp 260–270
16. Shilaja C, Ravi K (2017) Optimization of emission/economic dispatch using euclidean affine flower pollination algorithm (efpa) and binary fpa (bfpa) in solar photo voltaic generation. Renew Energy 107:550–566
17. Abdelaziz A, Ali E, Elazim SA (2016) Flower pollination algorithm to solve combined economic and emission dispatch problems. Eng Sci Technol Int J 19(2):980–990
18. Abdel-Baset M, Wu H, Zhou Y (2017) A complex encoding flower pollination algorithm for constrained engineering optimisation problems. Int J Math Model Numer Optim 8(2):108–126
19. Abdel-Raouf O, Abdel-Baset M et al (2014) A new hybrid flower pollination algorithm for solving constrained global optimization problems. Int J Appl Oper Res Open Access J 4(2):1–13
20. Goyal S, Patterh MS (2015) Flower pollination algorithm based localization of wireless sensor network. In: Recent Advances in Engineering & Computational Sciences (RAECS), 2015 2nd International Conference on. IEEE, pp 1–5
21. Kaur R, Arora S (2018) Nature inspired range based wireless sensor node localization algorithms. Int J Interact Multimed Artif Intell 4(Regular Issue)
22. Kayabekir AE, Bekdaş G, Nigdeli SM, Yang X-S (2018) A comprehensive review of the flower pollination algorithm for solving engineering problems. In: Nature-Inspired Algorithms and Applied Optimization. Springer, New York, pp 171–188
23. Alyasseri ZAA, Khader AT, Al-Betar MA, Awadallah MA, Yang X-S (2018) Variants of the flower pollination algorithm: a review:91–118
24. Pant S, Kumar A, Ram M (2017) Flower pollination algorithm development: a state of art review. Int J Syst Assur Eng Manag 8(2):1858–1866
25. Patnaik S, Yang X, Nakamatsu K (2017) Nature-inspired computing and optimization: theory and applications, modeling and optimization in science and technologies. Springer International Publishing, New York. https://books.google.com.sa/books?id=dFVKDgAAQBAJ
26. Abdel-Basset M, Shawky LA, Sangaiah AK (2017) A comparative study of cuckoo search and flower pollination algorithm on solving global optimization problems. Library Hi Tech 35(4):595–608
27. Awad N, Ali M, Liang J, Qu B, Suganthan P (2016) Problem definitions and evaluation criteria for the cec 2017 special session and competition on single objective bound constrained real-parameter numerical optimization. In: Technical Report. NTU, Singapore
28. Zhou Y, Wang R, Luo Q (2016) Elite opposition-based flower pollination algorithm. Neurocomputing 188:294–310
29. Tang K, Yáo X, Suganthan PN, MacNish C, Chen Y-P, Chen C-M, Yang Z (2007) Benchmark functions for the cec2008 special session and competition on large scale global optimization. In: Nature Inspired Computation and Applications Laboratory. USTC, p 24
30. Salgotra R, Singh U (2017) Application of mutation operators to flower pollination algorithm. Expert Syst Appl 79:112–129
31. Xu S, Wang Y, Huang F (2017) Optimization of multi-pass turning parameters through an improved flower pollination algorithm. Int J Adv Manuf Technol 89(1–4):503–514
32. Deb K (2000) An efficient constraint handling method for genetic algorithms. Comput Methods Appl Mech Eng 186(2–4):311–338
33. Otsu N (1979) A threshold selection method from gray-level histograms. IEEE Trans Syst Man Cybern 9(1):62–66
34. Abdel-Baset M, Hezam IM (2015) An effective hybrid flower pollination and genetic algorithm for constrained optimization problems. Adv Eng Technol Appl Int J 4:27–27
35. Lazim D, Zain AM, Bahari M, Omar AH (2017) Review of modified and hybrid flower pollination algorithms for solving optimization problems. Artif Intell Rev:1–31
36. Shambour MKY (2017) Dynamic search zones (dsz) for harmony search algorithm. In: 2017 8th International Conference on Information Technology (ICIT), pp 941–946
37. Shambour MKY (2018) Vibrant search mechanism (vsm) for numerical optimization functions. J Inf Commun Technol (JICT) 17:21
38. Abualigah LM, Khader AT, Hanandeh ES (2017) A new feature selection method to improve the document clustering using particle swarm optimization algorithm. J Comput Sci:11
39. Abualigah LM, Khader AT (2017) Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering. J Supercomput 73(11):4773–4795
40. Jamil M, Yang X-S (2013) A literature survey of benchmark functions for global optimisation problems. Int J Math Model Numer Optim 4(2):150–194
41. (2017) Virtual library of simulation experiments: test functions and datasets. https://www.sfu.ca
42. Ghosh-Dastidar S, Adeli H (2009) A new supervised learning algorithm for multiple spiking neural networks with application in epilepsy and seizure detection. Neural Netw 22(10):1419–1431
43. Bache K, Lichman M (2018) Uci machine learning repository

44. Abusnaina AA, Abdullah R, Kattan A (2014) Enhanced MWO training algorithm to improve classification accuracy of artificial neural networks. Springer International Publishing, New York, pp 183–194
45. Abusnaina AA, Abdullah R (2013) Mussels wandering optimization algorithm based training of artificial neural networks for pattern classification. In: Proceedings of the 4th International Conference on Computing and Informatics. ICOCI, pp 78–85
46. Gómez D, Rojas A (2016) An empirical overview of the no free lunch theorem and its effect on real-world machine learning classification. Neural Comput 28(1):216–228