Hindawi
International Journal of Antennas and Propagation
Volume 2017, Article ID 7158752, 11 pages
https://doi.org/10.1155/2017/7158752

Research Article

Pattern Synthesis of Linear Antenna Arrays Using Enhanced Flower Pollination Algorithm

Urvinder Singh and Rohit Salgotra

Department of ECE, Thapar University, Patiala, India

Correspondence should be addressed to Rohit Salgotra; cassonova.00@gmail.com

Received 18 October 2016; Revised 9 December 2016; Accepted 18 January 2017; Published 20 February 2017

Academic Editor: Shih Yuan Chen

In this paper, a new variant of flower pollination algorithm (FPA), namely, enhanced flower pollination algorithm (EFPA), has been proposed for the pattern synthesis of nonuniform linear antenna arrays (LAA). The proposed algorithm uses the concept of Cauchy mutation in global pollination and enhanced local search to improve the exploration and exploitation tendencies of FPA. It also uses dynamic switching to control the rate of exploration and exploitation. The algorithm is tested on standard benchmark problems and has been compared statistically with state of the art to prove its worthiness. LAA design is a tricky and difficult electromagnetic problem. Hence to check the efficacy of the proposed algorithm it has been used for synthesis of four different LAA with different sizes. Experimental results show that EFPA algorithm provides enhanced performance in terms of side lobe suppression and null control compared to FPA and other popular algorithms.

1. Introduction

Antenna arrays find their application in number of wireless applications such as radar, sonar, mobile, TV, and satellite. Antenna arrays are favored as these have high directivity, reduce power consumption, increase spectral efficiency and also have beam steering capability [1]. The design of antenna arrays with desired radiation pattern has found great interest in the literature. Antenna arrays design is intricate and nonlinear problem. Hence number of optimization techniques such as genetic algorithm (GA), differential algorithm (DE), particle swarm optimization (PSO), biogeography based optimization (BBO) and many others [2–21] have been used to synthesize these. It is required that antenna arrays radiate in desired directions so that these do no add to electromagnetic pollution. This can be achieved if the energy is maximum in the main lobe and minimum in side lobes. Moreover, to avoid interference from undesired directions and to circumvent jamming, null placement in radiation pattern has also gained importance. So, overall the main issues in designing the antenna arrays are reducing the side lobe level (SLL) and placing nulls in the desired directions of radiation pattern.

Linear antenna arrays (LAA) consist of number of antenna elements arranged in a straight line. LAA have become very popular because of their simple geometry and applications. LAA design has been investigated by numerous researchers using several optimization algorithms in the past. Recioui has used PSO for SLL reduction of LAA [2]. Khodier and Christodoulou have synthesized LAA for maximum SLL reduction and null placement. They have designed three arrays for different sizes and showed that PSO gives better results than uniform and quadrature programming method (QPM) [3]. Rattan et al. have applied GA [7] for the optimization of LAA. Their results were better than those obtained using PSO [3]. DE has been employed by Lin et al. to design unequally spaced LAA [8]. They have discussed about the impact of angular resolution on the final results. Dib et al. have compared the performance of self-adaptive DE (SADE) and Tagachi's method for optimization of LAA [9]. Cengiz and Tokat have used GA, memetic algorithm (MA) and tabu search (TS) to optimize three different LAA [10]. Singh et al. [11] have applied BBO to synthesize uniform and nonuniform LAA and found that BBO is better than PSO and other methods for the design of arrays. BBO has also been used by Sharaqa and Dib to design LAA [12]. Ant colony optimization has been used by Rajo-lglesias and Quevedo-Teruel for the synthesis of LAA

for SLL reduction and null placement [13]. A multiobjective approach using multiobjective DE (MOEA/D-DE) has been employed by Pal et al. to optimize LAA [14]. The authors have noted that MOEA/D-DE gives better trade-off curves between null placement and SLL. LAA synthesis using fitness adaptive differential evolution algorithm (fiADE) has been done by Chowdhury et al. [15]. Guney and Onay have utilized harmony search algorithm (HSA) for SLL minimization and null placement in different directions of radiation pattern of LAA [16]. A new algorithm based on the breeding strategy of cuckoos known as cuckoo optimization algorithm (COA) has been employed to optimize three different nonuniform LAA [17]. The authors have compared the results with the popular algorithms like DE, PSO, firefly algorithm (FA) and found that the results provided by COA in terms of SLL reduction and null placement are better than competitive algorithms. Moreover, convergence curves of different algorithms are compared and it is concluded that COA provides faster convergence than the other methods. Guney and Durmus have used back scattering algorithm (BSA) for the pattern nulling of LAA [18]. Khodier has used cuckoo search (CS) to optimize antenna arrays [19]. An enhanced version of PSO named as comprehensive learning PSO (CLPSO) has been used to design three different LAA [20]. Recently, a novel algorithm which mimics the behaviour of flowers known as flower pollination algorithm (FPA) has been used to synthesize LAA [21]. Though number of algorithms have been proposed for synthesis of LAA, these suffer from certain shortcomings like getting stuck in local minima, slow convergence speeds and require precise parameter tuning. So, in this work, the authors propose an enhanced version of FPA known as enhanced FPA (EFPA). The newly proposed algorithm uses the concept of Cauchy distribution to follow large steps in global pollination, enhanced local search and dynamic switch probability to control the rate of local and global pollination. It has better exploration and exploitation capability and is also less likely to stuck in local minima.

The rest of the paper is organized as follows: Section 2 gives details about the basic FPA algorithm, Section 3 proposes a new EFPA algorithm, Section 4 gives result and discussion, and Section 5 provides an extensive conclusion.

2. Flower Pollination Algorithm

Flowers are the most fascinating plant species and have been dominating earth from the cretaceous period, partly from about 125 million years. Almost 80% of plant species are flowering and it has been made possible by the process of pollination [22]. Pollination in flowers, refer to transfer of pollen grains from one flower to another via pollinators such as wind, diffusion, bees, birds, bats and other animals [23]. If the pollination process is through insects, it is called biotic pollination and if it takes place via wind or diffusion, it is called abiotic pollination. Overall, 200,000 varieties of flower pollinators exist in nature. Pollinators such as honey bee has been found to show some specific phenomenon known as flower constancy. This phenomenon helps them to visit only specific flower species and by pass others, hence maximizing the transfer of pollens from a particular plant species. This helps pollinators in finding better food sources and minimize the exploration or learning cost [24]. Pollination can further be classified based upon the pollens of same or different flowers. If a pollinator transfers pollens from one flower to another, it is called cross-pollination while if it does for the same species, it is self-pollination.

Based upon the phenomenon of pollination, flower constancy and behaviour of pollinators, flower pollination algorithm (FPA) was proposed by Yang [25]. He proposed four sets of rules for governing this algorithm as follows:

(1) For long distances, the pollination used is biotic cross-pollination; the process is called global pollination and is performed via Levy flights.

(2) Abiotic self-pollination process forms the basis of local pollination.

(3) The reproduction probability of different flowers involved in pollination is considered as flower constancy.

(4) A switch probability $p \in [0, 1]$ is defined for controlling local and global pollination.

For simplicity, a single flower producing only one pollen gamete is considered. This means a single solution for problem under test is equivalent to a pollen gamete or a flower. From the rules above, two key features of global and local pollination have been used to formulate FPA.

The first step in FPA is global pollination, where pollinators such as insects carry pollen to long distances. This process helps in pollination of the best fit flower or solution so far and is represented as R_*. Rule 1 with a combination of flower constancy is represented as

$$x_i^{t+1} = x_i^t + L(\lambda)\left(R_* - x_i^t\right), \tag{1}$$

where x_i^t is the ith solution of the problem in the tth iteration, L is the step size, and Levy flight is generally used to mimic this phenomenon [25]. Levy flight is drawn from a Levy distribution as

$$L \sim \left\{ \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi} \frac{1}{s^{1+\lambda}}, \quad \left(s \gg s_0 > 0\right), \right. \tag{2}$$

where $\Gamma(\lambda)$ is the standard gamma function with a step size of $s > 0$.

In the second step, Rule 2 is used along with flower constancy to represent local pollination as

$$x_i^{t+1} = x_i^t + \epsilon\left(x_j^t - x_k^t\right), \tag{3}$$

where x_j^t and x_k^t are pollen gametes of same plants but different flowers and ϵ is uniformly distributed in $[0, 1]$. This phase mimics glower constancy at a local scale or in a limited search space.

So, we can say that global and local pollination carry out pollination activities at large and small scale, respectively. In practice, we use a switching probability based on Rule 4 to define the extent of local and global pollination. In the next section, a new enhanced version of flower pollination algorithm is proposed.

3. Enhanced Flower Pollination Algorithm

In the recent past, a large number of researchers have focused on enhancing the basic capabilities of FPA. The algorithm

Start:
Initialize n random flowers
Define switch probability, $p \in [0, 1]$
Define d-dimensional objective function fro problem under test
Find current best solution R_* from the initial population
while iterations < maximum number of iterations
for $i = 1 : n$
if rand < p
Draw a d-dimensional Cauchy step vector obeying Cauchy distribution
Global Pollination: $x_i^{t+1} = x_i^t + C(\delta)(R_* - x_i^t)$
else
Local Pollination: $x_i^{t+1} = x_i^t + a(R_* - x_i^t) + b(x_j^t - x_k^t)$
end if
evaluate x_i^{t+1}
if x_i^{t+1} is better than x_i^t, update
end if
Update p by: $p = p - \dfrac{\text{maxiter} - t}{\text{maxiter}} \times (0.1)$
end for
Update the final best solution.
end while
end

Pseudocode 1: Pseudocode of EFPA.

due to its linear nature, make it suitable for deeper analysis. But it has been proved qualitatively and quantitatively in [26] that the FPA algorithm has a very limited scope for optimization problems at hand. Also, the performance of FPA has not been analyzed to a deeper level and the algorithm is still to prove its worthiness for becoming a state-of-the-art algorithm. Keeping in view the above analysis, a new version of FPA namely EFPA has been proposed. The newly proposed EFPA aims at providing three different modifications to the basic FPA. These include Cauchy based global pollination, enhanced local pollination based upon experience of current best flower pollinator as well as local flowers in proximity, and, thirdly, using dynamic switching probabilities.

(I) Cauchy Based Global Pollination. In the global pollination phase, instead of using a standard Levy distribution, a Cauchy based operator $C(\delta)$ is used. This operator is basically a Cauchy random variable with distribution given by

$$\delta = \frac{1}{2} + \frac{1}{\pi}\arctan\left(\frac{x}{g}\right). \tag{4}$$

The Cauchy density function is given by

$$f_{\text{Cauchy}(0,g)}(x) = \frac{1}{\pi}\frac{g}{g^2 + x^2}. \tag{5}$$

The general equation of global pollination becomes

$$x_i^{t+1} = x_i^t + C(\delta)\left(R_* - x_i^t\right), \tag{6}$$

where g is a scale parameter and its value is generally set to 1. The use of Cauchy operator allows for larger mutation by searching the search space at a faster pace and further accounts for avoiding premature convergence.

(II) Enhanced Local Pollination. The second modification is added in the local pollination phase. Here based upon the experience of local and current best pollinators, the position of new pollinators is updated. Further, if the fitness of new position is greater than the old one, each pollinator updates its position with respect to the previous one. The general equation is given by

$$x_i^{t+1} = x_i^t + a\left(R_* - x_i^t\right) + b\left(x_j^t - x_k^t\right), \tag{7}$$

where a and b are uniformly distributed random numbers in the range of $[0, 1]$. Also, the solutions x_j^t and x_k^t, corresponds to jth and kth flower pollinator in the group with $j \neq k$. This phase enhances the local search capabilities of FPA algorithm.

(III) Dynamic Switch Probability. In FPA, local and global pollinations are controlled by switching probability $p \in [0, 1]$. For a standard state-of-the-art algorithm, it is imperative to follow more global search at the start and as the algorithm progresses more intensive local search is followed. Using this basic concept, the value of switching probability is selected dynamically. The switch probability is updated by following a general formula as

$$p = p - \frac{\text{maxiter} - t}{\text{maxiter}} \times (0.1). \tag{8}$$

The above general equation decreases the value of p linearly with iterations and hence adds to intensive global search at the beginning and local search towards the end. Here maxiter corresponds to the maximum number of iterations and t is the current iteration. The pseudocode for the EFPA is given in Pseudocode 1.

Table 1: Description of test functions.

Test problems	Objective function	Search range	Optimum value	D
Sphere function	$f_1(x) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]$	0	10
Weierstrass function	$f_2(x) = \sum_{i=1}^{D} \sum_{k=0}^{k\max} \left[a^k \cos\left(2\pi b^k (x_i + 0.5)\right)\right] - D \sum_{k=0}^{k\max} \left[a^k \cos(2\pi b^k \cdot 0.5)\right]$; where $a = 0.5,\ b = 3,\ k\max = 20$	$[-0.5, 0.5]$	0	30
Schaffer function	$f_3(x) = \left[\frac{1}{n-1}\sqrt{s_i} \cdot \left(\sin\left(50.0 s_i^{1/5}\right) + 1\right)\right]^2 \quad s_i = \sqrt{x_i^2 + x_{i+1}^2}$	$[-100, 100]$	0	30
Griewank	$f_4(x) = \frac{1}{4000}\sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]$	0	30
Penalized 1 function	$f_5(x) = \frac{\pi}{n}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2\left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$; $y_i = 1 + \frac{x_{i+1}}{4}$; $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	$[-50, 50]$	0	30
Ackley function	$f_6(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e$	$[-100, 100]$	0	30
Rosenbrock function	$f_7 = \sum_{i=1}^{N-1}\left(100\left(x_{i+1} - x_i^2\right)2 + (x_i - 1)2\right)$	$[-100, 100]$	0	30

4. Results and Discussion

4.1. Results of Benchmark Functions. The performance of EFPA is evaluated by applying it to two set of optimization problems. One set includes test functions and other is the real-world application of antenna array design. The algorithm is also compared with other states of the art to prove its competitiveness. The simulations are performed on Intel Core i3 personal computer with Windows 10 operating system using MATLAB version 7.10.0 (R2010a). EFPA is tested on seven well-known benchmark problems [27] as shown in Table 1 with search range and optimal solution. For comparison purposes, artificial bee colony (ABC) [28], DE [29], bat algorithm (BA) [30] and FPA [25] are used. The maximum number of function evaluations for each algorithm is set to 30×1000, with 30 as the population size and 1000 as the maximum number of iterations. The parameter settings of all algorithms are given in Table 2. All the algorithms are run 50 times and results in terms of best, worst, mean and standard deviation are reported. Since because of the stochastic nature of algorithm, Wilcoxon rank-sum test has also been performed to significantly prove the better performance of EFPA. The performance in terms of time-complexity has also been shown in Table 5 to prove its competitiveness. Time-complexity is basically taken for each of the algorithms in the benchmark problems and it has been calculated for one run for each algorithm.

The comparison of comparison for various test functions is given in Table 3. The best and worst values give the most optimal and least optimal solution among the 50 solutions from the maximum number of runs. Mean gives the average values of results and is used when the best and worst values do not give comparable results. But mean may give same results for widely different results. So, standard deviation is also used to calculate results. For functions f_1, f_4, f_5, f_6, and f_7 it can be seen that EFPA performs much better than other algorithms except for f_2 where ABC gives better results. For function f_3 ABC and EFPA gives same results. Further statistical analysis from the Table 4 verify the results. In Table 5, time-complexity for each algorithm is shown. Here time-complexity show that EFPA takes comparatively less time for solving the benchmark problem for most of the benchmark functions in comparison to FPA, BA and ABC. For DE, only four functions give better time-complexity while, for the rest three, again EFPA is better. Overall, we can say that EFPA is superior to ABC, DE, FPA, and BA. In the next subsection, EFPA is applied for synthesis of LAA.

4.2. Results of LAA. A symmetrical LAA consists of $2N$ elements with N elements on both sides of the origin and

Table 2: Parameter settings of various algorithms.

Algorithm	Parameters	Values
DE	F	1.5
	CR	0.8
	Population size	30
	Maximum iterations	1000
	Stopping criteria	Max iteration
ABC	Colony size	30
	Number of food sources	Colony size/2
	Limit	100
	Maximum cycles	1000
	Stopping criteria	Max iteration
FPA	Population size	30
	Probability switch	0.6
	Maximum iterations	1000
	Stopping criteria	Max iteration
BA	Population size	50
	Loudness	0.5
	Pulse rate	0.5
	Maximum number of iterations	200
	Stopping criteria	Max iteration
EFPA	Population size	30
	Probability switch	Dynamic and linearly decreasing
	Maximum iterations	1000
	Stopping criteria	Max iteration

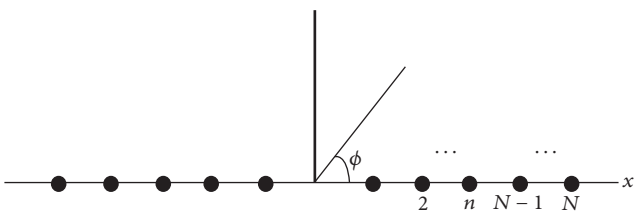

Figure 1: Geometry of a linear antenna array.

arranged in a straight line as shown in Figure 1. The nth element is excited with a current of amplitude I_n and phase φ_n. The position of nth element is represented by x_n. The array factor (AF) of such an antenna is given by [1]

$$\mathrm{AF}(\emptyset) = 2\sum_{n=1}^{N} I_n \cos\left[kx_n \cos(\emptyset) + \varphi_n\right]. \quad (9)$$

Here k is the wave number and is given by $2\pi/\lambda$. The symmetric nature of the array reduces the computational cost as only N elements are needed to be optimized. Moreover, these symmetric arrays give symmetric radiation pattern which is desired for number of applications.

The AF for LAA in (9) is a function of excitation currents (both I_n and φ_n) and positions x_n. If current excitations are held uniform (i.e., $I_n = 1$ and $\varphi_n = 0^\circ$), then the LAA is function of x_n only and array is called nonuniform LAA. Hence for nonuniform LAA, AF reduces to

$$\mathrm{AF}(\phi) = 2\sum_{n=1}^{N} \cos\left[kx_n \cos(\phi)\right]. \quad (10)$$

The design of nonuniform LAA has gained importance in the past and several researchers have synthesized it using different optimization techniques [3, 7, 9, 11–21]. The array factor of a nonuniform array is aperiodic by virtue of nonuniform spacing between the elements. This aperiodic nature of arrays helps in obtaining lower SLL with lesser number of elements for a given antenna size. Furthermore, the uniform element excitations help in reducing the feed network cost and complexity. However, the relationship between the array factor and element spacing is nonconvex and nonlinear. So, this makes the design of nonuniform arrays difficult.

For directing antenna arrays energy in a specific direction, it is desired that its radiation pattern should have low side lobes. Moreover, nulls are also required in the direction of interference. The desired radiation pattern can be achieved by suitably adjusting the positions of antenna elements for nonuniform LAA. The adjustment of positions is a tricky and nonlinear problem. So, optimization algorithms are well suited for this type of problems. For applying optimization to antenna problem, it is imperative to define a fitness function which is given as

$$\begin{aligned}\mathrm{fit} = \mathrm{SLL} + \alpha * \max\left\{0, \left|\mathrm{BW} - \mathrm{BW}_d\right| - 1\right\} + \alpha \\ * \left\{\sum_{k=1}^{K} \max\left\{0, \mathrm{AF}_{\mathrm{dB}}(\phi_k) - \mathrm{Nu}_{\mathrm{dB}}\right\}\right\}.\end{aligned} \quad (11)$$

In the above fitness function, SLL is the side lobe level, $\mathrm{AF}_{\mathrm{dB}}$ is the array factor in decibels, BW and BW_d are the calculated and desired beamwidth of the main lobe, ϕ_k gives the direction of the kth null, $\mathrm{Nu}_{\mathrm{dB}}$ is the required null depth in dB, K is the number of the required nulls, and α is a very large number. The fitness function given in (11) uses a penalty method which penalizes any antenna design which does not meet the required constraints of beamwidth and nulls (if desired). This is achieved by taking value α to be very large. The BW here is obtained computationally from the radiation pattern data.

4.2.1. Linear Antenna Optimization. This subsection presents the application of EFPA to optimization of nonuniform LAA. In the first example, a 12-element nonuniform antenna is taken for optimization. The goal of optimization is to minimize the SLL in the region $\phi \in [98^\circ, 180^\circ]$. Because of symmetry there are only six element positions which are to be optimized. The number of generations for EFPA are taken as 1000. The population size taken as 20 in order to make fair comparison between the results of FPA [21] and EFPA. The total number of function evaluations are $20 * 1000 = 2000$ for both the FPA [21] and EFPA. As the aim of the optimization is to suppress the SLL only, so the value of α is taken as 0 in the fitness equation of (11). The EFPA algorithm is run for 30 times and best results are listed in this work.

Table 3: Comparison of results for various test functions.

Objective function	Algorithm	Best	Worst	Mean	Standard deviation
$f_1(x)$	DE	$4.43E+03$	$2.97E+04$	$1.54E+04$	$6.22E+03$
	ABC	$2.74E+03$	$3.67E+04$	$2.74E+04$	$5.93E+05$
	BA	$8.08E+03$	$3.54E+04$	$1.96E+04$	$7.01E+03$
	FPA	$1.00E+02$	$1.00E+03$	$3.61E+02$	$1.66E+02$
	EFPA	$3.35E-15$	$4.72E-07$	$1.82E-08$	$6.85E-08$
$f_2(x)$	DE	$1.64E+01$	$3.82E+01$	$2.54E+01$	$4.51E+00$
	ABC	$2.53E+00$	$5.90E+00$	$3.90E+00$	$8.40E-01$
	BA	$2.51E+01$	$4.30E+01$	$3.36E+01$	$4.24E+00$
	FPA	$3.83E+01$	$5.72E+01$	$5.12E+01$	$4.64E+00$
	EFPA	$1.34E+01$	$2.66E+01$	$2.00E+01$	$3.10E+00$
$f_3(x)$	DE	$7.95E-05$	$2.29E-01$	$4.64E-02$	$5.37E-02$
	ABC	$0.00E-00$	$0.00E-00$	$0.00E-00$	$0.00E-00$
	BA	$3.06E-14$	$3.47E-01$	$8.95E-02$	$9.22E-02$
	FPA	$0.00E-00$	$1.98E-12$	$1.03E-13$	$3.39E-13$
	EFPA	$0.00E-00$	$0.00E-00$	$0.00E-00$	$0.00E-00$
$f_4(x)$	DE	$4.89E+01$	$3.68E+02$	$1.54E+02$	$7.19E+01$
	ABC	$1.06E+00$	$1.76E+00$	$1.25E+00$	$1.73E-01$
	BA	$2.94E+00$	$1.18E+01$	$5.77E+00$	$1.63E+00$
	FPA	$5.71E-01$	$1.04E+00$	$9.25E-01$	$9.76E-02$
	EFPA	$0.00E-00$	$1.45E-01$	$2.90E-03$	$2.06E-02$
$f_5(x)$	DE	$1.78E+06$	$6.92E+08$	$2.24E+08$	$2.27E+08$
	ABC	$3.49E-02$	$2.77E+07$	$1.07E+08$	$8.91E+07$
	BA	$3.40E+05$	$5.55E+07$	$1.58E+07$	$1.53E+07$
	FPA	$2.81E+00$	$1.11E+01$	$6.81E+00$	$1.87E+00$
	EFPA	$1.25E+00$	$3.1E+01$	$8.93E+00$	$6.40E+00$
$f_6(x)$	DE	$2.00E+01$	$2.00E+01$	$2.00E+01$	$0.00E+00$
	ABC	$2.00E+01$	$2.02E+01$	$2.00E+01$	$5.30E-03$
	BA	$1.97E+01$	$2.00E+01$	$1.99E+01$	$2.92E-02$
	FPA	$2.05E+01$	$2.11E+01$	$2.10E+01$	$8.37E-02$
	EFPA	$7.29E+00$	$1.33E+01$	$9.95E+00$	$1.29E+00$
$f_7(x)$	DE	$4.21E+08$	$3.78E+10$	$8.63E+09$	$1.00E+10$
	ABC	$2.18E+04$	$3.61E+05$	$2.96E+04$	$5.39E+04$
	BA	$8.62E+08$	$1.29E+10$	$3.07E+09$	$2.10E+09$
	FPA	$1.48E+05$	$6.49E+06$	$2.30E+06$	$1.72E+06$
	EFPA	$0.00E-00$	$0.00E-00$	$0.00E-00$	$0.00E-00$

Table 4: p-test values of various algorithms.

Objective function	BA	FPA	ABC	DE	EFPA
$f_1(x)$	$7.06E-18$	$7.06E-18$	$7.06E-18$	$7.06E-18$	NA
$f_2(x)$	$7.06E-18$	$7.06E-18$	NA	$3.25E-09$	$7.06E-18$
$f_3(x)$	$3.31E-20$	$6.89E-17$	NA	$3.31E-20$	NA
$f_4(x)$	$4.73E-20$	$4.73E-20$	$4.73E-20$	$4.73E-20$	NA
$f_5(x)$	$7.06E-18$	$7.70E-02$	$4.21E-04$	$7.06E-18$	NA
$f_6(x)$	$7.06E-18$	$7.06E-18$	$7.06E-18$	$3.31E-20$	NA
$f_7(x)$	$3.31E-20$	$1.06E-17$	$3.31E-20$	$3.31E-20$	NA

Table 5: Time-complexity comparison for various algorithms (time taken by each algorithm for single run in seconds).

Function	DE	ABC	BA	FPA	EFPA
$f_1(x)$	1.284464	2.527247	1.190053	2.328157	1.965927
$f_2(x)$	15.98023	15.62962	19.02841	15.22211	14.80168
$f_3(x)$	1.185347	2.470673	1.047533	2.183295	1.900515
$f_4(x)$	3.022755	5.167423	4.165333	2.183153	2.030502
$f_5(x)$	2.848487	5.988428	2.974728	3.755994	3.498411
$f_6(x)$	1.667705	3.186360	2.261759	2.346810	2.383599
$f_7(x)$	1.693011	3.265792	2.087382	3.384049	2.999886

Table 6: Geometry of 12-element LAA (with respect to $\lambda/2$) obtained using EFPA.

Element	1	2	3	4	5	6
Position	0.3765	0.9734	1.7376	2.5051	3.4827	4.6744

The optimum positions obtained using EFPA are shown in Table 6. The performance of EFPA is contrasted against other popular algorithms in terms of SLL in Table 7. The minimum SLL obtained by EFPA is −21.07 dB and is better than GA [10], FiADE [15], TS [10], MA [10], FPA [21] and PSO [15]. The comparison of convergence curve between EFPA and FPA is shown in Figure 2 which clearly shows faster convergence of EFPA than the FPA [21]. The FPA converged in about 700 iterations whereas EFPA took only 225 iterations to converge to the final solution. This is due to enhanced exploration due to Cauchy operator and better exploitation in EFPA. The radiation pattern of EFPA antenna array is shown in Figure 3. For comparison, the radiation patterns of GA [10], FiADE [15], and PSO [15] are also shown in Figure 3.

In the next example, a 22-element nonuniform LAA is optimized for reducing SLL in the region $\phi \in [98^\circ, 180^\circ]$ and also imposing a null in the direction of 99°. For achieving the required pattern, the value of $\mathrm{Nu}_{\mathrm{dB}}$ is set to −60 dB. It is established fact that as SLL is decreased the width of the main lobe in radiation pattern increases. So, for this design, the desired beamwidth BW_d is taken as 18°. The value of α is taken as 10^6 in fitness function. The optimum positions are shown in Table 8. The results are compared in Table 9 which clearly indicate better performance of EFPA as against BSA [18], MOEA/DE [14], HSA [16], PSO [15], GA [10], MA [10], TS [10], and FPA [21]. The SLL of EFPA antenna array is −26.31 dB which is much lower than that obtained using other algorithms. Furthermore, the null in the direction of 99° is −75.58 dB which is less than the desired value of −60 dB and deeper than MOEA/DE [14], GA [7], PSO [3] and MA [10]. In order to compare the performance of EFPA and FPA in terms of convergence speed, FPA is also run for the same objective with same population size. The convergence graphs of EFPA and FPA are plotted in the Figure 4. The convergence characteristics clearly substantiates the superior performance of EPFA in terms of faster convergence speed. The radiation plots of EFPA, HSA [16], BSA [18], and MOEA/DE [14] optimized arrays are shown in Figure 5.

In order to show the capability of EFPA in synthesizing the radiation pattern with multiple nulls, a 28-element nonuniform LAA is optimized for SLL reduction in the region $\phi \in [100^\circ, 180^\circ]$ and having nulls in the directions of 120°, 122.5°, and 125°. The desired beamwidth is set to the value of 8.35° with tolerance of ±1°. The optimum geometry

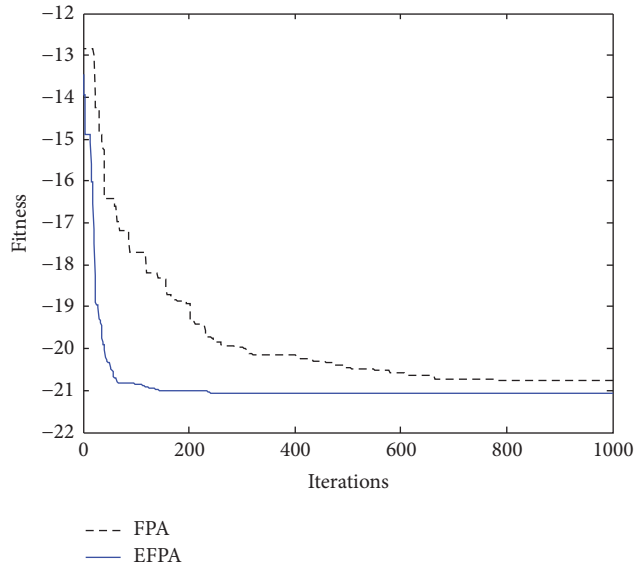

Figure 2: Convergence plots of 12-element LAA.

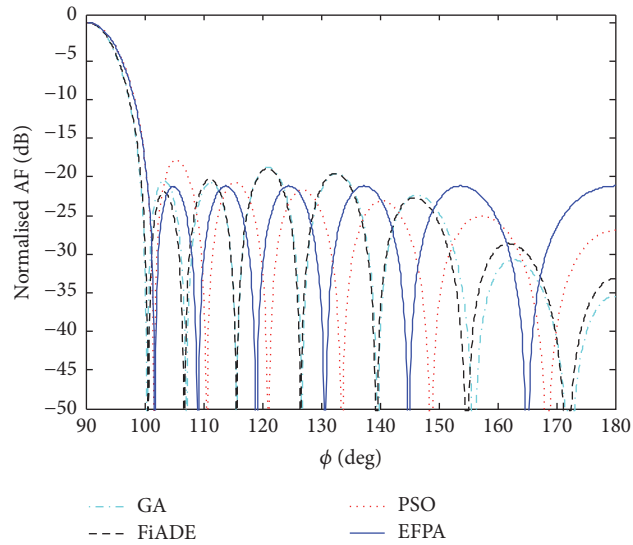

Figure 3: Radiation plot of 12-element EFPA optimized LAA.

Table 7: Performance of different algorithms for 12-element LAA.

Algorithm	FiADE [15]	MA [10]	GA [10]	PSO [15]	FPA [21]	TS [10]	EFPA
SLL (dB)	−18.96	−19.10	−18.77	−17.90	−20.764	−18.40	−21.07

Table 8: Geometry of 22-element (with respect to $\lambda/2$) LAA obtained using EFPA.

Element	1	2	3	4	5	6	7	8	9	10	11
Position	0.4937	0.7965	1.2153	2.1313	2.5003	3.1327	3.9654	4.5877	5.5435	6.6391	8.0603

Table 9: Performance of different algorithms for 22-element LAA.

Algorithm	BSA [18]	TS [10]	MOEA/D-DE [14]	HSA [16]	PSO [15]	GA [10]	MA [10]	FPA [21]	EFPA
SLL (dB)	−23.54	−17.17	−20.93	−23.28	−20.68	−15.73	−18.11	−23.81	−26.31
Null (dB) (99°)	−104.61	−67.94	−69.64	−103.3	−49.94	−54.29	−73.92	−101.71	−75.58

Table 10: Geometry of 28-element (with respect to $\lambda/2$) LAA obtained using EFPA.

Element	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Position	0.5880	1.2142	2.5291	2.8291	4.1205	4.9374	6.1006	6.9646	8.1374	9.5186	10.7399	12.4399	14.1399	15.8399

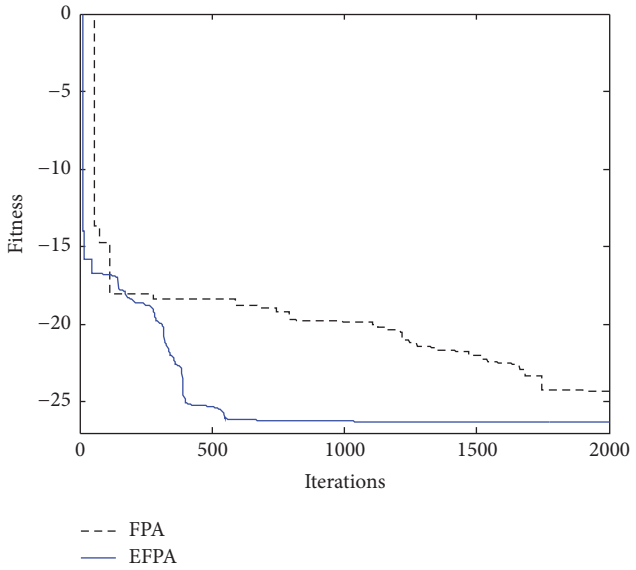

Figure 4: Convergence plots of 22-element LAA.

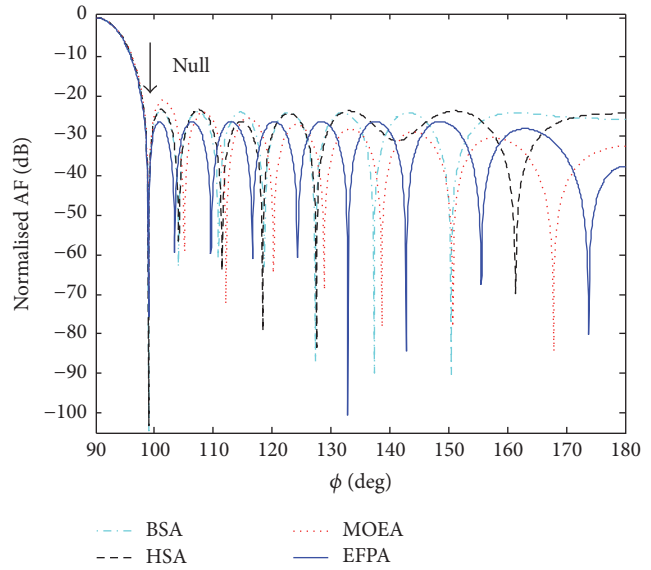

Figure 5: Radiation plot of 22-element EFPA optimized LAA.

obtained using EFPA is shown in Table 10. The comparison of convergence curves between EFPA and BBO [17], CS [17], DE [17], FA [17], and COA [17] is given in Figure 6 which shows that EFPA reaches the optimum solution in less number of iterations. The comparison of results in terms of SLL and null values achieved is given in Table 11 which again clearly indicates the superior performance of EFPA as against other methods. The SLL of EFPA is −22.90 dB which is lower by 1 dB, 9.24 dB, 9.3 dB, 1.04 dB, 8.26 dB, 9.24 dB, 1.27 dB, and 0.30 dB than BSA [18], QPM [3], COA [17], ACO [13], GA [7], CLPSO [20], and FPA [21], respectively. The null values are also equal to or less than −60 dB. Figure 7 shows the comparison of QPM [3], HSA [16], BSA [18], and EFPA optimized array radiation plots.

The last design example illustrates the use of EFPA to optimize the element positions for minimization of SLL and null placement for 32-element LAA. The SLL is to be minimized in the region $\phi \in [93^\circ, 180^\circ]$ and null is to be placed in the direction of 99°. The required beamwidth BW_d is 7.1°. The plot of convergence for EFPA is shown in Figure 8. Also for the sake of comparison, the convergence plots of BBO [17], CS [17], DE [17], FA [17], and COA [17] are shown in the same figure. The element positions for EFPA optimized array

Table 11: Performance of different algorithms for 28-element LAA.

Algorithm	BSA [18]	QPM [3]	PSO [3]	COA [17]	ACO [13]	GA [7]	CLPSO [20]	FPA [21]	EFPA
SLL (dB)	−21.90	−13.24	−13.6	−21.86	−14.64	−13.60	−21.63	−22.60	−22.90
Null (dB) (120°)	−82.49	−48.49	−52.74	−60.08	−52.74	−59.25	−60.04	−78.45	−60
Null (dB) (122.5°)	−93.59	−48.35	−51.66	−60.05	−59.20	−75.53	−60.01	−96.51	−60
Null (dB) (125°)	−80.49	−89.3	−61.46	−60.10	−43.58	−62.52	−60.00	−82.58	−67.66

Table 12: Geometry of 32-element (with respect to $\lambda/2$) LAA obtained using EFPA.

Element	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Position	0.5268	1.1101	2.2759	2.9840	3.8883	4.5802	5.7817	6.6474	7.7093	8.8370	9.7490	11.0885	12.6565	14.3043	16.0043	17.7043

Table 13: Performance of different algorithms for 32-element LAA.

Algorithm	BSA [18]	PSO [3]	HSA [16]	QPM [3]	CS [19]	GA [7]	CLPSO [20]	FPA [21]	EFPA
SLL (dB)	−20.50	−18.80	−19.51	−17.73	−22.83	−16.24	−22.75	−23.10	−23.73
Null (dB) (99°)	−107.50	−62.20	−88.08	−34.74	−62.63	−62.94	−60	−130.60	−60.00

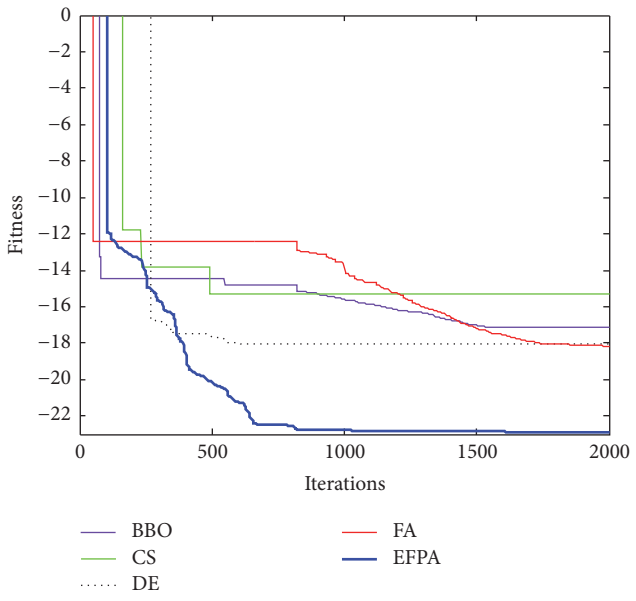

Figure 6: Convergence plots of 28-element LAA.

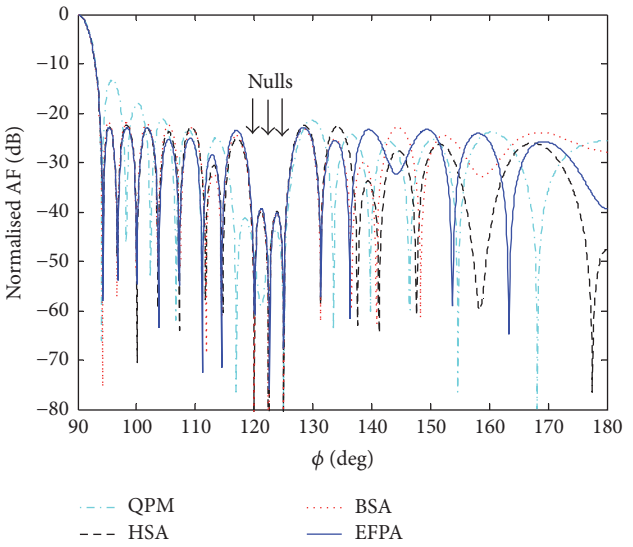

Figure 7: Radiation plot of 28-element EFPA optimized LAA.

are shown in Table 12. Table 13 compares the performance of different optimization methods for design of 32-element LAA. The SLL of proposed EFPA antenna array is −23.73 dB which is least in Table 13. The reduction is considerable as compared to BSA [18], QPM [3], PSO [3], HSA [16], and GA [7] methods. (Figure 9).

5. Conclusions

FPA is a novel nature inspired algorithm which mimics the pollination of flowers. But it has some weaknesses like poor exploitation capability and slow convergence speed. So, to overcome these problems an improved version of FPA called EFPA has been proposed in this paper. Three modifications in the basic FPA have been proposed in the enhanced version. The modifications in EFPA help in providing better search capability and help it to escape local minima. The proposed algorithm has been tested on seven benchmark functions. The results show that EFPA is able to find global optima of most of the benchmark functions. Moreover, the EFPA is utilized for pattern synthesis of nonuniform LAA. Four different antenna arrays of different sizes and different design constraints have been taken. When EFPA is used for SLL

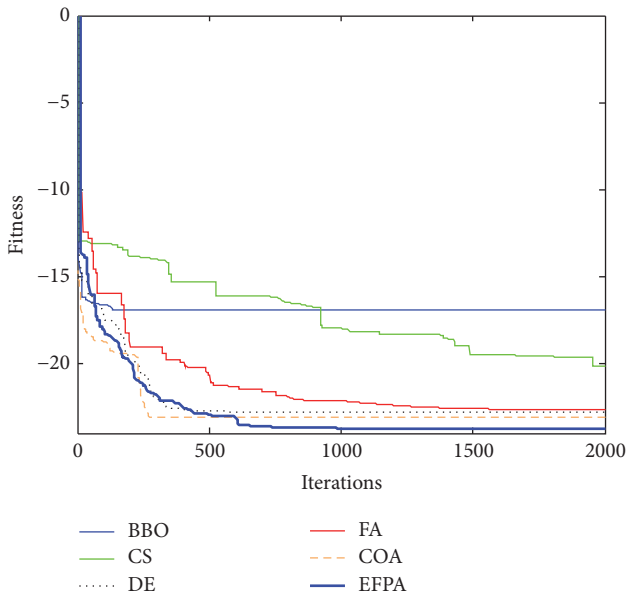

Figure 8: Convergence plots of 32-element LAA.

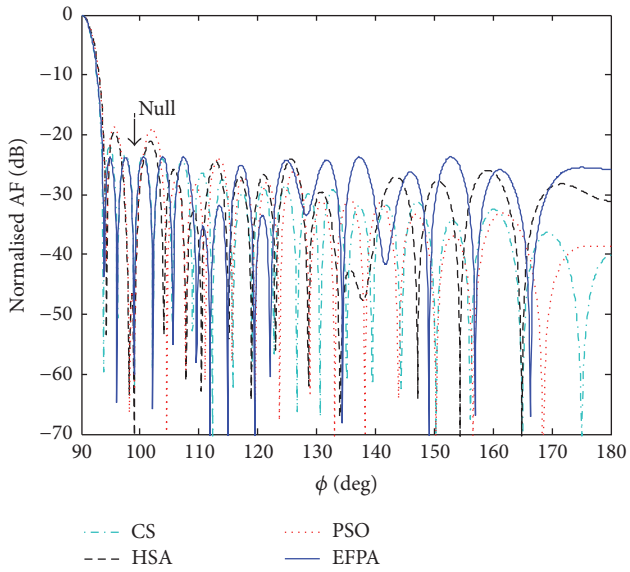

Figure 9: Radiation plot of 32-element EFPA optimized LAA.

minimization only, it reaches the optimum result in less number of iterations than FPA. For multiobjective design with SLL suppression, null, and beamwidth control, EFPA obtains better performance than the results of well-known algorithms reported in literature. The results indicate the potential ability of EFPA to be used for optimization for other antenna designs.

Competing Interests

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

References

[1] C. Balanis, *Antenna Theory-Analysis and Design*, John Wiley & Sons, 2nd edition, 1997.

[2] A. Recioui, "Sidelobe level reduction in linear array pattern synthesis using particle swarm optimization," *Journal of Optimization Theory and Applications*, vol. 153, no. 2, pp. 497–512, 2012.

[3] M. M. Khodier and C. G. Christodoulou, "Linear array geometry synthesis with minimum sidelobe level and null control using particle swarm optimization," *IEEE Transactions on Antennas and Propagation*, vol. 53, no. 8, pp. 2674–2679, 2005.

[4] M. Shihab, Y. Najjar, N. Dib, and M. Khodier, "Design of non-uniform circular antenna arrays using particle swarm optimization," *Journal of Electrical Engineering*, vol. 59, no. 4, pp. 216–220, 2008.

[5] N. Pathak, G. K. Mahanti, S. K. Singh, J. K. Mishra, and A. Chakraborty, "Synthesis of thinned planar circular array antennas using modified particle swarm optimization," *Progress In Electromagnetics Research Letters*, vol. 12, pp. 87–97, 2009.

[6] N. Dib and A. Sharaqa, "Synthesis of thinned concentric circular antenna arrays using teaching-learning-based optimization," *International Journal of RF and Microwave Computer-Aided Engineering*, vol. 24, no. 4, pp. 443–450, 2014.

[7] M. Rattan, M. S. Patterh, and B. S. Sohi, "Synthesis of aperiodic liner antenna arrays using genetic algorithm," in *Proceedings of the 19th International Conference on Applied Electromagnetics and Communications (ICECom '07)*, pp. 1–4, September 2007.

[8] C. Lin, A. Qing, and Q. Feng, "Synthesis of unequally spaced antenna arrays by using differential evolution," *IEEE Transactions on Antennas and Propagation*, vol. 58, no. 8, pp. 2553–2561, 2010.

[9] N. Dib, S. K. Goudos, and H. Muhsen, "Application of Taguchi's optimization method and self-adaptive differential evolution to the synthesis of linear antenna arrays," *Progress in Electromagnetics Research*, vol. 102, pp. 159–180, 2010.

[10] Y. Cengiz and H. Tokat, "Linear antenna array design with use of genetic, memetic and tabu search optimization algorithms," *Progress in Electromagnetics Research C*, vol. 1, pp. 63–72, 2008.

[11] U. Singh, H. Kumar, and T. S. Kamal, "Linear array synthesis using biogeography based optimization," *Progress in Electromagnetics Research M*, vol. 11, pp. 25–36, 2010.

[12] A. Sharaqa and N. Dib, "Design of linear and elliptical antenna arrays using biogeography based optimization," *Arabian Journal for Science and Engineering*, vol. 39, no. 4, pp. 2929–2939, 2014.

[13] E. Rajo-lglesias and Ó. Quevedo-Teruel, "Linear array synthesis using an ant-colony-optimization-based algorithm," *IEEE Antennas and Propagation Magazine*, vol. 49, no. 2, pp. 70–79, 2007.

[14] S. Pal, B.-Y. Qu, S. Das, and P. N. Suganthan, "Optimal synthesis of linear antenna arrays with multi-objective differential evolution," *Progress In Electromagnetics Research B*, no. 21, pp. 87–111, 2010.

[15] A. Chowdhury, R. Giri, A. Ghosh, S. Das, A. Abraham, and V. Snasel, "Linear antenna array synthesis using fitness-adaptive differential evolution algorithm," in *Proceedings of the 6th IEEE World Congress on Computational Intelligence (WCCI '10)—IEEE Congress on Evolutionary Computation (CEC '10)*, July 2010.

[16] K. Guney and M. Onay, "Optimal synthesis of linear antenna arrays using a harmony search algorithm," *Expert Systems with Applications*, vol. 38, no. 12, pp. 15455–15462, 2011.

[17] U. Singh and M. Rattan, "Design of linear and circular antenna arrays using cuckoo optimization algorithm," *PIER C*, vol. 46, pp. 1–11, 2014.

[18] K. Guney and A. Durmus, "Pattern nulling of linear antenna arrays using backtracking search optimization algorithm," *International Journal of Antennas and Propagation*, vol. 2015, Article ID 713080, 10 pages, 2015.

[19] M. Khodier, "Optimisation of antenna arrays using the cuckoo search algorithm," *IET Microwaves, Antennas & Propagation*, vol. 7, no. 6, pp. 458–464, 2013.

[20] S. K. Goudos, V. Moysiadou, T. Samaras, K. Siakavara, and J. N. Sahalos, "Application of a comprehensive learning particle swarm optimizer to unequally spaced linear array synthesis with sidelobe level suppression and null control," *IEEE Antennas and Wireless Propagation Letters*, vol. 9, pp. 125–129, 2010.

[21] U. Singh and R. Salgotra, "Synthesis of linear antenna array using flower pollination algorithm," *Neural Computing and Applications*, 2016.

[22] M. Walker, "How flowers conquered the world," BBC Earth News, July 2009, http://news.bbc.co.uk/earth/hi/earth_news/newsid_8143000/8143095.stm .

[23] B. Glover, *Understanding Flowers and Flowering: An Integrated Appraoch*, Oxford University Press, 2007.

[24] N. M. Waser, "Flower constancy: definition, cause, and measurement," *The American Naturalist*, vol. 127, no. 5, pp. 593–603, 1986.

[25] X. S. Yang, "Flower pollination algorithm for global optimization," in *Unconventional Computation and Natural Computation*, pp. 240–249, Springer, Berlin, Germany, 2012.

[26] A. Draa, "On the performances of the flower pollination algorithm—qualitative and quantitative analyses," *Applied Soft Computing*, vol. 34, pp. 349–371, 2015.

[27] M. Jamil and X.-S. Yang, "A literature survey of benchmark functions for global optimisation problems," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 4, no. 2, pp. 150–194, 2013.

[28] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Tech. Rep. TR-06, Computer Engineering Department, Engineering Faculty, Erciyes University, 2005.

[29] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[30] X.-S. Yang and A. H. Gandomi, "Bat algorithm: a novel approach for global engineering optimization," *Engineering Computations*, vol. 29, no. 5, pp. 464–483, 2012.