Progress In Electromagnetics Research B, Vol. 40, 361–380, 2012

AN INFORMATIVE DIFFERENTIAL EVOLUTION ALGORITHM WITH SELF ADAPTIVE RE-CLUSTERING TECHNIQUE FOR THE OPTIMIZATION OF PHASED ANTENNA ARRAY

D. Maity[1], U. Halder[1], S. Das[2,*], and A. V. Vasilakos[3]

[1]Department of Electronics and Telecommunication Engineering, Jadavpur University, Kolkata, India

[2]Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata, India

[3]Department of Computer and Telecommunication Engineering, University of Western Macedonia, Greece

Abstract—In this paper, we propose a new algorithm called An Informative Differential Evolution with Self Adaptive Re-clustering Technique to find the amplitude-phase excitation of a linear phased array to have the desired far field pattern. Here we consider three problems for three different far field patterns and each problem is optimized with this algorithm. This algorithm has a proper balancing of exploration and exploitation power which is achieved with the help of information exchange among the subpopulations. We also used an elitist local search algorithm for the fine tuning at the suspected optimal position, and that helps us from the unnecessary wastage of Function Evaluations (FEs).

1. INTRODUCTION

The main purpose of optimization is to find global minima or global maxima value in a given search range of a function that is being optimized. There may exist more than one global maxima or global minima for a given function in a given search range. The traditional techniques such as steepest decent, liner programming, dynamic programmings fail to find the maxima and minima of a function

Received 1 February 2012, Accepted 19 April 2012, Scheduled 10 May 2012

* Corresponding author: Swagatam Das (swagatamdas19@yahoo.co.in).

because of the discontinuity, high dimensionality, and obviously due to multimodality in a complex function. Here lies the importance of optimization techniques, and researchers have developed a lot of techniques along with some evolutionary algorithms. There are some powerful algorithms that have been proposed by researches in the last few decades. Among them DE [2, 3] is one of the most powerful algorithms that has been proposed by Storn and Price.

In past decades, many electromagnetic optimization problems have been solved by using EAs such as by using Invasive Weed Optimization [4–6], Particle Swarm Optimization [1, 7]. Genetic Algorithm [1, 8, 9]. Mouhamadou et al. used a new method called Sequential Quadratic Programming (SQP) algorithm to suppress multiple narrow and wide band interferences and track the desire signal by controlling only the phase has been presented in [12]. Mahanti et al. presented a new technique for designing thinned linear antenna arrays with fixed side-lobe level and fixed percentage of thinning using real-coded genetic algorithm [13] in the year 2007. In the same year, Guney and Onay used Bees algorithm for interference suppression of linear antenna array, controlling the amplitude in the paper [14]. Again in the year of 2008, Guney and Basbug used Bacterial Foraging algorithm for interference suppression of linear antenna array, only controlling the amplitude of the element excitation [11].

Optimization problems in electromagnetic domain generally involve with large number of designing parameters and the parameters may also include constrains in the search range. So for these reasons, these problems cannot be solved with the help of the traditional process that has been stated earlier. To have the most promising solution of these problems we have to go for evolutionary algorithms. The present problem is to find the current excitations (both magnitude and phase) of the phased antenna array for desired power pattern(s). The antenna under study is a twenty five element antenna array. Using this small number of antenna elements it is very difficult to have the desired power pattern such as very low side-lobe level, directivity, and notch characteristics. Using this antenna array it is a challenge to have the directivity of the antenna at a direction away from zero degree (in theta span). In this paper, we choose the desired direction to be -45 degree for the second problem which is really a challenge to suppress the side-lobes below -25 dB. In recent years, several optimization algorithms have been developed to solve these kinds of electromagnetic problems but most of the problems are concerned about the side-lobe levels, half power beam width, first null beam width etc. with less concern about the different main-lobe directions, also there is few works that deal with notch characteristics of a power pattern. In this paper, we find

the excitations of the antenna element where the antenna design does not change. Our purpose is without changing the design parameters (position of elements, distance etc.), how we can get desired patterns with satisfactory performance by only changing the excitations. The problems under consideration in this paper are difficult enough and thus need to be tackled carefully. We take the help of EAs to solve these problems. The main advantage of these optimization techniques lies in the efficiency in finding the globally best solution without being trapped in any local optima. In this paper, we have adopted DE as our main algorithm as it has a better explorative power with a satisfactory convergence rate. Sometimes a better convergence property can lead to a premature convergence and hence the global best solution is not found. Keeping these things in mind and according to the requirements of the problem under consideration, we developed our algorithm that self-adaptively tries to avoid the trapping in any local optima. We took the help of subpopulations instead of a single population-which is used in classical DE-along with the information exchange among the subpopulations to have a good convergence property. Whereas the use of the local search algorithm increases the exploration power, which is elaborately explained later on in this paper.

The rest of the paper is presented as follows, Section 2 contains a brief description about classical DE, Section 3 describes the proposed algorithm, Section 4 contains synthesis of a linear array antenna, Section 5 contains the problems we considered and their numerical results, and Section 6 concludes the paper.

2. DIFFERENTIAL EVOLUTION ALGORITHM

DE is a very simple but a very powerful algorithm for optimization problem. Let $S \in R^n$ be the search space of the problem under consideration. DE algorithm starts with an initial population of *NP,* n dimensional solution particles. These particles (solution vectors) are initially covering the search space (S) as much as possible by randomly initializing them through the search space. The particles are of the form $\vec{X}_i = (x_{i1}, x_{i2}, x_{i3}, \ldots, x_{in}) \in S$, where $i = 1, 2, \ldots, NP$ and are upgraded from one generation to next generation, where $x_{i1}, x_{i2}, \ldots, x_{in}$ are in between their respective upper and lower bounds x_j^{upper}, x_j^{lower} respectively. The population undergoes through Crossover, Mutation at each generation t and produces a new solution vector $\vec{U}_{i,t}$ for each vector $\vec{X}_{i,t}$.

2.1. Mutation

After initialization for each solution vector $\vec{X}_{i,t}$, a new vector $\vec{V}_{i,t}$ is generated at each generation t. There are several methods to generate $\vec{V}_{i,t}$, one such technique that we have used in our paper is,

$$\text{“DE/best/1”} : \ \vec{Y}_{i,t} = \vec{X}_{best,t} + F \cdot \left(\vec{X}_{r_1^i,t} - \vec{X}_{r_2^i,t} \right) \tag{1}$$

where the indices r_1^i and r_2^i, are mutually exclusive integers randomly chosen from the range [1, NP], and all are different from the base index i. F is a scaling factor for the differential vectors and $\overrightarrow{X}_{best,t}$ is the vector with best fitness in the generation t.

2.2. Cross-over

Crossover plays a major role to enhance the diversity of the population. In this phase, the generated vector $\vec{V}_{i,t}$ exchanges its component with its parent vector $\vec{X}_{i,t}$ to generate a new vector $\vec{U}_{i,t} = (u_{1i,t}, u_{2i,t}, u_{3i,t}, \ldots, u_{ni,t})$, where $u_{j,i,t}$ is found by the following procedure:

$$u_{j,i,t} = \begin{cases} V_{j,i,t} \ , & \text{if } (rand_{i,j}[0,1) \ < \ Cr \quad \text{or} \quad j = j_{rand} \\ X_{j,i,t} \ , & \text{otherwise} \end{cases}$$

where $rand_{i,j}[0,1)$ is a uniformly distributed random number. For have an elaborate discussion on DE the readers are directed to [17] and the references therein.

2.3. Selection Operation

The selection procedure is done by the following way:

$$\vec{X}_{i,t+1} = \begin{cases} \vec{U}_{i,t} & \text{if } f\left(\vec{U}_{i,t}\right) \leq f\left(\vec{X}_{i,t}\right) \\ \vec{X}_{i,t} & \text{otherwise} \end{cases}$$

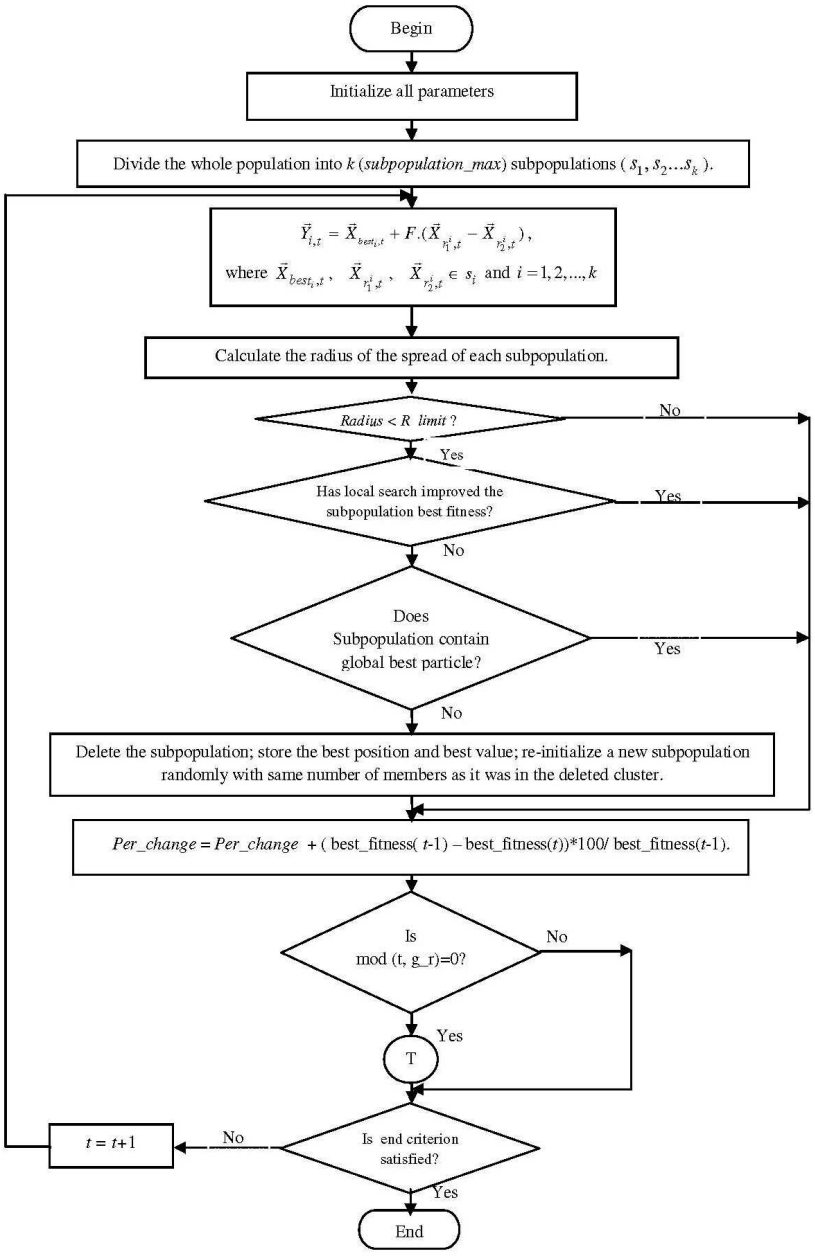
for minimization problems. Thus after every generation we find either a new solution which has better fitness (here for minimization problem) or the previous vector is kept. So after each generation the population gets better or remains unchanged but never deteriorates.

3. PROPOSED ALGORITHM

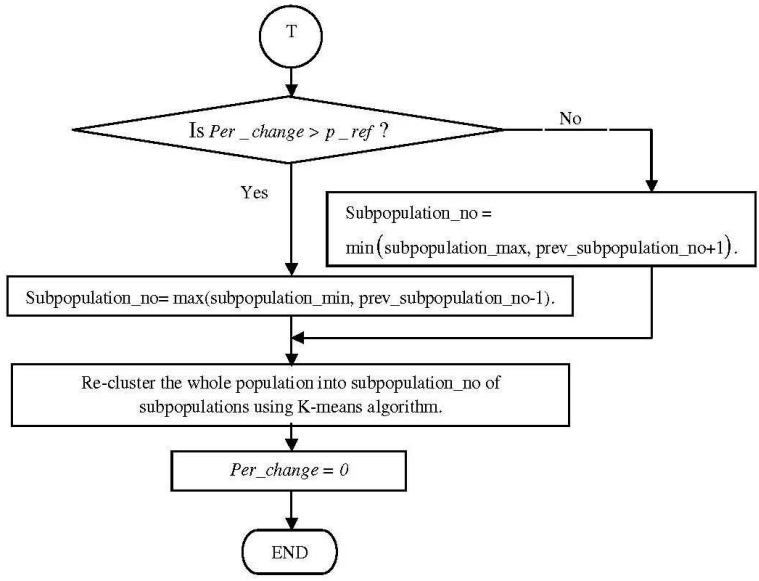
In this paper, we propose a modified differential evolution algorithm with an information exchange strategy by the use of multiple

populations. Though DE has a balancing between the explorative and exploitative power, but in some cases the exploitative nature leads to a premature convergence or trapping to a local optimum in a complicated and multimodal search space; which may be overcome by using multi-population scheme. Even if we assume that in this case also the subpopulation converges to any optimum then different subpopulation will converge to different optimum; whereas use of single population scheme converge the population to a single optimum. The total population is subdivided using K-means clustering algorithm and throughout the iterations the number of subpopulations is changed in a self-adaptive way for the balancing of the exploration and exploitation. Along with these strategies we have also used a local search technique for the better tuning near a suspected optimum. The local search technique is used only when the algorithm finds that any subpopulation has reached to a local optimum; this process is discussed later in this paper. The subpopulations separately use the DE/best/1 and find for better optimum solution, and they do not exchange any information. They exchange information after a specific number of iterations — which will be called *refreshing gap* (g_r) later on in this paper — and then the whole population is again re-clustered into a specific number of clusters. Each cluster denotes each subpopulation. The number of clusters (subpopulations) is determined in a self adaptive way by the algorithm. In this paper, we used a fitness feedback scheme to vary the subpopulation number. If the algorithm performs satisfactorily well i.e. the exploration is good then we decrease the subpopulation number that was used to increase exploration; and if the algorithm does not perform satisfactorily then the subpopulation number is increased. We have kept an upper and lower limit for the subpopulation number denoted as *cluster*_max and *cluster*_min. IIt may also happen that two or more subpopulations can come close to each other when they are improving themselves. In the next g_r when the algorithm re-clusters the whole population, the subpopulations those were very close to each other will belong to a same subpopulation now. This is because of the property of k-means Algorithm [15] that generates clusters based on the spatial distribution of the data in the n dimensional space. In this paper we have used DE/best/1strategy to each subpopulation separately because this variation of DE has a better convergence rate.

In multimodal landscapes when a subpopulation converge to a local optima then the difference vector in Equation (1) becomes very small and the improvement is very low for that subpopulation, and after some iterations the subpopulation is totally converged. Hence there will be only loss of FEs in evaluating the fitness of those particles after the convergence. Here lies the importance of the local search


Begin
Initialize all parameters
Divide the whole population into k (subpopulation_max) subpopulations ($s_1, s_2 \ldots s_k$).
$\vec{Y}_{i,t} = \vec{X}_{best_i,t} + F.(\vec{X}_{r_1^i,t} - \vec{X}_{r_2^i,t})$,
where $\vec{X}_{best_i,t}$, $\vec{X}_{r_1^i,t}$, $\vec{X}_{r_2^i,t} \in s_i$ and $i = 1, 2, ..., k$
Calculate the radius of the spread of each subpopulation.
Radius < R limit ?
No
Yes
Has local search improved the subpopulation best fitness?
Yes
No
Does Subpopulation contain global best particle?
Yes
No
Delete the subpopulation; store the best position and best value; re-initialize a new subpopulation randomly with same number of members as it was in the deleted cluster.
Per_change = Per_change + (best_fitness(t-1) – best_fitness(t))*100/ best_fitness(t-1).
Is mod (t, g_r)=0?
No
Yes
T
Is end criterion satisfied?
No
t = t+1
Yes
End

(a)



(b)

Figure 1. (a) Flowchart of the proposed algorithm, (b) flowchart of the re-clustering technique used.

technique; when the improvement of a subpopulation becomes less down the iterations then the algorithm suspects that the subpopulation has trapped to a local optimum and the local search technique is used. Actually the local search algorithm introduces some new particles randomly around the members within a specific radius. This process is known as the tuning of the suspected optimum. If the subpopulation is improved using the local search technique then the algorithm takes only the best members among the particles of that subpopulation including the particles generated by the local search technique. Otherwise the total subpopulation is deleted from the search space if it does not contain the global best particle, and the best solution found by the subpopulation is kept in an archive. The key parts of the algorithm are described in the following way and a flowchart of the complete algorithm is shown in Figure 1.

3.1. Initialization

The total population is initialized randomly in the search space to cover the search space as uniformly as possible. Then the population is divided into some subpopulations using the K-means algorithm.

Initially we take the subpopulation number to be *subpopulation_*max.

3.2. Subpopulation Interaction

For the particles of a cluster we use the DE/best/1 strategy and they can exchange information among the particles those belong the particular subpopulation only. They have no scope to interact with those particles belonging to a different subpopulation. Only after each g_r when the algorithm re-clusters the whole population, there is a mixing of members between two different subpopulations. For the next g_r the particles of different subpopulation which are now in same subpopulation will exchange their experiences that they have learned in the previous iterations and that will help them to find the optimum position. The number of subpopulation is determined in a self adaptive way. The algorithm counts the total percentage change in the fitness value of the whole population, and if the percentage change throughout a g_r is above a certain threshold percentage value then the subpopulation number is decreased and if the percentage change is below a certain threshold value then the subpopulation number is decreased. The subpopulation number is changed as follows:

$$\begin{aligned} &Subpopulation\_no \\ &= \min(subpopulation\_\max,\ prev\_subpopulation\_no + 1). \\ &Subpopulation\_no \\ &= \max(subpopulation\_\min,\ prev\_subpopulation\_no - 1). \end{aligned}$$

3.3. Local Search Technique

If the radius of the subpopulation becomes very less then we use the local search technique and depending on the response of the subpopulation after the application of local search technique the algorithm decides whether to keep the subpopulation or to delete. The radius of a subpopulation is defined as the mean Euclidian distance of the particle from the centre of the subpopulation. The radius is calculated as:

$$R = \sum_{i=1}^{m} \|\vec{x}_i - \vec{c}\|/m, \tag{2}$$

where m is the number of particles in the subpopulation, $\vec{c}$ is the centre of the subpopulation and it is calculated as:

$$\vec{c} = \sum_{i=1}^{m} \vec{x}_i/m. \tag{3}$$

If the radius of the subpopulation calculated using (2) becomes less than *R_limit* then the algorithm calls for the local search technique. The *R_limit* is defined as:

$$R\_limit = \sqrt{\left(\vec{X}_{upper} - \vec{X}_{lower}\right)^T \cdot \left(\vec{X}_{upper} - \vec{X}_{lower}\right)} \times 10^{-2}, \quad (4)$$

where $\vec{X}_{upper}$ is the upper bound of the search space and $\vec{X}_{lower}$ is the lower bound of the search space and $(\cdot)^T$ denotes the transpose of the matrix $(\cdot)$. The local search technique introduces some new particle known as *exploiters* to exploit the information from a suspected optimum position. We take only 40% best particle from the subpopulation and generate *exploiters* only around them within a specific radius known as *r_exploit*. The *r_exploit* is smallest for the best fit particle in the subpopulation and it increases with the fitness value for a minimization problem. The worst fit particle among the 40% chosen particle has the largest *r_exploit*. The number of generated *exploiters* is also varied with the fitness of the particles. We used the following relations to find *exploiters* number and *r_exploit*.

$$exploiters_i = round\left(\begin{array}{l}(exploiters\_\max - exploiters\_\min) \\ \cdot\left(1 - \left(\frac{f_i - best}{worst - best}\right)^2\right) + exploiters\_\min\end{array}\right), \quad (5)$$

where $exploiters_i$ denotes the number of exploiters generated around the particle i and f_i denotes the fitness value of the i-th particle, the *best* and *worst* denotes the best fit and worst fit particle among the chosen 40% particles respectively.

$$r\_exploit_i = \left(1 - k^d\right) \cdot \left(\frac{(f_i - best)}{worst - best}\right) + r\_ini, \quad (6)$$

where $k = \frac{best}{worst}$. The parameter d is chosen either 1 or 2 for best performance.

4. ARRAY FACTOR AND ARRAY OPTIMIZATION

The far field radiation pattern that has to be minimised is given by $FF(\theta) = AF(\theta) \cdot EP(\theta)$, [1] where $AF(\theta)$ is the array factor of the linear phased array of half-wavelength space radiators.

$$AF(\theta) = \sum_{i=1}^{N} I_i e^{j2\pi i\left(\frac{d}{\lambda}\right)\sin(\theta)}, \quad (7)$$

I_i is the excitation of the ith element, N = the number of elements. Realistic phased array using aperture elements such as slots or patches

typically have directive elements that give rise to a element pattern often approximated by [10].

$$EP(\theta) = \sqrt{\cos^n \theta} \tag{8}$$

where $n = 0$ would represent ideal isotropic array elements and $n > 0$ represents directive array elements. In this paper, n is taken to be 1.2 according to [1] and the element pattern becomes

$$EP(\theta) = \sqrt{\cos^{1.2} \theta}. \tag{9}$$

The excitation has amplitude and also a phase so we can write $I_i = A_i e^{j\beta_i}$ and the array factor becomes

$$AF(\theta) = \sum_{i=1}^{N} A_i e^{j\left(2\pi i\left(\frac{d}{\lambda}\right)\sin(\theta)+\beta_i\right)}. \tag{10}$$

Our objective is to optimize the Equation (10) for a desired far field radiation pattern. We chose the number of element in the array to be 25 thus the objective function is a 50 dimensional problem of which the first 25 dimensions are for the amplitude excitation and the rest are the corresponding phase excitations. Many works have been done before to optimize various parameters of an antenna array such as side-lobe levels, etc. but most of then uses a large number of array elements and thus in practical life it may not be suitable so here we take the number of elements to be only 25. These 25 elements are placed at a half wave length distance from each other. We try to find different excitations for this array so that the same array can produce more than one desired far field pattern. Here we consider three different far field patterns, such as side-lobe suppression below a -30 dB, direction of main beam to be $-45°$, and designing a notch from $50°$ to $60°$ in theta direction.

5. OBJECTIVE FUNCTIONS

The goal of pattern synthesis is to find the position and the excitation of each element in the array so that the side-lobe suppression, main-lobe direction, beam-width, null control is as per the designers choice. In most of the real cases we dont need any isotropic radiation pattern rather we choose that the transmitting antenna must transmit its major power along the direction that we prefer. In some cases like communication between two military base station, we have to keep it in mind that no signals should be transmitted to the enemy base station for that purpose we desire that the antenna radiation pattern should have very poor radiation along that direction and therefore we may design a notch antenna of which the radiation pattern contains a

sudden drop in the power pattern below a certain dB at some preferred direction. Similarly any transmitting antenna we desire that most of the radiation power should be concentrated within the main beam and for that we have to suppress the other minor lobes below a certain dB with respect to the main lobe. From the practical requirement of any transmitting and receiving antenna we have to control the side-lobe level, main lobe direction, positions of null and presence of notch etc. We take here three objective functions that optimize one or more than one constrains written above. When the number of constrain is more than one then constrains are combined with one cost function by computing their proportion sum. For the suppression of side-lobes we only try to suppress the peak side-lobe below a certain dB so that other side-lobes must lie below the assigned dB level. The peak side-lobe is the sidelobe which has the maximum dB among the side-lobes. In this paper we define the peak side-lobe as:

$$P_{SL} = \max\left(\frac{FF(\theta)}{|FF(\theta_{des})|}\right) \quad \forall\theta \in \Omega, \tag{11}$$

where Ω is the space spanned in θ direction but it excludes the main lobe that is in the direction of θ_{des}. For some problems we took θ_{des} to be zero and for some other we took some desired value along which we need to transmit or receive power.

5.1. Problem 1

In this problem, we consider an antenna that will transmit its maximum power along the zero degree direction of theta so for that purpose we need that the power transmitted by the side-lobes along other directions should be minimized. So here we consider the optimization goal is to suppress the side-lobes (peak side lobe will be below $-30\,\mathrm{dB}$). We need to minimize the P_{SL}.

5.2. Problem 2

The previous problem was to have the maximum radiation along the zero degree but now we consider that with the same physical arrangement of the antenna what should be the excitation so that the maximum radiation takes place along -45° direction. So here our aim is to find the far field pattern such that the main-lobe will be directed along the $\theta = -45^\circ$ and all the side-lobes will be below $-20\,\mathrm{dB}$.

5.3. Problem 3

In this problem, we incorporate the idea of having a notch in the radiation pattern. The presence of notch ensures us that there will be a very little or even no propagation of power from the source. In this problem we consider an designing of array that will give a far field radiation pattern such that the pattern will contain a $-60\,\mathrm{dB}$ notch in the θ direction from θ equals to 50° to θ equals to 60° and tried to suppress the side-lobes below $-15\,\mathrm{dB}$.

5.4. Results and Comparison

We took the maximum number of function evaluations to be 125000, which is quite a less number for solving these types of problems. The solution vector contains the amplitude and phase excitations for each antenna array element. We took the amplitude excitation to be in the range $[0.1, 1]$ so that for excitation ratio of maximum amplitude to minimum amplitude will be 10 which can be afforded easily compared to the case if the ratio is taken as 100 or more. The generated cost functions are also optimized with other powerful algorithms SADE [17] and CLPSO [16]. For the comparison of proposed algorithm with other algorithms we took the same cost functions with same goal (suppression of side-lobes for same amount etc.) and the number of function evaluations is also same.

5.4.1. Parameter Settings

In this section, we discuss about the optimum value of each parameter used in this paper. Before choosing the proper value of the parameter we have dome some experiments to get the optimum value. In this paper, some of the parameters are self adaptive so they change themselves according to the requirement, whereas some other parameters need to be set an optimum value.

A. Initial Population Size

The initial population size is one of the important parameter of this paper because if the initial population is very small then the population can easily get stuck to a local optimum and the explorative power of the algorithm also decreases. If the population size is kept high to increase diversity in population and to increase the explorative power then the number of function evaluation increases in each iteration. Thus we need to optimize the initial population size. The performance of the algorithm due to the different population size is given in the Table 1.

Table 1. Performance of the algorithm with different population sizes.

Population Size (NP)	FF (θ) in dB
50	−29.7
60	−33.2
70	−33.6
80	−34.3
90	−35.1
100	−36.9
120	−36.4

Thus from the above table we can see that the best value for the population size is 100 and as we stated before that due to increase in population size the number of function evaluation increases so number of generation decreases and thus the algorithms' performance decreases.

B. Number of Maximum and Minimum Subpopulations

The number of subpopulation plays a great role in the algorithms performance. Though we used a self adaptive strategy to change the subpopulation number but still we kept a maximum and minimum value so that the number of subpopulation does not increase huge of reduced to 1 or zero or even negative. The subpopulation number cannot be zero. Subpopulation number one is also excluded because if subpopulation reduces to one then the notion of multi-population is not valid there and the purpose of information exchanging among the subpopulation is lost. So we keep the minimum subpopulation number to be two. If we use large number of subpopulation then each subpopulation will contain small number of particles and as we used K-means algorithm, it may happen that one or more subpopulations can contain very few particles and the diversity of that subpopulation is lost, and the subpopulation is of no use. Similarly if the subpopulation number is low then the information exchange will not be effective properly, so we need to find a optimum maximum cluster number. Table 2 shows the variation of the performance of the algorithm with the number of subpopulations.

The Table 2 shows that for population size to be 100, the maximum subpopulation number has to be 7.

C. Scaling Factor (F)

Table 2. Performance of the algorithm on problem 1 for different values of the subpopulation.

*subpopulation*_max, *NP*	FF (θ) in dB
5, 100	−34.2
6, 100	−35.1
7, 100	−36.9
8, 100	−36.7
9, 100	−35.1
10, 100	−33.8

Table 3. Performance of the algorithm on problem 1 for different scheme in scaling factor.

Scaling Factor (F)	FF (θ) in dB
0.5	−35.8
rand (0,1)	−36.4
0.5 (rand $(0, 1) + 1$)	−36.9

In this paper, we have used a random scaling factor in the range $[0.5, 1]$ instead of a constant scaling factor. The effect of scaling factor on problem 1 is given Table 3.

D. Number of Maximum and Minimum Exploiter Particle

For the efficient use of the local search algorithm the parameters of the local search algorithm also has to be tuned, so here he find the proper value of the *exploiters*_max and *exploiters*_min for the best performance of our algorithm and a tabular form is given in Table 4.

From Table 4 we see the maximum and minimum *exploiters* number should be 7 and 2.

E. Search Range of Exploiter Particles

The search range for each exploiter particle is different and it depends on the fitness of the parent particle. The search range is a function of fitness of parent particle and it is given by Equation (6). The *r_ini* is a parameter here that has to be properly chosen. The physical significance of *r_ini* is that, the exploiters particles generated by the best fit particle will be generated in a sphere of radius *r_ini* in a hyper dimension space of dimension n, and the radius for the

Table 4. Performance of the algorithm on problem 1 for different values of *exploiters*_max and *exploiters*_min.

*exploiters*_max, *exploiters*_min	FF (θ) in dB
5, 1	−36.7
5, 2	−36.2
6, 1	−36.3
6, 2	−36.5
7, 1	−36.7
7, 2	−36.9
8, 1	−36.6
8, 2	−36.2
9, 1	−36.4
9, 2	−36.1

Table 5. Performance of the algorithm on problem 1 for different values of *r_ini*.

r_ini (*e-03)	FF (θ) in dB
0.25	−35.8
0.50	−36.9
1.00	−35.7

exploiters of other parents will be more that *r_ini*. We have chosen three different values of *r_ini* to see the performance of the algorithm on problem 1 which is given in Table 5. So the optimum value for the *r_ini* is 0.5e-03.

Thus from the above discussions regarding parameter setting we have found the optimum values of each parameter and it is tabulated in the Table 6.

For CLPSO and SADE we used the best parameter settings suggested in the corresponding papers.

5.4.2. Results and Comparison

The plots of the far field patterns given in Figures 2, 3, and 4 successfully illustrate that in what extent our proposed algorithm is better than the other two.

From the above plots it is clear that our algorithm performs much well than SADE and CLPSO. The problem 1 was to design an antenna

Table 6. Optimum value for each parameter.

Parameters	Values
NP	100
F	0.5 (rand +1)
Cr	0.9
g_r	20
*cluster*_max	6
*cluster*_min	2
*exploiters*_max	7
*exploiters*_min	2
r_ini	5e-03
p_ref	2

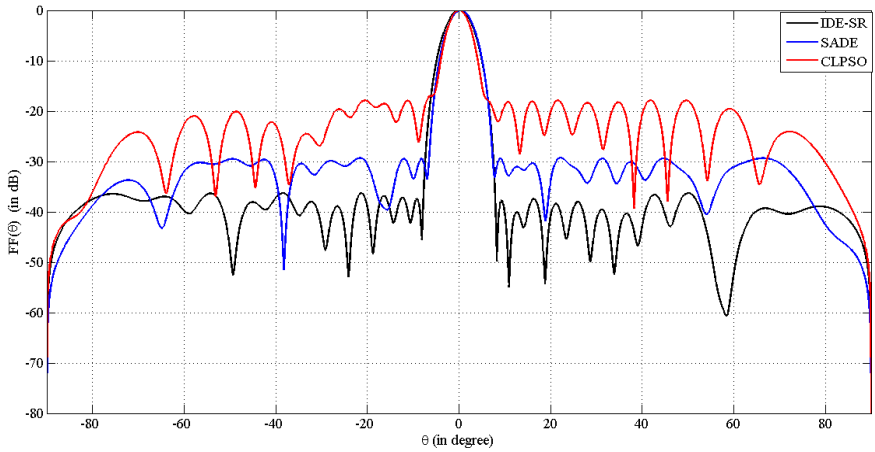


Figure 2. Far field radiation pattern for problem 1.

array that will give a far field pattern which will have suppressed side-lobes below −30 dB. Using SADE in problem 1 we get a pattern that suppresses the side-lobes below −29 dB whereas with CLPSO it is only −17 dB but our algorithm suppresses it below −37 dB which is given in Fig. 2. The problem 2 was to design the antenna parameters in such a way that the main-lobe is along the −45 degree in theta direction. From the Fig. 3 we notice that SADE can optimize the function in such a way that that the main-lobe is at −45 degree but with CLPSO it is at −46 degree, whereas using our algorithm we find the proper main-lobe

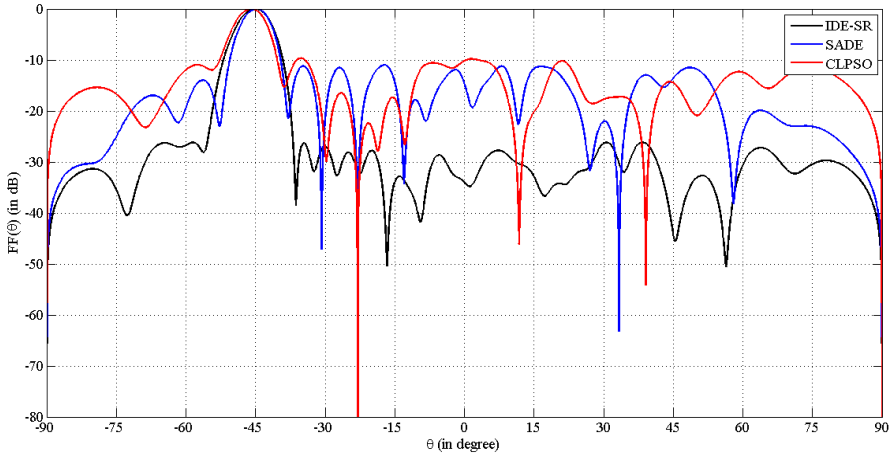


Figure 3. Far field radiation pattern for problem 2.

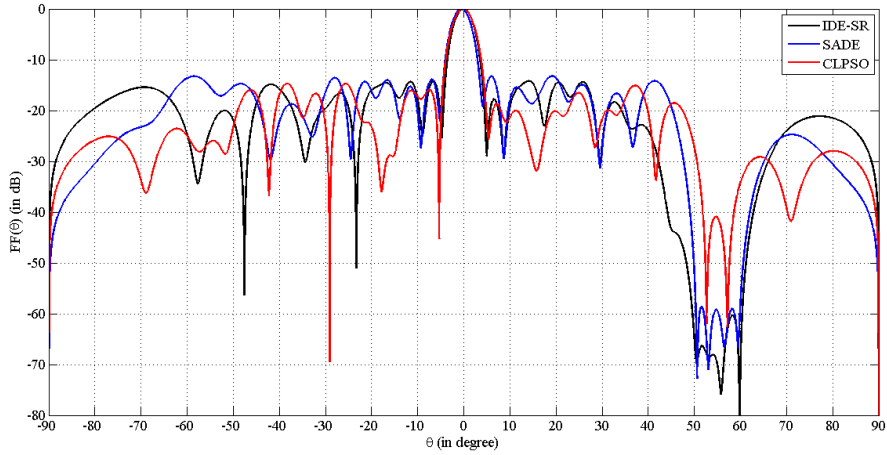


Figure 4. Far field radiation pattern for problem 3.

direction position. Moreover our algorithm has become successful to suppress the side-lobes below −27 dB whereas with SADE and CLPSO it's only −12 dB and −11 dB respectively. In problem 3 we had to design a notch antenna i.e., radiation pattern will contain a notch below −60 dB for the theta space from 50 degree to 60 degree. Optimization of problem 3 with SADE generates a notch of −59 dB and with CLPSO it is −40 dB but not for the whole region between theta 50 degree to 60 degree. In this case also our algorithm gives the desired result i.e., a notch of −60 dB that can be verified from Fig. 4.

Table 7. Comparative study of the performance of different algorithms with IDE-SR.

Problem no. \ Algorithm		**IDE_SR**	**SADE**	**CLPSO**
Prob. 1	Best	**−3.69e+01**#	−2.92e+01	−1.69e+01
	Mean	**−3.47e+01**	−2.51e+01	−1.45e+01
	Std.	**2.43e+00**	2.68e+00	1.36e+00
Prob. 2	Best	**−2.71e+01**#	−1.18e+01	−1.01e+01
	Mean	**−2.32e+01**	−9.84e+00	−9.45e+00
	Std.	**1.62e+00**	3.74e+00	9.69e−01
Prob. 3	Best	**−6.00e+01**#	−5.82e+01	−3.74e+01
	Mean	**−5.64e+01**	−4.97e+01	−3.55e+01
	Std.	**3.24e+00**	6.54e+00	1.04e+00

From Table 7 and the plots given in Figs. 2–4 we see that our algorithm out performs the rest two algorithms. To determine whether the result of proposed algorithm is different from the result of the other algorithms, the Wilcoxon rank sum test [18, 19] is conducted and whenever the result of our algorithm is statistically significant we put a '#' besides our result and '!' mark is given when algorithm is not statistically significant and when it is comparable to others then we put '0' mark.

6. CONCLUSION

In this paper, we try to find the optimal excitation of a phased antenna array to have the suppression of side-lobes, desired direction of main-lobe with proper suppression of side-lobes, having a notch characteristic in the radiation pattern. We designed all these using an array that consists of only twenty five elements. Here we optimize both the amplitude and Phase to have a better control on the desired pattern. The simulation results successfully show the better performance of proposed IDE-SR algorithm over the SADE and CLPSO. We used only 125,000 function evaluations for these three problems and it incurs considerably small computational cost for these types of problems. As a future work we can consider the multi-objective version of the problems instead of taking the weighted sum. Though in a first look to the algorithm it may seem that there are a large number of parameters but in some cases the performance of the algorithm is merely dependant on some of them. We can try to make the number of parameters lesser keeping the excellence of the performance.

REFERENCES

1. Boeringer, D. W. and D. H. Werner, "Particle swarm optimization versus genetic algorithms for phased array synthesis," *IEEE Transactions on Antennas and Propagation*, Vol. 52, No. 3, 771–778, Mar. 2004.
2. Storn, R. and K. V. Price, "Differential evolution — A simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, Vol. 11, 341–359, 1997.
3. Das, S. and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. on Evolutionary Computation*, Vol. 15, No. 1, 4–31, Feb. 2011.
4. Mehrabian, A. R. and C. Lucas, "A novel numerical optimization algorithm inspired from weed colonization," *Ecological Informatics*, Vol. 1, 355–366, 2006.
5. Mallahzadeh, A. R., S. Es'haghi, and A. Alipour, "Design of an E-shaped MIMO antenna using IWO algorithm for wireless application at 5.8 GHz," *Progress In Electromagnetics Research*, Vol. 90, 187–203, 2009.
6. Mallahzadeh, A. R., S. Es'haghi, and H. R. Hassani, "Compact U-array MIMO antenna designs using IWO algorithm," *International Journal of RF and Microwave Computer-aided Engineering*, Wiley-InterSscience, Jul. 2009, DOI: 10.1002/mmce.20379.
7. Kennedy, J. and R. C. Eberhart, "Particle swarm optimization," *Proceedings of IEEE International Conference on Neural Networks*, 1942–1948, Piscataway, NJ, 1995.
8. Holland, J., "Adaptation in natural and artificial systems," *University of Michigan Press*, Ann Arbor, 1975.
9. Jain, R. and G. S. Mani, "Dynamic thinning of antenna array using genetic algorithm," *Progress In Electromagnetics Research B*, Vol. 32, 1–20, 2011.
10. Grimaccia, F., M. Mussetta, and R. E. Zich, "Genetical swarm optimization: Self-adaptive hybrid evolutionary algorithm for electromagnetics," *IEEE Transactions on Antennas and Propagation*, Vol. 55, 781–785, 2007.
11. Guney, K. and S. Basbug, "Interference suppression of linear antenna arrays by amplitude-only control using a Bacterial Foraging algorithm," *Progress In Electromagnetics Research*, Vol. 79, 475–497, 2008.
12. Mouhamadou, M., P. Armand, P. Vaudon, and M. Rammal, "Interference suppression of the linear antenna arrays controlled by phase with use of SQP algorithm," *Progress In Electromagnetics*

Research, Vol. 59, 251–265, 2006.

13. Mahanti, G. K., N. Pathak, and P. Mahanti, "Synthesis of thinned linear antenna arrays with fixed sidelobe level using real-coded genetic algorithm," *Progress In Electromagnetics Research*, Vol. 75, 319–328, 2007.
14. Guney, K. and M. Onay, "Amplitude-only pattern nulling of linear antenna arrays with the use of BEES algorithm," *Progress In Electromagnetics Research*, Vol. 70, 21–36, 2007.
15. MacQueen, J. B., "Some methods for classification and analysis of multivariate observations," *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, 281–297, University of California Press, 1967, Retrieved April 7, 2009.
16. Liang, J. J., A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE T. on Evolutionary Computation*, Vol. 10, No. 3, 281–295, Jun. 2006.
17. Qin, A. K. and P. N. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," *Proceedings of the 2005 IEEE Congress on Evolutionary Computation*, Vol. 2, 1785–1791, 2005.
18. Wilcoxon, F., "Individual comparisons by ranking methods," *Biometrics*, Vol. 1, 80–83, 1945.
19. Derrac, J., S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, Vol. 1, No. 1, 3–18, Mar. 2011.