Linear Antenna Array Synthesis using Fitness-Adaptive Differential Evolution Algorithm

Aritra Chowdhury, Ritwik Giri, Arnob Ghosh, Swagatam Das, Ajith Abraham and Vaclav Snasel

Abstract — **Design of non-uniform linear antenna arrays is one of the most important electromagnetic optimization problems of current interest. In this article, an adaptive Differential Evolution (DE) algorithm has been used to optimize the spacing between the elements of the linear array to produce a radiation pattern with minimum side lobe level and null placement control. DE is arguably one of the best real parameter optimizers of current interest takes very few control parameters and is easy to implement in any programming language. In this study two very simple adaptation schemes are used to regulate the control parameters *F* and *Cr*, upon which the performance of DE is critically dependent. The adaptation schemes are based on the objective function values of the target vectors and donor vectors. The adaptive DE-variant has been used to solve three difficult instances of the design problem and the optimization goal in each example is easily achieved. The results of the proposed algorithm have been shown to meet or beat the recently published results obtained using other state-of-the-art metaheuristics like the Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Memetic Algorithms (MA), and Tabu Search (TS) in a statistically meaningful way.**

Keywords: Antenna array, differential evolution, genetic algorithms, particle swarm optimization, , metaheuristics

I. Introduction

Antenna arrays play an important role in detecting and processing signals arriving from different directions. Compared with a single antenna that is limited in directivity and bandwidth, an array of sensors can have its beam-pattern modified with an amplitude and phase distribution called the weights of the array. After preprocessing the antenna outputs, signals are weighted and summed to give the antenna array beam-pattern. The antenna array pattern synthesis problem consists of finding weights that satisfy a set of specifications on the beam-pattern [1 – 3].

The goal in antenna array geometry synthesis is to determine the physical layout of the array that produces a radiation pattern that is closest to the desired pattern. The shape of the desired pattern can vary widely depending on the application. Many synthesis methods are concerned with suppressing the Side Lobe Level (SLL) while preserving the gain of the main beam [4]. Other methods deal with the null control to reduce the effects of interference and jamming. For the linear array geometry, this can be done by designing the spacing between the elements, while keeping a uniform excitation over the array aperture. Other methods of controlling the array pattern use non-uniform excitation and phased arrays [1].

It is well known that the classical derivative-based optimization techniques need a starting point that is reasonably close to the final solution, or they are likely to be stuck in a local minimum. As the number of parameters and hence the size of the solution space increases, the quality of the solution strongly depends on the estimation of the initial values. If the initial values fall in a region of the solution space where all the local solutions are poor, a local search is limited to finding the best of these poor solutions. The computational drawbacks of existing numerical methods have forced the researchers all over the world to rely on metaheuristic algorithms founded on simulations of some natural phenomena to solve antenna problems. These algorithms use an objective function, optimization of which leads to the side lobe suppression and null control [5]. Metaheuristic algorithms such as GAs [6 - 8], Simulated Annealing (SA) [9], Tabu Search [10], Taguchi's method [11], Memetic Algorithms (MAs) [12], and PSO [13, 14] have already used in the design of antenna arrays, e.g. see [5, 15 – 19].

The Differential Evolution (DE) [20 – 23] algorithm emerged as a very competitive form of evolutionary computing more than a decade ago. The performance of DE is severely dependent on two of its most important control parameters: The crossover rate (*Cr*) and scale factor (*F*) [23]. Over the past decade many claims and counter-claims have been reported regarding the tuning and adaptation strategies of these control parameters. Some objective functions are very sensitive to the proper choice of the parameter settings in DE [24]. Therefore, researchers naturally started to consider some techniques to automatically find an optimal set of control parameters for DE [25 – 30]. Most recent trend in this direction is the use of self-adaptive strategies like the ones reported in [28] and [30]. However, self-adaptation schemes usually make the

Aritra Chowdhury is with Department. of Electronics and Telecommunication Engineering, Jadavpur University , India
Ritwik Giri is with Department. of Electronics and Telecommunication Engineering, Jadavpur University , India
Arnob Ghosh is with Department. of Electronics and Telecommunication Engineering, Jadavpur University , India
Swagatam Das is with Department. of Electronics and Telecommunication Engineering, Jadavpur University , India
Ajith Abraham is with Machine Intelligence Resarch Labs (MIR Labs), USA and Faculty of Electrical Engineering and Computer Science, VSB Technical University of Ostrava, Czech Republic (ajith.abraham@ieee.org)
Vaclav Snasel is with the Faculty of Electrical Engineering and Computer Science, VSB Technical University of Ostrava, Czech Republic

programming fairly complex and run the risk of increasing the number of function evaluations. This article suggests a simple automatic tuning method for the scale factor and crossover rate of population members in DE, based on their individual fitness. The key sense of this adaptation mechanism is that if a search-agent (DE-vector) moves near to the optimum, its mutation step-size decreases and during crossover, it passes more genetic information to its offspring (*trial vector* in DE terminology) so as to favor exploitation. However, if the agent moves away from the optima, then it is more perturbed and during DE-type crossover, the offspring inherits lesser genetic information from the parent, so that the agent may be able to explore alternate regions quickly.

The proposed DE-variant is applied to three numerical instantiations of the linear array design problem have been used to illustrate the application of the algorithm. Comparison with the results obtained with other best known metaheuristics like GA, PSO, TS, MA etc. reflects the superiority of this modified DE in a statistically significant way.

The rest of the paper is organized as follows. A formulation of the array pattern synthesis as an optimization task has been discussed in Section 2. Section 3 provides a comprehensive overview of the Classical DE algorithm. Section 4 gives the details of the modification of DE. Experimental settings have been discussed and the results have been presented in Section 5. Section 6 finally concludes the paper and unfolds a few future research issues.

II. Problem Formulation and Design

An antenna array is a configuration of individual radiating elements that are arranged in space and can produce direction radiation pattern. For a linear antenna array, let us assume that there are $2N$ isotropic radiators placed symmetrically along the x-axis. The array geometry is shown in Figure 1.

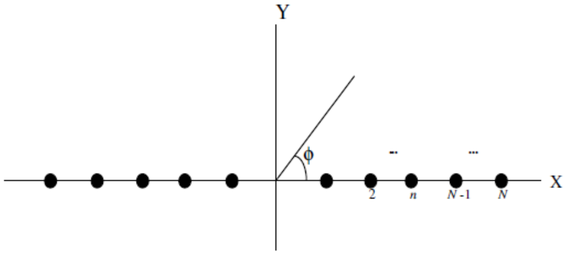

FIGURE 1: SYMMETRICALLY PLACED LINEAR ANTENNA

The radiation pattern of the array depends on the distance between the elements, the amplitudes and phase excitation of the elements and also the radiation of the individual elements [31]. Since the array elements are identical we can assume the radiation pattern of the array considering the sum of all contributing signals of individual elements. The above relation is often referred as *pattern multiplication*, which indicates that the total field of the array is equal to the product of the field due to a single element located at the origin and a factor called called array factor, AF, defined as

$$AF(\phi) = 2.\sum_{n=1}^{N}\left(I_n.\cos\left(k.x_n.\cos(\phi)+\phi_n\right)\right) \qquad (1)$$

where,

$k = \dfrac{2\pi}{\lambda}$ = wave number

I_n = Excitation magnitude of the n-th element

ϕ_n = Phase of the *n*-th element

x_n = Position of the *n*-th element

If we further assume a uniform excitation of amplitude and phase (i. e. $I_n = 1 = 1$ and $\phi_n = 0$ for all elements), the array factor can be further simplified as,

$$AF(\phi) = 2.\sum_{n=1}^{N}\cos\left[k.x_n.\cos(\phi)\right] \qquad (2)$$

Now the statement of the problem, addressed here simply reduces to: apply the DE algorithm to find the position x_n of the array elements that will result into an array beam with minimum Side Lobe Level (SLL) and if desired, nulls at specific locations. For side lobe suppression the objective function is,

$$f_1 = \sum_i \frac{1}{\Delta\phi_i}\int_{\phi_{li}}^{\phi_{ui}} \left|AF(\phi)\right|^2 d\phi \qquad (3)$$

And for null control we use:

$$f_2 = \sum_k \left|AF(\phi_k)\right|^2 \qquad (4)$$

To minimize both of them we use sum of (3) and (4) as the objective function of the proposed DE-variant. Note that in (3) $\Delta\phi_i$ represents the bandwidth to suppress where $\Delta\phi_i = \phi_{ui} - \phi_{li}$, and in (4) ϕ_k is the direction of the nulls.

III. Classical Differential Evolution – On Outline

DE is a simple real-coded evolutionary algorithm. It works through a simple cycle of stages, which are detailed below.

A. Initialization of the Parameter Vectors

It begins with a randomly initiated population of *NP D* dimensional real-valued parameter vectors. Each vector, also known as *genome/chromosome*, forms a candidate solution to the multi-dimensional optimization problem. We shall denote subsequent generations in DE by $G = 0,1...,G_{\max}$. Since the parameter vectors are likely to be changed over different generations, we may adopt the following notation for representing the *i*-th vector of the population at the current generation:

$$\vec{X}_{i,G} = [x_{1,i,G}, x_{2,i,G}, x_{3,i,G}, \ldots.., x_{D,i,G}]. \qquad (5)$$

The initial population (at $G = 0$) should cover the entire search space as much as possible by uniformly randomizing individuals within the search space constrained by the prescribed minimum and maximum bounds:

B. Mutation with Difference vectors

After initialization DE creates a *donor* vector $\vec{V}_{i,G}$, corresponding to each population member or *target* vector $\vec{X}_{i,G}$ in the current generation through mutation. It is the method of creating this donor vector, which differentiates between the various DE schemes. Among very commonly used five variants of mutation strategies very important one is :

"DE/best/1": $\vec{V}_{i,G} = \vec{X}_{best,G} + F \cdot (\vec{X}_{r_1^i,G} - \vec{X}_{r_2^i,G}).$ (6)

The indices $r_1^i, r_2^i$are mutually exclusive integers randomly chosen from the range [1, *NP*], and all are different from the index *i*. These indices are randomly generated once for each donor vector. The scaling factor *F* is a positive control parameter for scaling the difference vectors. $\vec{X}_{best,G}$ is the best individual vector with the best fitness (i.e. lowest objective function value for minimization problem) in the population at generation *G*.

C .Crossover

The donor vector exchanges its components with the target vector $\vec{X}_{i,G}$ under this operation to form the *trial* vector $\vec{U}_{i,G}$. The DE family of algorithms can use two kinds of crossover methods - *exponential* (or two-point modulo) and *binomial* (or uniform) [4]. In this article we focus on the widely used binomial crossover that is performed on each of the *D* variables. In this case, the number of parameters inherited from the donor has a (nearly) binomial distribution. The scheme may be outlined as:

$$u_{j,i,G} = \begin{cases} v_{j,i,G}, & \text{if } (rand_{i,j}[0,1) \le Cr \text{ or } j = j_{rand} \\ x_{j,i,G}, & \text{otherwise,} \end{cases} \qquad (7)$$

where, as before, $rand_{i,j}[0,1)$is a uniformly distributed random number, which is called a new for each *j*-th component of the *i*-th parameter vector. $j_{rand} \in [1,2,....,D]$ is a randomly chosen index, which ensures that $\vec{U}_{i,G}$ gets at least one component from $\vec{V}_{i,G}$.

D. Selection

The next step of the algorithm calls for *selection* to determine whether the target or the trial vector survives to the next generation i.e. at $G = G + 1$. The selection operation is described as:

$$\begin{aligned} \vec{X}_{i,G+1} &= \vec{U}_{i,G}, \quad \text{if } f(\vec{U}_{i,G}) \le f(\vec{X}_{i,G}) \\ &= \vec{X}_{i,G}, \quad \text{if } f(\vec{U}_{i,G}) > f(\vec{X}_{i,G}), \end{aligned} \qquad (8)$$

where $f(\vec{X})$ is the objective function to be minimized.

Note that throughout the article, we shall use the terms *objective function value* and *fitness* interchangeably. But, always for minimization problems, a lower objective function value will correspond to higher fitness.

IV. FITNESS BASED ADAPTATION OF DE (FiADE)

In this article we firstly aim at reducing *F* when the objective function value of any vector nears the minimum objective function value so far obtained in the population. In this case the vector is expected to suffer lesser perturbation so that it may undergo a fine search within a small neighborhood of the suspected optima. Equations (9) and (10) show two different schemes for varying the value of *F* for the *i*-th target vector and these schemes have been applied alternatively to determine the scale factor for each individual population member following a certain criteria to be discussed next.

Scheme 1: $F_i = 0.8 * \left(\dfrac{\Delta f_i}{\lambda + \Delta f_i} \right),$ (9)

where $\lambda = 1.00e - 14 + \Delta f_i / 10$ and $\Delta f_i = \left| f(\vec{X}_i) - f(\vec{X}_{best}) \right|$.

Clearly

$F_i = 0.8 * \left(\dfrac{\Delta f_i}{\lambda + \Delta f_i} \right) = 0.8 * \left(\dfrac{1}{1 + \lambda / \Delta f_i} \right) \to 0$ as $\Delta f_i \to 0$. Also when Δf_i is large $\lambda / \Delta f_i \to 0$ and consequently $F_i \to 0.8$.

Scheme 2: $F_i = 0.8 * \left(1 - e^{-\Delta f_i} \right)$ (10)

Figures 2 shows the variation of F with Δf in two different scales. As can be seen from Figure 2, the two plots intersect at approximately $\Delta f = 2.4$ .So from Figure 2, it is evident that as long as $\Delta f > 2.4$ scheme 2 results greater values of *F*, which helps the vector to explore larger search volume. But as soon as Δf falls below 2.4, scheme 1 starts reducing *F* drastically helping the vector to finely search the surroundings of the suspected optima.

Thus the adaptation of the scale factor for the *i*-th target vector takes place in the following way:

if $\Delta f_i > 2.4$,

$$F_i = 0.8 * \left(1 - e^{-\Delta f_i} \right),$$

else $F_i = 0.8 * \left(\dfrac{\Delta f_i}{\lambda + \Delta f_i} \right),$

Vectors that are distributed away from the current best vector in fitness-space have their *F* values large (due to scheme 2) and keeps on exploring the fitness landscape, maintaining adequate population diversity.

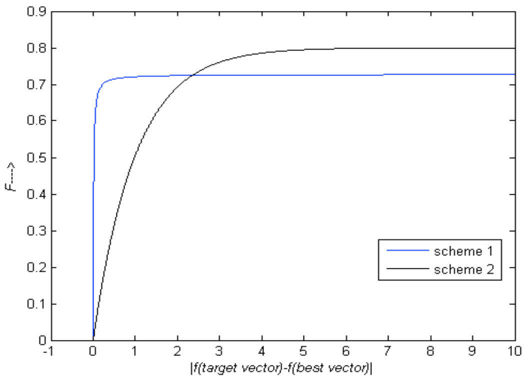

FIGURE 2: VARIATION OF F WITH Δf VARYING IN SCALE OF 1 TO 10 FOR SCHEMES 1 AND 2 OUTLINED IN EQUATIONS (9) AND (10).

Similarly we adapt the values of crossover rate *Cr* associated with each target vector according to the fitness of the donor vector produced. The key sense of this adaptation is that if *Cr* is higher, then more genetic information will be passed to the trial vector from the donor vector. So if the donor vector yields an objective function value lesser than the minimum value attained by the current population, then more of its features should be encompassed into the trial vector obtained after the crossover operation. If the solution of the donor vector is better, *Cr* should be larger; otherwise, it should be smaller. *Cr* updating strategy is divided into two cases: if donor's objective value is less than that of the best vector of the current population, a high constant value of *Cr* in equation (11a) is used, otherwise updating strategy in (11b) is used;

$$Cr_i = \begin{cases} Cr_{const}, & \text{(11a)} \\ Cr_{\min} + \dfrac{Cr_{\max}}{1+\Delta f_{donor\_i}}, & \text{(11b)} \end{cases}$$

where $\Delta f_{donor\_i} = \left| f(\vec{V}_i) - f(\vec{X}_{best}) \right|$, $Cr_{\min} = 0.1$ and $Cr_{\max} = 0.7$. Cr_{const} is fixed at 0.95.

Note that although calculating the objective function values of the donor vector increases the number of FEs per generation, as will be evident from the experimental results, the proposed adaptation scheme is still very much competitive against the latest and best known DE-variants, when all the algorithms are run for same maximum number of FEs. The adaptation scheme of *F* and *Cr* has been applied to the DE/best/1/bin algorithm and in what follows, we shall refer to this new DE-variant as FiADE (Fitness-Adaptive DE)

V. EXPERIMENTAL SET UP AND RESULTS

The sample problems on which the proposed algorithm (FiADE) is compared with other existing algorithms and a brief introduction of those algorithms is presented below.

A. Algorithms compared

The different algorithms with which the proposed algorithm has been compared are as follows:

1. DE/best/1/bin (DE)
2. Genetic Algorithm (GA)
3. Tabu Search Algorithm (TSA)
4. Particle Swarm Optimization Algorithm (PSO)
5. Memetic Algorithm (MA)

For the competitor algorithms PSO, GA, MA, and TSA, we used the best possible parametric setup as explained in the relevant literatures [5, 16]. Both DE/rand/1/bin and FiADE starts with same population with 50 vectors for each run on each problem. This is done to make the comparison fair enough, i.e. any difference in performance between FiADE and DE/rand/1/bin may be attributed to their internal search operators only. DE/rand/1/bin uses $F = 0.8$ and $Cr = 0.9$ over all tested problem instances.

B. Problem 1

In the first example a 12 element array is to be designed so that it has minimum side lobe level (SLL) in the band [0°,82°] and [98°,180°] and having no null direction. Following Figure 3 shows the Gain vs. Azimuth angle plot for this problem.

Table 1 shows the position coordinates of the array elements (normalized with respect to $\lambda/2$). Table 2 shows the mean objective function values and standard deviation obtained by different optimization techniques. A nonparametric statistical test called Wilcoxon's rank sum test (with 5% significance level) for independent samples has been done to check whether the final result obtained by the proposed algorithm differ in a statistically significant way from the result obtained by other competiting algorithms. P values obtained through the rank sum test between the best algorithm and each of the contestants over all the unconstrained problem instances are presented in the 3rd row of Table 2. Here *NA* stands for "*Not Applicable*" and occurs for the best performing algorithm itself in each case. The values corresponding to statistically insignificant value are marked bold. Here if P values are less than 0.05(5% significance level) then it is a strong proof against null hypothesis indicating that better final objective function values obtained by the best Algorithm I statistically significant and has not occurred by chance.

Table 2 also contains the maximum SLL values obtained by different algorithms. SLL_{max} is minimum for FiADE indicating that null suppression has been done best by FiADE. The directivity has also been calculated in decibel (db) for various algorithms and those values are listed in table 2. The directivity is defined as the ratio of the radiation intensity in a given direction to the antenna to radiation intensity averaged over all direction. Higher value of directivity indicates that the antenna radiates energy more in certain direction over the other. So in this problem higher value of directivity is desirable. Directivity obtained by the algorithm FiADE is maximum. So FiADE performs better in this case over the other considered algorithms.

Table 1: Geometry of the 12 element linear array normalized with respect to (median solution of 50 runs)

FiADE	±0.5110	±1.2036	±2.2486	±3.1370	±4.5307	±5.9999
DE/best/1/bin	±0.1327	±1.3386	±1.8480	±2.9350	±3.8348	±5.1019
GA	±0.5068	±1.2611	±2.2885	±3.1910	±4.5611	±5.9998
PSO	±0.2470	±1.3109	±1.9367	±2.9717	±3.9272	±5.1808
MA	±0.4819	±1.2781	±2.2881	±3.2357	±4.6025	±5.9990
TSA	±0.3881	±1.2588	±2.1076	±3.0525	±4.1881	±5.4507

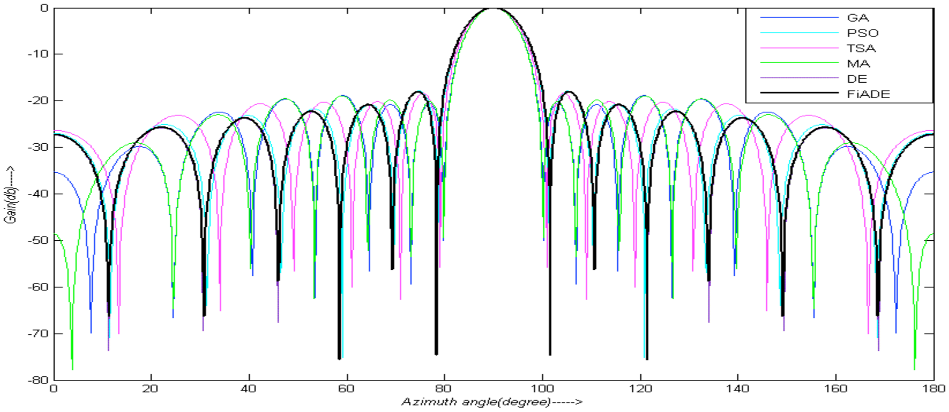

Figure 3: 12 element array for minimum SLL in bands $[0^o, 82^o]$ and $[98^o, 180^o]$ and no nulls

Table 2: Mean final objective function value, standard deviation, P-values for Wilcoxon's Test, SLL and Directivity

Algorithm	FiADE	DE/best/1/bin	GA	TSA	PSO	MA
Mean fitness	**0.008720**	0.010395	0.009061	0.011219	0.010843	0.009196
Std. Dev.	**0**	0.000284	0.000681	0.000712	0.000319	0.000521
P-values	NA	1.1894e-10	2.2917e-10	1.3493e-11	3.9034e-13	1.3493e-16
SLL_{max}(db)	**-13.1203**	-9.1696	-12.5396	-9.6334	-10.9892	-8.0410
Directivity(db)	**20.0300**	19.2668	19.9865	19.0554	19.2451	10.3722

C. Problem 2

In the second example a 22 element array is to be designed so that it has minimum side lobe level (SLL) in the bands [0^o, 82^o] and [98^o,180^o] and having nulls at 81^o and 99^o. Following Figure 4 shows the Gain vs. Azimuth angle plot for this problem. From Fig. 4 it is evident that FiADE algorithm has produced the minimized SLL to the greatest extent. In addition to it, it also produces the minimum gain value at the desired null direction in comparison with the other algorithms. Table 3 contains the array element positions in the array obtained through different algorithms and Table 4 lists the mean objective function value, standard deviation, the P values obtained through Wilcoxon's rank test and the maximum SLL and directivity.

Here for the proposed algorithm (FiADE) P value appears as *NA* indicating that it is the best performing algorithm. The maximum SLL obtained by FiADE is also the minimum among all, which means that the null suppression has been done best by FiADE. The directivity obtained by FiADE is also maximum, which indicates that the radiation intensity in a given direction is maximum and the energy radiated in other direction is minimum.

TABLE 3: GEOMETRY OF THE 22 ELEMENT LINEAR ARRAY NORMALIZED WITH RESPECT TO $\lambda/2$(MEDIAN SOLUTION OF 50 RUNS)

FiADE	±0.3634	±1.0660	±1.8542	±2.5501	±3.4666	±4.2949	±5.3570	±6.3996	±7.7696	±9.4318	±10.9011
DE/best/1/bin	±0.0773	±1.1668	±1.7876	±2.5822	±3.3893	±4.1471	±5.2329	±6.2659	±7.6473	±9.3065	±10.8308
GA	±0.0002	±1.061	±1.486	±2.419	±3.386	±4.285	±5.407	±6.846	±8.042	±9.136	±10.398
TSA	±0.6982	±1.071	±2.485	±2.541	±4.148	±5.479	±6.480	±7.573	±8.714	±10.211	±11.64
PSO	±0.3006	±1.177	±1.855	±2.685	±3.524	±4.428	±5.468	±6.580	±7.953	±9.552	±11.00
MA	±0.8113	±2.273	±3.157	±3.948	±4.770	±5.411	±6.432	±6.934	±7.896	±8.712	±10.124

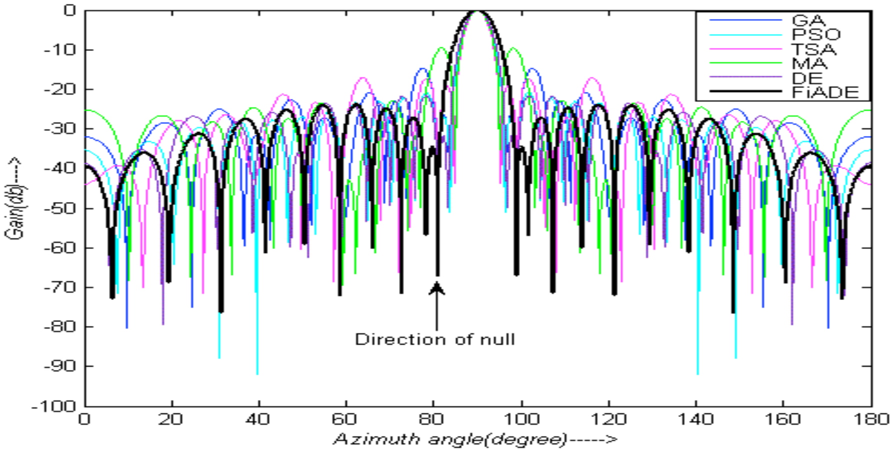

FIGURE 4: 22 ELEMENT ARRAY FOR MINIMUM SLL IN BANDS (0^0,82^0) AND (98^0, 180^0) WITH NULLS AT 81^0 AND 99^0

TABLE 4: MEAN OBJECTIVE FUNCTION VALUE, STANDARD DEVIATION, *P*-VALUE FOR WILCOXON'S RANK-SUM TEST, SLL, DIRECTIVITY PERTAINING TO PROBLEM 2

Algorithms	**FiADE**	**DE/best/1/bin**	**GA**	**TSA**	**PSO**	**MA**
Mean fitness	**0.01065**	0.011593	0.023662	0.021898	0.016755	0.021630
Std. Dev.	**0.001034**	0.001248	0.001201	0.001109	0.001613	0.001331
P-values	NA	3.2547e-10	6.1304e-12	5.5647e-12	9.6808e-15	8.0040e-17
SLL_{max} (db)	**-23.8866**	-21.5667	-14.6337	-20.6832	-17.2759	-23.5597
Directivity(db)	**24.4450**	24.1729	20.9657	24.4188	21.3125	19.3764

D. Problem 3

In the third example a 26 elements array is to be designed so that it has minimum side lobe level (SLL) in the bands [0^o, 80^o] and [100^o,180^o] and having nulls at 12^o,60^o,120^o, and 168^0. Following Figure 5 shows the Gain vs. Azimuth angle plot for this problem. Table 5 and 6 lists the mean objective function value, standard deviation and the P values obtained through Wilcoxon's rank test. From the Fig. 5 and Table 5 and 6 it is evident not only FiADE produces the best mean objective function value, but also it produces the minimum SLL and minimum gain value at the desired null directions. FiADE also yields the statistically better final accuracy than all its competitors. The maximum SLL obtained by FiADE is also the minimum among all, which means that the null suppression has been done best by FiADE and the directivity obtained by FiADE is maximum compared to other algorithms. So, these results establish the superiority of the proposed algorithm FiADE.

Table 5: Geometry of the 26 element linear array normalized with respect to (median solution of 50 runs)

FiADE	±0.429	±1.038	±1.835	±2.632	±3.322	±3.986	±4.781	±5.303	±6.328	±6.918	±7.950	±8.725	±9.907
DE/best/1/bin	±3.156	±4.462	±5.011	±5.834	±6.332	±6.836	±7.932	±8.053	±8.553	±9.106	±9.943	±10.44	±11.69
GA	±0.424	±0.847	±1.579	±2.468	±2.993	±4.391	±4.629	±5.640	±6.399	±7.791	±8.795	±9.974	±11.38
PSO	±0.579	±1.741	±2.806	±3.923	±4.885	±5.939	±7.100	±8.137	±9.171	±9.956	±10.75	±11.82	±13.00
TSA	±0.531	±1.491	±2.468	±3.445	±4.524	±5.501	±6.491	±7.268	±8.498	±9.514	±10.27	±11.44	±12.42
MA	±0.452	±0.851	±1.606	±2.497	±3.019	±4.397	±4.629	±5.687	±6.399	±7.792	±8.796	±9.976	±11.40

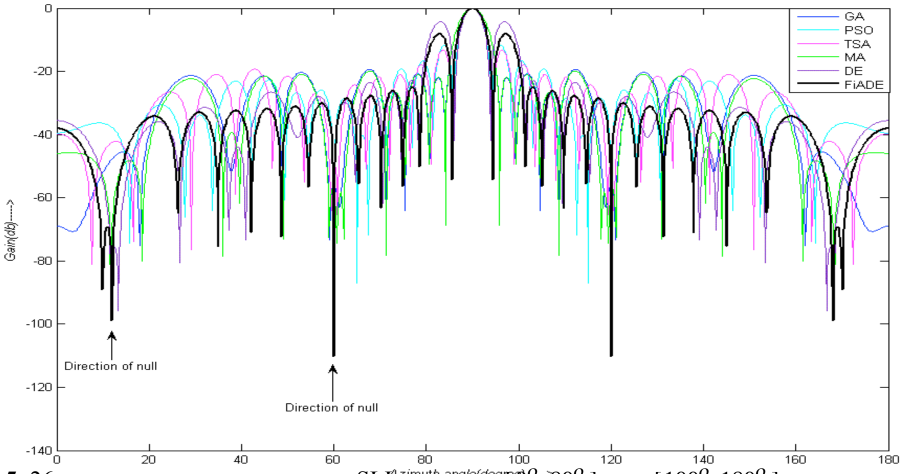

Figure 5: 26 element array for minimum SLL in bands $[0^o, 80^o]$ and $[100^o, 180^o]$ with nulls at 81^o, 99^o, 120^o, 168^o

Table 6: Mean objective function value, standard deviation, *P*-value for Wilcoxon's rank test, SLL, and Directivity

Algorithms	FiADE	DE/best/1/bin	GA	TSA	PSO	MA
Mean fitness	**0.0097**	0.010556	0.050279	0.061428	0.048131	0.027195
Std. Dev.	**0.000567**	0.000623	0.003163	0.008221	0.003331	0.001109
P-Values	NA	4.4826e-12	5.5647e-12	9.6808e-15	4.0040e-13	6.1304e-12
$\text{SLL}_{\max}$ (db)	**-22.8599**	-17.6550	-19.3634	-17.9091	-16.6816	-19.8770
Directivity(db)	**26.3671**	18.0029	21.4904	21.4840	23.3009	22.2302

VI. Conclusions

This paper illustrated the application of an improved adaptive DE-variant called FiADE in designing a non-uniform linear antenna arrays having suppressed side lobes and efficient null control in certain direction. FiADE has successfully found out the optimal array element location so that the array-pattern has either suppressed side lobe or null direction or both. Future research may focus on achieving more control of the array pattern using the FiADE algorithm to optimize not only the location, but also the excitation amplitude and phase of each element in the array, and exploring other array geometries.

Acknowledgement

This work was supported by the Czech Science Foundation under the grant no.102/09/1494.

References

[1] *Handbook of Antennas in Wireless Communications*, L. C. Godara, Ed., CRC, Boca Raton, FL, 2002.

[2] O.M. Bucci, D. D'Elia, G. Mazzarella, and G. Panatiello, Antenna pattern synthesis: A new general approach. *Proc. IEEE*, 82: 358-371, 1994.

[3] Y. Rahmat-Samii and E. Michielssen Eds., *Electromagnetic Optimization by Genetic Algorithms*. New York: Wiley, 1999.

[4] H. Lebret and S. Boyd, "Antenna array pattern synthesis via convex optimization," *IEEE*

Transactions on Signal Processing, Vol. 45, No. 3, March 1997.

[5] M. M. Khodier and C. G. Christodoulou, “Linear array geometry synthesis with minimum side lobe level and null control using particle swarm optimization,” *IEEE Transactions on Antennas and Propagation*, Vol. 53, No. 8, August 2005.

[6] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Harbor, 1975.

[7] T. Bäck, D. Fogel, Z. Michalewicz, *Handbook of Evolutionary Computation*, Oxford Univ. Press, 1997.

[8] A.E. Eiben and J.E. Smith, *Introduction to Evolutionary Computing*, Springer, 2003.

[9] S. Kirkpatrik, C. Gelatt, and M. Vecchi, Optimization by Simulated Annealing. *Science*, 220: 671–680, 1983.

[10] F. Glover and M. Laguna, *Tabu Search*, Kluwer, Norwell, MA, 1997.

[11] G. Taguchi, S. Chowdhury, and Y.Wu*, Taguchi’s Quality Engineering Handbook*. New York: Wiley, 2005.

[12] Y.-S. Ong, and A. J. Keane, “Meta-lamarckian learning in memetic algorithms,” IEEE *Transactions on Evolutionary Computation*, vol. 8, no. 2, pp. 99–110, 2004.

[13] J. Kennedy and R. Eberhart, “Particle swarm optimization,” *Proc. IEEE Int. conf. Neural Networks,* pp.1942-1948, 1995.

[14] J. Kennedy, R. C. Eberhart, and Y. Shi, *Swarm Intelligence*, Morgan Kaufmann, San Francisco, CA, 2001.

[15] A. Udina, N. M. Martin, and L. C. Jain, “Linear antenna array optimization by genetic means,” *Third International Conference on Knowledge-Based Intelligent Information Engineering Systems Adelaide*, Australia, Sept. 1999.

[16] Y. Cengiz and H. Tokat, Linear Antenna Array Design With use of Genetic, Memetic and Tabu Search Optimization Algorithms, *Progress In Electromagnetics Research (PIER) C*, Vol. 1, 63–72, 2008.

[17] W-C. Weng, F. Yang, and A. Z. Elsherbeni, Linear Antenna Array Synthesis Using Taguchi’s Method: A Novel Optimization Technique in Electromagnetics, *IEEE Transactions on Antennas and Propagation*, Vol. 55, No. 3, pp. 723 – 730, March 2007.

[18] F. J. Ares-Pena, A. Rodriguez-Gonzalez, E. Villanueva-Lopez, and S. R. Rengarajan, “Genetic algorithms in the design and optimization of antenna array patterns,” *IEEE Transactions on Antennas and Propagation*, vol. 47, pp. 506–510, Mar. 1999.

[19] Y. B. Tian and J. Qian, “Improve the performance of a linear array by changing the spaces among array elements in terms of genetic algorithm,” *IEEE Transactions on Antennas and Propagation.*, vol. 53, pp. 2226–2230, Jul. 2005.

[20] R. Storn and K. V. Price, “Differential Evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces”, *Technical Report TR-95-012,ICSI*,

[21] R. Storn and K. V. Price, “Minimizing the real functions of the ICEC 1996 contest by differential evolution”, *Proceedings of the 1996 IEEE international conference on evolutionary computation*, Nagoya, Japan, pp. 842–844. IEEE Press, New York, 1996

[22] R. Storn and K. Price, “Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces”, *Journal of Global Optimization*, 11(4) 341–359,1997

[23] K. Price, R. Storn, and J. Lampinen, *Differential evolution – A Practical Approach to Global Optimization,* Springer, Berlin, 2005

[24] J. Liu and J. Lampinen, “On setting the control parameters of the differential evolution method”, in: R. Matoušek and P. Ošmera, (eds.) *Proc. of Mendel* 2002, 8-th *International Conference on Soft Computing*, pp. 11–18, 2002

[25] J. Liu and J. Lampinen, “Adaptive parameter control of differential evolution”, : R. Matoušek and P. Ošmera, (eds.) *Proc. of Mendel* 2002, 8-th *International Conference on Soft Computing*, pp. 19–26, 2002.

[26] J. Liu and J. Lampinen, “A Fuzzy adaptive differential evolution algorithm”*, Soft computing- A Fusion of Foundations*, *Methodologies and Applications*, Vol. 9, No. 6, pp. 448-462, 2005

[27] J. Rönkkönen and J. Lampinen, “On using normally distributed mutation step length for the differential evolution algorithm”, 9th *Int. Conf. Soft Computing* (*MENDEL* 2003), Brno, Czech Republic, June 5-7, 2002, pp. 11–18, 2003

[28] A. K. Qin, V. L. Huang, and P. N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization", *IEEE Transactions on Evolutionary Computation,* Vol. 13, Issue 2,, pp. 398-417, April, 2009.

[29] M. M. Ali and A. Törn, “Population set based global optimization algorithms: some modifications and numerical studies,” *Computers and Operations Research, Elsevier*, no. 31, pp. 1703–1725, 2004.

[30] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Žumer, “Self-adapting Control parameters in differential evolution: a comparative study on numerical benchmark problems,” *IEEE Transactions on Evolutionary Computation*, Vol. 10, Issue 6, pp. 646 – 657, 2006

[31] B.D. Van Veen, and K.M. Buckley, “Beamforming: A versatile approach to spatial filtering”, IEEE Acoust, Speech Signal Process. Mag. 5, 4-24, 1988.