Research Article

Antenna Array Synthesis and Failure Correction Using Differential Search Algorithm

Kerim Guney, 1 Ali Durmus, 2 and Suad Basbug 3

1 Faculty of Engineering, Nah Naci Yazgan University, 38040 Kayseri, Turkey
2 Department of Electricity and Energy, Vocational College, Erciyes University, 38039 Kayseri, Turkey
3 Department of Computer Technologies, Vocational College, Nevsehir University, 50300 Nevsehir, Turkey

Correspondence should be addressed to Ali Durmus; alidurmus@gmail.com

Received 24 October 2013; Accepted 31 January 2014; Published 6 March 2014

Academic Editor: Ahmed A. Kishk

Copyright © 2014 Kerim Guney et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Differential search (DS) optimization algorithm is proposed for the synthesis of three different types of linear antenna array design examples. The first group of examples is that DS algorithm is used to locate wide nulls on the linear antenna array patterns by controlling amplitude-only. In these examples, sidelobe levels disposed to rise are also suppressed by using DS algorithm in the same optimization process. In the second group of examples, individual nulls are placed with the help of DS algorithm by controlling the amplitude-only, phase-only, and position-only. The last example is a linear antenna array failure correction example. In order to tolerate the element failures, DS is employed to recalculate the amplitude values of the remaining intact elements of the antenna array. The results show that DS is very capable to solve the linear antenna array optimization problems which have different characteristics.

1. Introduction

Antenna arrays find usage in a large number of application areas, particularly in communication and radar systems. One of the well-known advantages of antenna arrays is that they can be synthesized in a very flexible manner [1]. The main target of antenna array synthesis is to achieve desired radiation pattern by controlling amplitudes, phases, and positions of the antenna array elements. The elimination of unwanted signals which can cause interference is probably the most important ability of antenna array synthesis techniques. There are different kinds of studies on the interference cancellation. In some cases, the arrival direction of interference signal is not exactly known or its location is not permanent and may be slowly changed in time. The wide nulls on the sidelobe region of the array pattern can be required in these kinds of circumstances. If the interference direction is known, an individual null is placed at this certain direction to cancel interference and consequently to increase signal-to-noise ratio. By using this last method, it is possible to achieve very deep null levels. Nulling operations can be performed by controlling amplitude-only [2–14], phase-only [12, 13, 15–21], and position-only [12, 13, 22–28]. The synthesis techniques are not only used for the antenna arrays in good condition but also for the antenna arrays which have failed elements. In some cases, the perturbation on the pattern caused by failed antenna array elements can be tolerated by recalculating the other intact array elements [29–32].

Both deterministic and stochastic optimization algorithms have been widely used for the synthesis of antenna arrays. However, the popularity of the stochastic algorithms has been increased since they are more flexible than their counterparts. Additionally, the stochastic algorithms are very skillful to escape from local minima. Several versions of stochastic algorithms are utilized to synthesize antenna arrays in the literature. Genetic algorithm (GA) [3, 16, 17, 30, 31], differential evolution (DE) algorithm [6, 27], particle swarm optimization (PSO) algorithm [14, 23], tabu search algorithm [4, 18], touring ant colony algorithm [5, 22], clonal selection algorithm [7, 19, 24, 26], bees algorithm [8, 21, 25], bacterial foraging algorithm (BFA) [9, 20, 28], immune algorithm [10], plant growth simulation algorithm [11], seeker optimization
algorithm [12], harmony search algorithm [13], and firefly algorithm (FA) [32] can be given as the examples of these stochastic optimization algorithms.

In this paper, differential search algorithm (DS) [33] is employed to solve three different types of antenna array optimization problems. The first is the wide nulling on the array pattern by controlling amplitude-only. The second group of optimization problem is the single nulling on the array pattern by controlling amplitude-only, phase-only, and position-only. The failure correction is the last kind of optimization problem presented in this paper. DS originally proposed in [33] is a stochastic search optimization algorithm that mimics the migration behaviors of organisms which use the Brownian-like random-walk movement. In DS algorithm, the population members are represented by artificial organisms. The artificial organisms change their positions in the solution space by so-called migration movement. The population members stay on their new positions temporarily in an iteration cycle. They make their decision about whether or not to stay on the new position in regard to the cost function. This movement goes on iteratively until the stopping criteria are satisfied. In [33], the DS algorithm was used to solve the problem of transforming the geocentric cartesian coordinates into geodetic coordinates and its performance was compared with the performances of eight computational intelligence algorithms which consist of artificial bee colony, self-adaptive DE, adaptive DE, strategy adaptation based DE, DE with ensemble of parameters, gravitational search algorithm, PSO, and covariance matrix adaptation evolution strategy, through use of a set of test data containing 100,000 test points. The statistical tests realized for the comparison of performances indicated that the problem-solving success of DS algorithm in transforming the geocentric Cartesian coordinates into geodetic coordinates is higher than that of computational intelligence algorithms. In [34], DS was used to solve two famous economic dispatch problems with practical constraints for 6-unit and 15-unit systems. The results showed that DS can achieve more economical solutions than PSO, self-organizing hierarchical PSO, GA, BFA, FA, and modified FA.

2. Problem Formulation

The array factor of a linear antenna array in azimuth plane (x-y plane) with M elements placed along the x-axis is given by

\[
AF(\theta) = \sum_{m=1}^{M} I_m e^{j(kd_m \sin \theta + \delta_m)},
\]

where \( \theta \) is the scanning angle from broadside, \( k \) is the wavenumber (\( k = 2\pi / \lambda \)), and \( I_m, d_m, \) and \( \delta_m \) are the amplitude, location, and phase of the mth element, respectively. The array elements are considered as isotropic sources.

The array factor can be written as the following format in case that the antenna array elements are symmetrically located and excited around the center of the linear array:

\[
AF(\theta) = 2 \sum_{m=1}^{H} I_m \cos (kd_m \cos \theta + \delta_m),
\]

where the number of array elements is even and \( H = M/2 \).

The first cost function used for the wide nulling and failure correction examples in this paper is formulated as

\[
C_1 = C_{\text{MSL}} + C_{\text{FNBW}} + C_{\text{NULL}},
\]

where \( C_{\text{MSL}} \) is the function used to suppress the maximum side lobe level (MSL) values; \( C_{\text{FNBW}} \) function makes sure that the obtained the first null beam width (FNBW) value is confined in the range of determined maximum FNBW values; \( C_{\text{NULL}} \) is used to locate single or wide nulls on sidelobes. The function \( C_{\text{MSL}} \) is given as

\[
C_{\text{MSL}} = \int_{-\pi/2}^{\theta_{\text{null}1}} \chi_{\text{MSL}}(\theta) d\theta + \int_{\theta_{\text{null}2}}^{\pi/2} \chi_{\text{MSL}}(\theta) d\theta,
\]

where \( \theta_{\text{null}1} \) is the first null on the left side of the main beam and \( \theta_{\text{null}2} \) is the first null on the right side of the main beam. \( \chi_{\text{MSL}}(\theta) \) function is used to obtain the values which exceed the desired MSL (\( MSL_d \)) in the sidelobe region and it can be calculated by the following expression:

\[
\chi_{\text{MSL}}(\theta) = \begin{cases} 
(20 \log\left( AF(\theta)_{\text{normalized}} \right) - MSL_d)^2, & \text{for } 20 \log\left( AF(\theta)_{\text{normalized}} \right) > MSL_d\\ 
0, & \text{elsewhere}.
\end{cases}
\]

The FNBW function in (3) is given by

\[
C_{\text{FNBW}} = \begin{cases} 
(FNBW_{o} - C_{\text{FNBWmax}})^2, & \text{for } FNBW_{o} > C_{\text{FNBWmax}}\\ 
0, & \text{elsewhere},
\end{cases}
\]

where \( FNBW_{o} \) is the obtained FNBW value by DS and \( C_{\text{FNBWmax}} \) is the desired maximum FNBW.

In this paper two different nulling methods are presented: wide nulls and single nulls. In order to produce wide nulls, \( C_{\text{NULL}} \) in (3) can be written as follows:

\[
C_{\text{NULL}}(\theta) = \begin{cases} 
(20 \log\left( AF(\theta)_{\text{normalized}} \right) - \text{NULL}_d)^2, & \text{for } 20 \log\left( AF(\theta)_{\text{normalized}} \right) > \text{NULL}_d\\ 
0, & \text{elsewhere},
\end{cases}
\]

where \( \text{NULL}_d \) is the null depth level (NDL) which is desired at the predetermined angle. To produce wide nulls, \( C_{\text{NULL}} \) function is used for the every angle in the range of the wide null region and the total values achieved by \( C_{\text{NULL}} \) are added to the cost function.
(1) Set the initial random positions for the superorganism members
(2) Evaluate the cost functions for each members.
(3) while (stopping criterion is not satisfied)
(4) {
(5) Select the donor members by a random shuffling mechanism.
(6) Produce the scale value by using a gamma random number generator (10)
(7) Determine the Stopover Sites by using (11)
(8) Determine the population members to be used for the stopover process.
(9) Set new random positions in the solution space for the members which exceed the boundaries.
(10) }
(11) Display the best result.

Algorithm 1: Pseudocode of DS algorithm.

In this paper, the following second cost function will be used for the single nulls produced by controlling amplitude-only, phase-only, and position-only:

\[ C_2 = \sum_{\theta=\theta_{\min}}^{\theta_{\max}} [W(\theta)|\text{AF}_g(\theta) - \text{AF}_{\text{d}}(\theta)| + \text{ESL}(\theta)] , \]  

where \( \text{AF}_g(\theta) \) function produces the values obtained by using DS and \( \text{AF}_{\text{d}}(\theta) \) gives the desired array pattern values. \( W(\theta) \) and ESL(\( \theta \)) are employed to control the NDL and MSL, respectively.

3. Differential Search Algorithm

Superorganism is a collection of single creatures that together possess the functional organization implicit in the formal definition of organism \([33, 35]\). The superorganism having many individuals migrates to fruitful lands when they need much nutrition, usually on a seasonal basis. Some migration movements can be explained by Brownian-like random-walk. Superorganism continuously checks new locations to find more fertile areas. If a more fertile location is detected, the superorganism moves itself to this location.

In DS algorithm \([33]\), a superorganism is a population which consists of solution members distributed randomly to the solution space. This artificial superorganism tries to find the global minimum value in the solution space by performing migration movement. The superorganism is always in search of new locations which includes better solutions (much food) during the migration process. Searching activity mainly depends on a stochastic mechanism. The individual of the superorganism which finds better location than the previous one moves to this new location temporarily until the global minimum is found.

Artificial organisms as the members of the superorganism are denoted by \( X_i, i = 1, 2, 3, \ldots, P \), where \( P \) is the number of population. The iteration index is represented by \( g = 1, 2, 3, \ldots, G \), where \( G \) is the maximum iteration number. Assuming that \( D \) is the size of the problem, every individual of the superorganism can be described as \( x_{i,j}, j = 1, 2, 3, \ldots, D \).

The initial random values of the organisms are given as the following formulation:

\[ x_{i,j} = \text{rand} \cdot (\text{up}_j - \text{low}_j) + \text{low}_j, \]  

where \( \text{up}_j \) and \( \text{low}_j \) are upper and lower values of the solutions, respectively. In this way, the initial values are always in the range of permitted values. \( \text{rand} \) is a random number distributed uniformly.

In order to describe DS algorithm, one more term should also be defined: stopover sites. The locations where the organisms stay until finding better places are named as stopover sites. A donor is randomly selected from the population to discover new stopover sites. A scale value is needed to control the step size of superorganism. The scale value is produced as the following:

\[ \text{Scale} = \text{randg} \cdot [2 \cdot \text{rand}_{d1} \cdot (\text{rand}_{d2} - \text{rand}_{d3})], \]  

where \( \text{randg} \) is a gamma random number generator. \( \text{rand}_{d1}, \text{rand}_{d2}, \text{and rand}_{d3} \) functions generate uniform random numbers in the range of \([0, 1]\). After producing the scale value, the new stopover site is calculated as follows:

\[ \text{StopoverSite} = \text{Superorganism} + \text{Scale} \times (\text{donor} - \text{Superorganism}). \]

If the new stopover is out of the range, a random location is assigned as stopover location instead of the calculated value. If the new stopover site is better than the current location, superorganism moves to the new stopover site. Therefore, the members of the superorganism perform the searching activity until the global minimum is found. A pseudocode of the DS algorithm is shown in Algorithm 1. The detailed information about DS algorithm can be found in [33].

4. Numerical Results

In this paper, in order to test the capabilities of DS algorithm, three different kinds of examples are given. The first group of examples is to place wide nulls on the array pattern by controlling amplitude-only. For the second group of
Table 1: FNBW, MSL, NDL, and DRR values obtained by DS and CRPSO [14] for the first example (12 elements).

<table>
<thead>
<tr>
<th></th>
<th>FNBW (deg.)</th>
<th>MSL (dB)</th>
<th>NDL (dB)</th>
<th>DRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS</td>
<td>36.50</td>
<td>-18.37</td>
<td>-45.33</td>
<td>5.50</td>
</tr>
<tr>
<td>CRPSO [14]</td>
<td>38.84</td>
<td>-17.22</td>
<td>-41.78</td>
<td>5.52</td>
</tr>
</tbody>
</table>

Table 2: FNBW, MSL, NDL, and DRR values obtained by DS and CRPSO [14] for the second example (16 elements).

<table>
<thead>
<tr>
<th></th>
<th>FNBW (deg.)</th>
<th>MSL (dB)</th>
<th>NDL (dB)</th>
<th>DRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS</td>
<td>29.50</td>
<td>-20.96</td>
<td>-53.30</td>
<td>15.17</td>
</tr>
<tr>
<td>CRPSO [14]</td>
<td>30.60</td>
<td>-17.49</td>
<td>-53.04</td>
<td>Infinitive</td>
</tr>
</tbody>
</table>

Table 3: FNBW, MSL, NDL, and DRR values obtained by DS and CRPSO [14] for the third example (20 elements).

<table>
<thead>
<tr>
<th></th>
<th>FNBW (deg.)</th>
<th>MSL (dB)</th>
<th>NDL (dB)</th>
<th>DRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS</td>
<td>22.50</td>
<td>-19.65</td>
<td>-47.75</td>
<td>8.24</td>
</tr>
<tr>
<td>CRPSO [14]</td>
<td>23.02</td>
<td>-17.61</td>
<td>-45.56</td>
<td>57.88</td>
</tr>
</tbody>
</table>

4.1. Wide Nulls. DS is used for the synthesis of antenna array patterns with wide nulls by controlling amplitude-only values of the array elements. Three different linear antenna arrays which have 12, 16, and 20 elements with \(\lambda/2\) interelement spacing are considered for these three examples. The optimization processes take approximately 125, 165, and 207 seconds for the linear antenna arrays with 12, 16, and 20 elements, respectively. The function in (3) is used as cost function in the simulations.

The antenna array used in the first example simulations has 12 equispaced elements. DS is used to calculate the amplitude values of array elements in order to achieve an array pattern having wide nulls in the range of \([46^\circ, 71^\circ]\) and \([108^\circ, 134^\circ]\). The array pattern obtained by DS is compared with the pattern of craziness PSO (CRPSO) [14] in Figure 1. FNBW, MSL, NDL, and dynamic range ratio (DRR = \(|I_{\text{max}}/I_{\text{min}}|\)) values obtained by DS and CRPSO [14] are tabulated in Table 1. In Figure 1 and Table 1, it can be seen that FNBW, MSL, and NDL values achieved by DS are better than those of CRPSO [14], whereas DRR values are about the same for both algorithms.

In the second example, an antenna array with 16 equispaced elements is considered. The targeted null ranges of the array pattern are in \([60^\circ, 75^\circ]\) and \([105^\circ, 120^\circ]\). Figure 2 shows the patterns obtained by DS and CRPSO [14]. FNBW, MSL, NDL, and DRR values achieved by DS and CRPSO [14] are shown in Table 2. In Figure 2 and Table 2, it is very clear that FNBW, MSL, and DRR values achieved by DS are better than those of CRPSO [14], whereas NDL values calculated by both DS and CRPSO [14] are about the same.

The linear antenna array utilized in the simulations has 20 equispaced elements for the third example. The desired wide null ranges were determined as in \([66^\circ, 78^\circ]\) and \([101^\circ, 113^\circ]\). The patterns obtained by using DS and CRPSO [14] are illustrated in Figure 3. FNBW, MSL, NDL, and DRR values calculated by DS and CRPSO [14] are shown in Table 3. In Figure 3 and Table 3, it can be said that FNBW, MSL, NDL, and DRR values achieved by DS are better than those of CRPSO [14].
Figure 3: The radiation patterns with wide nulls at [66°, 78°] and [101°, 113°] obtained by DS and CRPSO [14] for the third example (20 elements).

Figure 4: Radiation pattern obtained by amplitude-only control with one imposed null at 104°.

Table 4: The amplitude values calculated by DS for the antenna arrays with 12, 16, and 20 elements.

<table>
<thead>
<tr>
<th>Arrays</th>
<th>Amplitude values</th>
</tr>
</thead>
<tbody>
<tr>
<td>12 elements</td>
<td>1.0000, 0.9996, 0.6034, 0.6055, 0.1816, and 0.1891</td>
</tr>
<tr>
<td>(first example)</td>
<td></td>
</tr>
<tr>
<td>16 elements</td>
<td>0.9745, 0.9999, 0.8518, 0.5440, 0.5587, 0.3664, 0.0659, and 0.1960</td>
</tr>
<tr>
<td>(second example)</td>
<td></td>
</tr>
<tr>
<td>20 elements</td>
<td>0.6604, 0.9929, 0.5991, 0.6066, 0.4668, 0.4443, 0.3396, 0.1298, 0.1205, and 0.1370</td>
</tr>
<tr>
<td>(third example)</td>
<td></td>
</tr>
</tbody>
</table>

The amplitude values calculated by DS in the first, second, and third examples are tabulated in Table 4. These examples clearly show the capacity of DS to synthesize the array pattern with wide nulls imposed at the direction of interference.

4.2. Single Null. DS is used to synthesize three different antenna array patterns with single nulls by controlling the amplitude-only, phase-only, and position-only of the array elements. For this aim, a 30 dB Chebyshev pattern of an antenna array having 20 equispaced isotropic elements with λ/2 interelement spacing is used as initial pattern for the simulations. The time consumption for all optimization processes of the single null examples is about 202 seconds. The cost function in (8) is employed for these three examples.

In the fourth example, the pattern nulling is achieved by controlling only the element amplitudes. The nulling direction θ_i is chosen as 104°. The cost function parameters given in (8) are defined as follows:

\[
AF_d(\theta) = \begin{cases} 
0, & \text{for } \theta = \theta_i \\
\text{Initial pattern}, & \text{elsewhere},
\end{cases}
\]

\[
W(\theta) = \begin{cases} 
200, & \text{for } \theta = \theta_i \\
1, & \text{elsewhere},
\end{cases}
\]

\[
ESL(\theta) = \begin{cases} 
50, & \text{if MSL} > -29 \text{ dB} \\
0, & \text{elsewhere}.
\end{cases}
\]

The pattern with null at 104° obtained by DS is shown in Figure 4. Due to the symmetrical structure of the array, another null with the same NDL can also be observed at 76°. The NDL and MSL values of the pattern in Figure 4 are −133 dB and −29.33 dB, respectively.

DS is utilized to obtain an optimized pattern with a single null at θ_i = 80° by controlling the phase-only in the fifth example. MSL is restricted with −24 dB. The other cost function definition parameters are determined as in the fourth example. Figure 5 shows the pattern achieved by using DS. It can be seen from Figure 5 that there is a single null on the pattern. The NDL value of this null is −182.5 dB and the MSL value of the pattern is −24.0 dB.

For the sixth example, the element positions of the array are calculated by DS algorithm to locate a null at θ_i = 105° on the array pattern. MSL value is limited by −28 dB. The pattern
Table 5: Antenna arrays parameters values obtained by DS for the fourth, fifth, and sixth examples.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Parameters values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amplitude (fourth example)</td>
<td>1.00000, 0.98124, 0.94043, 0.86849, 0.75833, 0.61671, 0.46289, 0.32721, 0.22770, and 0.29669</td>
</tr>
<tr>
<td>Phase (deg.) (fifth example)</td>
<td>80.62, 162.75, 182.03, 175.35, 113.63, −24.90, −210.79, −165.96, −176.45, and −166.45</td>
</tr>
<tr>
<td>Position (Å) (sixth example)</td>
<td>0.24982, 0.76064, 1.25961, 1.74163, 2.21348, 2.67524, 3.14991, 3.65736, 4.20858, and 4.75772</td>
</tr>
</tbody>
</table>

Table 6: MSL, DRR, and FNBW values obtained by DS and FA [32].

<table>
<thead>
<tr>
<th>Parameter</th>
<th>DS</th>
<th>FA [32]</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSL (dB)</td>
<td>−35.02</td>
<td>−34.19</td>
</tr>
<tr>
<td>DRR</td>
<td>45.74</td>
<td>49.75</td>
</tr>
<tr>
<td>FNBW (deg.)</td>
<td>19</td>
<td>19</td>
</tr>
</tbody>
</table>

Table 7: Corrected amplitude weights obtained by DS.

<table>
<thead>
<tr>
<th>Amplitude values</th>
</tr>
</thead>
<tbody>
<tr>
<td>32 elements</td>
</tr>
<tr>
<td>(seventh example)</td>
</tr>
<tr>
<td>1.00000, 0.9663, 0.8956, 0.8024, 0.6856, 0.5615, 0.4367, 0.3286, 0.2192, 0.1773, 0, 0, 0.0219, 0, 0, and 0</td>
</tr>
</tbody>
</table>

obtained by DS is shown in Figure 6. Due to the symmetry, it is possible to see that there is another null generated at 75°. The NDL and MSL values obtained by DS algorithm are −151.7 dB and −28.11 dB, respectively.

Table 5 presents the amplitude, phase, and position values calculated by DS in the fourth, fifth, and sixth examples, respectively. From the null depth and the maximum side lobe level points of view, the performances of the patterns in these examples are very good. The nulling technique based on DS algorithm preserves the characteristics of the initial Chebyshev pattern with little pattern disturbance except for the nulling directions.

4.3. Failure Correction. For the seventh example, a 35 dB Chebyshev antenna array having 32 equispaced elements with \( \lambda/2 \) interelement spacing is considered. The function in (3) is used as a cost function in this simulation. It is assumed that 1st, 2nd, 3rd, 5th, 6th, 27th, 28th, 30th, 31th, and 32th elements of the antenna array failed. Under this condition, the pattern of the antenna array is disturbed by raising its MSL value to −21.29 dB. DS algorithm is used to recalculate the values of the remaining intact elements in order to compensate for the element failures. The optimization process of the failure correction example takes about 325 seconds. Figure 7 shows the patterns of the Chebyshev array, failed array, and corrected arrays by DS. It can be seen in Figure 7 that DS succeeded to suppress the MSL value in an acceptable level. In Table 6, The MSL, DRR, and FNBW values of the pattern obtained by using DS are compared with those of the pattern obtained by using FA [32]. It is clear from Figure 7 and Table 6 that MSL and DRR values obtained by DS are better than those of FA whereas their FNBW values are the same. The amplitude values corrected by DS are tabulated in Table 7.

5. Conclusions

In this paper, DS optimization algorithm is employed to solve three different types of antenna array problems. In the first group of examples, wide nulls on the array patterns are attained by using DS algorithm. In the same nulling concept, for the second group of examples, DS is also utilized in order to achieve single nulls on the array patterns by controlling amplitude-only, phase-only, and position-only. In the last example, DS is used to mitigate the negative effects of the failed elements on the array pattern by optimizing amplitude values of the remaining intact elements. It can be concluded that DS has capability of solving different types of antenna array synthesis problems. DS can be used as an alternative to other antenna array synthesis algorithms.
Conflict of Interests
The authors declare that there is no conflict of interests regarding the publication of this paper.

References


Submit your manuscripts at
http://www.hindawi.com