Proceedings of the 9th International Conference on Neural Information Processing (ICONIP'02), Vol. 2
Lipo Wang, Jagath C. Rajapakse, Kunihiko Fukushima, Soo-Young Lee, and Xin Yao (Editors)

THE COMBINATION OF NEURAL NETWORKS AND GENETIC ALGORITHM FOR FAST AND FLEXIBLE WIDE NULLING IN DIGITAL BEAMFORMING

Yan Wang and Yilong Lu

School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore 639798

ABSTRACT

The paper presents an approach of using Neural Networks to apply slow genetic algorithm solutions for real time applications. Genetic Algorithms (GAs) are powerful optimization tools which have been applied to solve many complicated problems in an very wide range of areas. However, GA's slowness prevent it from being used in real-time systems. In this paper, radial basis function neural network (RBFNN) is exploited to approximate the generic algorithm's function. This GA-RBFNN approach makes powerful yet slow genetic algorithm solutions possible for real-time problems. As an example, we have successfully applied the proposed approach for fast solution of the GA based wide nulling problem in adaptive digital beamforming.

1. INTRODUCTION

Genetic Algorithms (GAs) are stochastic search algorithms that mimic the process of natural selection and genetics. They were originally developed by Holland to study the adaptive process of natural systems and to design software systems with adaptive behavior. In the simplest form, a genetic algorithm requires a string representation to code the parameter space, a fitness function to evaluate the strings, a set of genetic operators to create new strings, and a set of probabilities to control the genetic operators.

Since early 1990s, GAs have received ever increasing interests as effective tools for complicated problems that are difficult or impossible to be solved by other methods. GAs have been used in an very wide range of areas. However, one of the major drawbacks of genetic algorithms is the slow convergence, which prevents GAs to be used in in real time application system.

Antenna pattern null forming and steering are very important in some critical wireless communication and radar applications, especially when used in modern digital beamforming adaptive array systems [1]-[2]. In cellular mobile communication systems and radar systems, signals suffers distortions such as multipath fading, co-channel interference and noise. Therefore, to find a fast beamforming system, which could constantly adapt the radiation pattern of the antenna to direct multiple narrow beams to desired users and suppress the interference, is needed. Traditionally, adaptive nulling is within a fairly narrow angle to reject a strong interfering source, at a specific azimuth direction. But in most applications, the interfering signals could be incident from many different direction. Due to increasing electromagnetic pollution of the environment, beamforming techniques that allow the placement of more than one null in the antenna pattern at specific jamming directions are becoming more important, [3]. Since a wideband jammer will appear to cover an angular sector of the pattern due to the frequency dependence of the antenna. Hence, the term wide null has been used to describe this suppression in the sidelobe level (SLL) of an angular sector [4].

Usually, the wide nulls produced should be deep and wide enough to accommodate frequency fluctuations and interference from mobile jammers. This is overcome in convention techniques by placing two adjacent nulls in the radiation pattern [5].

In our previous studies [6], we have successfully applied GAs to solve some challenging wide nulling array beamforming problems.

As GA solutions are normally very slow and are traditionally considered not practical for real-time DBF application. In order to make GA solutions possible for real time problems, we propose, in this study, to use a suitable neural network (NN) to approximate the function of GA in complicated problem. The neural network could be trained by a finite number of off-line slow GA solutions and the trained neural work could be used for infinite number of real time solutions of the problem that could only be solved by GA.

Due to the best approximation property of radial basis function neural network (RBFNN), we chose RBFNN to approximate the function of GA. The low sidelobe wide nulling (LSWN) problem in digital beamforming is proved to be well solved by this GA-RBFNN approach.

2. LOW SIDELOBE WIDE NULLING (LSWN) BEAMFORMING FORMULATION

For an arbitrary array of M element, the pattern function of the array factor can be expressed by the general function,

$$F(\theta, \phi) = \mathbf{w}^T \mathbf{S}(\theta, \phi) \tag{1}$$

$$\begin{aligned}
\mathbf{w} &= \{w_1, w_2, ..., w_M\}^T, \\
\mathbf{S} &= \{e^{jk\mathbf{r}_m \cdot \hat{a}_r}\}^T, \\
\mathbf{r}_m &= \hat{a}_x x_m + \hat{a}_y y_m + \hat{a}_z z_m, m = 1, 2, ..., M \\
\hat{a}_r &= \hat{a}_x \sin\theta \cos\phi + \hat{a}_y \sin\theta \sin\phi + \hat{a}_z \cos\theta;
\end{aligned}$$

where $\mathbf{w}$ is a column vector with M complex weighting coefficients, S is the generic steering column vector, $\mathbf{r}_n$ is the element location vectors, $\hat{a}_r$ is unit vector of distance ray of the spherical coordinate, and θ and ϕ are the elevation and azimuth angles, respectively.

The objective of adaptive digital beamforming is to find proper weighting coefficients $\mathbf{w}$ to achieve desired pattern shape, including beam pointing direction, beamwidth, sidelobe level, null pointing direction, null depth, and etc. Fig. 1 shows an example of wide nulling.

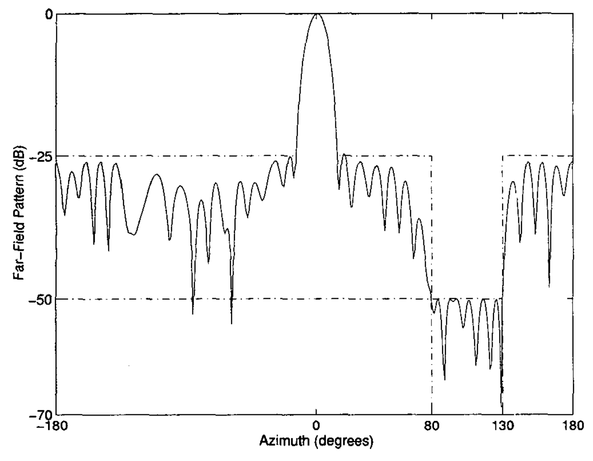


Figure 1: Low sidelobe wide nulling.

3. NEURAL NETWORKS (NN)

In recent years, a major result that has emerged with the growth of interest in neural networks is that a multilayer perceptron (MLP), with a single hidden layer, is capable of approximating any smooth nonlinear input-output mapping to an arbitrary degree of accuracy, provided that a sufficient number of hidden layer neurons is used [7], [8]. Girosi and Poggio proved that RBF networks which have been derived in a regularization frame-work have the property of best approximation [9]. Most importantly, they showed that MLPs do not possess the best approximation property for the class of continuous functions, while RBF networks possess this property in a unique sense.

Motivated by these inherent advantages, this paper presents the development of a radial basis function neural network (RBFNN)-based algorithm to compute the weights of an adaptive array antenna.

In this study, a finite number of GA solutions are used to train NN. The trained NN is used to generate infinite number of solutions for arbitrary desired beam pattern. The trained NN can directly generate the array weights if given the desired beam pattern template. Because NN is a fast tool, it solves the slow problem in GA.

A radial basis function neural network (RBFNN) generally consists of two weight layers - the hidden layer and the output layer. They can be described by the following equation:

$$y = \mathbf{b}_2 + \sum_{i=1}^{N_h} \varpi_i f(||\mathbf{p} \quad \mathbf{c}_i||\mathbf{b}_1) \tag{2}$$

where f are the radial basis functions, ϖ_i are the output layer weights, $\mathbf{b}_2$ is the output offset, $\mathbf{p}$ are the inputs to the network, $\mathbf{c}_i$ are the centers associated with the radial basis functions, n_h is the number of basis functions in the network, and $|| \cdot ||$ denotes the Enclidean norm.

The nonlinear basis function f can be formed using a number of different functions. One common example is: Gaussian function, as

$$f(x) = \exp\{\frac{(x \quad a)^2}{r^2}\} \tag{3}$$

where $a \in \mathbf{R}$ is the center of the basis function which has radius r.

In our experiment of LSWN problems, we conduct a simulation of a RBFNN model to approximate the GA solution. The input of RBFNN is the desired array pattern template of N_p variables and output is the antenna weighting vector of N_e dimensions. This is a complicated problem which has so far only be solved by GA. Through the experiment, RBFNN is proved effective to resolve such N-M dimensional approximation problem. After the RBFNN is trained by a number of off-line GA solutions, it can work as a fast optimization tool that can generate the proper weights in the M dimensional space for other arbitrary wide nulling without further time consuming GA solutions. we used RBFNN as function approximation tool, as Fig. 2 shows.

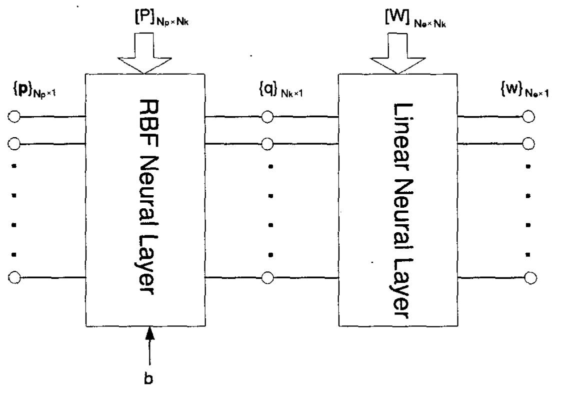


Figure 2: RBFNN application in experiment.

- $\{\mathbf{p}\}_{N_p \times 1}$ is the neural network input vector. In our experiment, it is the desired array pattern.
- $[\mathbf{P}]_{N_p \times N_k}$ is radial basis function centers. In our experiment, it is the N_k groups GA template aggregations.
- $\{\mathbf{q}\}_{N_k \times 1}$ denotes the hidden radias neural layer output.
- $[\mathbf{W}]_{N_e \times N_k}$ is the linear neural layer weights matrix. In our experiment, it is set as the N_k groups of solutions from GA.
- $\{\mathbf{w}\}_{N_e \times 1}$ is the neural network output. In our experiment, it is the desired antenna weighting vector.

Where N_p is the number of sampling points of a template pattern. N_k is the number of known solutions. N_e is the number of array elements.

In Fig. 2,

$$q_k = \exp\{\ \ b^2 \sum_{i=1}^{N_p} (p_i \quad P_{ik})^2\}, k = 1, 2, ..., N_k \qquad (4)$$

$$w_i = \sum_{k=1}^{N_k} q_k W_{ik}, i = 1, 2, ..., N_e \qquad (5)$$

4. NEURAL BEAMFORMER

This section briefly describes how to use NN to solve the LSWN beamforming problem.

4.1. Getting Weights from GA as Training Data for NN

The proposed approach is based on GA. Firstly, a number of solutions for LSWN beamforming are calculated by GA [6]. Most GA's use binary coding and binary genetic operations. However, in our experiment, the GA applies floating-point genetic operations on array weight vectors. Hence, each chromosome is a vector of numbers and the dimension of the vector is equivalent to the number of array elements.

An initial population of at least 100 random chromosomes is generated. The weighting vectors $\mathbf{w}^T$ of a Taylor synthesized array with an identical side lobe as the original patter is added to replace one weakest individual among the initial population. The insertion helps to improve the rate of convergence.

The Emperor-Selective (EMS) mating scheme is used in experiment. In EMS, the best individual in the parent population gets to mate with every other even-numbered individual in the population. If one or more near-solutions are added to an initial population of random individuals, EMS usually yields the best chromosome.

Extrapolation Crossover (EPX) and Non-uniform Upslope Mutation (NUM) are the crossover operator and mutation operator uesd in GA respectively.

EPX takes two parents, $P1$ and $P2$, to produce two offsprings, $C1$ and $C2$, that lie outside the range, a, of the two parents. The offsprings have equal probability to lie within the range a, extended in both directions from $P1$ and $P2$. $C1$ will then lie on the same side as $P1$ and $C2$ on the same side as $P2$.

The range, δ, of the parents is defined by

$$\delta = (P2 \quad P1), where\ P2 > P1 \qquad (6)$$

$$\begin{aligned} C1 &= P1 + a \cdot \delta, \\ C2 &= P2 \quad a \cdot \delta; \end{aligned}$$

where δ is a randomly chosen number between 1 and 2.

NUM changes several individuals of the population based on a non-uniform distribution. This non-uniform distribution starts from very low value, and increases exponentially as the current generation approaches the maximum number of generations.

A template, formed by null pointing direction, null width, null depth, sidebel level, major beam pointing direction, beam width, is cast over the array pattern produced by each candidate to compute their cumulative difference as a form of fitness measure in decibels.

4.2. RBFNN Beamformer

The neural beamformer consists of antenna array input preprocession, an artificial neural network, and output postprocessing. Here, we emphasize to introduce the part of artificial neural network.

In the whole antenna array system, Neural Network is used as a tool to generate the array weights(w). This Neural Network should have been well trained in advance. When presented a desired antenna beam pattern(c) to the input layer of trained RBFNN, the array weighting vector **w** is produced at the output layer consisting of Ne nodes(Ne equals to the array element number in experiment). The RBFNN is designed to perform an input-output mapping trained with examples{ $\mathbf{c}^m, \mathbf{w}^m$; m=1,2,...,Nm},where Nm stands for the number of examples contained in the training set. The trained RBFNN is supposed to generalize outputs when given inputs beam patterns are not among the GA solutions.

5. NUMERICAL EXPERIMENT

The data for training the Neural Network is a set of finite number of solutions from GA. First, given desired pattern matrix$\{F^{(i)}; i = 1, 2, ..., N_k\}$, Using GA, the array weighting vectors $\{\mathbf{w}^{(i)}; i = 1, 2, ..., N_k\}$ are evaluated to produce the required training input/output pairs of the training set; that is $\{(F^{(i)}; \mathbf{w}^{(i)}); i = 1, 2, ...N_k\}$. The desired antenna pattern $F^{(i)}$ can be denoted by such 5 eigenvalues: null pointing direction, null width, null depth, sidebel level, major beam pointing direction, and beam width.

In simulation, a 32-element half-wavelength spaced linear array is used as the working example. In this example, two different null widths (20 and 30 degrees) are considered. To train the RBFNN, 108-sets of GA solutions are obtained first as input/out training data for the RBFNN. As shown in Table I, the 108 cases can be presented in 12 groups. In each group, 9 discrete null pointing directions are chosen, i.e., the left edges of nulls (θ_1) are from 10 degree to 50 degree at a step of 5 degrees.

After training, the resultant RBFNN should have stored the characters and relationships among these data. Because of the approximation function of the neural network, the original low sidelobe wide null steering problem can be approximately represented by the trained RBFNN. The successfully trained NN should have the follow character: when given the NN any other desired antenna pattern that is not among the training data, the NN should output the array weights that is able to produce the desired pattern or in other words they are good approximations of the array weights from GA solution of the same pattern.

Table 1: Twelve groups of input training examples

Null width (degree)	Null depth (dB)	Sidelobe level (dB)
20	20	-20
20	20	-25
20	20	-30
20	30	-20
20	30	-25
20	30	-30
30	20	-20
30	20	-25
30	20	-30
30	30	-20
30	30	-25
30	30	-30

Note: Each with 9 θ_1 (from 10 to 50 with a step of 5).

Properly-trained neural network tends to give reasonable answers when presented with inputs that they have never seen. This generalization property makes it possible to train a network on a representative set of input/target pairs and get good results without training the network on all possible input/output pairs.

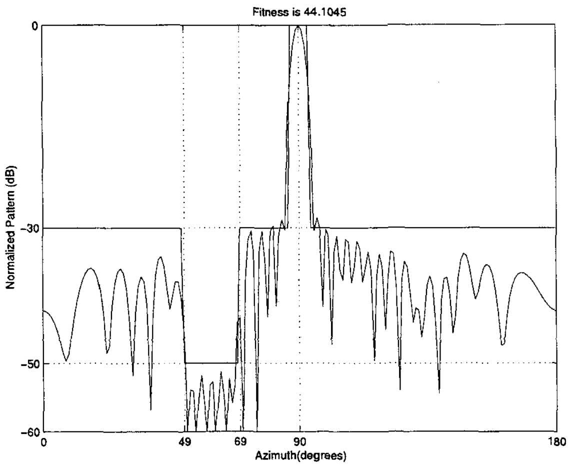
The desired benefit is that the trained NN can generate infinite number of fast solutions that can be implemented for on-line real time application.

Present a new beam pattern **p** which is unseen in training examples to the input layer of the trained RBFNN. The output layer of the trained RBFNN will produce an output as the optimum result from Genetic Algorithm. One advance of this approach is that trained NN can generate infinite number of solutions. Properly-trained neural network tends to give reasonable answers when presented with inputs that they have never seen. This generalization property makes it possible to train a network on a representative set of input/target pairs and get good results without training the network on all possible input/output pairs. Moreover, because NN is a real time tool, the approach solve the slow problem existing in GA.

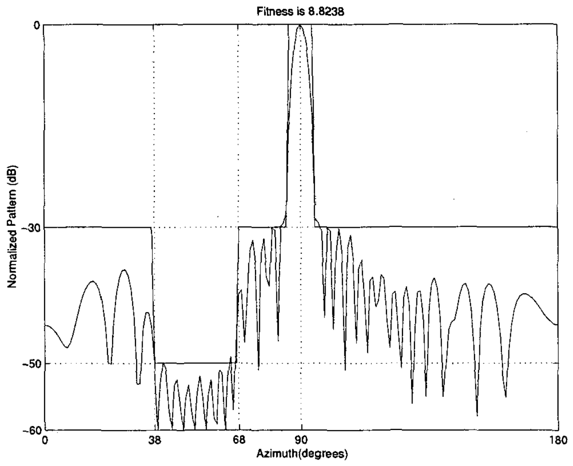
Fig. 3 shows the test results of the low sidelobe wide nulling to illustrate the effectiveness of the above trained RBFNN. The rectangular lines in the Figures show the templates of the desired patterns.

In Fig. 3, the input pattern presented to the RBFNN is denoted by the input parameter vectors [*null pointing direction, null width, null depth, sidebel level, major beam pointing direction, beam width*] = [49, 20, 20, 30, 90, 7], and [38, 30, 20, 30, 90, 9], respectively.

Be noticed that those input patterns are not among the training examples. We have also conducted exhaust test for θ_1 from 10 degrees to 50 degrees at a step of 1 degree and the test results are very good. The experiment result is also quit good.



(a)



(b)

Figure 3: Examples of low sidelobe wide null steering with null width of 20 degrees.(comp)

The computational time for GA solutions depends on the desired fitness level or the maximum number of generations. Typically, the solution for a satisfactory weighting vector takes about 1 to 2 hours under Matlab environment on a Pentium 700 MHZ PC. While the time cost using trained RBFNN to compute each weighting vector is less than 0.02 second.

6. CONCLUSION

An approach combining Radial Basis Function Neural Network and Genetic Algorithm (GA-RBFNN) is proposed for the independent wide nulling of linear array. The weights were computed using an RBFNN that approximates the GA solution. The slow drawback of GA is solved effectively.

7. REFERENCES

[1] Special Issue on adaptive antennas *IEEE Trans. Antennas Propagat.*,vol. AP-24,1976.

[2] J. E. Hudson, *Adaptive Array Principles,* Chippenham, Wiltshire:Antony Rowe Ltd, 1989.

[3] K. A. Al-Mashouq, I. El-Azhary, M. S. Afifi, A. S. Al-Ruwais, and P. S. Excell, 'Multiple nulls fromatin and low interference of futre arrays for small earth stations", *Proc. ARABSAT Symp., King Saud Univ.,* Riyadh, Saudi Arabia, pp. 434-443, 1987.

[4] El-Azhary, M. S. Afifi and P. S. Excell, "Fast cancellation of sidelobes in the pattern of a uniforml excited array using external elements", *IEEE Trans. Antennas Propagat.*,vol. 38, no.12, Dec. 1990.

[5] P. Owen and J.C. Mason, "The use of linear programming in the design of antenna patterns with prescribed nulls and other constraints", *Int. J. Comput. Math. Elec. Electron. Eng.,* vol. 3, no. 4, pp.201-215, 1984.

[6] Y. Lu, B. K. Yeo , "Adaptive wide null steering for digital beamforming array with the complex coded genetic algorithm", *Proc. of 2000 IEEE International Conference on Phased Array Systems and Technology,* pp. 557-560, 2000.

[7] G. Cybenko, Approximation by superposition of a sigmoidal function, *Mathematical Control Systems Signals,* 2(4):303-314, 1989.

[8] A. R. Barron, Universal approximation bounds for superpositions of a sigmoidal function, *IEEE Transactions on Information Theory,* 39(3):930-945, 1993.

[9] F. Girosi, T .Poggio, Networks and the best approximation property. *Technical Report AIM-1164,* Massachusetts Institute of Technology,October 1989.