2014

Design of Non-Uniform Linear Array via Linear Programming and Particle Swarm Optimization and Studies on Phased Array Calibration

Hua Bai
University of Massachusetts Amherst

DESIGN OF NON-UNIFORM LINEAR ARRAY VIA LINEAR PROGRAMMING AND PARTICLE SWARM OPTIMIZATION AND STUDIES ON PHASED ARRAY CALIBRATION

A Thesis Presented

by

HUA BAI

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL AND COMPUTER ENGINEERING

September 2014

Electrical and Computer Engineering

DESIGN OF NON-UNIFORM LINEAR ARRAY VIA LINEAR PROGRAMMING AND PARTICLE SWARM OPTIMIZATION AND STUDIES ON PHASED ARRAY CALIBRATION

A Thesis Presented

by

HUA BAI

Approved as to style and content by:

____

Ramakrishna Janaswamy, Co-chair

____

Do-Hoon Kwon, Co-chair

____

Paul Siqueira, Member

____

Christopher V. Hollot, Department Chair
Electrical and Computer Engineering

To my parents.

ACKNOWLEDGMENTS

Studying at University of Massachusetts Amherst and the Center for dvanced Sensor and Communication Antennas (CASCA) is very lucky for me and this will be a memorable experience in my life.

I would like take this chance to thank every professor, colleague and friend who helped me in the past two years. First of all, I want to express my deepest gratitude to my advisor Professor Ramakrishna Janaswamy who gave me the opportunity to work with him at CASCA and lead me to the area of electromagnetic field and antenna, the thesis cannot be completed without his constant guidance and encouragement. Secondly, I would like to thank Professor Do-Hoon Kwon for his advice and support during my Master's program. I also appreciate Professor Paul Siqueira for being the member of my thesis committee, I learned a lot from him especially the attitudes towards teaching and research.

My gratitude is extended to the colleagues in the lab especially to Narayan, Selman and Bayo who helped me a lot when I was studying and living at Amherst and to the staff of the ECE department, especially Mary McCulloch and Barbara Barnett.

Finally, I would like thank my parents and my girlfriend for their patience, support and love, and for making me believe everything will be fine.

ABSTRACT

DESIGN OF NON-UNIFORM LINEAR ARRAY VIA LINEAR PROGRAMMING AND PARTICLE SWARM OPTIMIZATION AND STUDIES ON PHASED ARRAY CALIBRATION

SEPTEMBER 2014

HUA BAI

B.Sc., SICHUAN NORMAL UNIVERSITY, CHINA

M.S.E.C.E., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Ramakrishna Janaswamy

For a linear array, the excitation coefficients of each element and its geometry play an important role, because they will determine the radiation pattern of the given array. Side Lobe Level (SLL) is one of the key parameters to evaluate the radiation pattern of the array. Generally speaking, we desire SLL to be as low as possible. For the linear array with uniform spacing, there are some classic methods to calculate the excitation coefficients to make the radiation pattern satisfy the given requirements. For the linear array with non-uniform spacing, linear programming and particle swarm optimization are proposed to calculate the excitation coefficients to make the array get minimum SLL in this thesis. They are demonstrated for symmetric and asymmetric array in the first part of this thesis. In the second part of this thesis, a simple method is proposed for correcting excitation coefficients of a linear phased array. This proposed method corrects the coefficients through using the Normalized Least Means Squares

(NLMS) algorithm, dither signal and a near-field sensor being used for sensing the field emitted by the array. The advantage of this proposed method is that it avoids the problem of estimating the largest eigenvalue of the coefficient matrix to get optimal step size. Its robustness in different environments is demonstrated as well as the effect of noise with various Signal-to-Noise Ratio (SNR), and mutual coupling. In addition, the effect of using discrete dither signal to the array is considered, because the continuous dither signal cannot be generated in practice.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 Overview

Beamforming is a kind of technology used in phased arrays to get directional radiation pattern at specified angles. The main lobe is the lobe with maximum magnitude, it means the array can transmit more energy in the direction of main lobe when it is used as a transmitter or the signal from that direction will be received better if the array works as a receiver. Often we need to adjust the radiation pattern of the phased array to satisfy our various requirements. The radiation pattern of the phase array is determined by several parameters including the nature of the elements in the array, the geometry of the array and the excitation coefficients of each element, etc. The nature of the elements are determined after they are produced, so we are more interested in adjusting the excitation coefficients and the geometry of the phased array. This thesis focuses on the excitation coefficients of the linear array. For the linear array with uniform spacing, the problem of excitation coefficients has already being solved well, so the linear array with non-uniform spacing will be designed here. This thesis consists of two parts, the first part (Ch.2 and Ch.3) focuses on figuring out the optimal excitation coefficients with respect of Side Lobe Level (SLL), the second part (Ch.4 and Ch.5) solves the problem of correcting the excitation coefficients when they drift due to environmental conditions.

1.2 Motivation

With the development of more than one hundred years, phased arrays are widely used in a lot of areas now, such as weather forecasting, detecting or tracking targets and so on [1] [2]. For a phased array, its excitation coefficients play an important role, because they determine the radiation pattern of the system. Considering its importance, various classic methods have been proposed to generate the array's excitation coefficients to design the array meeting certain goals. For example, the Taylor method is one of the popular methods in designing the array. It was first introduced by Taylor in 1955 [3] and the design makes a compromise between beamwidth and SLL[4]. An example of using the Taylor design is illustrated in Fig.1.1, which shows the electric field in far field region of a 32-element array with spacing of $\lambda/2$, the SLL is around -24 dB. Those classic designs are easy to implement; however most of them can only be used for the linear array with uniform spacing. Here we want to solve the problem of designing the excitation coefficient of a *non-uniform* linear array by methods of the linear programming and the Particle Swarm Optimization (PSO).

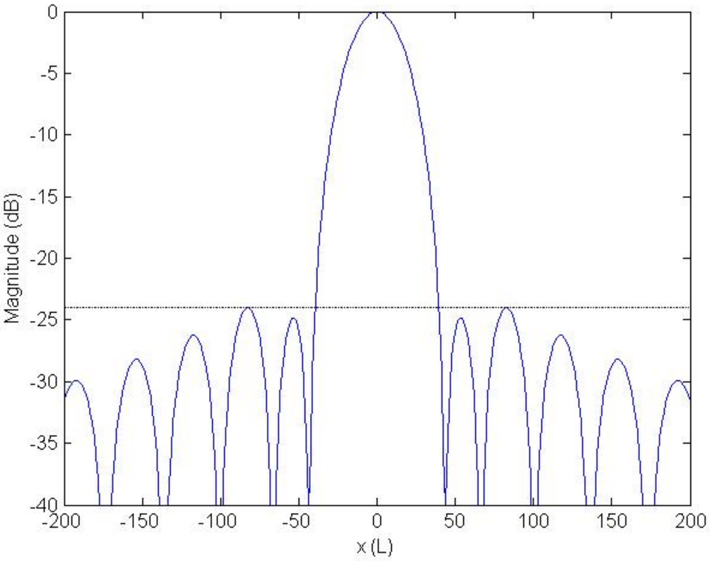

Fig. 1.1. The electric field of the array in far-field region

The excitation coefficients are of great importance to the array as mentioned before, but sometimes they may be altered due to the change of working conditions, errors during fabricating and so on. This will lead to the change of the radiation pattern of the array, which might degrade the gain, directivity of the array and then the array could not meet the requirement. The second part of the thesis is concerned with finding a method to calibrate these excitation coefficients. One correction method via dithering and the Least Mean Squares (LMS) algorithm has been proposed in [5]. In that paper, the LMS algorithm is used to correct the actual coefficients. In using the LMS algorithm to correct the coefficients, one necessary step is to estimate the largest eigenvalue of the coefficient matrix to update the coefficients by following a gradient based procedure, which makes the method more complicated and increases execution time. In this thesis, we implement the Normalized Least Mean Squares (NLMS) algorithm which bypasses finding the minimum eigenvalue, thereby, making the algorithm faster and more efficient, and demonstrate the robustness of the algorithm.

1.3 Outline of This Thesis

This thesis will be organized as follows: in Chapter 2, we will use linear programming to determine the optimal excitation coefficients with the respect of SLL for a special kind of non-uniform spacing array—the symmetric array. In Chapter 3, the array is extended to the more general case—asymmetric array, and the PSO algorithm will be introduced for finding out the optimal excitation coefficient. Both of the two methods will be demonstrated by constructing arrays that meet various performance requirements. The calibration of the linear array is presented in Chapter 4, the algorithm is demonstrated in noiseless and noisy environment, respectively, and the effect of mutual coupling and using discrete dithering signal are also considered.

The final results of the proposed algorithm is presented in Chapter 5. In Chapter 6, the thesis work is summarized.

CHAPTER 2

LINEAR PROGRAMMING FOR SYMMETRIC ARRAY

2.1 Introduction

An array which consists of multielements is used for improving a radiation pattern without changing the characteristics of individual elements. Compared with individual elements, an array can achieve higher directivity, higher gains and lower SLL [6]. The most common geometry for arrays are linear, rectangular and circular [7] [8]. This thesis only focuses on linear arrays. For simplicity and practice, we will only study the performance of the array consisting of identical elements. An example of a linear array is shown in Fig.2.1

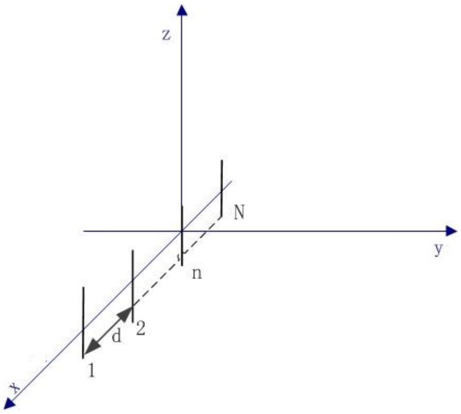

Fig. 2.1. Linear array with N elements

As illustrated in the above figure, the number of the elements is denoted by N ($N \geq 2$). To start with, the simplest situation is considered: the array consists of 2 infinitesimal dipoles, and the two elements are in phase. For far field ($kr \gg 1$, k is the wavenumber, r is the distance from origin to observation point), the electric field can be written in the xy-plane as [4]

$$E(\phi) = j\eta \frac{kI_o le^{-jkr}}{4\pi r}(w_1 e^{-jkd_1 cos\phi} + w_2 e^{-jkd_2 cos\phi}) \tag{2.1}$$

where η is the wave impedance, I_o is the current, l is the length of the dipole, d_n is the position of the dipole, and w_n is the excitation coefficient for the element, which is also known as the array weighting characteristics. In 2.1, the element factor is denoted by $j\eta\frac{kIle^{-jkr}}{4\pi r}$, because it is only dependent on the characteristics of elements in the array. The remaining term, which is inside the parentheses, represents the array factor and is related to the excitation coefficients of the elements and the geometry of the array. In order to extend the 2-element array to an N-element array. Equation 2.1 can be rewritten as

$$E(\phi) = j\eta \frac{kI_o le^{-jkr}}{4\pi r} \sum_{n=1}^{N} w_n e^{-jkd_n cos\phi} = EF * AF \tag{2.2}$$

where EF is short for element factor, AF is short for array factor. The total radiation of the array is affected by the element factor and array factor simultaneously, however, once the element is chosen, the element factor is determined and the electric field is proportional to AF. In this thesis, we are more interested in the array factor.

When we design the array's excitation coefficients, some parameters are available for referring to such as gain, directivity, etc. In this thesis, the SLL is a design specification meaning that the excitation coefficient which produces the minimum SLL is optimal. The symmetric broadside array will be designed at first.

2.2 Symmetric Array

Symmetric array means the position of the elements are symmetric with respect to one axis or some other reference. In this section, the array is symmetric with respect to the y-axis, which is illustrated in Fig.2.2. For such an array, the following equations are valid

$$AF(\phi) = w_1 e^{-jkd_1 cos\phi} + w_2 e^{-jkd_2 cos\phi} + ... + w_N e^{-jkd_N cos\phi} \tag{2.3}$$

$$d_n = -d_{N+1-n}, n \leq \frac{N}{2} \tag{2.4}$$

$$w_n = w_{N+1-n}, n \leq \frac{N}{2}. \tag{2.5}$$

Via (2.4) and (2.5), (2.3) can be written as

$$AF(\phi) = \begin{cases} \sum_{n=1}^{\frac{N}{2}} 2w_n cos(kd_n cos\phi), \text{N is even} \\ \sum_{n=1}^{\frac{N-1}{2}} 2w_n cos(kd_n cos\phi) + w_{\frac{N+1}{2}} cos(kd_{\frac{N+1}{2}} cos\phi), \text{N is odd} \end{cases} \tag{2.6}$$

when N is odd, $d_{\frac{N+1}{2}}$ must be zero to guarantee the array is symmetric, equation (2.6) is rewritten as

$$AF(\phi) = \begin{cases} \sum_{n=1}^{\frac{N}{2}} 2w_n cos(kd_n cos\phi), \text{N is even} \\ \sum_{n=1}^{\frac{N-1}{2}} 2w_n cos(kd_n cos\phi) + w_{\frac{N+1}{2}}, \text{N is odd} \end{cases} \tag{2.7}$$

According to the definition of Side Lobe Ratio (SLR) [9], we can get

$$SLR = |AF(\phi)|_{max}, \phi \notin \text{main beam} \tag{2.8}$$

the relation between SLL and SLR is $SLL = 20 * log_{10}(SLR)$. Assuming that the element factor is represented by an isotropic radiator (i.e. EF=1), the desired angle for the array is denoted by ϕ_d, which corresponds to the center point of the main beam of the radiation pattern; when $\phi = \phi_d$, AF should reach its peak value such that E in (2.2) is a maximum. The peak value of the array factor is assumed to be equal to 1, $AF(\phi_d) = 1$. In order to guarantee $AF(\phi_d)$ is at the peak, its derivative is also forced to be equal to 0 at ϕ_d. Now the problem can be summarized as finding the minimum SLR with the constraints $AF(\phi_d) = 1$ and $\frac{\partial AF(\phi)}{\partial \phi}|\phi_d = 0$.

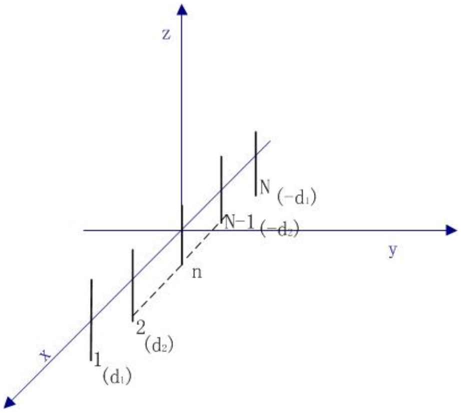

Fig. 2.2. A symmetric array

2.3 Linear Programming

In the symmetric case, there exists a linear relationship between the array factor and the excitation coefficients, considering that the liner programming method is appropriate for solving this kind of problem. Linear programming was first introduced by Leonid Kantorovich in 1939 [10], this method has been widely used for solving the linear optimal problem in a number of fields [11]. This method relies on the apparent or potential linear relationships between different parameters. How to use linear programming in symmetric array is what we are going to do next. The first step is to convert the problem of obtaining the minimum sidelobe to a linear optimal problem.

The general linear programming problems can be expressed as

$$\text{Maximize/Minimize}: \quad A^T\mathbf{x} \tag{2.9}$$

$$\text{Subject to}: \quad B\mathbf{x} \leq C \tag{2.10}$$

$$D\mathbf{x} = E \tag{2.11}$$

where $\mathbf{x}$ is a vector denoting the unknown coefficients, letters A, B and D represent known matrices, symbol $(\cdot)^T$ denotes the transpose of a matrix, letters C and E denote two known vectors. Equation 2.9 represents the function we want to solve, subject to the constraints of (2.10) and (2.11). Fig.2.3 depicts a simple example of the linear programming, the x-axis, the y-axis and the two solid lines constitute the shaded region which is called feasible region, x-axis, y-axis and the two lines are the constraints, then we need to find out the optimum result in the feasible region depending on the given requirement.

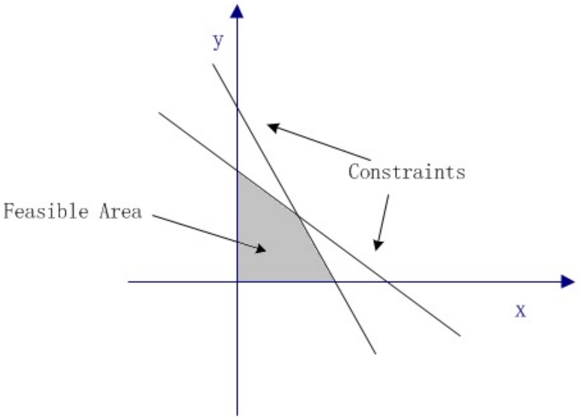

Fig. 2.3. General linear programming problem

For a symmetric array, since its weights are the variable we need to determine, we can set [12]

$$\mathbf{x} = \begin{bmatrix} w_1 \\ w_2 \\ \cdot \\ w_N \end{bmatrix} \tag{2.12}$$

the array factor is the objective function, we can get

$$A^T = \begin{cases} [2cos(kd_1cos\phi), 2cos(kd_2cos\phi) \cdots 2cos(kd_{\frac{N}{2}}cos\phi)], \text{N is even} \\ [2cos(kd_1cos\phi), 2cos(kd_2cos\phi) \cdots 2cos(kd_{\frac{N-1}{2}}cos\phi), 1], \text{N is odd} \end{cases}. \tag{2.13}$$

Assuming the minimum $SLR = m$, we can get the following inequality

$$|A^T\mathbf{x}| \leq m, \phi \notin \text{main beam} \tag{2.14}$$

another two equations with respect to the array factor at desired angle are

$$A^T_{\phi_d}\mathbf{x} = 1 \tag{2.15}$$

$$\frac{\partial A^T_\phi \mathbf{x}}{\partial \phi}|\phi_d = 0. \tag{2.16}$$

Now, the problem reduces to, given the beamwidth and element locations

$$\text{Minimize :}m \tag{2.17}$$

$$\text{Subject to :}|A^T\mathbf{x}| \leq m, \phi \notin \text{main beam} \tag{2.18}$$

$$A^T_{\phi_d}\mathbf{x} = 1 \tag{2.19}$$

$$\frac{\partial A^T_\phi \mathbf{x}}{\partial \phi}|\phi_d = 0. \tag{2.20}$$

Once the problem is converted to a linear programming problem, we can adopt some algorithms such as the simplex algorithm to solve it. The simplex algorithm is derived from the concept of simplex, its appearance brought great improvements to solving the linear programming problem, you can find more details about this algorithm in [13]. In this thesis, Matlab is used to figure out the problem, the numerical results gotten by Matlab are shown in next section.

2.4 Numerical Results

A ten-element linear array is used to verify our method. The desired main beam angle of the array is set as 90 degrees, and the beamwidth is 40 degrees. Then, the positions of each element are set as [-2λ, -1.3λ, -0.85λ, -0.5λ, -0.2λ, 0.2λ, 0.5λ, 0.85λ, 1.3λ, 2λ]. By (2.13), $A^T = [2, 2, 2, 2, 2]$ at $\phi = 90°$, then using (2.19), it is clear to show that

$$2(w_1 + w_2 + ... + w_5) = 1. \tag{2.21}$$

It is not hard to see the above equation satisfies (2.20) at the same time when $\phi_d = 90°$. There are just 5 excitation coefficients, because the array is symmetric, the corresponding elements share the weight and the weights are set to be real numbers for the broadside array. $\mathbf{x}'$ is set as

$$\mathbf{x}' = \begin{bmatrix} m \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \end{bmatrix} \tag{2.22}$$

where m is just the minimum SLR. Using (2.21) and (2.22), we get

$$\boldsymbol{a} \cdot \mathbf{x}' = 1 \tag{2.23}$$

where $\boldsymbol{a} = [0\ 2\ 2\ 2\ 2\ 2]$. As the beamwidth is chosen as 40 degrees, the side lobe occurs in the angles range [0°,70°] and [110°,180°]. The side lobe region is observed finely every degree so as to not miss the peak of the side lobe. Equation 2.18 can be seen as two inequalities without the absolute value sign, which is helpful for future calculation

$$A^T\mathbf{x} \leq m \tag{2.24}$$

$$A^T\mathbf{x} \geq -m. \tag{2.25}$$

Converting the above two inequalities to the following two inequalities

$$\boldsymbol{b} \cdot \mathbf{x}' \leq 0 \tag{2.26}$$

$$\boldsymbol{c} \cdot \mathbf{x}' \leq 0 \tag{2.27}$$

where $\boldsymbol{b} = [-1,\ 2cos(kd_1cos\phi),\ 2cos(kd_2cos\phi),\ ...\ 2cos(kd_5cos\phi)]$, $\boldsymbol{c} = [-1,\ -2cos(kd_1cos\phi),\ -2cos(kd_2cos\phi),\ ...\ -2cos(kd_5cos\phi)]$. For ϕ, it is chosen from the side lobe region, and as mentioned before, the pattern in side lobe region is sampled every degree. If the observation number is too small, it may lead to a wrong result because the peak of any particular side lobes may not be captured by sparse observation. The optimal excitation coefficients for each element we get finally are [0.0535, 0.1152, 0.0888, 0.1224, 0.1201, 0.1201, 0.1224, 0.0888, 0.1152, 0.0535], the SLL is -30.343 dB. The array factor of the array is illustrated in Fig.2.4, the position and optimal excitation coefficients determined for the array with different number of elements is shown in Table 2.1 and Table 2.2 respectively. In Table 2.1, Min. and Max. denote the minimum and maximum spacing, respectively. Notice that for N=6,8,10, the spacing exceeds the traditional limit of $\lambda/2$.

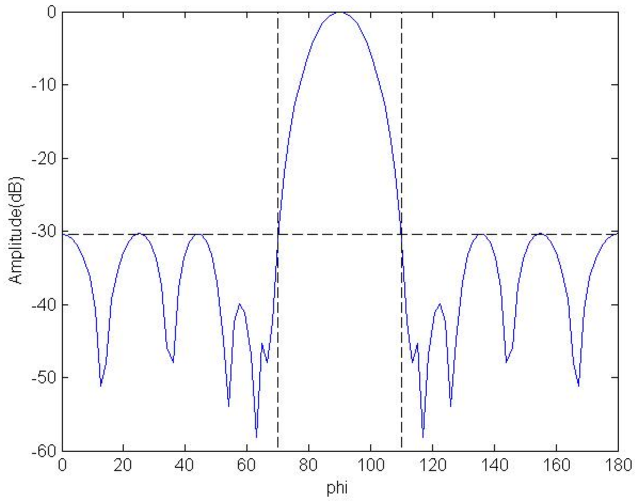

Fig. 2.4. The array factor with optimal excitation coefficients

Number of Elements	d_1/λ	d_2/λ	d_3/λ	d_4/λ	d_5/λ	Min./λ	Max./λ
2	0.2					0.4	0.4
4	0.2	0.5				0.3	0.4
6	0.2	0.5	0.85			0.3	0.4
8	0.2	0.5	0.85	1.3		0.3	0.55
10	0.2	0.5	0.85	1.3	2	0.3	0.7

Table 2.1. Element spacing of the symmetric array

Number of Elements	w_1	w_2	w_3	w_4	w_5	Minimum SLL(dB)
2	0.5					-0.828
4	0.1504	0.3496				-4.344
6	0.1861	0.0657	0.2482			-11.204
8	0.1466	0.1228	0.1098	0.1209		-19.948
10	0.1201	0.1224	0.0888	0.1152	0.0535	-30.343

Table 2.2. Optimum weights for symmetric arrays, broadside and BW=40°

For the same broadside arrays with beamwidths being 50 degrees and and 60 degrees, the optimum weights and the minimum SLL are shown in Table 2.3 and Table 2.4, respectively.

Number of Elements	w_1	w_2	w_3	w_4	w_5	Minimum SLL(dB)
2	0.5					-1.276
4	0.1966	0.3034				-6.282
6	0.2112	0.0958	0.1930			-15.603
8	0.1733	0.1221	0.1216	0.083		-26.859
10	0.1433	0.1215	0.1051	0.0971	0.033	-32.111

Table 2.3. Optimum weights for symmetric arrays, broadside and BW=50°

Number of Elements	w_1	w_2	w_3	w_4	w_5	Minimum SLL(dB)
2	0.5					-1.841
4	0.2361	0.2639				-8.259
6	0.2308	0.1140	0.1552			-20.265
8	0.1924	0.1233	0.1237	0.0606		-34.067
10	0.1756	0.1225	0.1163	0.0725	0.0132	-34.846

Table 2.4. Optimum weights for symmetric arrays, broadside and BW=60°

From Table 2.2 to Table 2.4, it can be seen that the SLL can be improved with adding more elements and extending beamwidth. Fig.2.5 and Fig.2.6 illustrate the array factor of the ten-element array with BW=50° and BW=60°.

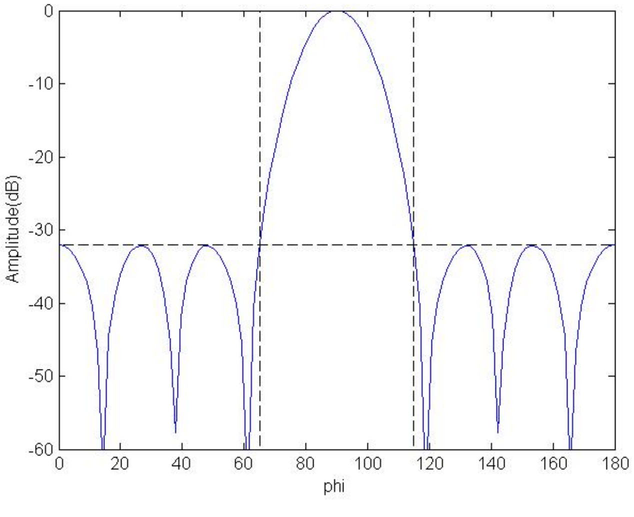

Fig. 2.5. The array factor with optimal excitation coefficients and BW=50°

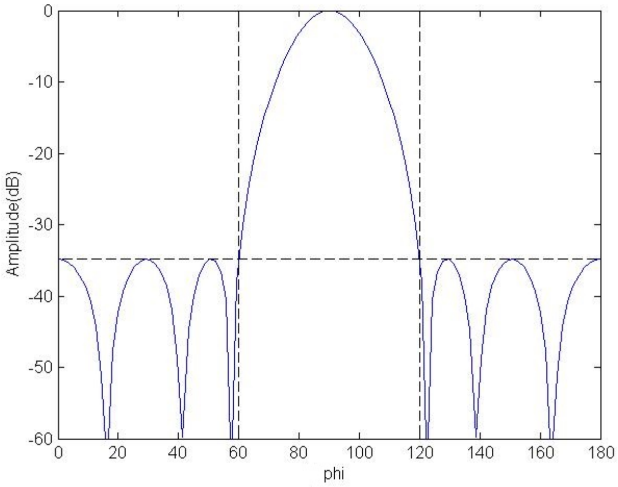

Fig. 2.6. The array factor with optimal excitation coefficients and BW=60°

For the non-broadside arrays, the phase part of the excitation coefficients for the elements should be considered to make the array scanning to other degrees. For that kind of arrays, we will use the Particle Swarm Optimization(PSO) method to design them, which will be discussed in next chapter.

CHAPTER 3

PARTICLE SWARM OPTIMIZATION FOR ASYMMETRIC ARRAY

3.1 Analysis of asymmetric array

The symmetric array was designed in the preceding chapter. In this chapter, the asymmetric array will be designed. Compared with the symmetric case, the asymmetric array will make the problem more complicated because the positions of certain elements need not to be symmetric. But we can use the same method to analyze it and solve it.

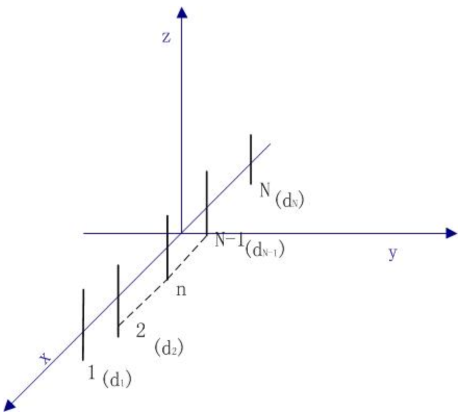

Fig. 3.1. An asymmetric array

The problem begins with the basic expression of array factor

$$AF(\phi) = w_1 e^{-jkd_1 cos\phi} + w_2 e^{-jkd_2 cos\phi} + ... + w_N e^{-jkd_N cos\phi} \tag{3.1}$$

according to the Euler's formula, (3.1) can be written as

$$\begin{aligned} AF(\phi) =& w_1(-jsin(kd_1cos\phi) + cos(kd_1cos\phi)) + ... \\ &+ w_N(-jsin(kd_Ncos\phi) + cos(kd_Ncos\phi)) \end{aligned} \tag{3.2}$$

since the expression cannot be reduced to the form as the symmetric case, so it is better to rewrite it as

$$\begin{aligned} AF(\phi) =& w_1cos(kd_1cos\phi) + ... + w_Ncos(kd_Ncos\phi) \\ &- j(w_1sin(kd_1cos\phi) + ... + w_Nsin(kd_Ncos\phi)) \end{aligned} \tag{3.3}$$

For the array factor, it is the magnitude that is usually important. So the objective function becomes $|AF(\phi)| = \sqrt{AF(\phi)AF(\phi)^*}$. As we assumed in the case of symmetric array, when $\phi = \phi_d$, $AF(\phi_d) = 1$. We can finally get the first two equations that govern the performance of the asymmetric array

$$w_1cos(kd_1cos\phi_d) + ... + w_Ncos(kd_Ncos\phi_d) = 1 \tag{3.4}$$

$$w_1sin(kd_1cos\phi_d) + ... + w_Nsin(kd_Ncos\phi_d) = 0. \tag{3.5}$$

The weights are assumed to be real numbers now because the broadside arrays is designed at first, the non-broadside array with complex weights will be discussed later. The above two equations cannot guarantee that the magnitude of the array factor reaches the peak at ϕ_d, one more constraint regarding its derivative should be added:

$$\frac{\partial AF(\phi)AF(\phi)^*}{\partial \phi}|\phi_d = 0. \tag{3.6}$$

Equation 3.6 can be expanded as

$$\frac{\partial AF(\phi)AF(\phi)^*}{\partial \phi}|\phi_d = AF(\phi)\frac{\partial AF(\phi)^*}{\partial \phi}|\phi_d + AF(\phi)^*\frac{\partial AF(\phi)}{\partial \phi}|\phi_d \tag{3.7}$$

$$\frac{\partial AF(\phi)^*}{\partial \phi}|\phi_d = R' - j \cdot I' \tag{3.8}$$

$$\frac{\partial AF(\phi)}{\partial \phi}|\phi_d = R' + j \cdot I' \tag{3.9}$$

where R denotes the real part of the array factor, I is the imaginary part, their derivatives are denoted by R' and I' respectively. Through some basic algebra we can get

$$R' = w_1 k d_1 sin(\phi) sin(k d_1 cos\phi) + ... + w_N k d_N sin(\phi) sin(k d_N cos\phi)|\phi_d \tag{3.10}$$

$$I' = w_1 k d_1 sin(\phi) cos(k d_1 cos\phi) + ... + w_N k d_N sin(\phi) cos(k d_N cos\phi)|\phi_d. \tag{3.11}$$

Since R is set as 1 and I is set as 0 at ϕ_d, equation 3.7 can be reduced to

$$\frac{\partial AF(\phi)AF(\phi)^*}{\partial \phi}|\phi_d = 2 \cdot R'. \tag{3.12}$$

Therefore another equation of the asymmetric array becomes

$$w_1 k d_1 sin(\phi) sin(k d_1 cos\phi) + ... + w_N k d_N sin(\phi) sin(k d_N cos\phi)|\phi_d = 0. \tag{3.13}$$

Now the problem can be summarized as minimizing $|AF(\phi|$ ($\phi \notin$ main beam), subject to (3.4), (3.5) and (3.13). Compared with symmetric array, the difference is the objective function changes into a complex number, which makes it difficult to build the linear relationship between the magnitude of array factor and the excitation coefficients, so linear programming is not the most appropriate method in this case. Here the Particle Swarm Optimization (PSO) method [14] will be used to determine the optimal excitation coefficients for this kind of array.

3.2 Particle Swarm Optimization

Particle Swarm Optimization(PSO) was originally introduced by Kennedy, Eberhart and Shi [14] [15], it has been used in areas such as power system and antenna designs [16] [17]. The PSO method is easy to implement, but its disadvantage is also apparent that it cannot guarantee that the final result is globally optimal [18]. The basic idea behind the algorithm is simulating the movement of a group of animals. For example, when a bird flock goes to find food, the most important question for them is where to go. The rational behavior for a bird is to decide its destination by comparing the places it went previously to places other birds visited. If it finds more food at the last location it visited, it will continue to fly there or moving a little around at the point to try to find more food. If not, it will fly to other locations found by other birds because the possibility of finding more food there is generally higher. As the process continues, more and more smart birds will come to the location with most food. In a word, the movement of the bird depends on its own best known location and the best known location of the whole group.

Let's extend the example to a more general model, assuming there are N birds in each flock, each bird can be seen as one particle in the algorithm, and the space is extended to M dimensions, the total number of the particles is $M * N$. The movement of each particle will be updated by following two equations

$$v_{mn}^{k+1} = wv_{mn}^{k} + c_1\xi(p_{mn}^{k} - x_{mn}^{k}) + c_2\eta(p_{gm}^{k} - x_{mn}^{k}) \tag{3.14}$$

$$x_{mn}^{k+1} = x_{mn}^{k} + rv_{mn}^{k+1} \tag{3.15}$$

where v_{mn}^{k+1} denotes the velocity of n^{th} particle in m^{th} dimension at $(k+1)^{\text{th}}$ time, x_{mn}^{k} denoted the position of the n^{th} particle in m^{th} dimension at k^{th} time, the initial velocity and position of each particle are generated randomly. Inertia weight is denoted by w, it relates the next velocity with previous one and is empirically set to 1.2 [15]. The cognitive constant and social constant are denoted by c_1 and c_2, respectively.

They build a relation between velocity of next time with its own best known value and the best known value of the whole group. Both of these are set to a value of two [14]. There are two random numbers denoted by ξ and γ, which are updated every time. The best known outputs of its own and the group are denoted by p_{mn}^k and p_{gm}^k respectively, and are similar to the quantity of the food of the location for birds. They are calculated by the best known position of the individual particle and the best known position of the group. In (3.15), the parameter r is the update constant, it is always set as 1 [14]. In total, this describes one basic model of PSO method.

With the several decade's development of the algorithm, some more efficient PSO methods have appeared, for example, one PSO method with nonconstant c_1 and c_2 has already been proposed in order to improve the efficiency and speed of the algorithm, more details about these advanced PSO methods can be found in [19] [20] [21].

The following flowchart shows the procedure of the basic PSO method.

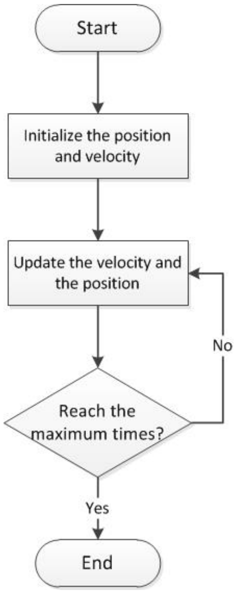
Start
Initialize the position and velocity
Update the velocity and the position
No
Reach the maximum times?
Yes
End

Fig. 3.2. The flowchart of the PSO method

A simple problem solved by the PSO method is shown, the objective function is given $y = 2 - cos(3 * x) * exp(-x)$, we would like to find its maximum value in the range $x \in [0, 4]$. The number of the particle is chosen as 2 and the maximum iteration number is set as 50, which means each particle can try 50 times and every time there are 2 particles looking for the answer, the final result is shown in Fig.3.3. Even though the exact maximum value of y and the value of x corresponding to the peak are not known, what can be seen is after 50 iterations, more points are found in the peak region.

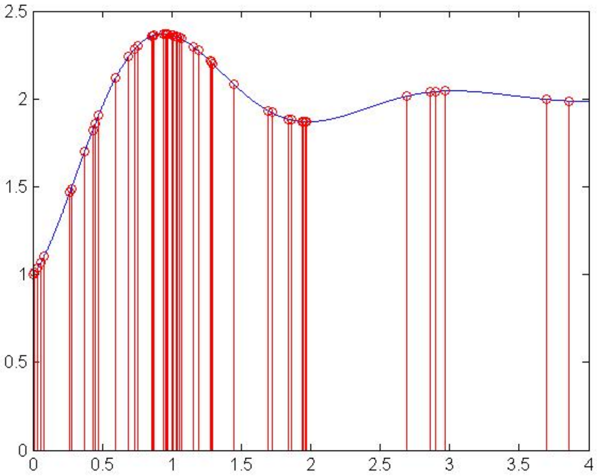

Fig. 3.3. An example of the PSO method

3.3 Some other popular algorithms

Besides the PSO method, there exist other methods which are also popular in the area of beamforming, such as the Genetic Algorithm (GA) method[22], the Simulated Annealing (SA) method [23] and so on. They are also used to get the approximate solution to the nonlinear and complicated problems.

The GA method was developed by simulating the process of natural selection. In the GA method, the potential optimal weights are seen as a population, a population consists of several individuals which is similar to the weight for each element in the array, a fitness function as the SLL in this thesis is chosen for evaluating the population. As evolution affects the populations in nature, in this method, the population will also go through selection, recombination and mutation, then the next generation will be generated. After appropriate iterations(the number of the iterations depends on the complexity of the problem), the solution will approach to the optimal solution.

The SA method was introduced by Scott Kirkpatrick, C. Daniel Gelatt and Mario P. Vecchi in 1983 [24], it is a kind of algorithm based on the Monte Carlo method to find out the optimal solution. This method comes from the procedure of annealing in metallurgy. When the temperature of the metal is cooled down, the inner particles will tend to be stable, and the thermodynamic free energy will decrease, if the thermodynamic free energy reaches the lowest level, the state is called the stable state. The whole procedure is controlled by cooling schedule which includes the iteration number, the termination requirement and so on. When it is used for beamforming, first of all, we generate a random solution, then we evaluate the solution, if it does not satisfy the specification (e.g., the SLL should be below -30 dB) or it does not reach the iteration number, it will generate a new solution, after the new solution is generated, we need to compute its ”energy” (the energy is related with current temperature) and decide to accept it or not, then we evaluate the new solution. If the solution meets the specification or the it reaches the iteration number, then the solution is just the final solution. More details about the GA and the SA methods in beamforming can be found in [25] [26].

3.4 Numerical results

In this section, we will use some examples to demonstrate the PSO method in designing the array. To begin, the PSO method was used to design a ten-element symmetric broadside array which was solved by linear programming previously, the beamwidth is 40°. The comparison of the results using the two methods are shown Table 3.1 and Fig.3.4.

Methods	w_1	w_2	w_3	w_4	w_5	Minimum SLL(dB)
LP	0.1201	0.1224	0.0888	0.1152	0.0535	-30.343
PSO	0.121	0.1226	0.0858	0.1165	0.0538	-29.37

Table 3.1. Comparison between LP and PSO method

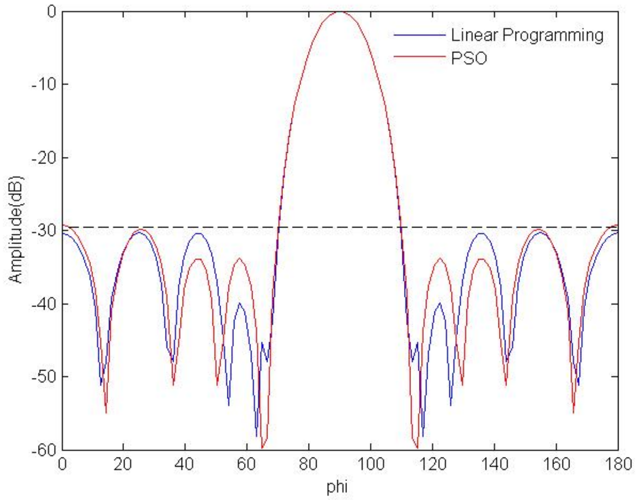

Fig. 3.4. Array factor of symmetric broadside array, 40°

From Table 3.1, it can be seen that using the PSO method we get similar weights compared with using linear programming method. In this case, the maximum iteration number is set as 20000 and the number of particles is 4 for the PSO method. Fig.3.4 shows the corresponding array factor, since the SLL is the criterion for designing, the two methods get the approximate result in term of the SLL. Next, an eight-element asymmetric array is used to verify the method in asymmetric case, the desired angle is 90°, the beamwidth is 40°. In the side lobe region, the array factor it is sampled every degree. The positions of an eight-element array are set as $d_1 = 1.2\lambda$, $d_2 = 0.85\lambda$, $d_3 = 0.4\lambda$, $d_4 = 0.15\lambda$, $d_5 = -0.1\lambda$, $d_6 = -0.45\lambda$, $d_7 = -0.9\lambda$, $d_8 = -1.4\lambda$. By (3.4), (3.5) and (3.13), we can get the following equation

$$w_1 + w_2 ... + w_8 = 1. \tag{3.16}$$

For an array with eight elements, the number of the dimensions are set to 8, and for each dimension 4 particles are chosen and the maximum iteration number is chosen as 500000. In order to avoid the shortfall of partial optimization, a block of computer code is added into the program to fix it. The optimal excitation coefficients that were determined are: $w_1 = 0.085, w_2 = 0.0895, w_3 = 0.148, w_4 = 0.1149, w_5 = 0.0836, w_6 = 0.182, w_7 = 0.1543, w_8 = 0.1431$, which took 631 seconds to get the solution with a quad-core 3.2 GHz CPU. In this case, the SLL is approximate -19dB. The array factor with optimal excitation coefficient is shown in Fig.3.5, and the change of minimum SLL versus the iteration is shown in Fig.3.6. It can be seen that the minimum SLL keeps decreasing with iteration number increasing at first, then after around 170,000 times, the minimum SLL remains unchanged until the maximum iteration number is reached.

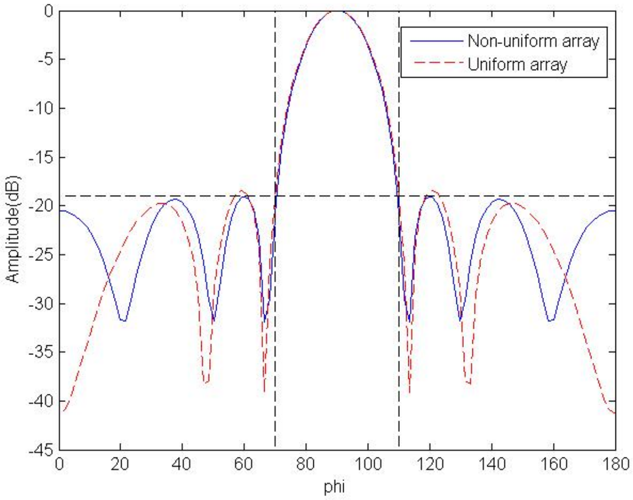

Fig. 3.5. Array factor of the asymmetric array

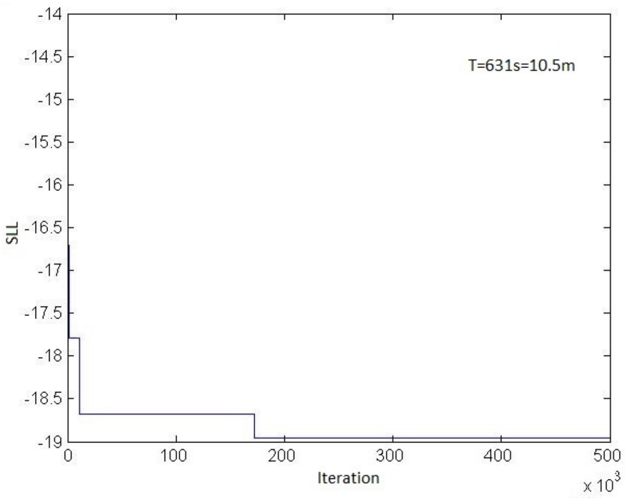

Fig. 3.6. The minimum SLL versus iteration

Since the PSO method cannot guarantee that the final result is globally optimal, it is necessary to consider whether the result is reasonable or not. One approach would be to use the Taylor method to redesign the array. This method is specific for the uniform spacing arrays. When the total length of the array is 2.6λ, an eight-element with uniform spacing of 0.371λ is generated. Using the Taylor method, the SLL we can achieve is -18.4 dB for the broadside uniform spacing array with BW=40°, which is also shown in Fig.3.5, considering the difference of the nature of the two arrays, our results for the asymmetric array is reasonable. Then the SLL versus iteration number is checked, as mentioned before, the SLL drops at first with iteration number increasing, then it is kept being constant for a long time. For that two reasons, our results can be seen as the reasonable result.

The optimum weights and the minimum SLL are gotten via the PSO method for asymmetric array with different numbers of elements and various beamdwidths are shown in Table 3.3-3.5. The position of the elements is shown in Table 3.2.

N	d_1/λ	d_2/λ	d_3/λ	d_4/λ	d_5/λ	d_6/λ	d_7/λ	d_8/λ	d_9/λ	d_{10}/λ
2	0.15	-0.1								
4	0.4	0.15	-0.1	-0.45						
6	0.85	0.4	0.15	-0.1	-0.45	-0.9				
8	1.2	0.85	0.4	0.15	-0.1	-0.45	-0.9	-1.4		
10	1.7	1.2	0.85	0.4	0.15	-0.1	-0.45	-0.9	-1.4	-2.2

Table 3.2. Elements positions of the asymmetric array

w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}	SLL(dB)
0.5	0.5									-0.318
0.454	0.003	0.13	0.416							-3.606
0.216	0.151	0.107	0.142	0.13	0.255					-11.454
0.099	0.115	0.149	0.087	0.09	0.198	0.134	0.126			-18.923
0.064	0.086	0.151	0.126	0.112	0.092	0.142	0.134	0.081	0.011	-23.09

Table 3.3. Optimum weights for asymmetric array, broadside and BW=40°

w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}	SLL(dB)
0.5	0.5									-0.488
0.422	0.004	0.2	0.377							-5.371
0.2	0.154	0.119	0.159	0.199	0.169					-15.869
0.062	0.076	0.203	0.009	0.203	0.163	0.181	0.103			-23.836
0.075	0.112	0.137	0.205	0.053	0.16	0.129	0.105	0.015	0.009	-25.11

Table 3.4. Optimum weights for asymmetric array, broadside and BW=50°

w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}	SLL(dB)
0.5	0.5									-0.688
0.421	0.002	0.264	0.333							-7.353
0.165	0.161	0.169	0.144	0.225	0.137					-20.528
0.025	0.084	0.22	0.026	0.2	0.212	0.176	0.057			-26.267
0.075	0.165	0.158	0.27	0.001	0.167	0.083	0.041	0.021	0.021	-28.29

Table 3.5. Optimum weights for asymmetric array, broadside and BW=60°

From Table 3.3 to 3.5, we can also conclude that with the beamwidth extending, the potential SLL can be lower. The final array factors of the ten-element arrays with various beamwidths are shown in Fig.3.7 through 3.9. In these figures, the letter I denotes the iteration number, P denotes the number of the particles in each dimension and T denotes the total time needed to perform the optimization.

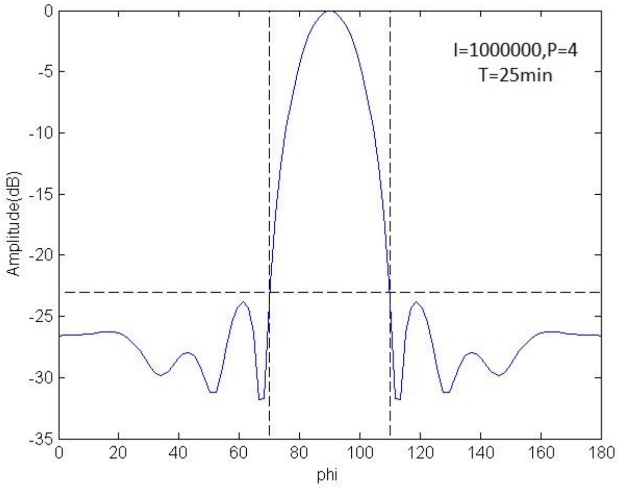

Fig. 3.7. Array factor of the ten-element asymmetric array, BW=40°

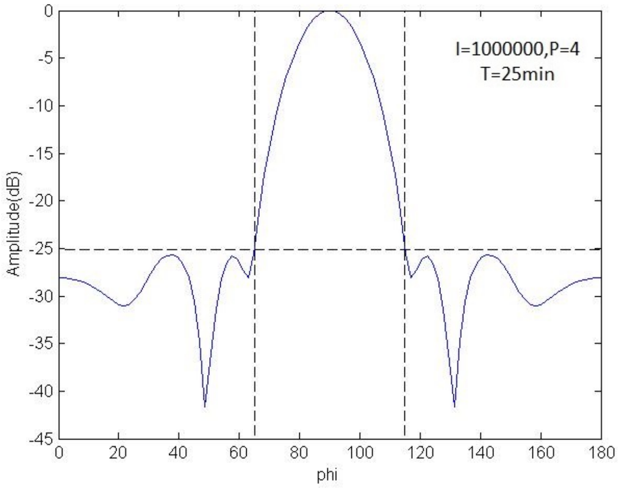

Fig. 3.8. Array factor of the ten-element asymmetric array, BW=50°

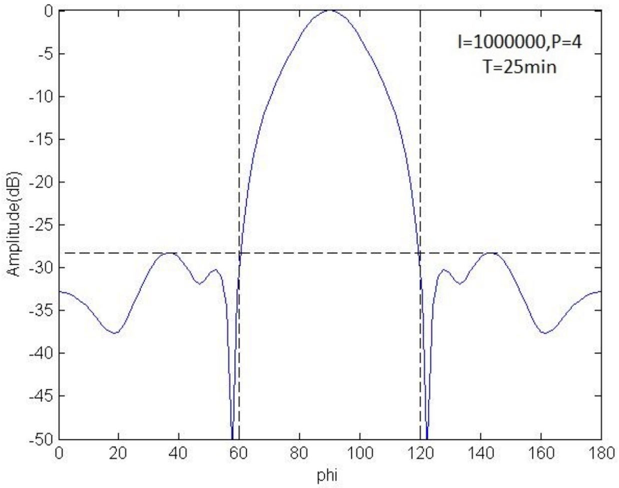

Fig. 3.9. Array factor of the ten-element asymmetric array, BW=60°

For the non-broadside array, the weights have two parts, the amplitude and the phase. When this kind of problem is formulated for the PSO method, the weights are set as $w_n + j * w'_n$ and the equation constraints in this case are

$$\sum_{n=1}^{N} w_n cos(kd_n cos\phi_d) + w'_n sin(kd_n cos\phi_d) = 1 \tag{3.17}$$

$$\sum_{n=1}^{N} -w_n sin(kd_n cos\phi_d) + w'_n cos(kd_n cos\phi_d) = 0. \tag{3.18}$$

The derivative of the real part of the array factor in this case is

$$R' = \sum_{n=1}^{N} w_n kd_n sin(\phi) sin(kd_n cos\phi) - w'_n kd_n sin(\phi) cos(kd_N cos\phi) \tag{3.19}$$

an additional equation constraint in this case becomes

$$\sum_{n=1}^{N} w_n kd_n sin(\phi) sin(kd_n cos\phi) - w'_n kd_n sin(\phi) cos(kd_N cos\phi) | \phi_d = 0. \tag{3.20}$$

The remaining part of the algorithm is kept the same as the broadside case. Comparing with the broadside array, it can be found that the problem in this case becomes more complicated with the introduction of the imaginary part of the weights, so the iteration number and the number of the particles in each dimension under the same condition in broadside case is increased. The positions of the elements of the array are shown in Table 3.6.

Number of Elements	d_1/λ	d_2/λ	d_3/λ	d_4/λ	d_5/λ	Min./λ	Max./λ
2	0	0.35				0.35	0.35
3	0	0.35	1.4			0.35	0.95
4	0	0.35	1.4	1.85		0.35	0.95
5	0	0.35	1.4	1.85	2	0.15	0.95

Table 3.6. Element spacing of the non-broadside array

The desired angle is set as 45°, and the range of the beamwidths is from 40° to 60°. The iteration number(I) and the number of the particles (P) for each dimension we set in different cases are shown in Table 3.7, the time the code consumed (T/m) is also presented in this table.

Number of Elements	I	P	T/m
2	200000	4	10.4
3	500000	6	87.4
4	1000000	8	263.3
5	1500000	15	631.9

Table 3.7. Some parameters we set for the non-broadside array

The optimum weights and the SLL determined in this case are shown in Table 3.8-3.10, where w_n represents the real part of the weights and w_n' represent the imaginary part.

w_1	0.5	0.324	0.249	0.278
w'_1	0	-0.1	-0.052	0.0002
w_2	0.008	-0.13	-0.088	-0.089
w'_2	0.499	0.205	0.288	0.162
w_3		0.47	0.163	0.14
w'_3		-0.063	-0.104	-0.151
w_4			-0.156	0.371
w'_4			0.257	0.284
w_5				-0.495
w'_5				-0.284
SLL(dB)	-0.21	-3.24	-5.05	-6.59

Table 3.8. Optimum weights for non-broadside arrays, θ_d=45°, BW=40°

w_1	0.5	0.311	0.289	0.15
w'_1	0	-0.109	0.006	-0.138
w_2	0.008	-0.142	-0.036	-0.182
w'_2	0.499	0.265	0.329	0.293
w_3		0.423	0.191	0.156
w'_3		-0.063	-0.176	0.008
w_4			-0.173	0.181
w'_4			0.127	0.333
w_5				-0.249
w'_5				-0.109
SLL(dB)	-0.287	-4.3	-6.516	-7.292

Table 3.9. Optimum weights for non-broadside arrays, θ_d=45°, BW=50°

w_1	0.5	0.293	0.261	0.265
w'_1	0	-0.117	-0.032	-0.013
w_2	0.008	-0.151	-0.074	-0.104
w'_2	0.499	0.32	0.378	0.221
w_3		0.386	0.196	0.198
w'_3		-0.064	-0.148	-0.215
w_4			-0.137	0.078
w'_4			0.116	0.352
w_5				-0.16
w'_5				-0.261
SLL(dB)	-0.356	-4.954	-7.21	-8.316

Table 3.10. Optimum weights for non-broadside arrays, θ_d=45°, BW=60°

Additional degrees are selected as the scan angles, in this thesis, 30° and 60° are also selected as the scan angles to verify our method, the beamwidth is set as 60°, the optimum weights we got are shown in Table 3.11-12.

w_1	0.5	0.451	0.386	0.299
w_1'	0	-0.058	-0.012	-0.068
w_2	-0.164	-0.103	-0.006	-0.216
w_2'	-0.472	0.061	0.082	0.078
w_3		0.126	0.079	0.167
w_3'		0.44	0.113	-0.123
w_4			-0.293	-0.367
w_4'			-0.258	0.084
w_5				0.321
w_5'				-0.432
SLL(dB)	-0.1	-1.489	-2.499	-3.153

Table 3.11. Optimum weights for non-broadside arrays, θ_d=30°, BW=60°

w_1	0.5	0.254	0.075	0.063
w_1'	0	-0.129	-0.06	-0.137
w_2	0.227	0.029	0.053	0.003
w_2'	0.446	0.437	0.305	0.379
w_3		-0.147	-0.186	-0.135
w_3'		-0.314	-0.385	-0.257
w_4			0.195	0.182
w_4'			-0.07	-0.142
w_5				0.101
w_5'				-0.017
SLL(dB)	-0.723	-5.506	-6.24	-6.35

Table 3.12. Optimum weights for non-broadside arrays, θ_d=60°, BW=60°

The array factor of the five-element non-broadside array with various beamwidths are shown in Fig.3.10-3.14.

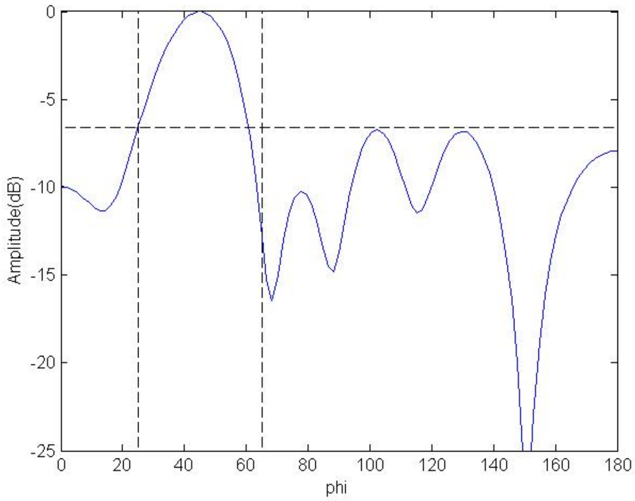

Fig. 3.10. Array factor of the five-element asymmetric array, θ_d=45°, BW=40°

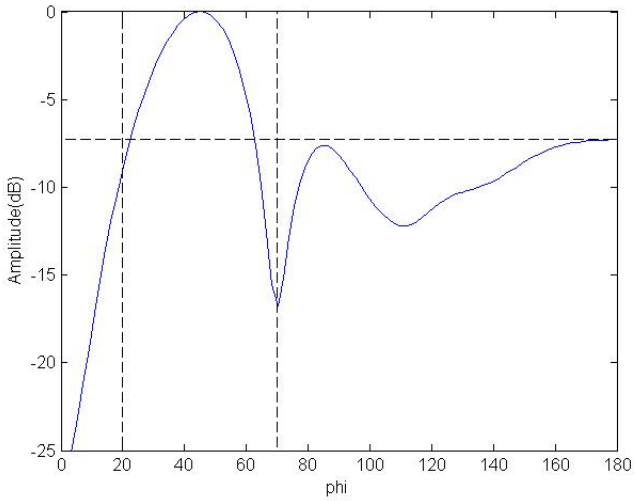

Fig. 3.11. Array factor of the five-element asymmetric array, θ_d=45°, BW=50°

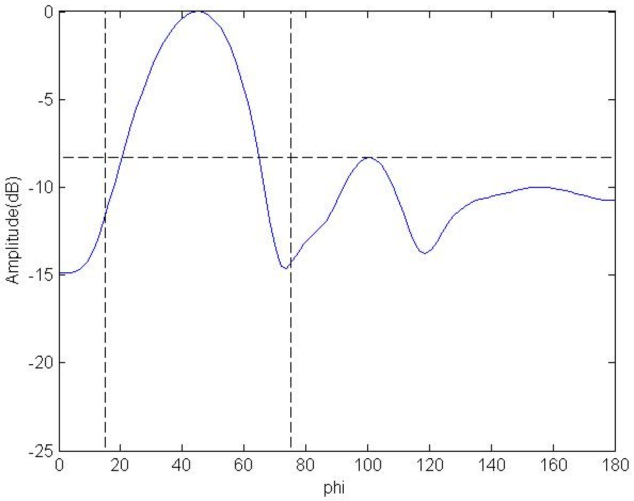

Fig. 3.12. Array factor of the five-element asymmetric array, θ_d=45°, BW=60°

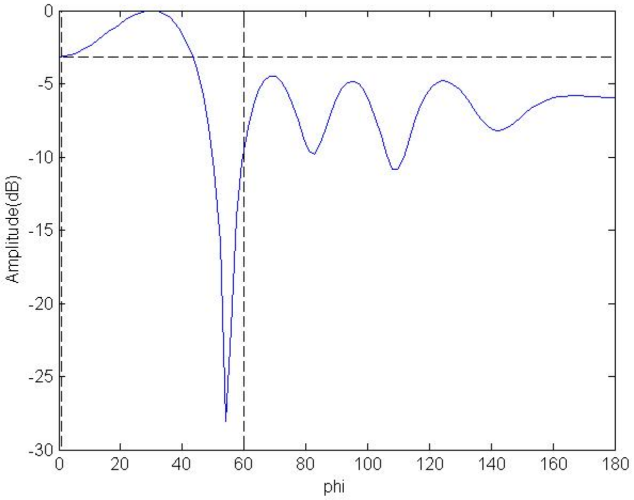

Fig. 3.13. Array factor of the five-element asymmetric array, θ_d=30°, BW=60°

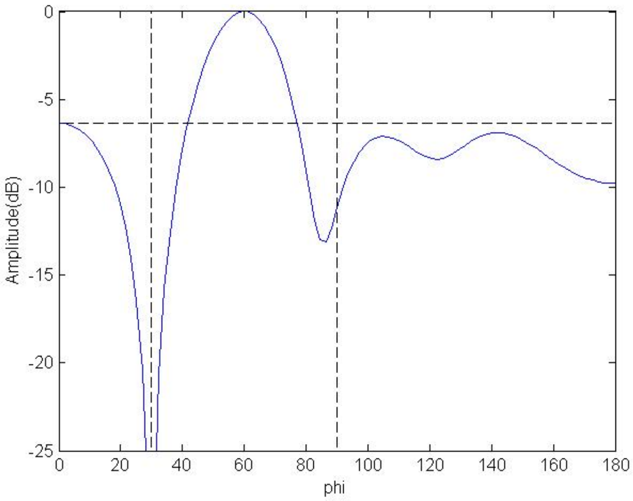

Fig. 3.14. Array factor of the five-element asymmetric array, θ_d=60°, BW=60°

How can we say the SLL determined now is the lowest with the given conditions, or what is the limit for the SLL with the given conditions? In fact, for an arbitrary linear array and an arbitrary beamwidth, it is difficult to find out its limit about the SLL, and unlike linear programming method, the PSO method can not guarantee that its final result is the globally optimal result. A useful intermediate step is to prove that the final result is the approximate optimal result. Like what is done in broadside case, the final result is compared with the SLL achieved by the Taylor method for uniform spacing arrays, the SLL achieved by the Taylor method is shown in Table 3.13, the number inside the parentheses is the SLL we achieved previously for the asymmetric array via the PSO method under the same condition. The comparison of the array factor of the two kinds of the arrays are shown in Fig.3.15-17. As is mentioned before, the Taylor method is only effective for uniform spacing arrays, the total length is kept being same as the asymmetric arrays, but their positions to are changed in order to make them into the uniform spacing arrays. The result gotten by the Taylor method is used as the reference, but it is not right to say the uniform spacing can always get better performance, and vice a versa. By comparing the two groups of the results and considering the difference of the positions of some elements, we conclude that our results are reasonable. Secondly, the SLL versus the iteration is studied in different cases; we used arrays with θ_d=45° and BW=60° as the example, which are shown in Fig.3.18-20. It can be seen that the final minimum SLL is being kept constant for a long time in each case. Of course the SLL may drop with setting a larger iteration number, but we do not think it will drop a lot even with a much larger iteration number. Considering the above two respects, the final results we get are the approximate optimal results.

θ_d,BW / N	45°,40°	45°,50°	45°,60°
2	-0.2 dB (-0.21 dB)	-0.3 dB (-0.29 dB)	-0.4 dB (-0.36 dB)
3	-3.2 dB (-3.24 dB)	-4.4 dB (-4.3 dB)	-5.5 dB (-4.95 dB)
4	-5.4 dB (-5.05 dB)	-7.1 dB (-6.52 dB)	-7.8 dB (-7.21 dB)
5	-5.9 dB (-6.59 dB)	-7.4 dB (-7.29 dB)	-8.6 dB (-8.32 dB)

Table 3.13. Comparison of the SLL of the uniform spacing non-broadside arrays and the non-uniform spacing non-broadside arrays

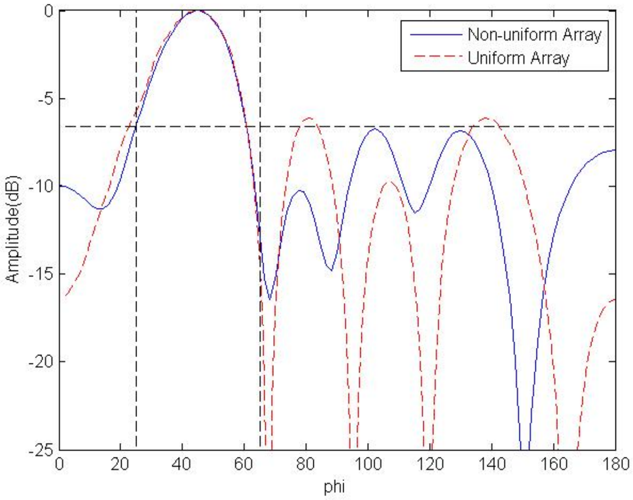

Fig. 3.15. Comparison of the array factor, θ_d=45°, BW=40°

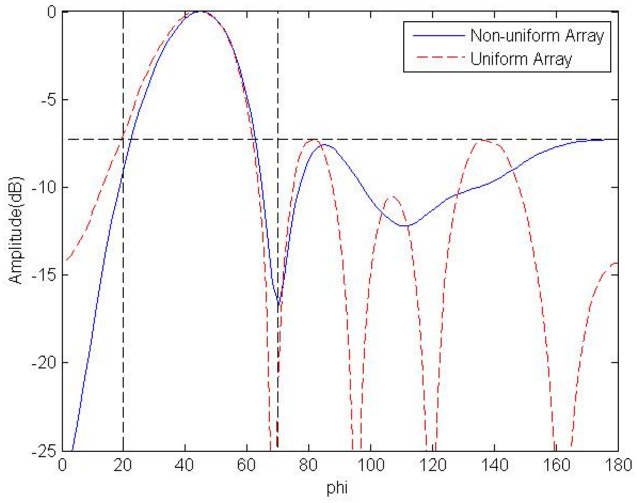

Fig. 3.16. Comparison of the array factor, θ_d=45°, BW=50°

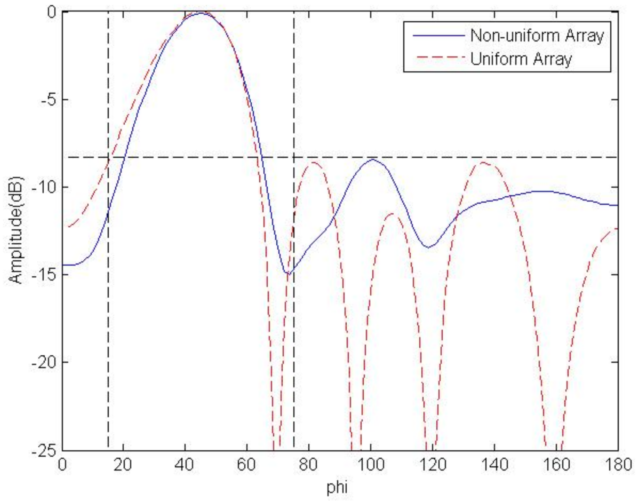

Fig. 3.17. Comparison of the array factor, θ_d=45°, BW=60°

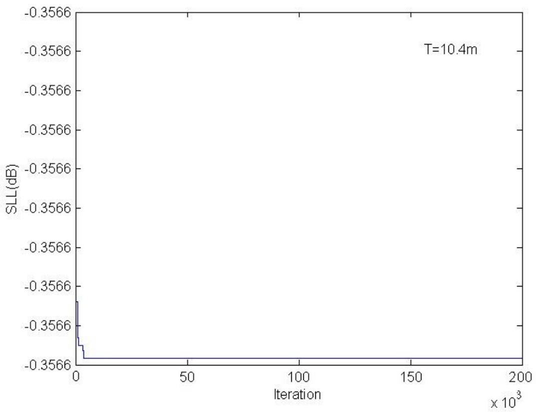

Figure 3.18. The SLL vs. iteration, N=2

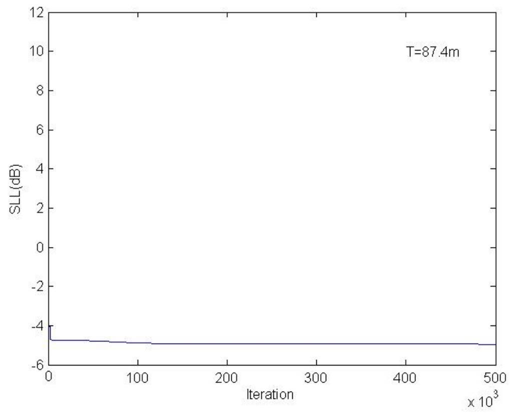

Figure 3.19. The SLL vs. iteration, N=3

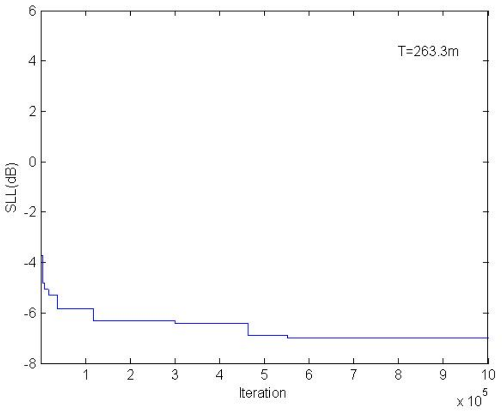

Figure 3.20. The SLL vs. iteration, N=4

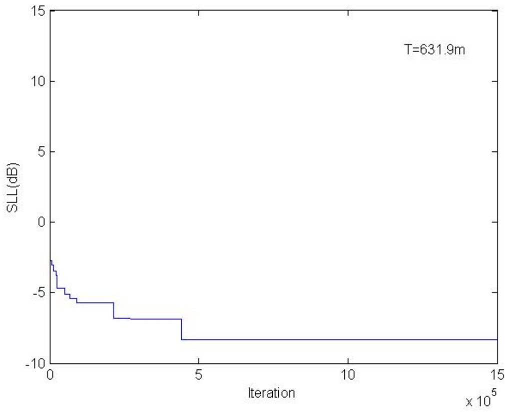

Figure 3.21. The SLL vs. iteration, N=5

From the case of broadside and non-broadside asymmetric arrays, it can be seen that PSO method is helpful for designing this kind of arrays. Compared with the LP method, the PSO method can be used for solving more complicated problems. However, its disadvantages cannot be ignored, one has been mentioned several times in this chapter that the PSO method cannot guarantee that the result is globally optimal. Another one is when the PSO method is used to design the array, it takes longer compared with the LP method, especially for non-broadside array. For example it costs more than ten hours to get the final result for the five-element non-broadside asymmetric array, that is another disadvantage of the PSO method.

CHAPTER 4

ADAPTIVE ARRAYS VIA THE NORMALIZED LEAST MEAN SQUARES ALGORITHM

4.1 Background of adaptive arrays

The Adaptive array is also called smart antenna, it is widely used in astronomy and communication engineering [27] [28]. It is used for calibrating or adjusting the excitation coefficients of the array to make the array get optimal beamforming performance in different cases. Why is it important for phased array? As it was mentioned in the proceeding part, the excitation coefficients are vital to the array, they can determine the radiation pattern of the array. In this chapter, a method is proposed to help us calibrate the excitation coefficients. We will use dither signal and a near field sensor to accomplish this. Our method will be demonstrated in noiseless environment, noisy environment, and the effect of mutual coupling and discrete dither signal will be considered.

One correction method via dithering and the Least Mean Squares (LMS) algorithm has been proposed in [5]. The LMS algorithm was developed by Widrow in 1960 [29]. In that method of [5], white noise is deliberately introduced to the coefficients as the dither signal, then the LMS algorithm is used to correct the actual coefficients. In using the LMS algorithm to correct the coefficients, one thing we have to do is estimating the largest eigenvalue of the coefficient matrix to update the coefficients in gradient based procedure, that is described in [30]. This necessary step in the LMS algorithm is one drawback since it will make the method more complicated and take a longer time. In order to avoid this problem, the Normalized Least Mean Squares

(NLMS) algorithm [31] is proposed to update the coefficients in this part of this thesis.

4.2 Theory of adaptive arrays via the NLMS algorithm

‘

4.2.1 Introduction

An array consisting of N half-wavelength dipoles with all dipoles are arranged along the x-axis with an inter-element spacing of d is illustrated in Fig.4.1. A sensor is set in the near-field of the array to measure the array's radiation. The normalized current excitation coefficient for n-th element is denoted by w_n, it is same as the weights in proceeding part, $w_n = a_n e^{j\psi_n}$, where a_n represents the magnitude and ψ_n is the phase. Once the array is excited, the array will radiate power, which will be received by the sensor, the z-component of the electric field E_z in the x-y plane is given by [4]

$$E_z = \frac{j\eta I_0 l}{4\pi} \sum_{n=1}^{N} w_n \frac{1}{R_n} \left(1 + \frac{1}{jk_0 R_n} - \frac{1}{k_0^2 R_n^2}\right) e^{-jk_0 R_n} \tag{4.1}$$

$$= \frac{j\eta I_0 l}{4\pi} \mathbf{g}\mathbf{w} \tag{4.2}$$

where $R_n = \sqrt{(x_s - x_n)^2 + y_s^2}$, x_s is the value of the sensor's position in x-axis, y_s is the value of the sensor's position in y-axis, x_n is the location of n-th element in the array, k_0 is the wavenumber in free space. In (4.2), $\mathbf{w}$ is a $1 \times N$ matrix, $\mathbf{w} = [w_1, w_2 .... w_n .... w_N]^T$, $\mathbf{g}$ is an $N \times 1$ matrix, $\mathbf{g} = [g_1, g_2 .... g_n .... g_N]$, and g_n is defined as

$$g_n = \frac{1}{R_n} \left(1 + \frac{1}{jk_0 R_n} - \frac{1}{k_0^2 R_n^2}\right) e^{-jk_0 R_n}. \tag{4.3}$$

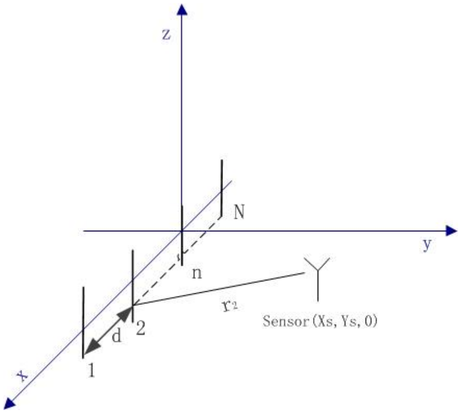

Fig. 4.1. Array consisting of N elements and the sensor

The NLMS algorithm will be used to calibrate the excitation coefficients. The general model of the NLMS algorithm is depicted in Fig.4.2. The actual coefficients that we will correct is represented by vector $\widehat{\mathbf{s}}$, and vector $\mathbf{s}$ represents the desired or reference coefficients, $\mathbf{h}$ is the input vector, y represents the desired output, while $\widehat{y}$ is the actual output, e denotes the difference between y and $\widehat{y}$.

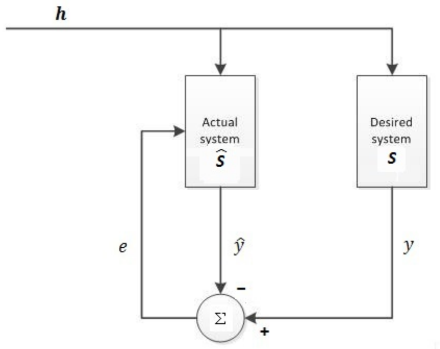

Fig. 4.2. The general model of NLMS algorithm

The general NLMS algorithm can be summarized as

$$\mathbf{h} = [h_1, h_2......h_N] \tag{4.4}$$

$$\widehat{\mathbf{s}}^{(k)} = [\widehat{s}_1^{(k)}, \widehat{s}_2^{(k)}......\widehat{s}_N^{(k)}]^T \tag{4.5}$$

$$\mathbf{s} = [s_1, s_2......s_N]^T \tag{4.6}$$

$$\widehat{y}^{(k)} = \widehat{\mathbf{s}}^{(k)} \cdot \mathbf{h} \tag{4.7}$$

$$y = \mathbf{s} \cdot \mathbf{h} \tag{4.8}$$

$$e^{(k)} = \widehat{y}^{(k)} - y \tag{4.9}$$

$$\widehat{\mathbf{s}}^{(k+1)} = \widehat{\mathbf{s}}^{(k)} - \mu \frac{(e^{(k)})^* \mathbf{h}^{(k)}}{(\mathbf{h}^{(k)})^H \mathbf{h}^{(k)}} \tag{4.10}$$

where the superscript k in (4.5) represents the iteration number and in (4.10) the symbol $*$ denotes complex conjugation, H denotes Hermitian conjugation, and μ is the step size. The initial condition is always set as $\widehat{\mathbf{s}}^{(0)} = [0, 0......0]$ if it is not given, then with iteration number k increasing, $\widehat{\mathbf{s}}$ will converge to the desired coefficients $\mathbf{s}$ step by step, the proof of the NLMS algorithm is in the Appendix. In our research, the actual current excitation coefficients can be seen as $\widehat{\mathbf{s}}$ in the model, the desired current excitation coefficients can be seen as $\mathbf{s}$, all terms except $\mathbf{w}$ can be seen as the input vector $\mathbf{h}$, by (4.2) we can get $h_n = \frac{j\eta I_0 l}{4\pi} \frac{1}{R_n} \left(1 + \frac{1}{jk_0 R_n} - \frac{1}{k_0^2 R_n^2}\right) e^{-jk_0 R_n}$, for simplicity, the constant factor is ignored, so $h_n = \frac{1}{R_n} \left(1 + \frac{1}{jk_0 R_n} - \frac{1}{k_0^2 R_n^2}\right) e^{-jk_0 R_n}$, the actual electric field which can be sensed by the sensor is the actual output in the model, and the desired electric field pattern which can be seen as the desired output is already known. Then the algorithm will be verified in different situations.

4.2.2 Noise Free Case

First of all, let us consider the ideal and simplest situation that the array works in a noise-free environment; the effect of noise will be talked later. Before correcting the actual coefficients, a dither signal should be added into the actual coefficients which are denoted by $\widehat{\mathbf{w}}$. The desired coefficients are denoted by $\mathbf{w}$. Dithering has been used for correcting signals in a lot of areas, especially in the area of digital audio and video processing [32], the effect of dithering on correction to phased array coefficients has been studied in [5] and the comparison between the coefficients and field without dithering and with dithering are shown in Fig.4.3 and Fig.4.4. Once the dither signal is added, the algorithm will be used to correct the actual coefficients. In this thesis, desired coefficients $\mathbf{w}$ are known, they are current excitation coefficients of the 32-element array with spacing of $\frac{\lambda}{2}$, λ is the wavelength, the location of each element is $x_n = -L + (n-1)\frac{\lambda}{2}$, L is the half length of the whole array, and the array is designed by the Taylor method [3] with side lobe level of -24dB. The weights $\widehat{\mathbf{w}}$ are generated randomly to denote an arbitrarily excitation coefficients of the array, $\mathbf{w}$ and $\widehat{\mathbf{w}}$ are shown in Fig.4.3. Since the current excitation coefficients in our research have two parts, the magnitude part and the phase part, the dither signals also consist of two parts, for the magnitude part, it is subject to log-normal distribution with the standard deviation of σ dB, and the phase part is subject to uniform distribution with deviation of Δ. The magnitude part and the phase part of the dither signal are independent of each other, and they are independent of the original coefficients $\widehat{\mathbf{w}}$. Vector $\widetilde{\mathbf{w}}$ is denoted as the desired coefficients with the dither signals added to it, $\widetilde{\mathbf{w}} = [\widetilde{w}_1, \widetilde{w}_2....\widetilde{w}_n....\widetilde{w}_N]^T$, and vector $\widetilde{\widehat{\mathbf{w}}}$ is denoted as the actual coefficients with the dither signals added to it, $\widetilde{\widehat{\mathbf{w}}} = [\widetilde{\widehat{w}}_1, \widetilde{\widehat{w}}_2....\widetilde{\widehat{w}}_n....\widetilde{\widehat{w}}_N]^T$, $\widetilde{w_n}$ and $\widetilde{\widehat{w}}_n$ are defined as

$$\widetilde{w_n} = w_n e^{\alpha_n} e^{j\beta_n} \tag{4.11}$$

$$\widetilde{\widehat{w}}_n = \widehat{w}_n e^{\alpha_n} e^{j\beta_n} \tag{4.12}$$

where the superscript n represents the n$^{\text{th}}$ element in the array, $e^{\alpha_n} e^{j\beta_n}$ is the dither signal, $\alpha_n = 0.05 \cdot ln(10) \cdot \sigma \cdot v_n$, $\beta_n = \mu_n \Delta$, where v_n is a normal distribution variable with zero mean and unit variance, μ_n is a variable subject to uniform distribution between -1 and 1. In this thesis σ is set as 3 dB and Δ is set as 12 degrees [5]. In the standard LMS algorithm, the coefficients are corrected by the following equation [5]

$$\widehat{\mathbf{c}}^{(k+1)} = \widehat{\mathbf{c}}^{(k)} - \mu \left\langle (\widehat{E}^{(k)} - E)\mathbf{E}_0^* \right\rangle \tag{4.13}$$

where $E = \mathbf{g}\widetilde{\mathbf{w}}$, $\widehat{E}^{(k)} = \mathbf{g}\widetilde{\widehat{\mathbf{w}}^{(k)}}, \widetilde{\widehat{\mathbf{w}}^{(k)}} = \left[\widehat{w}_1^{(k)} e^{\alpha_1} e^{j\beta_1}, \widehat{w}_2^{(k)} e^{\alpha_2} e^{j\beta_2} ....\widehat{w}_N^{(k)} e^{\alpha_N} e^{j\beta_N}\right]^T$, μ is the step size, the optimal step size μ_{opt} is determined by the largest eigenvalue of coefficients matrix, $\mathbf{E_0} = [t_1, t_2..t_n..t_N]^T$, $t_n = g_n \cdot e^{\alpha_n} e^{j\beta_n}$, $\mathbf{E_0}$ can be seen as the input vector in Fig.4.2, g_n can be calculated by the location of each element in the array and the sensor from (4.3). The location for each element of the array x_n is already known and the location of the sensor is set as $x_s = 0$, $y_s = 4\lambda$, $\langle \cdot \rangle$ denotes $\frac{1}{M}\sum_{m=1}^{M}(\cdot)$, M represents the number of realizations, the values of α and β will update in each realization and the actual coefficients are expected to converge to the desired ones with increasing M under the same iteration. In order to avoid the step of calculating μ_{opt}, the NLMS algorithm is proposed to update the actual coefficients, equation 4.13 will be written as

$$\widehat{\mathbf{w}}^{(k+1)} = \widehat{\mathbf{w}}^{(k)} - \frac{\mu \left\langle (\widehat{E}^{(k)} - E)\mathbf{E}_0^* \right\rangle}{\left(\mathbf{E}_0^*\right)^H \mathbf{E}_0^*}. \tag{4.14}$$

In using the NLMS algorithm, the optimal step size μ_{opt} can be obtained easily, because μ_{opt} in NLMS is constant instead of depending on other parameters, and it is always equal to 1 [33]. Once the value of μ_{opt} is set, the next step is choosing an appropriate value for the realization number M, as it was mentioned before, the value of M larger, the actual coefficient will converge to the desired coefficients better and the results will become stable. However, a larger M means the algorithm will take longer time to get the final results. By observing and comparing the final corrected coefficients with different values of M (from 1 to 100), when M is around 10, it generates satisfactory and stable data. Compared to the results with M equal to 20 or larger, the differences can be ignored. Fig.4.3 shows the actual and desired excitation coefficients and Fig.4.4 shows the electric field with dither signal.

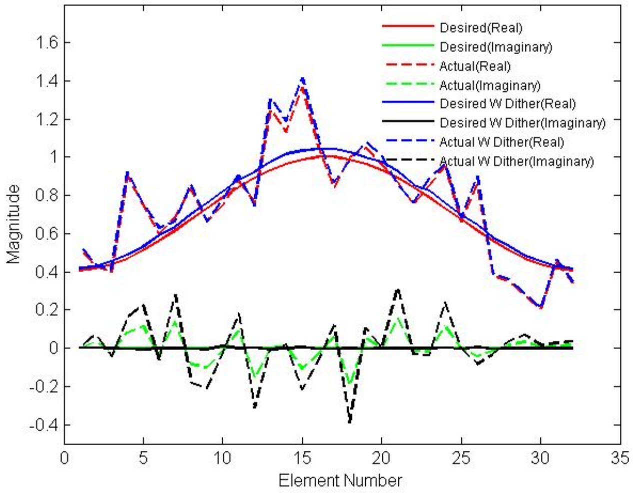

Fig. 4.3. Original coefficients and dithered coefficients

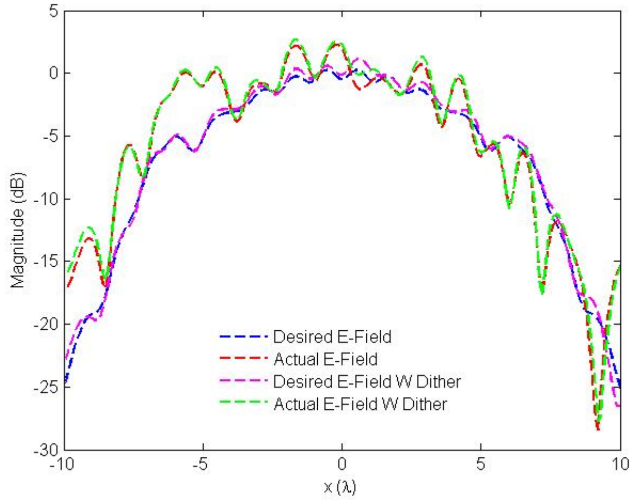

Fig. 4.4. Original and dithered electric field in near-field region

4.2.3 With Gaussian White Noise

In practice, it is impossible to find an environment without noise or interference. In this section the effect of noise to our algorithm will be considered.

The noise being added is assumed to be with zero mean and it is independent of the dithering process. In addition, the noise is supposed to be generated within the receiver. After the noise is considered, the actual electric field component that the sensor receives is written by $\widehat{E}_d$, $\widehat{E}_d^{(k)} = \mathbf{g}\widetilde{\widehat{\mathbf{w}}^{(k)}} + \mathbf{p}\widetilde{\widehat{\mathbf{w}}^{(k)}}$, k denotes the iteration number, and $\mathbf{p}$ represents the noise part, $\mathbf{p} = [p_1, p_2......p_N]$, variable p_n is subject to Gaussian distribution with zero mean and standard deviation of γ. For the desired electric field pattern with the noise added to, it is denoted by E_d, $E_d = \mathbf{g}\widetilde{\mathbf{w}} + \mathbf{p}'\widetilde{\mathbf{w}}$, $\mathbf{p}'$ shares the identical statistical characteristics with $\mathbf{p}$, but they are independent because the noise changes all the time randomly, the input vector with unit excitation coefficients is denoted as $\mathbf{E}_1$, $\mathbf{E}_1 = [t'_1, t'_2..t'_n..t'_N]^T$, $t'_n = g_n \cdot e^{\alpha_n} e^{j\beta_n} + p_n \cdot e^{\alpha_n} e^{j\beta_n}$. Then, the coefficients can be corrected by

$$\widehat{\mathbf{w}}^{(k+1)} = \widehat{\mathbf{w}}^{(k)} - \frac{\mu \left\langle (\widehat{E}_d^{(k)} - E_d)\mathbf{E}_1^* \right\rangle}{(\mathbf{E}_1^*)^H \mathbf{E}_1^*}. \tag{4.15}$$

For the noisy system, the noise part is added to the correcting equation, and this noise part keeps changing as the real noise. Since the algorithm works based on the gradient of the square error, it will bring deterioration to the final results with the introduction of noise, which will be demonstrated in next section.

When it comes to noisy system, one important factor is the signal-to-noise ratio (SNR). In this thesis, the noise part is $\mathbf{p}$, and $\mathbf{g}$ can be seen as the signal part, the SNR for this system is defined as $10 \cdot log(\frac{\|\mathbf{g}\|^2}{\|\mathbf{p}\|^2})$, where $\|\mathbf{g}\|^2 = \sum_{n=1}^{N} g_n^2$, for $\mathbf{p}$, since each element in the vector is independent with each other and subject to Gaussian distribution with zero mean and standard deviation of γ, $\|\mathbf{p}\|^2 = \sum_{n=1}^{N} \gamma^2 = N\gamma^2$. When the algorithm is used under the noisy environment, the value of M should be adjusted with the change of SNR, the detail will be presented in next section.

4.3 Effect of Mutual Coupling

For an array, mutual coupling is one factor that cannot be ignored. In this section, the effect of mutual coupling will be studied on the performance of our algorithm. Like what was shown in the situation with noise, mutual coupling does not alter the essence of the algorithm, but it will change some parameters in the algorithm that will have an impact on the convergence of our algorithm. The impedance of the array is denoted by an N by N matrix $\mathbf{Z}_m$ and defined as [34]

$$\mathbf{Z}_m = \mathbf{Z}_\infty + \Delta\mathbf{Z} \tag{4.16}$$

$\mathbf{Z}_\infty$ is an N by N diagonal matrix with entries Z_∞, Z_∞ represents the input impedance of an isolated element. In this thesis, the array consists of half-wavelength dipoles, thus $Z_\infty = 73 + j42.5$, $\Delta\mathbf{Z}$ denotes the mutual impedance, it is also an N by N matrix, and for side-by-side configuration $\Delta Z_{mn}(m \neq n)$ is calculated by [4] as

$$\Delta Z_{mn} = \frac{\eta}{4\pi}\left[2C_i(\mu_0) - C_i(\mu_1) - C_i(\mu_2)\right] + j\left(-\frac{\eta}{4\pi}\left[2S_i(\mu_0) - S_i(\mu_1) - S_i(\mu_2)\right]\right) \tag{4.17}$$

where $\mu_0 = kd$, $\mu_1 = k\left(\sqrt{d^2+l^2}+l\right)$, $\mu_2 = k\left(\sqrt{d^2+l^2}-l\right)$, $d = |m-n|\frac{\lambda}{2}$, $l = \frac{\lambda}{2}$ and $C_i\left(x\right) = \int_\infty^x \frac{cos(\tau)}{\tau}d\tau, S_i\left(x\right) = \int_0^x \frac{sin(\tau)}{\tau}d\tau$. The coupling matrix is denoted by $\mathbf{c}$ [34]

$$\mathbf{c} = \left(\mathbf{Z}_m + \mathbf{Z}_s\right)^{-1} \cdot \left(\mathbf{Z}_\infty + \mathbf{Z}_s\right) \tag{4.18}$$

where $\mathbf{Z}_s$ denotes the source matrix, it is an N by N diagonal matrix with entries Z_{sn}, Z_{sn} is set as 50. With mutual coupling being considered, the algorithm will be changed by replacing $\mathbf{g}$ with $\mathbf{gc'}$ and the rest part of the algorithm is the same as the case without mutual coupling, equation 4.15 will be rewritten as

$$\widehat{\mathbf{w}}^{(k+1)} = \widehat{\mathbf{w}}^{(k)} - \frac{\mu\left\langle(\widehat{E}_m^{(k)} - E_m)\mathbf{E}_m^*\right\rangle}{\left(\mathbf{E}_m^*\right)^H\mathbf{E}_m^*} \tag{4.19}$$

where $E_m = \mathbf{gc'}\widetilde{\mathbf{w}}$, $\widehat{E}_m^{(k)} = \mathbf{gc'}\widetilde{\widehat{\mathbf{w}}^{(k)}}$, $\mathbf{E}_m = \mathbf{c'E_0}$, for the optimal step size in the algorithm, in using the LMS algorithm, the largest eigenvalue of the coefficient matrix will be recalculated every time because the input is not constant. In the NLMS, μ_{opt} is kept being as 1, but the efficiency of the algorithm will be affected. One special case is when the algorithm is used in the case with mutual coupling and noise, if $\left\|\mathbf{c}^T\mathbf{g}\right\|^2$ is larger than $\|\mathbf{g}\|^2$, according to the expression of SNR in the preceding subsection, it will increase SNR which will increase the final error level.

4.4 Dithering with Discrete Magnitudes and Phases

In reality, the dither signal is generated by certain digital attenuator and phase shifter, thus the real dither signal is to be discrete instead of continuous with infinite states as used in our algorithm. Considering this, studying the effect of using the discrete dither signal to our algorithm is of practical value. Assuming the digital attenuator and phase shifter are both n-bits, it means that for the magnitude part and the phase part, they both have $S = 2^n$ states and each state is selected with equal chance. The magnitude part of the continuous dither signal is denoted by e^{α_n}, where $\alpha_n = 0.05 \cdot ln(10) \cdot \sigma \cdot v_n$. For the discrete dither signal, $e^{\alpha'_n}$ is denoted as the magnitude part, the prime on α_n emphasizes that it is different from the continuous one, $\alpha'_n = 0.05 \cdot ln(10) \cdot \sigma \cdot v'_n$, v'_n has only S states. An arithmetic progression with S terms for v'_n and the probability of each state is $\frac{1}{S}$. Considering that the mean value of the normal distribution v_n is 0, the range of the discrete sequence is set from $-W$ to W to produce the same mean value. Thus the difference of the two successive terms is $\frac{2W}{S-1}$. Then the value of W will be determined such that the variance of the magnitude part is close to 1 as in the continuous case. For variables with S elements subject to the discrete uniform contribution from $-W$ to W, the variance is calculated by

$$\sigma^2 = \sum_{n=1}^{S} \frac{1}{S} \cdot (-W + \frac{2}{S-1} \cdot W \cdot n)^2 \tag{4.20}$$

by equation 4.19, the value of W can be determined. Assuming the digital attenuator and phase shifter are both 4-bit, $S = 16$, then W will be set as 1.62 by (4.20), the variance is 0.9914 now. In this way, the arithmetic progression is set as: $[1.62, 1.404, 1.188, 0.972, 0.756, 0.54, 0.324, 0.108, -0.108, -0.324, -0.54, -0.756,$ $-\ 0.972, -1.188, -1.404, -1.62]$. The phase part will be adjusted by the similar method as that used for the magnitude part. In the continuous dither signal, $e^{j\beta_n}$ is denoted as the phase part, where $\beta_n = \mu_n\Delta$. In the discrete dither signal, $e^{j\beta'_n}$ is denoted as the phase part, $\beta'_n = \mu'_n\Delta$. What is altered is setting 16 fixed terms with constant difference between two successive terms for μ'_n ranging between -1 and 1. Since μ_n is uniformly distributed from -1 to 1 and the variance is $\frac{1}{3}$ for the continuous case, μ'_n will be picked up from $[-\frac{15}{16}, -\frac{13}{16}, -\frac{11}{16}, -\frac{9}{16}, -\frac{7}{16}, -\frac{5}{16}, -\frac{3}{16}, -\frac{1}{16}, \frac{1}{16}, \frac{3}{16}, \frac{5}{16}, \frac{7}{16}, \frac{9}{16}, \frac{11}{16}, \frac{13}{16},$ $\frac{15}{16}]$ with equal probability. The variance is 0.332 for the discrete case by (4.20). Using the discrete signal, the whole dither signal is written as $e^{\alpha'_n}e^{j\beta'_n}$, compared with the continuous dither signal, since the values for the discrete dither signal are just picked up from those certain fixed values and the variance is a little different, the efficiency of the algorithm will be affected somewhat.

CHAPTER 5

NUMERICAL RESULTS

5.1 Noiseless Environment

The purpose of this section is to demonstrate the capability of our proposed algorithm in different cases and present the numerical results to show the effect of noise, mutual coupling and discrete dither signal to our algorithm. The coefficients without the added dither signal (which are called Original) and the coefficients with dither signal are shown in Fig.4.3, the corresponding electric fields in near-field are shown in Fig.4.4, by dithering the coefficients, both the real part and the imaginary part are increased a little. Then, Fig.5.1 and Fig.5.2 show the corrected coefficients with 2000 iterations in the noise-free environment and the corresponding electric filed pattern in far-field region.

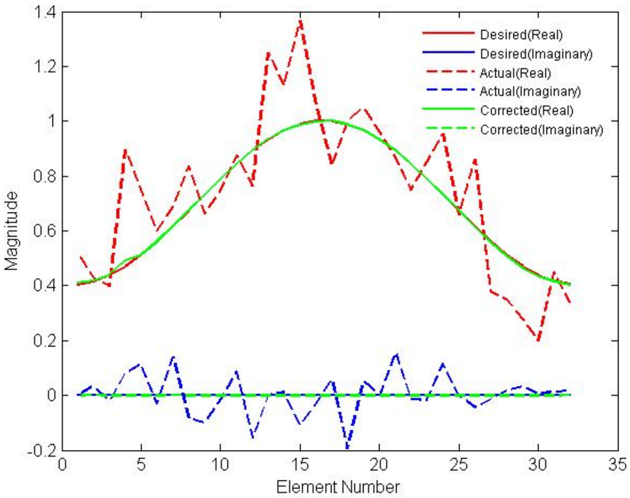

Fig. 5.1. Desired, corrected and actual coefficients

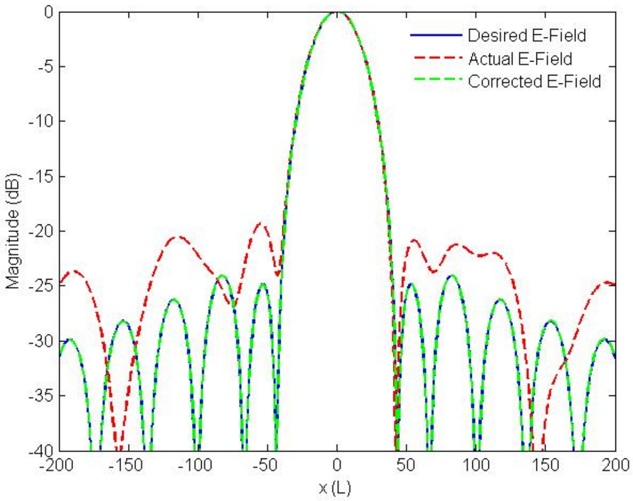

Fig. 5.2. Desired, corrected and actual electric fields in far-field region

From Fig.5.1, it is clear that after 2000 iterations, the coefficients have been corrected very well, not only the real part, but also the imaginary part. In this way, it is natural that the far-field electric field pattern is also corrected well in Fig.5.2. The relative error as a function of iteration number is shown in Fig.5.3, the relative error being defined as $\frac{\sum_{n=1}^{N}|(\widehat{w}_n - w_n)|}{\sum_{n=1}^{N}|w_n|}$, $N = 32$.

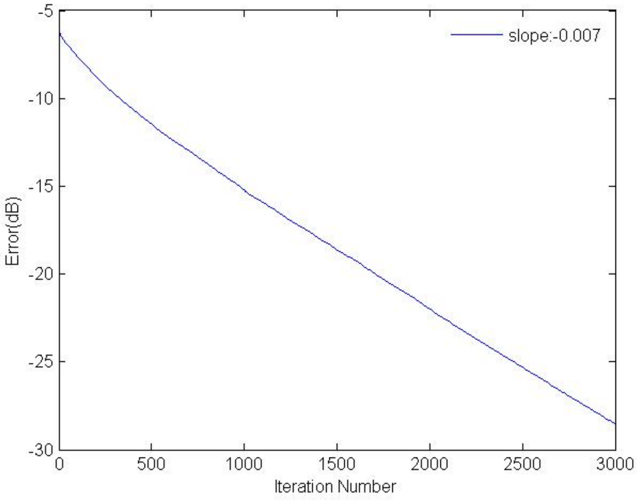

Fig. 5.3. The error vs. the iteration number

From Fig.5.3, it can be seen that with iteration number increasing, the error decreases quickly, the slope of the line is around -0.007.

5.2 Noisy Environment

Our algorithm will next be demonstrated in the noisy environment, in order to find the influence of the SNR to the algorithm, four groups of noise are selected, the values of their SNR being 30dB, 25dB, 20dB and 10dB respectively. When the value of their SNR is changed, the realization number M should be adjusted at the same time, because a larger M is necessary to get the stable corrected coefficients and the relative error with the noise increasing. By doing the comparisons with different values of M, M is set as 100 for 30dB, 200 for 25dB, 500 for 20dB and 1000 for 10dB. The far-field electric field patterns (the iteration numbers are all set as 2000) are shown in Fig.5.4-5.7, the error analysis ($\frac{\sum_{n=1}^{N}|(\widehat{w}_n - w_n)|}{\sum_{n=1}^{N}|w_n|}$) of the cases with various SNR and free-noise case (SNR=∞) are shown in Fig.5.8.

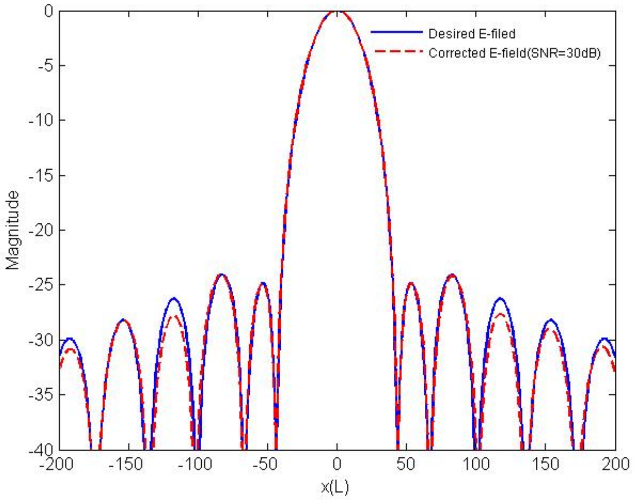

Fig. 5.4. The electric field patterns in far-field region, SNR=30dB

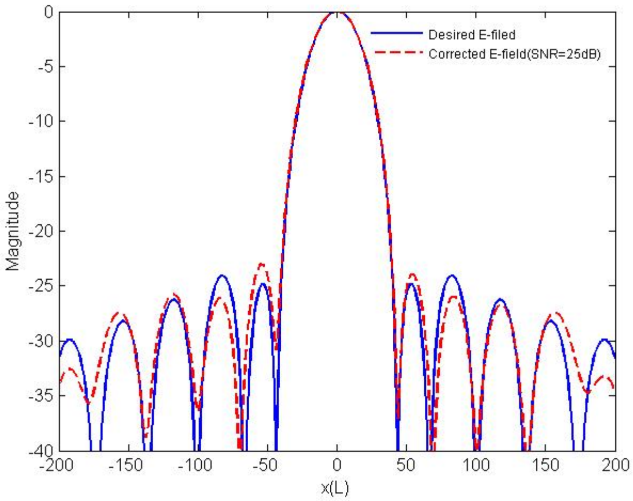

Fig. 5.5. The electric field patterns in far-field region, SNR=25dB

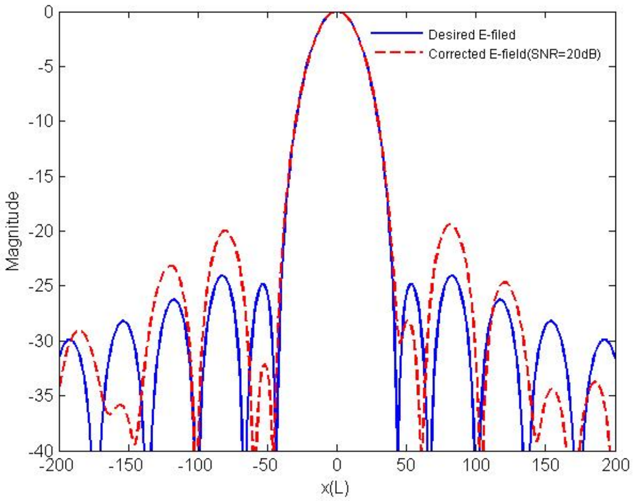

Fig. 5.6. The electric field patterns in far-field region, SNR=20dB

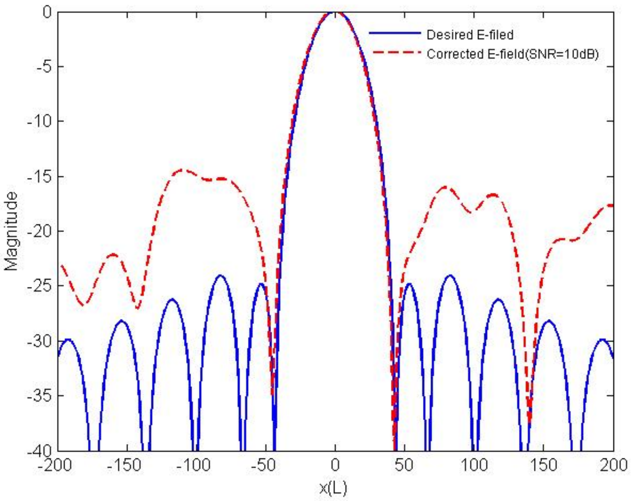

Fig. 5.7. The electric field patterns in far-field region, SNR=10dB

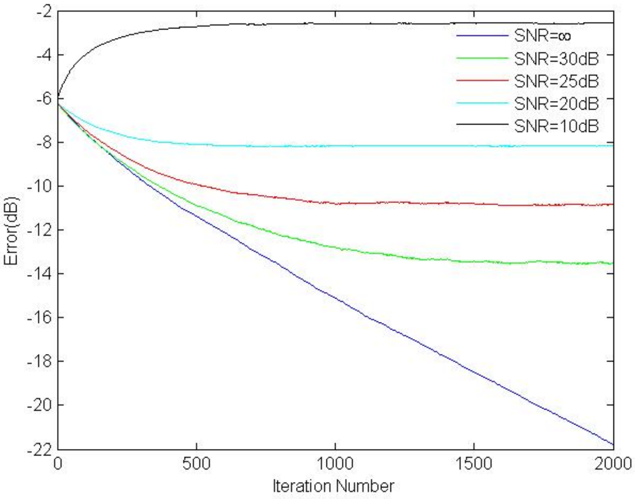

Fig. 5.8. The error analysis vs. the iteration with various SNR

From Fig.5.4-5.7, it can be seen that with the SNR decreasing, the difference between the corrected electric field pattern and the desired electric field pattern becomes more and more obvious. When it is 30dB, the electrical pattern is still being corrected well, but it is not as good as that under the noise-free environment, when the SNR reaches 10dB, the corrected electric field is even worse than the actual electric field in Fig.5.2, it means in that case the algorithm does not work at all. By Fig.5.4-5.7, it can be concluded that the noise plays an important role in our algorithm and there exists the limit can make the algorithm stop working. It is more straightforward to see the effect of noise to our algorithm in Fig.5.8, when there is no noise, the error keeps decreasing with iteration number increasing. However, after the noise is added, the performance of the error changes. If the SNR is above 10dB, the residual error decreases with iteration number increasing at first, then they become constant even the iteration number is still increasing. If the SNR is set as 10dB, the error increases with iteration number increasing at first, then it is kept constant; it explains why the corrected field is even worse than the uncorrected field in Fig.5.7. Another important point is that when SNR is 30dB the final value of the error is around −13.4dB, then it will increase to −11dB and −8dB with SNR decreasing to 25dB and 20dB. This explains why the corrected electric field pattern becomes worse with SNR decreasing in Fig.5.4-5.6.

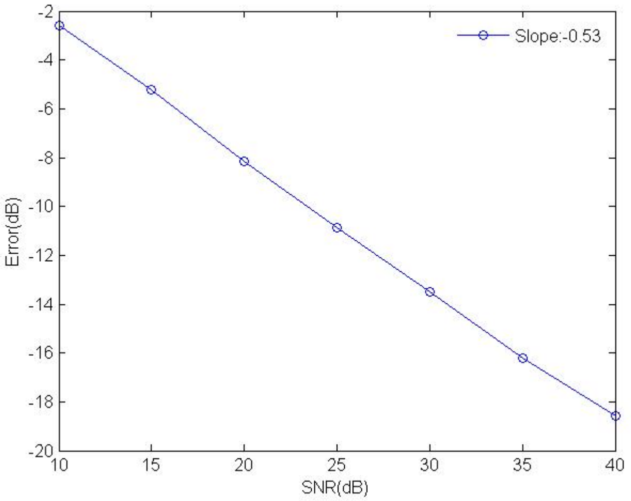

Fig. 5.9. The error analysis vs. SNR

Fig.5.9 plots the final error levels for various SNR, it gives the approximate answer to the question that for a given SNR, what is the accuracy of the algorithm? The results are generated with 3000 iterations when all the errors are already stable and the slope of the line is around -0.53, it means if the SNR decreases by 10dB, the error will increase by 5.3dB.

5.3 The Effect of Mutual Coupling

Thirdly, the effect of mutual coupling to the proposed algorithm is presented. The corrected far-field electric field with considering the effect of mutual coupling in noiseless environment is shown in Fig.5.10, Fig.5.11 shows the field with mutual coupling and noise (SNR=25dB), both of them are gotten with 2000 iterations , the error analysis for the case with mutual coupling is shown in Fig.5.12.

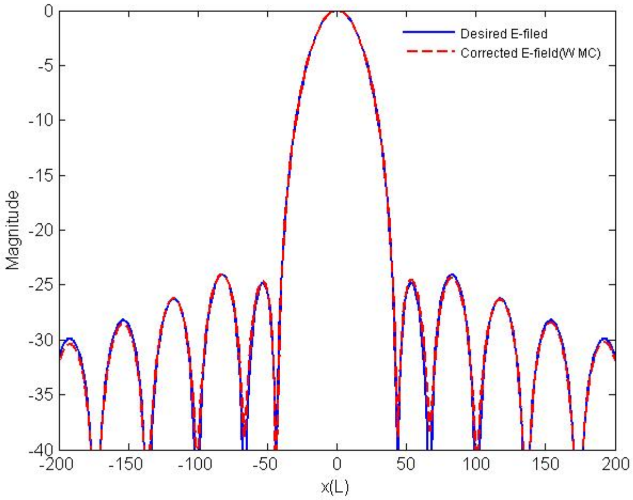

Fig. 5.10. The electric field patterns in far-field region with mutual coupling

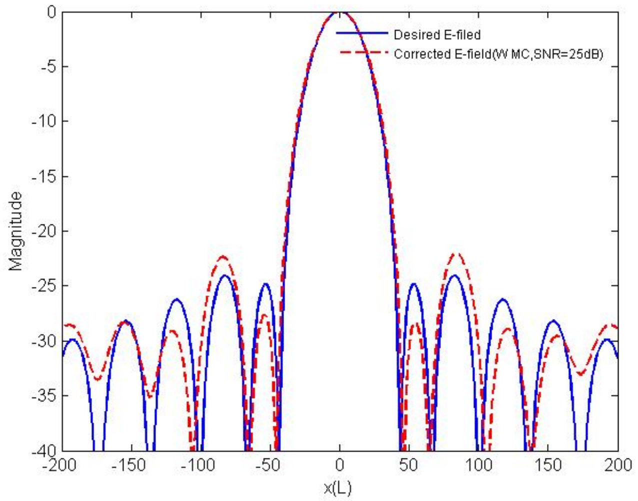

Fig. 5.11. The electric field patterns in far-field region with mutual coupling and noise

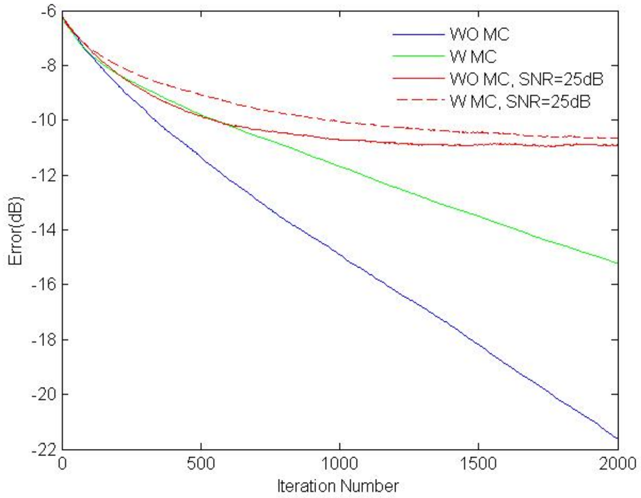

Fig. 5.12. The error analysis vs. the iteration for mutual coupling

By comparing Fig.5.10 and Fig.5.11 with the corrected electric field patterns under the environment without mutual coupling (Fig.5.2 and Fig.5.3), it can be seen that the final results become worse in the case with mutual coupling under the same condition. Fig.5.12 describes the change of the relative error with introduction of mutual coupling, it makes the speed of converging of the algorithm slow down and it will make the final error stay in a higher level (for SNR=25dB, the error is increased by around 0.2dB) compared with the case without mutual coupling, this is the same as what we expected in the preceding chapter. Fig.5.13 shows the accuracy of the algorithm with mutual coupling, it can be seen that mutual coupling increases the final error level a little.

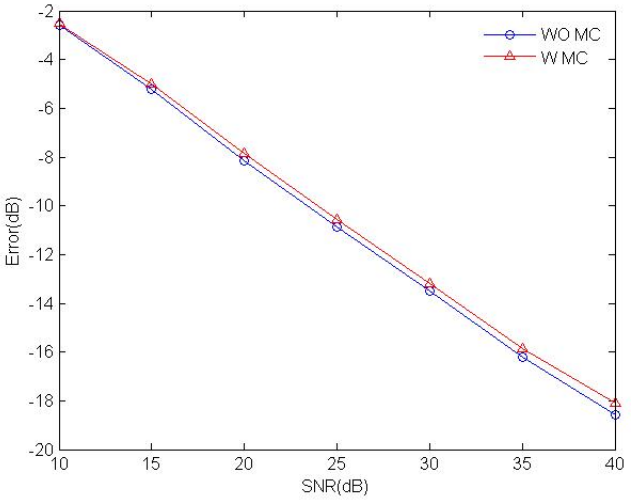

Fig. 5.13. The error analysis vs. SNR with mutual coupling

5.4 The Effect of Discrete Dither Signal

Finally, the corrected electric filed using the discrete dither signal and the error analysis are shown in Fig.5.14-Fig.5.17. In Fig.5.14, the corrected electric filed using the discrete dither signal in noiseless environment is shown, we can see it is corrected well after 2000 iterations. Fig.5.15 and Fig.5.16 show the effect of using the discrete dither signal to the algorithm with noise and mutual coupling, compared with cases using the continuous dither signal, the difference is not that obvious. The error analysis is presented in Fig.5.17 to support that.

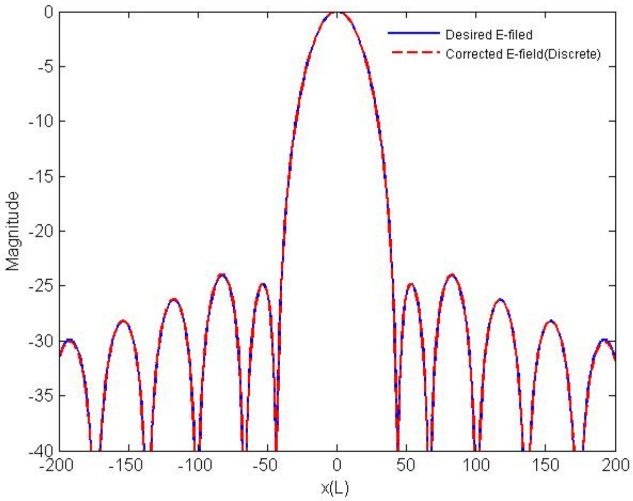

Fig. 5.14. The electric field patterns in far-field region with discrete dither signal

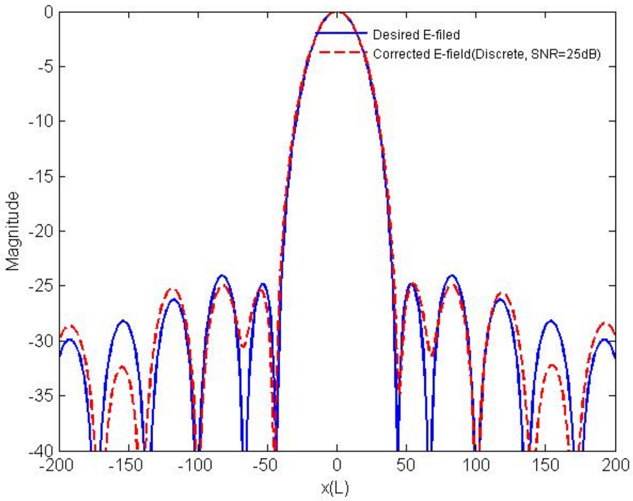

Fig. 5.15. The electric field patterns in far-field region with discrete dither signal and noise

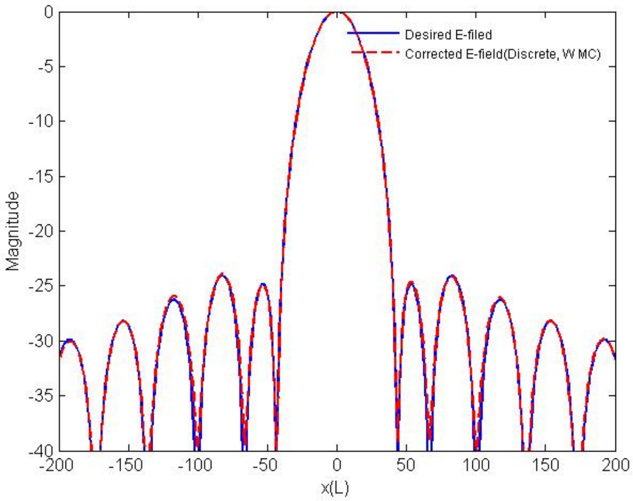

Fig. 5.16. The electric field patterns in far-field region with discrete dither signal and mutual coupling

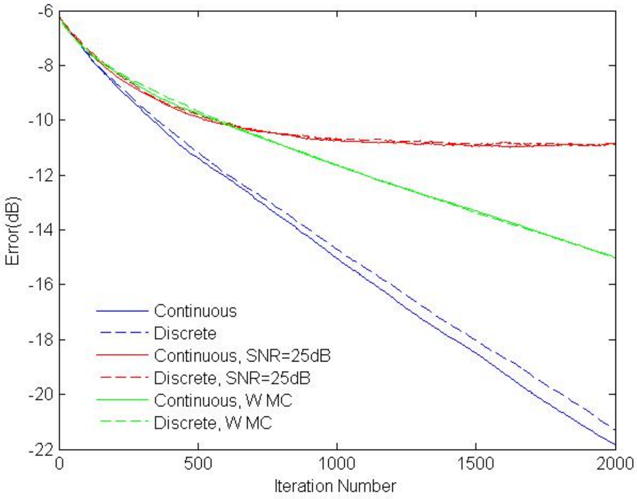

Fig. 5.17. Error analysis with discrete dither signal

From Fig.5.14-5.16, we can get the conclusion that using the discrete dither signal does not change the corrected fields so much compared with the results we got using the continuous dither signal. In Fig.5.17, it is clear to see that for the cases with noise and mutual coupling, the corresponding dash line which represents results using the discrete dither signal in each case is almost the same as the solid line, for noiseless case, the difference is larger with iteration number increasing, but they both keep decreasing. The effect of using the discrete dither signal is not so obvious because the mean value we set for the discrete dither signal remains the same as it for the continuous dither signal and the variance is very close to it in the continuous dither signal, it ensures the statistical characteristic of the two dither signal being similar. Even though sometimes using the discrete dither signal will slow down the speed of converging for our algorithm, the difference is not so obvious and it can be ignored.

CHAPTER 6

CONCLUSION

The Linear programming (LP) and the Particle Swarm Optimization (PSO) methods have been used to design the arrays in the first part of this thesis, respectively. The LP method is demonstrated by broadside symmetric arrays with element number from 2 to 10. The advantage of this method that it is easy to implement and fast have been presented in this thesis, however its disadvantage that it cannot be used to solve complicated nonlinear problem is also shown. Then the PSO method is introduced, the PSO method is used to design the asymmetric arrays and the non-broadside arrays, the two cases demonstrate the robustness of the PSO method in designing this kind of array, the disadvantage that the PSO method is time-consuming is also shown in this part.

In the second part, an adaptive method for correcting the current excitation of the phased array via using dither signal and the NLMS algorithm has been proposed, and it is demonstrated by a 32-element broadside array designed by the Taylor method with side lobe level of -24dB. The advantage of the NLMS algorithm in determining the optimal step size is demonstrated and the effect of noise and mutual coupling on our algorithm have been shown. After the noise is added, it will make the corrected coefficients stop converging to the desired coefficients when the iteration number reach some certain numbers. In other words, the noise will decrease the accuracy of our algorithm and there exists the limit for the noise. When it exceeds the limit, our algorithm will not work, so when we use the algorithm we should pay attention to it. For mutual coupling, it will not alter the essence of the algorithm, but it will affect the speed of converging and the accuracy of the algorithm. When the noise and mutual coupling are considered together, they will affect the efficiency at the same time. In the end, the discrete dither signal is considered because the continuous dither signal cannot be generated by the attenuator and phase shifter in practice. Compared with using the continuous dither signal, the discrete dither signal will slightly decrease the speed of our algorithm, but if the values are set properly, the effect can be ignored.

APPENDIX

PROOF OF NLMS ALGORITHM

The magnitude of the difference between the actual coefficients and the desired coefficients is defined as: $\Gamma(n) = |\mathbf{d} - \widehat{\mathbf{d}}^{(k)}|$, ρ is defined as $\rho(n) = \Gamma(n)^2 = |\mathbf{d} - \widehat{\mathbf{d}}^{(k)}|^2$.

$$E[\rho(n+1)] = E[|\widehat{\mathbf{d}}^{(n)} + \mu\frac{(e^{(n)})^*\mathbf{h}^{(n)}}{(\mathbf{h}^{(n)})^H\mathbf{h}^{(n)}} - \mathbf{d}|^2] \tag{A.1}$$

$$E[\rho(n+1)] = E[|\widehat{\mathbf{d}}^{(n)} + \mu\frac{((y^{(n)})^* - (\widehat{y}^{(n)})^*)\mathbf{h}^{(n)}}{(\mathbf{h}^{(n)})^H\mathbf{h}^{(n)}} - \mathbf{d}|^2]. \tag{A.2}$$

Let $\gamma(n) = \mathbf{d}^{(n)} - \mathbf{d}$.

$$E[\rho(n+1)] = E[|\gamma(n) + \mu\frac{((y^{(n)})^* - (\widehat{y}^{(n)})^*)^*\mathbf{h}^{(n)}}{(\mathbf{h}^{(n)})^H\mathbf{h}^{(n)}}|^2] \tag{A.3}$$

$$E[\rho(n+1)] = E[(\gamma(n) + \mu\frac{((y^{(n)})^* - (\widehat{y}^{(n)})^*)^*\mathbf{h}^{(n)}}{(\mathbf{h}^{(n)})^H\mathbf{h}^{(n)}})^H(\gamma(n) + \mu\frac{((y^{(n)})^* - (\widehat{y}^{(n)})^*)^*\mathbf{h}^{(n)}}{(\mathbf{h}^{(n)})^H\mathbf{h}^{(n)}})]. \tag{A.4}$$

Assuming the independence, we can get

$$E[\rho(n+1)] = \rho(n) + E[(\mu\frac{(e^{(n)})^*\mathbf{h}^{(n)}}{(\mathbf{h}^{(n)})^H\mathbf{h}^{(n)}})^H(\mu\frac{(e^{(n)})^*\mathbf{h}^{(n)}}{(\mathbf{h}^{(n)})^H\mathbf{h}^{(n)}})] - 2E[\mu\frac{|(e^{(n)})^*|^2}{(\mathbf{h}^{(n)})^H\mathbf{h}^{(n)}}] \tag{A.5}$$

$$E[\rho(n+1)] = \rho(n) + \mu^2\frac{E[|(e^{(n)})^*|^2]}{(\mathbf{h}^{(n)})^H\mathbf{h}^{(n)}} - 2\mu E[\frac{|(e^{(n)})^*|^2}{(\mathbf{h}^{(n)})^H\mathbf{h}^{(n)}}]. \tag{A.6}$$

The optimal step is gotten at $\frac{dE[\rho(n+1)]}{d\mu} = 0$, then we can get

$$2\mu E[|(e^{(n)})^*|^2] - 2E[|(e^{(n)})^*|^2] = 0 \tag{A.7}$$

$$\mu = 1. \tag{A.8}$$

BIBLIOGRAPHY

[1] M. Yeary, G. Crain, A. Zahrai, R. Kelley, J. Meier, Y. Zhang, I. Ivic, C. Curtis, R. Palmer, T.-Y. Yu, and R. Doviak. An update on the multi-channel phased array weather radar at the national weather radar testbed. pages 971–973. Radar Conference, May 2011.

[2] W.D Wirth. Signal processing for target detection in experimental phased-array radar elra. *Communications, Radar and Signal Processing, IEE Proceedings F*, 128(5):311–316, October 1981.

[3] T.T.taylor. Design of line-source antenna for narrow beamwidth and low sidelobes. *IEEE Trans. on Antenna Propagat.*, 3(1):16–28, Jaunary 1955.

[4] C.A. Balanis. *Antenna Theory: Analysis and Design, 2nd ed.* Wiley, New York, 1997.

[5] R. Janaswamy, Dev V. Gupta, and Daniel H. Schaubert. Adaptive correction to array coefficients through dithering and near-field sensing. *IEEE Trans. on Antennas and Propagat.*, 58(11):3558–3567, November 2010.

[6] Mou ping Jin and Cheng He. Study of side lobe levels for wideband phased arrays. pages 1–4, Shenzhen, China, May.5-8 2012. Microwave and Millimeter Wave Technology Conf.

[7] Longyang Huang, En Cheng, and Mengxing Li. Rectangular array pattern synthesis with simple method. pages 183–186, Beijing, China, Oct.26-29 2008. Signal Processing Conf.

[8] Lizhong Song, Ma Ning, Li Chongshen, and Wu Qun. Simulation and analysis of a microstrip circular array antenna at 15ghz. pages 1–4, Dalian, China, Oct.12-14 2008. Wireless Communications, Networking and Mobile Computing.

[9] *IEEE Standard Definitions of Terms for Antennas.* IEEE, 1993.

[10] L.V. Kantorovich. A new method of solving some classes of extremal problems. *Doklady Akad Sci USSR*, 28:211–214, 1940.

[11] George B. Dantzig and Mukund N. Thapa. *Linear Programming 1: Introduction.* Springer, New York, 1997.

[12] Peter Joseph Bevelacqua. *Performance Limits and GEeometry Optimization.* PhD thesis, Arizona State University, 2008.

[13] Robert J Vanderbei. *Linear programming : foundations and extensions.* Springer, New York, 2008.

[14] J. Kennedy and R. Eberhart. Particle swarm optimization. volume 4, pages 1942–1948, Perth, USA, Nov.27-Dec.01 1995. IEEE Neural Networks Conf.

[15] Yuhui Shi and R. Eberhart. A modified particle swarm optimizer. pages 69–73, Anchorage, USA, May.4-9 1998. IEEE Evolutionary Computation Conf.

[16] Boquan Zhang, Yimin Yang, and Lu Gan. Dynamic control of wind/photovoltaic hybrid power systems based on an advanced particle swarm optimization. pages 1 – 6, Chengdu, China, Apr.21-24 2008. IEEE Industrial Technology Conf.

[17] Nanbo Jin and Y. Rahmat-Samii. Advances in particle swarm optimization for antenna designs: Real-number, binary, single-objective and multiobjective implementations. *IEEE Antennas and Propagation*, 55:556–567, March 2007.

[18] M. Clerc and J. Kennedy. The particle swarm - explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. on Evolutionary Computation*, 6:58–73, February 2002.

[19] S. Doctor and G.K. Venayagamoorthy. Improving the performance of particle swarm optimization using adaptive critics designs. pages 363–369. Swarm Intelligence Symposium, Jun.8-10 2005.

[20] Na Li and Song Zhu. Modified particle swarm optimization and its application in multimodal function optimization. pages 375–378, Changchun, China, Dec.16-18 2011. Transportation, Mechanical, and Electrical Engineering Conf.

[21] Mengqi Hu, T. Wu, and J.D. Weir. An adaptive particle swarm optimization with multiple adaptive methods. *IEEE Trans. on Evolutionary Computation*, 17:705–720, 2013.

[22] Bocheng Zhu, Weijun Cheng, Lei Li, and Lezhu Zhou. Low sidelobe wide nulling for linear antenna array with an improved genetic algorithm assisted beamforming. pages 953–957. Communications, Circuits and Systems Proceedings Conf., Jun.25-28 2006.

[23] Penghan Xie, K. Chen, and Zishu He. Synthesis of thinned conical arrays using simulated annealing algorithm. pages 756–758. Communications, Circuits and Systems Conf., Jul.23-25 2009.

[24] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):670–680, May 1983.

[25] Tao Su and Hao Ling. Array beamforming in the presence of a mounting tower using genetic algorithms. *IEEE Antennas and Propagation*, 53:2011–2019, June 2005.

[26] Peng Chen, Yayu Zhang, and Wei Zhu. Optimized simulated annealing algorithm for thinning and weighting large planar arrays in both far-field and near-field. *IEEE Oceanic Engineering*, 36:658–664, October 2011.

[27] R. Levanda and A. Leshem. Adaptive selective sidelobe canceller beamformer with applications in radio astronomy. pages 839–843, Eliat, Israel, Nov.17-20 2010. Electrical and Electronics Engineers in Israel (IEEEI).

[28] T.W. Tai and J.J.S. Marciano. A study on the application of adaptive minimum bit error rate algorithms for wcdma. pages 1–4, Taipei, Oct.30-Nov.2 2007. IEEE TENCON.

[29] B. Widrow. Thinking about thinking: the discovery of the lms algorithm. *Signal Processing Magazine*, 22:100–106, Jaunary 2005.

[30] Simon S. Haykin. *Adaptive Filter Theory.* Prentice Hall, New Jersey, 2002.

[31] NEIL J. Bershad. Analysis of the normalized lms algorithm with gaussian inputs. *IEEE Trans. on Acoustics, Speech and Signal Processing*, 34(4):793–806, August 1986.

[32] L. Ranathunga, N.A. Abdullah, and R. Zainuddin. Analysis of video content in multi codec formats with compacted dither coding. pages 48–54. Information and Automation for Sustainability Conf., Dec.12-14 2008.

[33] H. Shin, A.H. Sayed, and W. Song. Variable step-size nlms and affine projection algorithms. *IEEE Signal processing Lett.*, 11(20):132–135, February 2004.

[34] R. Janaswamy. Effect of element mutual coupling on the capacity of fixed length linear arrays. *IEEE Antennas Wireless Propagat. Lett.*, 1:157–160, 2002.