Chapter - 3

OPTIMIZATION METHODS FOR LINEAR ARRAY SYNTHESIS

In recent years, various optimization methods for array pattern synthesis has come into light, explaining the possibility to achieve the desired patterns more or less with negligible deviations. From Chapter 2, it is evident that the array pattern depends on the coefficients of the elements and they play a major role in pattern synthesis, beam steering and in maintaining an allowable side lobe levels. A few optimization methods based on choosing the array coefficients are discussed in this chapter with the main focus on Taylor distribution. Statistics based weighting methods such as Minimum Mean Square Error (MMSE), Least Mean Square Algorithm (LMS) which employs a gradient based error minimization technique, Kalman filtering, H-infinity filtering are discussed in detail. A comparative study is made by applying Kalman filtering, H-infinity filtering to analyse, which technique suits best resulting in least error.

3.1. OPTIMIZATION USING WEIGHTING METHODS

From the Section 2.3, the variations in the pattern of the linear array has resulted from the variations in the magnitude and the phase of the coefficients. Hence, optimisation of the array coefficients in order to have the desired pattern from the array is discussed here with the help of few weighting methods in brief.

The word “Optimization” refers to a process of finding the minimum (or maximum) value at which the desired array pattern is resulted, but under the consideration of some constraints. The constraints set at this level are to minimise the error between the actual array pattern and the true array pattern, which in particular involves the proper maintenance of the side lobe levels in the presence of random errors. Chebyshev method and Taylor method are discussed in the context of linear array optimization.

3.1.1. Chebyshev Method

The unequal sidelobe levels resulting in linear arrays with uniform weights are often undesirable due to high side lobe levels. The optimal low sidelobe level at the expense of raising the lower side lobes, for a given beam width is discussed [87]. The array
pattern will have the side lobes, which are all equal in magnitude. Chebyshev method treated as a classical method for synthesizing array antennas, generates the narrowest major lobe with a definite sidelobe level.

Briefly, the method obtains weights for uniformly spaced broadside linear array and helps in obtaining the minimum possible null-null beam width. The main idea of this method is to match the unknown array coefficients with the known Chebyshev polynomial coefficients. This is achieved assuming a symmetric linear array antenna having $N$ elements, where every antenna element is uniformly separated by a distance $d = \lambda/2$. Further, assuming that the array lies along the $x$-axis, centered at $x = 0$, the array factor will be in the form given by:

$$AF = \sum_{n=1}^{M} W_n e^{-jk(2n-1)d/2 \cos \theta} + \sum_{n=-M}^{-1} W_n e^{-jk(2n-1)d/2 \cos \theta}$$

(3.1)

when $N$ is even

$$AF = \sum_{n=-M}^{M} W_n e^{-jknd \cos \theta}$$

(3.2)

when $N$ is odd

Since the cosine function can be expressed in the complex-exponential form given by: $\cos(x) = \frac{e^{jx} + e^{-jx}}{2}$

The array factors given in Equation (3.1) and Equation (3.2) can be rewritten as:

$$AF = \sum_{n=1}^{M} W_n \cos[(2n - 1)u]$$

(3.3)

$$AF = \sum_{n=0}^{M} W_n \cos(2nu)$$

(3.4)

These equations help in computing the array factor related with the parameter $u$. 

If we substitute the above expressions into the Antenna Array Factor given in Equation (1.1) and Equation (1.3) and introducing a substitution:

\[ \cos u(t_0) = t \]  

(3.5)

\( AF \) becomes a polynomial. Matching this Array Factor, to the corresponding Tschebyschev polynomial and determine the corresponding weights \( W_n \), optimal sidelobes are obtained using \( t_0 \).

The far field of the array using Tschebyschev coefficients or weights \( W_n \), with \( N = 32 \) and a sidelobe level \(-25\, dB\) which is computed similarly using the Equation (2.1) at a far field distance of 10 \( R_f \) is as shown in Figure 3.1. The sidelobe levels remain constant at \(-25.01\, dB\) in this figure. A line indicating the equal sidelobe levels in this method is shown at \(-25.01\, dB\) for a clear picture of what has discussed under this method.

![Figure 3.1. Far field pattern observed for Chebyshev method with \( N = 32 \) and sidelobe level \(-25\, dB\)](image)

Though the sidelobe levels are limited to the desired level, sometimes the “incentive coefficient has rebound phenomenon at the ends of array elements” [65]. These coefficients of the last elements in the array are larger than that of the closer one, going against the principles of array antennas. This method proposed by Chebyshev is not that
effective, when the interelement spacing of array elements is greater than or equal to half of $\lambda$ i.e., $(d \geq \lambda/2)$. Hence a thought of optimizing these disadvantages of Chebyshev is considered and a modified approach is given by Taylor [14, 88 and 89].

### 3.1.2. Taylor Method (Tschebysheff error)

The array pattern has minor lobes which decay monotonically in a broadside array is proposed by Taylor [88]. The array factor is referred to as “Space Factor” by Taylor. The current distribution along the elements of the linear array antenna is thereby optimised. The space factor proposed by Taylor [14] is given in Equation (3.6)

$$SF(u, A, \bar{n}) = \prod_{n=1}^{\bar{n}-1} \left[ \frac{1 - \left(\frac{u}{u_n}\right)^2}{1 - \left(\frac{n}{\bar{n}}\right)^2} \right] \frac{\sin(u)}{u} \quad (3.6)$$

where $u = \pi \frac{i}{\lambda} \cos \theta$, $u_n = \pi \frac{i}{\lambda} \cos \theta_n$ and $\bar{n}$ is an integer

When $u = u_n$, the $SF(u, A, \bar{n})$ function has nulls at $\theta = \theta_n$, and when $u = n\pi$ the function is supposed to have poles. When the perturbations exist due to random errors, the location of nulls and the poles tend to deviate from these expressions. The parameter $\bar{n}$ is selected in order to fix the number of minor lobes to be limited with desired amplitude of $1/R_0$ and are referred to as inner lobes, where $R_0$ is the desired side lobe level ratio (SLL ratio). Other than these minor lobes are referred to as outer lobes which also decay progressively at a rate of $1/u$. The parameter $\sigma$ called as scaling factor is introduced by Taylor, to have smooth transition between all types of minor lobes may it be inner or outer lobe and is given by

$$\sigma = \frac{\bar{n}}{\sqrt{A^2 + \left(\frac{n}{\bar{n}} - \frac{1}{2}\right)^2}} \quad (3.7)$$

The location of the nulls are obtained by relating them with $\sigma$ as given below:

$$u_n = \pi \frac{i}{\lambda} \cos \theta_n = \begin{cases} \pm \pi \sigma \sqrt{A^2 + \left(\frac{n}{\bar{n}} - \frac{1}{2}\right)^2} & 1 \leq n < \bar{n} \\ \pm n\pi & \bar{n} \leq n \leq \infty \end{cases} \quad (3.8)$$
where the constant $A$ and $R_0$ are related using Equation (3.9)

$$\cosh(\pi A) = R_0$$  \hspace{1cm} (3.9)

The normalized line source current distribution corresponding to the space factor of (3.6) is given by

$$I(z') = \frac{1}{l} \left[ 1 + 2 \sum_{p=1}^{\tilde{n}-1} SF(p, A, \tilde{n}) \cos \left( 2\pi p \frac{z'}{l} \right) \right]$$ \hspace{1cm} (3.10)

The coefficients $SF(p, A, \tilde{n})$ represent samples of the Taylor pattern, and they can be obtained from (3.10) with $u = \pi p$. In order to extract the excitation coefficients for the linear array, the line source distribution is sampled and are assumed as $a_n$ throughout the analysis. Figure 3.2 shows the excitation coefficients calculated by this method and Figure 3.3 shows the far field array pattern.

![Normalised Current Distribution](image)

**Figure: 3.2.** Normalised Current Distribution for Taylor method with $N = 32$, sidelobe level $-25 \, dB$ and $(\tilde{n} = 5)$.

The far field of the array using Taylor coefficients $a_n$ with $N = 32$ and a sidelobe level $-25 \, dB$ which is computed using the Equation (2.1) at a far field distance of
10 \( R_f \) is as shown in Figure 3.3. The sidelobe levels are below \(-25.48\) \( dB \) but with varying levels always below \(-25\) \( dB \).

In Taylor method, which can be considered as upgraded or Chebyshev method, the sidelobe levels decrease monotonically and hence increases the directivity. This method is considered to be efficient only with linear array synthesis but tend to drop its efficiency when the planar arrays are considered [65].

### 3.2. STATISTICS BASED WEIGHTING METHODS

The weighting methods where the theory of statistics is applied in order to extract the information or signal out of a randomly characterised environment or when the information is corrupted with some random noise, interference or clutter is referred to as “Statistics based Weighting methods”. The extraction of the signal of interest, when corrupted by noise, is calculated by estimation or by expectation of the signal where it is always the approximated form of the signal with an error.

Performance comparison on different types of arrays is made using LMS and RMS algorithms applied to a smart antenna which is proposed to receive a QPSK modulated signal corrupted with AWGN, and interference. Suppression of interference with the
help of an automatic convergence rate parameter is performed using the algorithms along with some useful analysis and explaining some limitations of Particle Swarm Optimization (PSO) algorithm. By LMS method, “rate of convergence factor” and by RLS method, “forgetting factor is considered” [69].

A technique to minimize the sidelobes in the co-polarization pattern that adjusts the orientation of each dipole in an array while compelling the gain is presented [90]. Alternatively, altering the element weighting or interelement spacing to obtain low side lobes is achieved using rotation of the elements. A genetic algorithm (GA) for rotation of the dipoles to acquire anticipated sidelobe levels is represented [90] and also similar hybrid approach is discussed for Sub-arrays using Monopoles [91]. The detailed steps in this procedure are given in Appendix-2 along with flow chart in the Figure A2.1.

Direct search methods which do not use random decision making are also proposed for array antenna optimization [92]. These random decision making methods involving GA and PSO are compared with the gradient and non-gradient based methods. Thereby, all these basics are put together progressing towards a hybrid algorithm. After the comparison and analysis, it is concluded that the Gradient and Non-Gradient methods without randomness are significantly efficient [92].

Highlighting the advantages of the array antennas that they can precisely control array incentive to produce low sidelobe pattern or close to the desired pattern, the process “Synthesis” is explained using GA, which can overcome the Chebyshev method’s inadequacy to achieve the lower sidelobe level array antenna pattern using its “stronger global search capability”. Low sidelobe levels resulted from the GA models of the linear array antennas and the planar array antennas are conferred in [65].

GA is sometimes restricted due to its slow search, high computation times for simulating each accurate and possible solution and often due to the premature convergence problem. Adaptive GAs are developed to avoid these problems by adjusting control parameters of GAs through “Population diversity measurements (PDM)” [93-99]. A new method based on fuzzy genetic algorithms (FGAs) by the optimization of the complex excitation coefficients to meet the specifications of the desired array patterns in a best and dynamical way is discussed [100, 101]. FGAs are much better than the Standard GAs having a better approach towards the specified
desired radiation pattern with high speed. Various optimization techniques especially for array antennas are also presented [102].

Initial goals of sidelobe level suppression are easily achieved using FGAs and further beam width problems associated with a linear antenna array is examined and then fixed by the Real Coded Genetic Algorithm (RGA) for the optimum element locations and excitation. The minimization of the objective function “Misfitness” (MF) is calculated using the evolutionary algorithm which optimizing the current excitation weights and the interelement spacing is reported [66].

The five controls used to shape the array pattern are identified as the geometry of the array, relative displacement between elements, excitation amplitude, excitation phase and the relative pattern of the individual elements. Simulated results reveal that the non-uniform excitations in linear antenna array with optimal interelement distance offers pleasurable sidelobe level reduction using a new technique namely Improved Particle Swarm Optimization (IPSO) [99].

A similar technique, namely Novel Particle Swarm Optimization (NPSO) proves to be fast and a robust technique than PSO, yielding minimum values of SLL for types of array geometry [67]. Giving prominence to null placement control, a derivative-free, algorithm called as Invasive Weed Optimisation (IWO) is compared with other best known metaheuristic algorithms like GA, PSO, etc. This comparative analysis reflect the superiority of IWO requiring minimum mathematical pre-processing [103].

3.2.1. Gradient-Based Optimization - Least Mean Square (LMS) Algorithm

Estimating the Gradient of the error resulting from the actual output and the desired output and thereby proceeding on optimizing this estimated gradient is referred to as “Gradient-based optimization technique”. These algorithms are used far and wide, because of their proven simplicity, effectiveness and convergent local optimization.

The least mean squares (LMS) algorithm, uses a method called steepest decent is presented by Widrow and Hoff in 1959. It is an adaptive algorithm which adjusts the array coefficients minimising the random errors discussed in the (Section 2.3). The
LMS algorithm actually involves no matrix operations, and is famous due to its requirement of less computational resources and memory.

Also, the implementation of the LMS algorithm is less complicated than any other algorithms which are discussed later. The eigenvalues of the correlation matrix of the input signal sometimes affect the performance of the algorithm in terms of convergence speed in order to adaptively result in the desirable array patterns.

To explain this in a form of simple expression, let us assume $x(n)$ is the signal from the individual antenna which is passed through a filter with weights $w(n)$ where $n = 1, 2, 3, ..., N$ and the output of the filter is $y(n)$. The error between the desired array pattern $d(n)$ and the filter output $y(n)$ is $e(n)$.

The equations related to the above assumptions are given below:

$$ y(n) = w(n)^T x(n), $$

The error is given by $e(n) = d(n) - y(n) = d(n) - w(n)^T x(n) \quad (3.11)$

Now using the least mean square algorithm, the filter weights are given by

$$ w(n + 1) = w(n) + \mu x(n)e(n) \quad (3.12) $$

where $\mu$ is the step size parameter of the LMS algorithm in the Equation (3.12).

The performance of the algorithm depends on the correlation function between the true array pattern and the actual array pattern (generated between the $d(n)$ and $x(n)$) and the maximum Eigen value $\lambda_{max}$ of the correlation matrix is chosen to determine the step size parameter $\mu$, whose value is within the interval $(0, 1/\lambda_{max})$. The convergence speeds of the LMS based algorithms are more rapid when the gradient is calculated from this interval and do not suffer from the chance of diverging.

3.2.2. Kalman Filtering

The Kalman filter estimates the “states of a linear system” to be explained in terms of mathematical analysis. Unlike the simple LMS, it estimates the variables of interest which belongs to a wide range of practices. Instead of the simple error minimisation,
Kalman filter works towards minimisation of the variance involved in estimation error [63, 71].

The usage of Kalman filter is limited only to processes that are involved within a linear system. It works as a recursive linear estimator which optimises all measurements input to the filter, in order to estimate the present value of the variables of interest. This method of minimising the variance of the estimation of the error requires any information available related to the variables like knowledge of the system noise levels, \textit{a priori} measured errors and any initial conditions of the variables [72-77].

Here the estimation of the desired array pattern is based on all near field observations at once and often it is considered to be advantageous to proceed with estimation from a subset of the observations made and then update the estimation process with new observations. This updation related to the estimator is done recursively leading to optimisation with a good number of observation random vectors during the process of estimation. As the signals tend to vary randomly w.r.t time, we must consider the estimation process which involves timely computation of the varying estimates.

3.2.2.1. \textit{Kalman Filter Estimation Process}

The estimation process involves the calculation of the Kalman filtered state estimates and outputs at a given instant of time. This process also computes the gain of the Kalman filter and the covariance matrix of the estimation error. The detailed estimation procedure is given in Appendix-2.

The Kalman filter is supposed to estimate the state $x$ of a discrete-time process that is referred to as state equation and described by the Equation (3.13),

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}$$ \hspace{1cm} (3.13)

with a measurement $z$ which holds the measurement equation

$$z_k = Hx_k + v_k$$ \hspace{1cm} (3.14)

The matrix $A$ in Equation (3.13) relates the state $x$ at the previous step $k - 1$ to that of the current step $k$, assuming the absence of noise. The matrix $B$ describes the optional
control input \( u \) according to the state \( x \). The matrix \( H \) in the measurement Equation (3.14) relates the state \( x \) to the measurement \( z_k \) and all the matrices obey proper size to fit the above operations in the equations. The random variables \( w_{k-1} \) and \( v_k \) represent the process and measurement noise respectively with their covariance matrices \( Q \) and \( R \) respectively.

### 3.2.2.2. Optimization by Kalman Filtering

The filter is a powerful tool which provides estimation and correction based on past, present and even future states of \( x \). It works on the principles based on recursive algorithms to achieve the correction equations and is dynamic when the additive noise is Gaussian.

The dynamic model for completing the prediction process of the Kalman filter can also be described assuming that the state vector and equations transform w.r.t time. Then,

\[
x(t) = F \cdot x(t) + n(t)
\]

where \( F \) is the dynamic matrix and remains constant with \( x(t) \) state vector and \( n(t) \) is the dynamic noise associated with the process which is assumed as Additive Gaussian White Noise (AWGN) and Equation (3.15) has the covariance matrix \( Q(t) \). The observation model represents the relationship between the state and the measurements.

\[
l(t_i) = H \cdot x(t_i) + w(t_i)
\]

where \( l(t_i) \) is the vector at given instant of time \( t_i \), \( H \) is the observation matrix and \( w(t_i) \) is the Gaussian noise involved in the measurement process described by Equation (3.16) with the covariance matrix \( R(t_i) \).

The predicted state, also referred to as apriori state of the Equation (3.15) is described by

\[
x^{-}(t) = F \cdot x^{-}(t)
\]

where \( x^{-}(t) \) is the predicted state.
The Kalman gain of the filter $K$ is chosen so that it minimizes the posteriori error covariance and is given by

$$K = P^-H^T(HP^-H^T + R(t_i))^{-1} \quad (3.18)$$

The corrected state, also referred to as posteriori state of the Equation (3.16) is described by

$$x^+(t_i) = x^-(t_i) + k(t_i)(l(t_i) - l^-(t_i)) \quad (3.19)$$

The covariance matrix of the posteriori state is described by the Equation (3.20) according to law of propagation by

$$p^+(t_i) = (I - k(t_i)H)p^-(t_i) \quad (3.20)$$

In the Equation (3.19) the predicted state and the measurements are weighted and combined to obtain the corrected state given in Equation (3.20). Hence, if the covariance of the measurements is smaller than that of the predicted state, the measurement’s weight will be high and thus the predicted states will be low. Thereby the uncertainty can be removed drastically.

### 3.2.2.3. The Discrete Kalman Filtering Algorithm

The Kalman filter estimates using the typical feedback control, where the estimation process is performed based on the state $x$ at an instant of time along with the feedback obtained in terms of measurements. Hence the filter has to update timely the state Equations and measurement equations. The current state and error covariance estimates are used in order to obtain the a-priori estimates for the next time run or step. The measurement equations incorporate a new measurement a-priori estimate for every time run or step to get the next posteriori estimate. This complete process can be thought of similar to predictor equations, while the measurement equations calculated can be thought of similar to corrector equations. In one phrase this can be treated as a correction algorithm based on predicted equations.

### 3.2.2.4. Steps involved in Synthesis using Kalman Filtering

1. Generate the array pattern using Taylor method.
2. Using the sensor, plot the actual array pattern which is perturbed due to random errors.
3. Define all state definition fields: $F$, $H$, $Q$, and $R$.
4. Define initial state estimate.
5. Obtain observation matrix.
6. Call the filter to obtain updated state estimate & find Kalman gain.
7. Return to step (5) and repeat.
8. Find the mean square error.
9. Plot the final corrected array pattern.

### 3.2.3. H- Infinity Filtering

Kalman filtering, also known as H2 filter, actually minimizes the “average” estimation error based on \textit{a-priori} and \textit{posteriori} estimates. It minimizes the variance of the existing error during the process of estimation. Few limitations of this filter are mainly:

a. The noise is assumed to be known and most of the cases it is the Gaussian noise.
b. Since it minimizes only the “average estimation error”, the performance of the filter in extreme case of maximum errors remains doubtful.

When the focus is made to resolve these limitations of Kalman filter, “H-infinity filter” also known as “Minimax filter” came into light in late 1980s. Using this filter, the extreme case of minimising the maximum error called as the “worst-case” is performed. This filter minimises the so called “minimax estimation error” other than the “Average estimation error”. There is no requirement of knowledge of the noise characterisation during estimation using this filter.

#### 3.2.3.1. H-Infinity Filter Estimation Process

These two features highlighted the application of H-infinity estimation algorithm widespread compared to Kalman filter estimation algorithm in the recent decades. Whenever the uncertainty in the process or system is high and unmanageable by simple algorithms, H-infinity algorithm is proposed. This filter also has similar steps to that of
the Kalman filter optimisation, and so the complexity involved in estimation and implementation remains complex similar to that of the Kalman algorithm.

The steps involved in the estimation of the state for H-Infinity filter are similar to Kalman filtering and are given as follows:

\[ L_k = (IQP_k + C^TV^zCP_k)^s \]  \hspace{1cm} (3.21)

\[ K_k = AP_kL_kC^TV^z \]  \hspace{1cm} (3.22)

\[ \hat{X}_{k+1} = A\hat{X}_k + Bu_k + K_k(yC\hat{X}_k) \]  \hspace{1cm} (3.23)

\[ P_{k+s} = AP_kL_kA^T + W \]  \hspace{1cm} (3.24)

This set of Equation (3.21) to Equation (3.24) are called as H-filter equations. They resemble the Kalman filter equations, but their details are quite different.

\( I \) is the identity matrix from matrix theory with ones in its diagonal. The initial state estimate \( X_0 \), and the initial value \( P_0 \) should be set to give acceptable filter performance. \( P_0 \) is chosen to be small enough when high confidence lies in initial state estimate.

Since, no known noise is considered, the H infinity estimator will consider all possible kinds of noise behaviours. It may be a worst-case estimator, when the estimator is over conservative. This approach considers the gain of the filter as similar to Signal to Noise Ratio and surprisingly it is always possible for this gain of an adaptive filter to be unity and LMS filter achieves it. The optimum value of LMS algorithm should be therefore in accordance with the H infinity computations rather than mean square error computation.

### 3.2.3.2. Optimization by H-infinity Filtering

Similar steps of Kalman filter are repeated in this H-Infinity filter and a parameter \( \gamma \) is introduced to improvise the reconstructed signal, where \( \gamma \) is the noise attenuation level. No requirement of covariance matrices, as no prior knowledge of noise is required. The filter gain \( K_k \) is computed such that the maximum singular value is always less than \( \gamma \), minimizing the worst-case estimation error. \( \gamma \) value must be chosen such that there exists a solution. In such a case the filter can be solved using a constant gain.
K which is found by solving the following given set of simultaneous equations as given below from Equation (3.25) to Equation (3.27)

\[
\text{Gain} = H^T P (P / \gamma^2 + I)^{-1} \tag{3.25}
\]

\[
m = FP F^T + I \tag{3.26}
\]

\[
P^- = m^- - \frac{I}{\gamma^2} + HH^T \tag{3.27}
\]

3.2.3.3. Steps involved in Synthesis using H-infinity Filtering

1. Generate the array pattern using Taylor method.
2. Using the sensor, plot the actual array pattern which is perturbed due to random errors.
3. Define all state definition fields: \( F, H, I \) & \( \gamma \) (Noise attenuation level).
4. Define initial state estimate.
5. Call the filter to obtain the estimate value.
6. Find gain of the filter.
7. Return to step (5) and repeat.
8. Find the mean square error.
9. Plot the final corrected array pattern.

3.2.4. Comparison of Gradient Based Technique, Kalman and H-Infinity Filtering

The statistical based weighting methods discussed in this chapter are simulated and their results are analysed and compared. For the simulations, the true array and actual array discussed and plotted in Chapter 2 are considered to look at the minimisation of the error discussed in Section (2.3). In brief the Taylor array designed for \( N = 32 \) elements, \( SLL = -25 \text{ dB} \) and \( \langle \bar{n} \rangle = 5 \), with an interelement spacing \( 0.5\lambda \) is chosen. The total length of the array for the given above values is \( 15.5 \lambda \) and the error which resulted in the actual array (perturbed) is due to when \( \sigma = 2 \text{ dB} \) and \( \delta = 10^0 \). The Figure 3.4 shows the normalised current distribution of the true and actual arrays where the real and imaginary parts of these coefficients are also plotted to analyse the efficiency of the algorithms in correcting the coefficients and the far field patterns.
Figure: 3.4. Normalised Current Distribution for Real and Imaginary parts of the true and actual array coefficients

Figure: 3.5. Far field patterns for the true (reference) and actual array (input for algorithm)

The far field pattern of the true and the actual arrays is once again given in this section for quick reference, as shown in Figure 3.5, where the algorithms always tries to correct the perturbed pattern of the actual array towards the array pattern of the true array, minimizing the effects of random errors.
3.2.4.1. Results using Gradient Based Algorithm - LMS

The Least Mean Square (LMS) Algorithm, which estimates the weights for correction of the coefficients and thereby the array pattern is simulated with the actual array coefficients as the input signal and the true coefficients to be the desired set of values. The Figure 3.6 shows the current distribution along the true array, actual array and the corrected array.

The real part of the true array coefficients is smooth and the imaginary part of the true array is zero as it is not associated with any phase changes. The actual array coefficients are assumed to be fluctuated due to the random errors with $\sigma = 2\ dB$ and $\delta = 10^0$. Now these errors are minimized using the LMS algorithm.

After estimation and correction, the corrected array coefficients are plotted in the Figure 3.6 and in Figure 3.7 which are almost overlapped with that of the true array. They are not visible as separate set of coefficients or as a different distribution curve as the algorithm is set to converge when the residual error is 0.2%.

![Normalized Current Distribution - True, Actual and Corrected](image)

Figure: 3.6. Normalized Current Distribution for Real and Imaginary parts of the true, actual and corrected arrays (Corrected coefficients almost merge with true ones)
Figure: 3.7. Expanded plot of current distribution showing the corrected coefficients’ curve almost overlapping with that of true coefficients

Hence in Figure 3.7 the initial part of the distribution is focused and shown on an expanded scale to understand the accuracy in computing the corrected coefficients’ curve which have almost overlapped with that of true coefficients. The LMS algorithm has resulted in this set of corrected array coefficients after 2095 iterations which is considered usually to be high and results in corrected far fields in Figure 3.8.

Figure: 3.8. Corrected array pattern using LMS algorithm which is optimized towards the true array pattern
The far field patterns of the true array and the corrected array are plotted in Figure 3.8 where the corrected array pattern almost overlaps and seems to be exactly the same as the true array pattern.

The optimization using the LMS algorithm has given good result where it converged towards 0.02% of residual error at 2095\textsuperscript{th} iteration. The performance of the algorithm also depends on the noise present in the input signal and this is analysed in detail in Chapter 4, using a Modified Gradient Based Algorithm which minimises the number of iterations, attaining the similar accuracy in correction.

### 3.2.4.2. Results using Kalman Filtering

Using the Equation (3.17) to Equation (3.20), the Kalman filter’s initial state estimate is calculated and thereby the observation matrix is updated and the final predicted states resulted in the corrected values. Thus, the algorithm for Kalman filtering leads to the correction of array pattern which follows the true array pattern only after the first half of the array pattern in the desirable way.

The Figure 3.9 shows the corrected array pattern which is considered to be optimized after the implementation of Kalman filtering on the actual array.
The pattern in Figure 3.9 follows the true pattern in the first half but with raised sidelobe levels where the first sidelobe level is exactly $-25 \, dB$ but with next sidelobe level still above with $-24.05 \, dB$. These values depend on the initial prediction values of the algorithm and may be minimised by working more on the initial values in the algorithm. The original pattern described in the Figure 3.10 is the true array pattern and is plotted for comparison with that of the “Estimated” curve alone.

![Comparison of Estimated Field with Original Field of the Array - Kalman Filter](image)

*Figure: 3.10. Corrected array pattern using Kalman filter which is optimized towards the true array pattern*

### 3.2.4.3. Results using H-infinity Filtering

H-Infinity filter as mentioned previously proves to be efficient than that of the Kalman filter as the corrected array pattern almost overlaps with the true array pattern similar to LMS algorithm.

This corrected array pattern is shown in Figure 3.11 along with the original or the true array pattern and the actual pattern. The sidelobe levels which have deteriorated to $-19.26 \, dB$, due to the random errors in the actual array in Figure 3.11, are corrected towards the original sidelobe level value of $-26.63 \, dB$ in the true array.
This is clearly shown in the Figure 3.12 where the corrected array pattern is overlapped with the true or original array pattern. The dashed blue colour curve indicating the corrected pattern completely overlaps and hides the true array pattern which should be in red colour.

Figure: 3.11. Corrected array pattern using H-Infinity filter compared with original and actual array patterns

Figure: 3.12. Corrected array pattern using H-Infinity filter which is optimized towards the true array pattern
The H-Infinity algorithm optimizes the corrected array pattern towards the true array with an initial error which seems almost imperceptible. This error is again due to the initialisation of the estimates at the start point of the algorithm. Similar to the case of LMS algorithm, where the step size $\mu$ value has the complete control on the convergence speed, the initial estimate in the H-Infinity and in the Kalman filter has impact on the final corrected array pattern.

From the simulated results it is clearly evident that the array patterns can be corrected with proper choice of algorithm depending on the trade-off between complexity in execution of the algorithm and the slow convergence of the algorithm. Giving a thought to minimise the complexity in the algorithm keeping in view for real time hardware implementation, a modified LMS algorithm is proposed which is supposed to converge at a faster rate when compared to that of the familiar and simple LMS algorithm.

**Summary**

The implementation of the algorithms discussed in this chapter are simulated using MATLAB Software. The concepts of optimisation are used in various fields like econometrics, bioprocessing, signal processing, etc. Application of these concepts to optimise the array antenna patterns is discussed and simulated.

While using the weighting methods for optimisation, the Taylor method was proven to be more optimum when compared to that of the Chebyshev. Later, the statistical methods used for optimisation resulted in removal of noise or random errors from the array pattern, and are compared to understand which algorithm suits the analysis of the problem in a proper direction.

Strictly by performance, the array pattern is more optimised when H-Infinity filtering is applied on the array coefficients, out of simple LMS, Kalman filtering and H-Infinity filtering. The LMS is simple to implement but has slow convergence rate. The Kalman filter proved to be better than LMS but at the cost of simplicity of implementation as the Kalman filter requires priori and posteriori estimate vectors along with the current states of the process. But it minimizes “average” estimation error only, and more precisely, the variance of the estimation error, H-infinity filtering is
studied and simulated which actually minimizes the “worst-case” estimation error. Both Kalman and H-Infinity are equally complex when compared with LMS, except that the H-Infinity filter does not require the knowledge of noise or the perturbing random errors.

Also the H-infinity algorithm is treated to be much less complex than that of the Kalman filtering, speaking in the point of the matrix computations since it does not require a large number of past measurements to be stored.

In the later chapters, a Modified Gradient Based LMS Algorithm is proposed which has got both the advantages of simplicity and as well as fast convergence and with more robustness to varied conditions. It is supposed not to require large number of measurements to be stored for the purpose of prediction and correction.