CHAPTER-6

SIDELOBE REDUCTION
6.1 INTRODUCTION

In a dense target environment or in situations where there are large undesired scatterers (point clutter), it is often desirable to reduce the time sidelobes of phase coded sequences to a prescribed low level. In principle, there is no difference between the problems of resolving a target in the interference from other targets and the detection of target in clutter. In previous chapters the reduction of the range sidelobes of the compressed pulse has been of much concern in the application of matched filter techniques to radar systems. In fact the mutual interference between targets or self clutter imposes rather fundamental limitations on resolution performance. So far, the reduction of the range sidelobes has been approached via waveform design. However, the attainable sidelobes levels for phase coded pulse trains might be inadequate for specific applications. Although it is possible to use a.m.p.h.m. pulse trains such as Huffman Codes (or codes discussed in previous chapter) to obtain the desired degree of discrimination, in most high power radar systems this method is not readily available. Thus, the designer has to resort to other sidelobe reduction techniques.

In principle sidelobe reduction can be achieved by either

(i) Amplitude-or Phase weighting in the Frequency Domain

(ii) Amplitude-or Phase weighting in the Time Domain

The weighting may be accomplished at the transmitter or receiver or at both. Furthermore, the shaping can be performed at the RF, IF or video stages.

Sidelobe suppression in the frequency domain requires the design of a filter such that the spectrum of the filtered waveform has a linear phase-frequency relationship and that the spectrum magnitude be proportional to one of the many available weighting functions such as Taylor or Chebyshev, [22], [31]. The rapid advance of digital hardware, alongwith the pipeline FFT configuration, does permit practical realization of the required transfer function as depicted in Fig. 2.9.

Amplitude weighting may be introduced into a pulse-compression system either entirely at the receiver, entirely at the transmitter, or at both simultaneously. Equal weighting at both the transmitter and receiver is equivalent to altering the transmitted
waveform. In this case the system is still considered as matched. However, it can be shown that in the peak power limited case, the SNR which assumes weighting at the receiver alone is greater or equal to the SNR which assumes matched weighting at the transmitter and receiver, [35]. An additional reason for unilateral weighting at the receiver is due to the advantage of operating the transmitter at its peak power limit (no expensive amplitude modulators required). Furthermore, amplitude weighting solely at the receiver can be maintained conveniently, due to the accessibility of the components and the low power levels involved. For these reasons it is henceforth assumed that weighting is performed solely at the receiver at the expense of a lower SNR.

A convenient way of weighting is at the IF stage in the time domain. Most of the sidelobe reduction techniques which have been proposed depend on cascading a weighting filter (tapped delay line) after the MF or by providing a suitable band shaping network, [76], [96]. However, instead of placing a sidelobe reduction filter after the MF it is probably more straightforward to design a mismatched filter (MMF) under some conditions of optimality, [78]. The amount of mismatch from the matched conditions is usually characterized by the loss factor $L_s$, given by, [76]

$$ L_s = \frac{\text{SNR(weighted)}}{\text{SNR(matched)}} $$

For an input sequence $a(n)$ of length $(N+1)$ and a filter weighting sequence $h(n)$ of length $(M+1)$, $L_s$ becomes

$$ L_s = \frac{\left| \sum_{n=0}^{M} h(n) a(j - n) \right|^2}{\sum_{n=0}^{N} |a(n)|^2 \sum_{n=0}^{M} |h(n)|^2} \leq 1 \quad \text{(6.1)} $$

Where $j \geq N$ denotes the time delay for which the output is maximum. (It is assumed that $(M+1) \geq (N+1)$. The basis of Eqn.6.1 is that for coherent summation signal components add as voltage levels while the noise components add as power levels.

87
6.2 INVERSE FILTERS

The reduction of the sidelobe interference can be accomplished through the use of inverse or deconvolution filters. These filters (equalizers) have been of much concern in removing inter symbol interference in communication and they are also of interest in spectrum analysis. What is required ideally is a digital filter which is inverse to the code sequence. That is, the response of the filter to the code would ideally be a sequence of which all the elements except one are zero. The position, L, of this nonzero element, which can be chosen to have unity magnitude, is immaterial. If the code sequence is \((a_0, a_1, \ldots, a_M)\) its Z-transform is given by

\[
A(Z) = a_0 + a_1Z^{-1} + \ldots + a_MZ^{-M}
\]

The filter will have the desired response if the transfer functions \(Z^L/A(Z)\). If \(A(Z)\) has all its zeroes within the unit circle, \(Z^L/A(Z)\) will have all its poles there and the filter will be stable and the transfer function will be realizable by a recursive filter or it can be closely approximated by a transversal filter. However, it has been shown in the preceding chapter that sequences with high energy efficiency (such as binary sequences) must have approximately half of their zeroes inside and half outside the unit circle. The inverse filter will thus have poles outside the unit circle and consequently will be unstable. Thus \(Z^L/A(Z)\) will not be physically realizable for the signals which are of interest as it will have poles outside the unit circle. Nevertheless, provided that \(L\) is large enough, a good approximation to \(Z^L/A(Z)\) can be obtained which is physically realizable.

One approach is to factor \(1/A(Z)\) into the product of a non-physically realizable part \(1/A_-(Z)\), whose poles are all outside the unit circle, and a realizable part \(1/A_+(Z)\) whose poles are all within the unit circle. This factorization can be done by the use of a polynomial root finding routine and a digital computer. The task, however, becomes onerous for sequences of length more than about forty. \(1/A(Z)\) can be expanded by polynomial division into a convergent series in powers of \(Z\) and if the expansion is truncated after the term in \(Z^L\), then the result, after multiplication by \(Z^L\), is a physically realizable transfer function. \(1/A_+(Z)\), having all its poles within the unit circle,
can be directly realized as the transfer function of a recursive filter. Alternatively, it can be approximately realized by expanding it as a convergent series in powers of $Z^{-1}$ which is truncated at some point and then approximately realized as the transfer function of a non-recursive filter.

Another approach to obtaining, to a degree of approximation, the weighting sequence of an ideal inverse filter is to augment the sequence $(a_0, a_1, \ldots, a_M)$ to $(P+1)$ points with a large number of zeroes and to compute the discrete Fourier transform $(A_0, A_1, \ldots, A_P)$ of the augmented sequence. The sequence $(1/A_0, 1/A_1, \ldots, 1/A_P)$ represents the sample values of the transfer function of the inverse filter. By computing its inverse discrete Fourier transform an aliased or folded version of the ideal inverse filter weighting sequence is obtained.

A filter designed in one of these ways has a weighting sequence which is either a truncated or a folded version of the weighting sequence of the ideal non-physically realizable inverse filter. However, for a given weighting sequence length it is possible to design filters with a lower mean square sidelobe level than can be achieved by this method.

Robinson and Trietel, [77] have comprehensively studied the problem of designing a non-recursive digital filter so that its response to a given input sequence approximates a desired output sequence in the least squares sense. In the present case the ideal output is a sequence whose elements are all zero except for one unity element occurring at some position $L$. The actual response of a filter having the weighting sequence $(f_0, f_1, \ldots, f_N)$ to the input $(a_0, a_1, \ldots, a_M)$ will be some sequence $(C_0, C_1, \ldots, C_{M+N})$. What is required is that the sum

$$V = C_0^2 + C_1^2 + \cdots + (C_L - 1)^2 + \cdots + C_{M+N}^2$$

should be minimized by proper choice of the weighting sequence. The sum $V$ represents the 'energy' of the difference between the actual response and the ideal response sequences. Robinson and Treitel show that $V$ is minimized when the weighting sequence is given by the solution of the vector matrix equation.

$$\mathbf{Rf} = \mathbf{g}$$
Where \( f \) is a column vector whose elements are the elements of the weighting sequence that is to be calculated and \( g \) is a vector containing the discrete cross-correlation function of the input sequence and the desired output sequence. \( R \) is a matrix each row of which consists of a shifted version of the auto-correlation of the input sequence (Section 5.5.2).

Robinson, [43], [62] gives a most useful set of Fortran programs for efficiently solving such matrix equations. In addition he gives a routine which computes the optimum filter and evaluates \( V \) for each position \( L \) of the response peak. His programs thus enable one to find the response peak position \( L \) for which the least squares inverse filter of weighting sequence length \( M \) produces the best approximation to the desired response and to compute the filter weighting sequence.

There are three characteristics of a sidelobe suppressing pulse compression filter which are of particular interest. One is the total response sidelobe energy given by

\[
V_s = \sum_{k=0}^{M+N} C_k^2 - 1
\]

Where filter weighting sequence has been normalized so that the peak of its response to input sequence is unity. In a dense uncorrelated target environment, the self clutter power at the filter output is proportional to this quality. Another property of concern is the peak sidelobe level, \( \max |C_k| \), which determines the ability of the system to resolve a weak target which is adjacent to a strong one. Finally, the total 'filter energy', given by

\[
V_s = \sum_{k=0}^{N} f_k^2
\]

is important since with white noise at the receiver input the noise power at the filter output is proportional to this quantity.

As the optimum filter weighting sequence length is increased, its behaviour approaches that of the ideal inverse filter (i.e. response sidelobes are suppressed), [78]. The Table 6.1 shows the decrease in signal to noise ratio (i.e. loss factor \( L_s \), equ 6.1) That results when the matched filter is replaced by the ideal inverse filter for the Barker
sequences of length from four to thirteen. It is to be remarked that a loss of only 0.2dB in
SNR is incurred when the sidelobes of the 13-element Barker sequence are completely
suppressed a fact which was first pointed out by Key et al, [76] and Messe et al, [96].

6.2.1 An Iterative Method for Range Side Lobe Suppression for Binary Codes

In section 5.5.2, it has been shown that the inverse filter weighting sequence or
coefficients obtained in the manner as discussed above, has better autocorrelation
function, smaller than that of the input sequence itself, [98], [99]. The same concept is
used here for sidelobe suppression by taking the filter coefficients as the input
sequence, evaluating its inverse filter and each time evaluating its response.

Table 6.2 and Fig. 6.1, show the variations of three quantities of a sidelobe
suppressing filter (V_n, max C(k) and V_n) for each iteration or computer run, when the
input sequence is the Barker sequence of length 13. The corresponding matched filter
output is also shown for comparison. It can be seen from the table and the fig. that
quantities V_n and max|C(k)| are decreasing with the number of iterations. The quantity
V_n is approximately constant. It can also be observed that as the iteration is increased
the performance of inverse filter and the matched filter becomes almost identical. The
advantage of the method proposed for sidelobe suppression is that the desired
suppression is achieved without increasing the length of the filter. The proposed
scheme can also be used for codes other than Barker codes.

Table: 6.1

<table>
<thead>
<tr>
<th>Barker Sequence Length</th>
<th>Loss in SNR when Matched Filter is replaced by Inverse filter (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1.7</td>
</tr>
<tr>
<td>5</td>
<td>0.6</td>
</tr>
<tr>
<td>7</td>
<td>1.5</td>
</tr>
<tr>
<td>11</td>
<td>1.5</td>
</tr>
<tr>
<td>13</td>
<td>0.2</td>
</tr>
</tbody>
</table>
Fig. 4.1 Variation of $V_1$, $V_4$, and $\max |C_1|$ at different iterations for Barker code of length 13.

- Sidelobe energy (IF output); $V_1$
- Sidelobe energy (corresponding MF)
- $\max$ sidelobe (IF); $\max |C_1|
- Filter energy (IP); $V_4$
| Iteration | IF output SL energy V | max SL \( |C_k| \) max \(|C_k|^2\) Matched filter | max SL (MF) Filter energy \( V_n \) (IF) |
|-----------|-----------------------|----------------------|---------------------|---------------------|
| 1         | 0.026                 | 0.063                | 0.071               | 0.077               | 0.980               |
| 2         | 0.010                 | 0.055                | 0.017               | 0.063               | 0.985               |
| 3         | 0.00858               | 0.056                | 0.0117              | 0.061               | 0.985               |
| 4         | 0.0075                | 0.052                | 0.0097              | 0.052               | 0.986               |
| 5         | 0.00717               | 0.052                | 0.0086              | 0.053               | 0.987               |
| 6         | 0.0067                | 0.049                | 0.0079              | 0.050               | 0.987               |
| 7         | 0.0062                | 0.048                | 0.0072              | 0.049               | 0.988               |
| 8         | 0.0059                | 0.047                | 0.0067              | 0.048               | 0.988               |
| 9         | 0.0055                | 0.046                | 0.0062              | 0.046               | 0.989               |
| 10        | 0.0052                | 0.045                | 0.0059              | 0.045               | 0.989               |
| 11        | 0.0050                | 0.044                | 0.0055              | 0.044               | 0.990               |
| 12        | 0.0047                | 0.043                | 0.0052              | 0.043               | 0.990               |
| 13        | 0.0045                | 0.042                | 0.0049              | 0.042               | 0.991               |
| 14        | 0.0043                | 0.041                | 0.0046              | 0.041               | 0.991               |
| 15        | 0.0041                | 0.040                | 0.0044              | 0.040               | 0.991               |
| 16        | 0.0039                | 0.039                | 0.0042              | 0.040               | 0.992               |
| 17        | 0.0038                | 0.038                | 0.0040              | 0.039               | 0.992               |
| 18        | 0.0036                | 0.038                | 0.0038              | 0.038               | 0.992               |
| 19        | 0.0035                | 0.037                | 0.0037              | 0.037               | 0.993               |
| 20        | 0.0034                | 0.036                | 0.0035              | 0.037               | 0.993               |
6.3 SIMPLE SCHEME FOR SIDE LOBE ELIMINATION

In this section, a simple approach is developed which completely eliminates the effects of the sidelobes.

Let \( \{a_i\} \) is the input sequence to a matched filter, matched to the input sequence \( \{a_i\} \) with discrete weighting sequence \( \{f_i\} \). With reference to Fig. 6.2, the output of the matched filter \( \{C_i\} \) is the convolution of \( \{a_i\} \) with \( \{f_i\} \). In terms of Z-transform.

\[
A(Z) = a_0 + a_1 Z^{-1} + \ldots + a_{N-1} Z^{N-1}
\] (6.2)

\[
F(Z) = a_{N-1} Z^{-N+1} + a_{N-2} Z^{-N+2} + \ldots + a_0 Z^{N+1}
\] (6.3)

\[
C(Z) = A(Z)F(Z) = Co + C_1 Z^{-1} + C_2 Z^{-2} + \ldots + C_{N-1} Z^{N-1} + \ldots + C_{2N-2} Z^{2N-2}
\] (6.4)

\( C(Z) \) is also the Z-transform of the auto-correlation function of the sequence \( \{a_i\} \) which is of the length \((2N-1)\). The main lobe in the output of the filter occurs at a delay of \((N-1)T\) seconds or in the centre, \(1/T\) being the clock frequency. The output should ideally be a pulse at \(t = NT\) seconds and zero elsewhere. However, it is not the actual case. The output usually contains some side-lobes in addition to the main lobe. Suppose, the ideal or desired output is denoted by \(D(Z)\) and the undesired output (having only the side-lobes) is denoted by \(B(Z)\), such that

\[
C(Z) = B(Z) + D(Z)
\] (6.5)

where

\[
B(Z) = C_0 + C_1 Z^{-1} + \ldots + 0.Z^{N+1} + \ldots
\] (6.6)

and

\[
D(Z) = 0 + 0.Z^{-1} + \ldots + C_{N-1} Z^{N+1} + 0.Z^{N+2} + \ldots
\] (6.7)

Where the length of the \(B(Z)\) and \(D(Z)\) is the same as that of \(C(Z)\) i.e. equal to \((2N-1)\).
The undesired effects represented by $B(Z)$, can now be eliminated after subtracting it from the output $C(Z)$, as shown in fig. 6.3 so that only the desired output is left.

$$D(Z) = C(Z) - B(Z)$$

$$= A(Z) F(Z) - B(Z) \quad \text{.......................................................... (6.8)}$$

The undesired effects represented by $B(Z)$ can be generated after subtracting $D(Z)$ from $C(Z)$ as shown in Fig. 6.4 i.e.

$$B(Z) = C(Z) - D(Z) \quad \text{.......................................................... (6.9)}$$

The overall scheme for side-lobe elimination is shown in Fig. 6.5 where both the matched filters are identical. As can be seen from Fig. 6.5 the output contains no side-lobes. If the auto-correlation function of the input sequence $\{a_i\}$ (output of the matched filter) is normalized such that the main lobe has a value equal to unity, then the $C_{N-1}$ in equation 6.7 is equal to unity.

The over-all transfer function of the scheme proposed for side-lobe elimination can be evaluated with reference to figures 6.6 to 6.8. As can be seen from these figures the over-all transfer function of the scheme is $1/A(Z)$ which is same as that of an ideal inverse filter.

From the above discussions it can be concluded that the scheme proposed for side-lobe elimination may be regarded as some kind of ideal inverse filtering operation, without actually using an inverse filter.

However, the noise performance of above scheme Fig. 6.5 comes out to be poor, because at the output of the subtracter of the upper channel both the noise terms will be added. If the noise is white Gaussian with zero mean and variance $\sigma^2$, Then noise output at both the channels will be proportional to $\sigma^2 \cdot \sqrt[n]{\sum_{i=0}^{n-1} f_i^2} = \eta$. Therefore after subtracting the outputs of both the channels the noise terms will fail to cancel, therefore overall effect of noise will increase. However, the effect of noise can be decreased by inserting a squarer circuit in both the channels as shown in Fig. 6.9.
Fig. 6.2

Fig. 6.3

Fig. 6.4
Fig. 6.5

Fig. 6.6
Fig. 6.7

\[ F(Z) - F(Z) + \frac{D(Z)}{A(Z)} \]

Fig. 6.8

\[ \frac{D(Z)}{A(Z)} \]
Fig. 6.9
The input to the squarer is $C(Z) + \eta$. Therefore the output will be

$$[C(Z) + \eta]^2 = C^2(Z) + \eta^2 + 2\eta C(Z)$$ \hspace{1cm} (6.10)

Modified $B(Z)$ will be

$$B(Z) = C^2(Z) + \eta^2 + 2\eta C(Z) - D(Z)$$ \hspace{1cm} (6.10)

The output of the subtracter of the upper channel will be the difference of Equ. 6.10 & 6.11 which will be equal to $D(Z)$, which is free from undesired sidelobes and noise.

6.4 NEURAL NETWORK FOR SIDELOBE SUPPRESSION

6.4.1 Introduction

In this section various aspects of using Artificial Neural Networks (ANN) for sidelobe reduction are explored. After two decades of eclipse, interest in artificial neural networks has grown rapidly over the past few years. This resurgence of interest has been fired by both theoretical and application successes. ANN has found wide application in pattern recognition. The pattern recognition is applicable in radar signal detection or classification processes. In the present case Radar signals can be treated as data patterns. The basic theories of statistical hypothesis, testing, decision theory etc. apply in these situations. Indeed, many pattern recognition techniques employ likelihood ratios, when the statistics are known apriori, and pattern recognition or classification is then merely an application of multiple hypothesis testing. Nevertheless, some interesting techniques presented under the name of pattern recognition, learning machines, artificial intelligence etc. are applicable to radar system and should be available for the radar designers consideration. In the following sections brief theory of neural networks and their use for sidelobe suppression is discussed.
### 4.2 Fundamentals of Artificial Neural Networks

The basic implementation mechanism can be explained with reference to Figure 6.10(a). Here, a set of inputs labeled \( x_1, x_2, \ldots, x_n \) is applied to the artificial neuron. These inputs, collectively referred to as the vector \( X \), correspond to the signals into the synapses of a biological neuron. Each signal is multiplied by an associated weight \( w_1, w_2, \ldots, w_n \) before it is applied to the summation block, labelled \( \Sigma \). Each weight corresponds to the 'strength' of a single biological synoptic connection. (The set of weights is referred to collectively as the vector \( W \)). The summation block, corresponding roughly to the biological cell body, adds all of the weighted inputs algebraically, producing an output that is called \( \text{NET} \). This may be compactly stated in vector notation as follows:

\[
\text{NET} = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n
\]

\[= \mathbf{X} \mathbf{W}\]

#### Activation Functions

The \( \text{NET} \) signal is usually further processed by an activation function \( F \) to produce the neuron's output \( \text{OUT} \). This may be a simple linear function

\[
\text{OUT} = K(\text{NET})
\]

Where \( K \) is a constant, a threshold function,

\[
\text{OUT} = 1, \text{ if } \text{NET} > T
\]

\[
\text{OUT} = 0 \quad \text{otherwise}
\]

Where \( T \) is a constant threshold value, or a function. That more accurately simulates the non-linear transfer characteristics of the biological neuron and permits more general network functions.
Fig. 6.10 (a) Artificial Neuron

\[ \text{NET} = XW \]

Fig. 6.10 (b) Artificial Neuron with Activation Function

\[ \text{OUT} = F(\text{NET}) \]

Fig. 6.10 (c) Sigmoidal Logistic Function

Fig. 6.11 Single Layer Neural Network
In figure 6.10(b) the block labeled F accepts the NET output and produces the signal labeled OUT if the F processing block compresses the range of NET, so that OUT never exceeds some low limits regardless of the value of NET, F is called a 'squashing function'. The squashing function is often chosen to be the logistic function or 'sigmoid' (meaning S-shaped) as shown in Fig. 6.10(c). This function is expressed mathematically as \( F(x) = \frac{1}{1 + e^{-x}} \). Thus

\[
\text{OUT} = \frac{1}{1 + e^{-\text{NET}}}
\]

By analogy to analog electronic systems, the activation function may be considered as a non-linear gain for the artificial neuron. This gain is calculated by finding the ratio of the change in OUT to a small change in NET. Thus gain is the slope of the curve at a specific excitation level. It varies from a low value at large negative excitations to a high value at zero excitation, and it drops back as excitation becomes very large and positive. Grossberg (1973) found that this non-linear gain characteristic solves the noise saturation dilemma that he posed; that is, how can the same network handle both small and large signals, small input signals require high gain through the network if they are to produce usable output; however, a large number of cascaded high gain stages can saturate the output with the amplified noise (random variation) that is present in any realizable network. Also, large input signals will saturate high gain stages, again eliminating any usable output. The central high gain region of the logistic function solves the problem of processing small signals, while its regions of decreasing gain at positive and negative extremes are appropriate for large excitations. In this way, a neuron performs with appropriate gain over a wide range of input levels.

Although a single neuron can perform certain simple pattern detection functions, the power of neural computation comes from connecting neurons into networks. The simplest network is a group of neurons arranged in a layer as shown in Figure 6.11. The set of inputs X has each of its elements connected to each artificial neuron through a separate weight.
Training the Network

The objective of training the network is to adjust the weights so that application of a set of inputs produces the desired set of outputs. These input-output sets can be referred to as vectors. Training assumes that each input vector is paired with a target vector representing the desired output. Before starting the training process, all of the weights must be initialized to small random numbers. This ensures that the network is not saturated by large values of the weights and prevents certain other training pathologies. Training the back propagation network requires the steps that follow:

1. Select the next training pair from the training set, apply the input vector to the network input.
2. Calculate the output of the network.
3. Calculate the error between the network output and the desired output.
4. Adjust the weights of the network in a way that minimizes the error.
5. Repeat steps 1 through 4 for each vector in the training set until the error for the entire set is acceptably low.

6.4.3 Results and Discussion

The above method is applied in the present case for suppressing the undesired sidelobes at the output of the matched filter. Three possible schemes have been proposed and studied they are as follows.

Scheme 1: The neural network is used after the matched filter for further suppressing the response sidelobes. Neural network can suppress the sidelobe to any desired value (ideally zero) but as the sequence length increases number of neurons and weights also increases making it practically difficult to realise the scheme for longer codes. For example in the case of Barker sequence of length 13, the matched filter output (which is also the auto-correlation function of the code) is of length 25. The number of weights of
the network becomes $25 \times 25 = 625$ which is quite large and obviously have practical limitations for implementing the scheme using appropriate hardware. As the sequence length increases, number of weights also increases accordingly.

**Scheme II**: In this scheme the matched filter is replaced by a single neural network. Input is the received code sequence and output is the auto-correlation function of the received sequence the desired output is ideal i.e. $[0 \ 0 \ 0 \ 0 ... \ 0 \ 1 \ 0 \ 0 ... \ 0 \ 0]$ (if the input sequence is normalized). This scheme is used with Barker Code of length 5. The results are given in Table 6.4(a). Table (a) shows the sidelobe energy (SLE) and Peak sidelobe levels with number of iterations for various value of $n$ (the learning rate of ANN). With the increase in $n$ the number of iterations are decreasing, while SLE and PSL almost remains constant. Table 6.4(b) shows the results when the input code sequence is of length 5 will all element values given as +1. Table 6.4(c) shows the number of iterations with respect of $\beta$ (Degree of non-linearity used). As $\beta$ increases number of iterations decreases, sidelobe energy and peak sidelobe level also decreases. This table also shows the variation of summation of the square of weights corresponding to the nth output, which also decreases as $\beta$ increases. Table-6.5 gives these results when the input sequence is of length 13 (all element values +1). Table (a) shows the variation w.r.t. $\eta$ and Table (b) shows the variation w.r.t $\beta$. Fig. 6.12 shows the variation of SLE, PSL, and number of iteration corresponding to various values of $\eta$ & $\beta$. From Tables and Figures it is clear that the process is converging. However, as the sequence length increases number of weights of the neural network also increases. When the sequence length is quite large say 100 then number of weights required are $100 \times 199 = 19900$ which may be quite large putting serious limitations for practically implementing this scheme. Therefore, in order to reduce the number of weights on alternate scheme is being suggested and is being given as follows.
<table>
<thead>
<tr>
<th>S.No.</th>
<th>n</th>
<th>iterations</th>
<th>SLE (Nor)</th>
<th>PSL (Nor)</th>
<th>SLE (Nor)</th>
<th>PSL (Nor)</th>
<th>E_n_i</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.2</td>
<td>358</td>
<td>0.048</td>
<td>0.111</td>
<td>0.049</td>
<td>0.110</td>
<td>20.67</td>
</tr>
<tr>
<td>2</td>
<td>0.6</td>
<td>119</td>
<td>0.048</td>
<td>0.111</td>
<td>0.048</td>
<td>0.110</td>
<td>6.86</td>
</tr>
<tr>
<td>3</td>
<td>1.0</td>
<td>72</td>
<td>0.048</td>
<td>0.110</td>
<td>0.047</td>
<td>0.110</td>
<td>3.32</td>
</tr>
<tr>
<td>4</td>
<td>1.2</td>
<td>60</td>
<td>0.047</td>
<td>0.110</td>
<td>0.046</td>
<td>0.110</td>
<td>2.35</td>
</tr>
<tr>
<td>5</td>
<td>1.6</td>
<td>45</td>
<td>0.047</td>
<td>0.110</td>
<td>0.045</td>
<td>0.109</td>
<td>1.65</td>
</tr>
<tr>
<td>6</td>
<td>2.0</td>
<td>36</td>
<td>0.047</td>
<td>0.110</td>
<td>0.043</td>
<td>0.109</td>
<td>1.29</td>
</tr>
<tr>
<td>7</td>
<td>2.4</td>
<td>30</td>
<td>0.047</td>
<td>0.110</td>
<td>0.037</td>
<td>0.107</td>
<td>1.19</td>
</tr>
<tr>
<td>8</td>
<td>3.0</td>
<td>24</td>
<td>0.047</td>
<td>0.110</td>
<td>0.025</td>
<td>0.104</td>
<td>1.14</td>
</tr>
<tr>
<td>9</td>
<td>5.0</td>
<td>14</td>
<td>0.048</td>
<td>0.110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>8</td>
<td>0.048</td>
<td>0.110</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S.No.</td>
<td>n</td>
<td>No. of iterations</td>
<td>SLE (Nor)</td>
<td>PSL (Nor)</td>
<td>$E_{w1}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>-----</td>
<td>-------------------</td>
<td>-----------</td>
<td>-----------</td>
<td>----------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.6</td>
<td>126</td>
<td>0.276</td>
<td>0.110</td>
<td>6.40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.8</td>
<td>95</td>
<td>0.273</td>
<td>0.110</td>
<td>6.43</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1.0</td>
<td>76</td>
<td>0.273</td>
<td>0.110</td>
<td>6.44</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1.2</td>
<td>63</td>
<td>0.274</td>
<td>0.110</td>
<td>6.43</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1.6</td>
<td>47</td>
<td>0.276</td>
<td>0.110</td>
<td>6.44</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>2.0</td>
<td>38</td>
<td>0.267</td>
<td>0.109</td>
<td>6.34</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2.4</td>
<td>32</td>
<td>0.267</td>
<td>0.109</td>
<td>6.54</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3.0</td>
<td>25</td>
<td>0.269</td>
<td>0.109</td>
<td>6.62</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>5.0</td>
<td>15</td>
<td>0.274</td>
<td>0.109</td>
<td>6.50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.089</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>8.0</td>
<td>15</td>
<td>0.269</td>
<td>0.068</td>
<td>1.54</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>10.0</td>
<td>33</td>
<td>0.327</td>
<td>1.53</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
TABLE 6.6

<table>
<thead>
<tr>
<th>S.No.</th>
<th>n</th>
<th>No. of iterations</th>
<th>SLE (Nor)</th>
<th>PSL (Nor)</th>
<th>E w_{ni}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.6</td>
<td>125</td>
<td>0.048</td>
<td>0.110</td>
<td>6.04</td>
</tr>
<tr>
<td>2</td>
<td>0.8</td>
<td>93</td>
<td>0.048</td>
<td>0.111</td>
<td>6.03</td>
</tr>
<tr>
<td>3</td>
<td>1.0</td>
<td>75</td>
<td>0.048</td>
<td>0.110</td>
<td>6.06</td>
</tr>
<tr>
<td>4</td>
<td>1.2</td>
<td>62</td>
<td>0.048</td>
<td>0.110</td>
<td>6.05</td>
</tr>
<tr>
<td>5</td>
<td>1.6</td>
<td>47</td>
<td>0.047</td>
<td>0.110</td>
<td>6.10</td>
</tr>
<tr>
<td>6</td>
<td>2.0</td>
<td>37</td>
<td>0.048</td>
<td>0.111</td>
<td>6.08</td>
</tr>
<tr>
<td>7</td>
<td>2.4</td>
<td>31</td>
<td>0.048</td>
<td>0.110</td>
<td>6.11</td>
</tr>
<tr>
<td>8</td>
<td>3.0</td>
<td>25</td>
<td>0.047</td>
<td>0.109</td>
<td>6.17</td>
</tr>
<tr>
<td>9</td>
<td>5.0</td>
<td>15</td>
<td>0.046</td>
<td>0.108</td>
<td>6.28</td>
</tr>
</tbody>
</table>

SLE = Side Lobe Energy = Total output Energy - Main Lobe Energy

PSL = Peak Side Lobe Level

E w_{ni} = Summation of the Squares of Weights Corresponding to the nth output

Tolerance = Actual output - Desired Output

n = Learning rate of the Artificial Neural Network

θ = Degree of non-linearity used.
**Scheme III**: In this scheme the output of the neural network is kept as 5 in all the cases, independent of the length of the input code (instead of the ACF of the code as given in Scheme II). This reduces the number of weights considerably. Table 6.6 gives the variation of SLE, PSL, \( \sum w^2 \) and number of iterations for various values of \( n \) for a code of length 13 (all element values \(+1\)). Comparing these results with the results obtained in Scheme II (Table 6.5(a)) it is clear that the results are almost similar except the case of SLE which is much smaller in this case which is quite obvious as the length of the output is reduced from 25 to 5 only. This shows that the results are better in this case, and this achievement is obtained using much less number of weights \((13 \times 5)\) in this case as compared to \((13 \times 25)\) used in Scheme II. Tolerance in all the cases is kept at 0.1.

### 6.5 SUMMARY

If processor complexity is not of overriding concern, the use of a MMF may be justified in those cases where improvements in clutter performance are of significant magnitude. In general, there exists a trade-off between resolution and detection performance which sets practical limits on the sidelobe suppression that can be obtained. However, the degradation in SNR is usually small if the input waveform is optimized for matched conditions. Moreover, as printed out by Rihaczek, [35] the approach of optimum waveform design for an MF receiver also implicitly solves the problem of waveform design for the optimum filter in the presence of clutter. Therefore, whenever possible it is preferable to transmit a wider spectrum to achieve the desired resolution rather than widening the spectrum of the receiver filter.