

# **APPLICATION OF SOFT COMPUTING TECHNIQUES TO RADAR PULSE COMPRESSION**

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE  
DEGREE OF

***Bachelor of Technology***

IN

**ELECTRONICS AND INSTRUMENTATION ENGINEERING**

By

**NISHANT PANIGRAHI**

**10607012**

**SUDEEP TRIPATHY**

**10607013**



---

**Department of Electronics and Communication Engineering,  
National Institute of Technology, Rourkela.**

***May, 2010***

# **APPLICATION OF SOFT COMPUTING TECHNIQUES TO RADAR PULSE COMPRESSION**

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE  
DEGREE OF

***Bachelor of Technology***

IN

**ELECTRONICS AND INSTRUMENTATION ENGINEERING**

By

**NISHANT PANIGRAHI**

**10607012**

**SUDEEP TRIPATHY**

**10607013**

Under the guidance of

**Prof. Ajit Kumar Sahoo**



---

**Department of Electronics and Communication Engineering,  
National Institute of Technology, Rourkela.**

***May, 2010***



**National Institute Of Technology,**

**Rourkela-769008, Orissa**

## **CERTIFICATE**

This is to certify that the thesis entitled, “**APPLICATION OF SOFT COMPUTING TECHNIQUES TO RADAR PULSE COMPRESSION**”, submitted by **Nishant Panigrahi** and **Sudeep Tripathy** in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in **Electronics and Instrumentation Engineering**, Department of **Electronics and Communication Engineering** at **National Institute of Technology, Rourkela** is an authentic work carried out by them under my supervision and guidance.

They have fulfilled all the requirements for the submission of this project, which has reached the requisite standards.

DATE:

**Prof. Ajit Kumar Sahoo**  
Dept. of Electronics and Communication  
Engineering  
National Institute of Technology  
Rourkela-769008

# ACKNOWLEDGEMENT

Like every project worthy of reference, this one has been the handiwork of many dedicated souls. We take this opportunity to express our heartfelt thanks to everybody who has contributed in any measure and at any stage of this work.

At the outset, we convey our deepest gratitude to our project guide, **Prof. Ajit Kumar Sahoo** for providing us with the kernel of our project work. His invaluable and untiring support at crucial junctures helped materialize this project. His supreme belief in our abilities to conquer new frontiers kept us motivated. Never satisfied with mediocre results, he made us strive till we arrived at the best results, all of which have been included in this thesis.

We are also indebted to the **PhD** scholar, **Ms. Anangi Sailaza** for sparing us her valuable time to help with much needed inputs during this work. We also feel privileged to express our heartiest thanks and gratitude to **Prof G.S. Rath, Prof U.C. Pati & Prof T.K. Dan** for their invaluable and generous suggestions during the periodic project evaluation. This acknowledgement would be incomplete without tendering our sincere regards and heartfelt thanks to **Dr. S.K.Patra**, HOD, Electronics and Communication Engineering for assigning us the project work under *Prof. A.K.Sahoo*.

This project work was aided by the innumerable references on this topic all of which have been duly acknowledged in the reference section.

We also thank all of our friends for extending us their helping hands during this work.

Date:

Place: NIT, Rourkela

**Nishant Panigrahi**

**Roll no.-10607012**

**Sudeep Tripathy**

**Roll no.-10607013**

# CONTENTS

	Page no.
<b>Abstract</b>	<b>1</b>
<b>List of Figures</b>	<b>2</b>
<b>List of Tables</b>	<b>4</b>
<b>1. Introduction</b>	<b>5</b>
1.1.PULSE COMPRESSION	6
1.2.RADAR	8
1.3.RADAR PULSE COMPRESSION	10
1.3.1. Why RADAR Pulse Compression?	10
1.3.2. The Concept of RADAR Pulse Compression	11
1.3.3. Matched filter	13
1.3.4. Barker Code	14
<b>2. Adaptive Filters and their application to RADAR Pulse Compression</b>	<b>18</b>
2.1.ADAPTIVE SYSTEMS	19
2.1.1. OPEN- AND CLOSED-LOOP ADAPTATION	19
2.2.THE ADAPTIVE LINEAR COMBINER	20
2.2.1. INPUT SIGNAL AND WEIGHT VECTORS	21
2.2.2. DESIRED RESPONSE AND ERROR	22
2.2.3. THE PERFORMANCE FUNCTION	22
2.3.ADAPTIVE FILTER ALGORITHMS	23
2.4.SYSTEM IDENTIFICATION	27
2.4.1. COMPARISON OF CONVERGENCE RATE OF THE LMS AND RLS ALGORITHMS IN SYSTEM IDENTIFICATION	28
2.5.RADAR PULSE COMPRESSION using ADAPTIVE FILTER ALGORITHMS	29
<b>3. Artificial Neural Networks and their application to RADAR Pulse Compression</b>	<b>33</b>
3.1.BIOLOGICAL INSPIRATION	34
3.2.DEFINITION and PROPERTIES	35
3.3.NEURON MODEL	36
3.3.1. Single-Input Neuron	36
3.3.2. Transfer Functions	37

3.3.3. Multiple-Input Neuron	38
3.4.NETWORK ARCHITECTURES	39
3.4.1. A Layer of Neurons	39
3.4.2. Multiple Layers of Neurons	40
3.5.PERCEPTRON LEARNING RULE	41
3.5.1. LEARNING RULES	41
3.5.2. PERCEPTRON ARCHITECTURE	42
3.6.FUNCTION APPROXIMATION	44
3.7.THE BACK-PROPAGATION ALGORITHM	46
3.7.1. RADAR PULSE COMPRESSION using BPA MLP	46
4. CONCLUSION AND FUTURE WORK	54
BIBLIOGRAPHY	57

# ABSTRACT

*Soft Computing* is a term associated with fields characterized by the use of inexact solutions to computationally-hard tasks for which an exact solution cannot be derived in polynomial time. Almost contrary to conventional (Hard) computing, it is tolerant of imprecision, uncertainty, partial truth, and approximation to achieve tractability, robustness and low solution cost. Effectively, it resembles the Human Mind. The Soft Computing Techniques used in this project work are *Adaptive Filter Algorithms* and *Artificial Neural Networks*.

An *adaptive filter* is a filter that *self-adjusts* its transfer function according to an optimizing algorithm. The adaptive filter algorithms used in this project work are the LMS algorithm, the RLS algorithm, and a slight variation of RLS, the Modified RLS algorithm.

An *Artificial Neural Network (ANN)* is a mathematical model or computational model that tries to simulate the structure and/or functional aspects of biological neural networks. It consists of an interconnected group of artificial neurons and processes information using a connectionist approach to computation. Several models have been designed to realize an ANN. In this project, Multi-Layer Perceptron (MLP) Network is used. The algorithm used for modeling such a network is Back-Propagation Algorithm (BPA).

Through this project, there has been analyzed a possibility for using the Adaptive Filter Algorithms to determine optimum Matched Filter Coefficients and effectively designing Multi-Layer Perceptron Networks with adequate weight and bias parameters for RADAR Pulse Compression. Barker Codes are taken as system inputs for Radar Pulse Compression. In case of Adaptive Filters, a *convergence rate analysis* has also been performed for System Identification and in case of ANN, *Function Approximation* using a 1-2-1 neural network has also been dealt with. A comparison of the adaptive filter algorithms has been performed on the basis of Peak Sidelobe Ratio (PSR). Finally, SSRs are obtained using MLPs of varying neurons and hidden layers and are then compared under several criteria like Noise Performance and Doppler Tolerance.

**Keywords:** Adaptive, Artificial Neural Network, Matched Filter, System Identification Barker Codes, LMS, RLS, MLP, Function Approximation, Back Propagation Algorithm, PSR, SSR, SNR, Doppler Shift.

# **LIST OF FIGURES**

- 1.1 Example of 5-bit Bi-phase Code**
- 1.2 Compressed Phase-Coded Waveform through a Matched Filter**
- 1.3 Radar block diagram**
- 1.4 Duplexer**
- 1.5 Parabolic Antenna**
- 1.6 Transmitter and Receiver ultimate signals**
- 1.7 Phase-modulated waveform after Compression**
- 1.8 Phase Code Short Segments**
- 1.9 Chirp**
- 1.10 Matched Filter**
  - a) Matched filter at the transmitter side**
  - b) Matched filter at the receiver side**
- 1.11 Barker Code**
  - a) Matched filter coefficient of a Barker code of length 7 at the transmitter side**
  - b) Stretched signal**
  - c) Matched filter coefficient of a Barker code of length 7 at the receiver side**
  - d) Matched filter processing a received signal**
  - e) Compressed Signal**
- 1.12 Binary Phase decoder for Pulse Compression**
- 2.1 Open Loop Adaptive System**
- 2.2 Closed Loop Adaptive System**
- 2.3 General Form of Adaptive Linear Combiner**
- 2.4 Adaptive Linear Combiner in the form of Single-input adaptive transversal filter**
- 2.5 Multiple Input Adaptive Linear Combiner with bias weight  $w_{0k}$**
- 2.6 Multiple Input Adaptive Linear Combiner with desired response and error signals**
- 2.7 Adaptive transversal filter structure**
- 2.8 Block Diagram for System Identification**
- 2.9 Simulation demonstrating faster convergence of RLS over LMS**
- 2.10 Filter Output Signals for Barker Code of Length 2**
- 2.11 Filter Output Signals for Barker Code of Length 3**



- 2.12 Filter Output Signals for Barker Code of Length 4**
- 2.13 Filter Output Signals for Barker Code of Length 5**
- 2.14 Filter Output Signals for Barker Code of Length 7**
- 2.15 Filter Output Signals for Barker Code of Length 11**
- 2.16 Filter Output Signals for Barker Code of Length 13**
- 3.1 Schematic Diagram of Biological Neurons**
- 3.2 Single Input Neuron**
- 3.3 Multiple Input Neuron**
- 3.4 Multiple Input Neuron, Abbreviated Notation**
- 3.5 Layer of S Neurons**
- 3.6 Layer of S Neurons, Abbreviated Notation**
- 3.7 Three-Layer Network**
- 3.8 Three-Layer Network, Abbreviated Notation**
- 3.9 Perceptron Network**
- 3.10 Two-Input/Single Output Perceptron**
- 3.11 Example of Function Approximation Network**
- 3.12 Effect of Parameter Changes on Network Response**
- 3.13 Mean Square Error plots for (a) 3 neurons and (b) 9 neurons in the hidden layer**
- 3.14 Mean Square Error plots for (a) 3-5 and (b) 7- 9 neurons in MLP containing 2 Hidden Layers**
- 3.15 SSR vs. SNR plots for (a) 3-5 and (b) 7- 9 neurons in MLP containing 2 Hidden Layers**
- 3.16 Compressed Waveforms for Doppler Tolerance taking (a) 3 neurons and (b) 11 neurons in the hidden layer**
- 3.17 Compressed Waveforms for Doppler Tolerance taking (a) 3-11 and (b) 7- 1 neurons in the 2 hidden layers**

# **LIST OF TABLES**

**2.1 Filter Comparison Parameters [Peak Side Lobe (PSR)] for the LMS, RLS and modified RLS method evaluation for Barker Codes of different lengths with filter lengths taken same as that of the code**

**3.1 SSR Comparison in dB for different number of neurons in MLP containing 1 Hidden Layer**

**3.2 SSR Comparison in dB for different number of neurons in MLP containing 2 Hidden Layers**

**3.3 SSR Comparison in dB for different SNRs for different number of neurons in MLP containing 1 Hidden Layer**

**3.4 SSR Comparison in dB for different SNRs for different number of neurons in MLP containing 2 Hidden Layers**

**3.5 Doppler Shift Performance in dB for MLP containing 1 Hidden Layer**

**3.6 Doppler Shift Performance in dB for MLP containing 2 Hidden Layers**

# Chapter 1

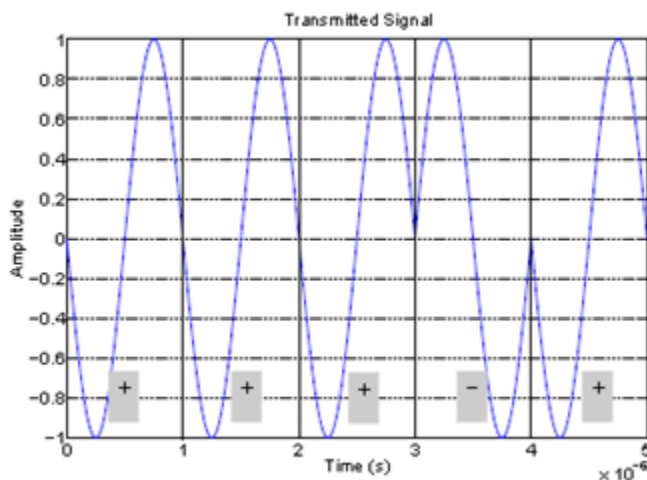
## INTRODUCTION

# 1.1 PULSE COMPRESSION

**Pulse Compression** is a signal processing technique mainly used in radar, sonar and echography to enhance the range resolution as well as the signal to noise ratio. This is achieved by modulating the transmitted pulse and then correlating the received signal with the transmitted pulse [45].

The technique of Pulse Compression consists of transmitting a pulse which sweeps through a range of frequencies with a carefully designed relationship of frequency to time. The received reflected pulse is cross-correlated with the digital version of the transmitted pulse and the resulting waveform is a narrow pulse of large magnitude.

Pulse compression involves transmitting a coded, wideband signal and compressing the return signal through filtering, which results in increased signal power and enhanced range resolution. Phase codes partition the transmitted pulse into equal segments, or sub pulses, and then switch the phase of the signal at specified intervals. In particular, binary phase codes switch the phase between two values where an example of a 5-bit bi-phase code is shown in Figure 1.1. Such types of bi-phase codes are known as BARKER CODES which are described in detail in Section 1.3.4. This waveform represents a carrier frequency being modulated in phase every sub pulse between 0 and  $\pi$  according to the code  $[+ + + - +]$  where + represents a phase of  $e^{i0}$  and represents  $e^{i\pi}$ . A sub pulse is defined as the time duration of one bit so a 5-bit code as shown which is 5  $\mu$ s in duration will have five 1 $\mu$ s sub pulses.

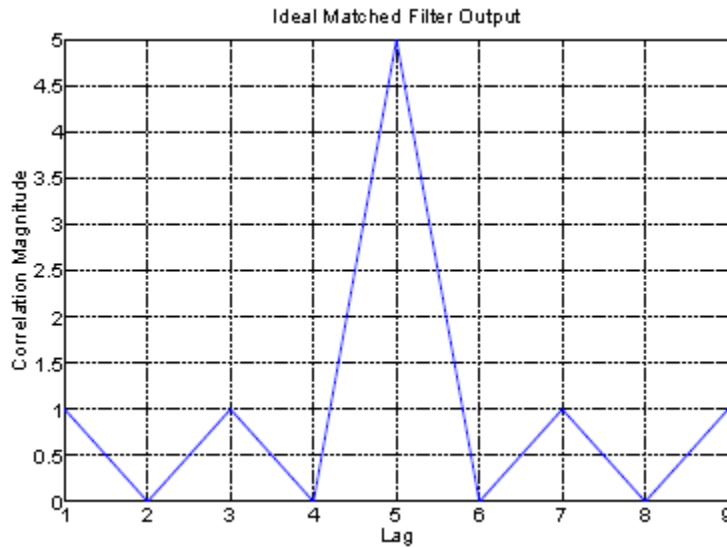


**Figure 1.1: Example of 5-bit Bi-phase Code**

The amount of compression possible is equivalent to the time-bandwidth product (BT) of the code, which is the product of the signal bandwidth and signal total duration. Bandwidth of a phase-coded signal is calculated via  $B=1/\tau$  where  $\tau$  is taken to be the code sub pulse length. The returned signal power increase is proportional to the code length while the range resolution is inversely related to bandwidth. This implies that decreasing sub pulse duration results in a corresponding enhancement in range resolution.

$$\Delta R = c \frac{\tau}{2} = c/2B$$

The weakness of such systems is in the creation of range sidelobes which are artifacts produced by the compression process whereby returns from other ranges contaminate the signal at the desired range. The resulting output can cause erroneous estimations of reflectivity, mean velocity, and spectral width. Figure 1.2 shows the decoded output for the waveform shown in Figure 1.1 if it were passed through a matched filter. In particular, this is one example of a set of codes known as Barker codes which have uniformly distributed sidelobes about the mainlobe [46]. Barker codes also have the property of producing mainlobes that are higher than the sidelobes by a factor of the code length. In this case, the code length is 5 bits so the mainlobe is 5 times higher than the sidelobes [42].



**Figure 1.2: Compressed Phase-Coded Waveform through a Matched Filter**

## 1.2 RADAR

RADAR is an acronym for Radio Detection and Ranging. It is a system that uses radio waves to locate a moving or fixed object and primarily determine its range, bearing and height. It transmits radio waves that are reflected back from the target and are detected by a receiver.

### Components of Radar

Usually, radar equipments are pulse modulated. Pulse modulated radar consists of the following equipments:

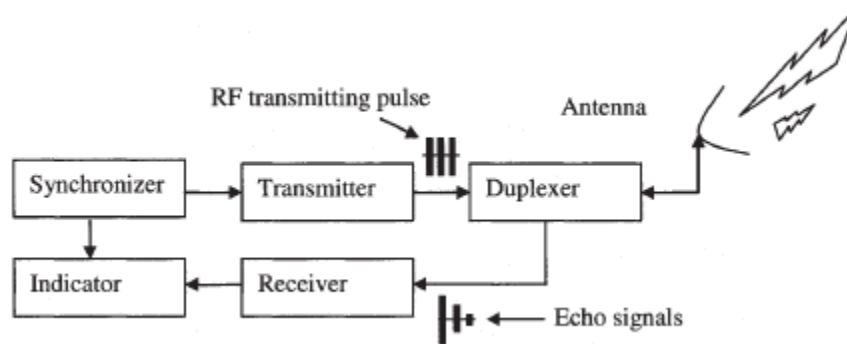


Figure 1.3: Radar block diagram

- **Transmitter**

Transmitter is responsible for producing short-duration, high power pulses of radio frequency energy at given periodic frequency. An oscillator produces the pulses and a pulser provides the repetition frequency. The pulse duration is of the order of 0.1 and 50 microseconds. During each pulse duration, the transmitter produces a very high peak output of 1 MW or more. Transmitter devices used are:

- I. Magnetron
- II. Crossed field amplifier
- III. Klystron
- IV. Travelling wave tube
- V. Solid state amplifier

- **Duplexer**

In pulse modulated radars, transmitter and receiver share a common aerial. As the transmitter and receiver work mutually exclusively, it creates no problem. Basically, duplexer is an electronics transmit-receive switch.

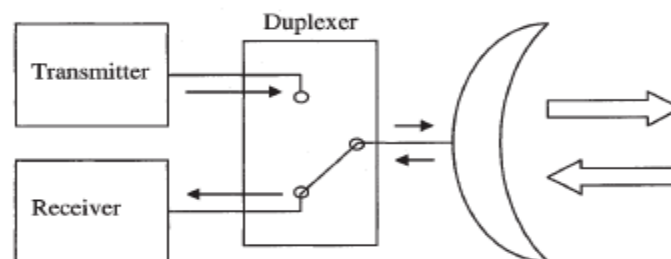
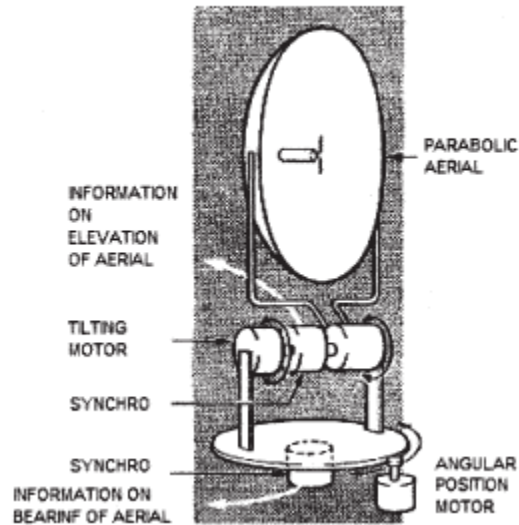


Figure 1.4: Duplexer

- **Aerial system**

In order that the high peak power output from the transmitter focuses on a small region of space, the aerial system produces a very narrow beam of RF energy to scan and locate an object. This is done to obtain accurate bearings in both azimuth and elevation.



**Figure 1.5: Parabolic Antenna**

- **Receiver**

This is used to amplify those reflected pulses which are noisy and weak that is of the order of few micro-volts, which might not be sufficient to display on the indicator. In order to design a receiver with a low noise factor, the receiver has to accept very narrow pulses.

- **Indicator**

This is usually a system of two cathode ray tubes (CRT) used to display an image of the receiver input. One CRT shows range and bearing while the other shows range and height.

Radars in general fall into two categories, monostatic and bistatic. Monostatic radars have their transmitter and receiver in the same place assembled together at the same location. Bistatic radars are opposite to monostatic ones in that they have their transmitter and receiver away from each other. Either way, by means of radar, the existence or presence of an object is discovered and detected. This is done through the release of waves from the radar transmitter and by the analysis of the returned echo through the receiver. Radar is not only used to determine a target position with respect to a fixed point, a reference, but is also used to calculate the target speed, shape and size, this is done by extraction of information, which usually needs a matched filter [26].

## 1.3 RADAR PULSE COMPRESSION

### 1.3.1 Why RADAR Pulse Compression?

To determine a target's shape and size, radar should have sufficient resolution. Resolution is proportional to the pulse width, so for high resolution applications a short pulse needs to be utilized. The shorter the pulse, the more accurate the range measurement is. Also the maximum range (pulse energy), which is proportional to both peak power and time duration of the pulse, has a serious drawback. The drawback is that when the pulse width gets shorter to improve resolution, the pulse energy is also reduced. Keeping the peak power fixed degrades the range performance. This can be solved easily by increasing the power. But, unfortunately, increasing the peak power creates severe problems in the design of high resolution radars, because the transmitter technological limitations affect peak power more than they affect the average power or the energy of the single pulse.

The sensitivity of radar depends on the energy transmitted in the radar pulses. This can be expressed in terms of the average transmitted power, i.e., the peak power multiplied by the transmitter duty cycle. Although the peak transmitter power may be as high as several hundred kilowatts, since most radars transmit very short pulses, the average transmitted power may be much less than 1% of this value. Clearly this is not an efficient use of the available transmitter power.

Without the use of pulse compression, pulse widths cannot be reduced indefinitely. Extremely narrow pulse widths result in wide receiver bandwidths and the associated problems with noise. Large receiver bandwidths effectively de-sensitize the radar receiver and either force the transmitter to transmit higher levels of peak power to compensate, or accept the consequential reduction in range. There are always limits on the amount of peak power available from the transmitter, as high power transmitters suffer from the following problems:

1. They need high voltage power supply of the order of kilowatts (kW).
2. They face the reliability problems like cooling problems and other thermal issues.
3. Safety issues always arise from both electrocution and irradiation of these equipments.
4. They are huge in size, weigh more and are obviously very expensive.

Invariably, a reduction in pulse width leads to a reduction in the maximum range of the radar. In short, narrow pulse widths are desirable, but they are not always feasible. Pulse compression radars use specific signal processing techniques to provide most of the advantages of extremely narrow pulses widths whilst remaining within the peak power limitations of the transmitter. The advantages of narrow pulses enjoyed by pulse compression radar are superior range resolution and range accuracy.



### 1.3.2 The Concept of RADAR Pulse Compression

Radar pulse compression, also known as pulse coding, is a signal processing technique designed to maximize the sensitivity and resolution of radar systems.

Radar pulse compression refers to a family of techniques used to increase the bandwidth of radar pulses. In the radar receiver, these echo pulses are 'compressed' in the time domain, resulting in a range resolution which is finer than that associated with an uncoded pulse.

A pulse compression radar transmits a long pulse with pulse width  $T$  and peak power  $P_t$ , which is coded using frequency or phase modulation to achieve a bandwidth  $B$ , that is large as compared to that of an uncoded pulse with the same duration. The transmit pulse width is chosen to achieve the single pulse transmit energy, given by  $E_{t1} = P_t T$ , that is required for target detection or tracking. The received echo is processed using a pulse compression filter to yield a narrow compressed pulse response with a mainlobe width of  $1/B$ , that doesn't depend on the duration of the transmitted pulse.

The ratio of the transmit pulse width to the compressed main lobe width is defined as the pulse compression ratio. The pulse compression is approximately  $TB$ , where  $TB$  is defined as the time bandwidth product of the waveform. Typically, the pulse compression ratio and the time bandwidth product are large as compared to unity [37].

In simple words, energy content of long-duration, low-power pulse will be comparable to that of the short-duration, high-power pulse [24].

$$P_1 \tau_1 \approx P_2 \tau_2$$

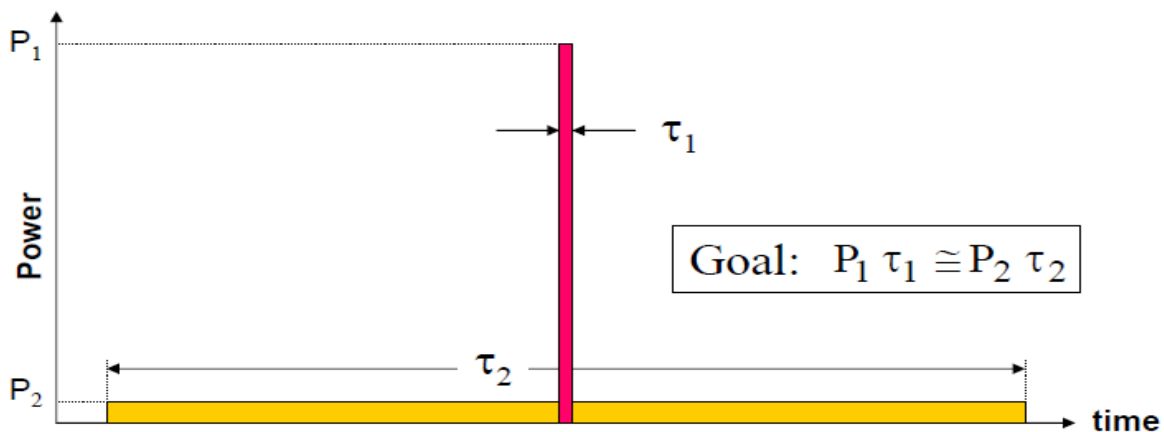


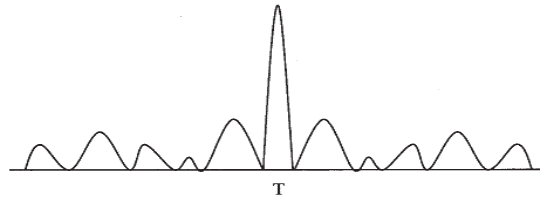
Figure 1.6: Transmitter and Receiver ultimate signals

## PULSE COMPRESSION WAVEFORM TYPES

Out of the various ways to code a long duration pulse to the desired bandwidth, the following waveforms are more frequently used:

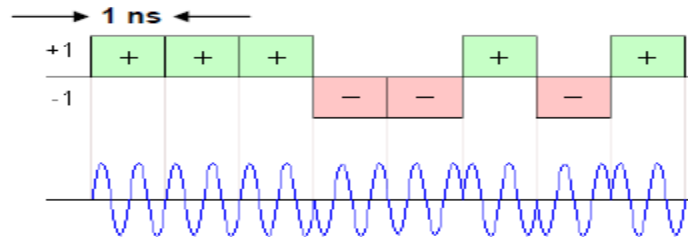
- **Phase Coded Waveform**

The phase-coded waveform divides the pulse into sub pulses of equal duration, each having a certain phase. The code sequence selects the phase of each sub pulse as shown in Figure 1.7.



**Figure 1.7: Phase-modulated waveform after Compression**

Another notation assigns a plus to a nominal carrier phase and a minus to a 180° phase-shift. Each segment is assigned unit amplitude and one of the two phases. In Figure 1.8, each segment is of duration 1ns.



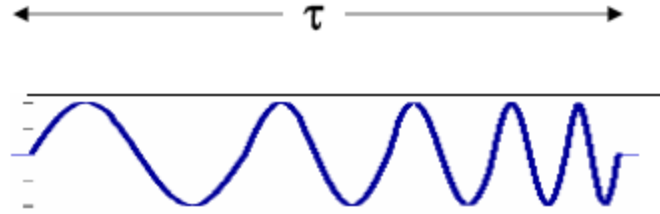
**Figure 1.8: Phase Code Short Segments**

- **Linear Frequency-Modulated Waveform**

The linear frequency modulation, or chirp waveform has a rectangular amplitude modulation with pulse width  $\tau$  and a linear frequency modulation with a swept bandwidth  $B$  applied over the pulse.

$$S(t) = A \cos(2\pi f_c t + 0.5 k t^2 + \phi_c) \text{ for } 0 \leq t \leq \tau$$

Here,  $f_c$  is starting frequency (Hz),  $k$  is the chirp rate (Hz/s) and  $B = k\tau = 1$  GHz.



**Figure 1.9: Chirp**

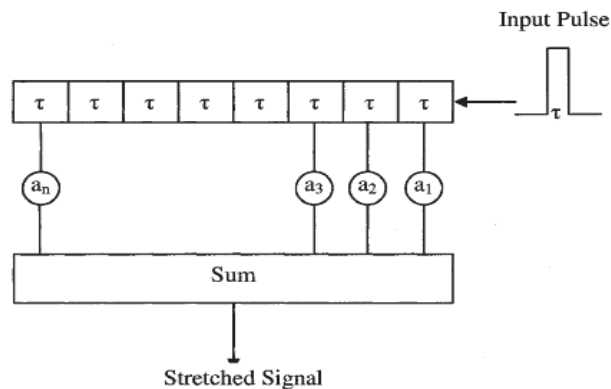
Choice is driven largely by required complexity of receiver electronics [24].

### 1.3.3 Matched filter

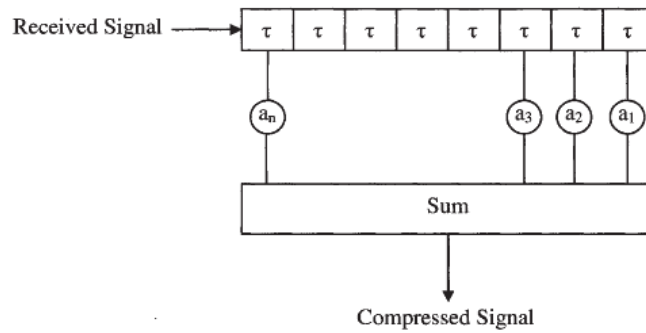
Matched filter provides optimum SNR when one is trying to detect a signal in white (Gaussian) noise. The matched filter is composed of delay elements, multipliers, adders and coefficients,  $a_i$ . The matched filters at the transmitter and receiver are conjugates of each other for the expansion and compression processes.

To generate a phase coded signal, a narrow pulse is fed to the matched filter, which is continuously clocked into a delay component whose number of stages is equal to the number of elements in the sequence. Then output of each stage is multiplied by weights,  $a_i$  which is either +1 or -1 according to the coding or reference sequence. The summation circuit provides the output correlation function or stretched pulse.

The received echo is processed by compression or matched filter which is part of the receiver that is specifically designed to maximize the output SNR and to compress then received phase coded signal to a sub-pulse width. The co-efficient of the compression filter is the inverse of the received signal [26].



**Figure 1.10 (a): Matched filter at the transmitter side**

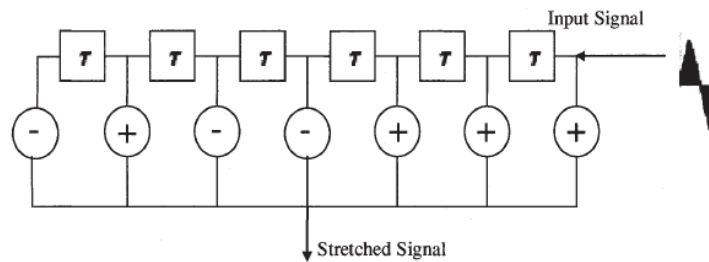


**Figure 1.10 (b): Matched filter at the receiver side**

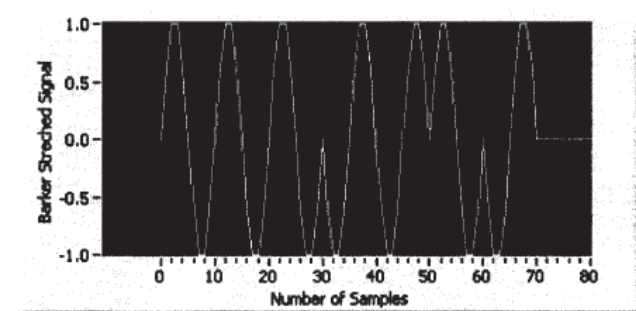
### 1.3.4 Barker Code

A special class of binary code is the Barker code. The peak of the autocorrelation function is  $N$ , and the minimum peak side lobe magnitude is 1, where  $N$  is the number of sub pulses or length of the code [26].

The process of stretching and then compressing a Barker code of length 7 using matched filters at the transmitter and the receiver is shown in Figure 1.11.



**Figure 1.11 (a): Matched filter coefficient of a Barker code of length 7 at the transmitter side**



**Figure 1.11 (b): Stretched signal**

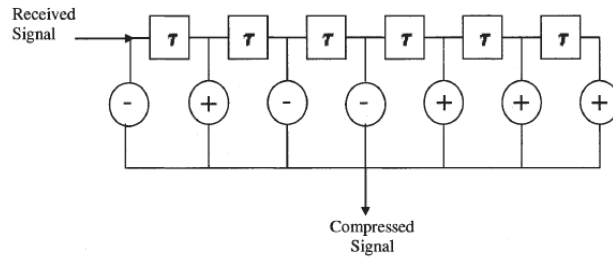


Figure 1.11 (c): Matched filter coefficient of a Barker code of length 7 at the receiver side

	1	1	1	-1	-1	1	-1						
-1	-	-	-	+	+	-	+						
1		+	+	+	-	-	+	-					
-1			-	-	-	+	+	-	+				
-1				-	-	-	+	+	-	+			
1					+	+	+	-	-	+	-		
1						+	+	+	-	-	+	-	
1							+	+	+	-	-	+	-
	-1	0	-1	0	-1	0	7	0	-1	0	-1	0	-1

Figure 1.11 (d): Matched filter processing a received signal

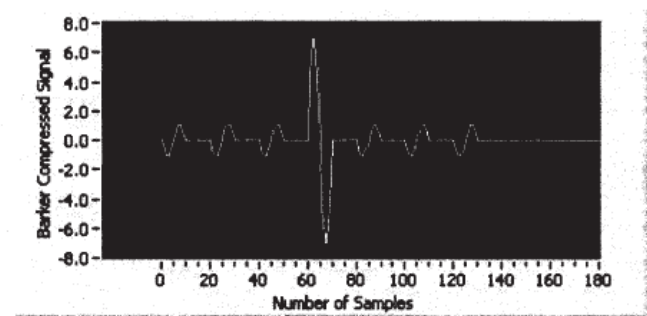
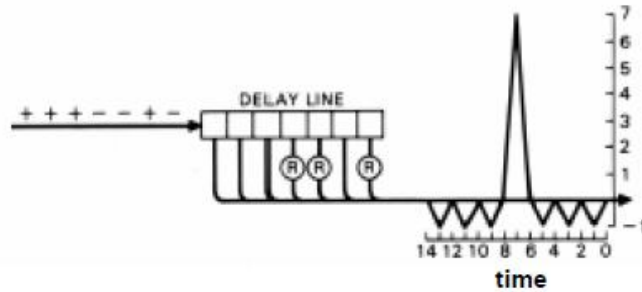


Figure 1.11 (e): Compressed Signal

For Barker Codes,

$$PSL = -20 \log (SL/P) = -20 \log (N)$$



**Figure 1.12: Binary Phase decoder for Pulse Compression**

Correlation process may be performed in analog or digital domain. A disadvantage of this approach is that the data acquisition system (A/D converter) must operate at the full system bandwidth.

Other codes and filters used in case of RADAR Pulse Compression are:

- ***Ternary***  
They include 0, which corresponds to the absence of a segment, in addition to +1 and -1 in Barker Codes.
- ***Combined Barker Code***  
It uses whatever codes are available; then it modulates the transmitted pulse at multiple levels so that each segment of the code is again coded with another Barker code.
- ***Complementary Sequences***  
They consist of two sequences or codes of the same length  $N$  whose autocorrelation functions are added together resulting in an output with a peak value of  $2N$  with no sidelobes.
- ***Mismatched Filter***  
It is used to suppress the sidelobes. Its coefficients are real numbers unlike matched filter, so multipliers are used instead of adders and subtractors. It has greater length than the matched filter to get acceptable results. The greater the length, the higher is the compression ratio.
- ***Inverse Filter***  
It is used for minimizing the Integrated Sidelobe Level (ISL), which compares the total power contained within the sidelobes to the mainlobe. It can be implemented directly on the output of the matched filter.
- ***Nonmatched Filter***  
It has been demonstrated that a weighted network can be designed to reduce sidelobes to an arbitrary low level. The matched filter is designed by a sub pulse filter whose impulse response resembles the code sub pulse, and a correlator that is achieved as a tapped delay line whose weights are matched to the coded pulse phases.

- ***Two-Sample Sliding Window Adder TSSWA***

It reduces the side lobe for the polyphase codes. TSSWA compresses the pulse not to the width of a single sub pulse but to that of several sub pulses by reducing the bandwidth. It is added after the auto-correlator of the binary code.

The thesis is organized as follows:

Chapter 2 first deals with Adaptive Filters and the LMS, RLS and the modified RLS algorithms. Further, MATLAB simulations regarding *convergence analysis* of LMS and RLS algorithms with respect to System Identification are performed. This is followed by *Comparison* of the three algorithms on the basis of PSR levels obtained by using Barker codes as inputs to Matched Filter.

Chapter 3 deals with Artificial Neural Networks. The virtues and limitations of ANN, different types of Neuron Models, Network Architecture, Perceptron Learning Rule, Implementation of Function Approximation using ANN, the Back Propagation Algorithm and its application in training different MLPs to determine the optimum values of weights and biases for Radar Pulse Compression are described in detail. Finally, Tables and MATLAB simulations corresponding to *Convergence Performance*, *SSR Performance*, *Noise Performance* and *Doppler Tolerance* are provided for comparison of different MLPs containing 1 and 2 hidden layers of neurons.

Chapter 4 consists of Conclusion and Future Work.

Finally, all possible References have been listed in Bibliography.

# Chapter 2

## Adaptive Filters and their application to RADAR Pulse Compression



## 2.1 ADAPTIVE SYSTEMS

An adaptive automation is a system whose structure is alterable or adjustable in such a way that its behavior or performance improves through contact with its environment. The objective of this circuit is to adjust the sensitivity of the receiver as the average incoming signal strength. The receiver is thus able to adapt to a wide range of input levels and to produce a much narrower range of output intensities.

The general characteristics of adaptive systems are:

1. They can automatically adapt to the changing environments and changing system requirements.
2. They can be trained to perform specific filtering and decision-making tasks.
3. Due to the above reasons, they do not need the elaborate synthesis procedures unlike nonadaptive systems.
4. They can extrapolate a behavioral model to deal with new situations after having been trained on a finite and often small number of training signals and patterns.
5. They can repair themselves to a limited extent, i.e., they can adapt around certain kinds of internal defects.
6. They are structurally nonlinear systems with time-varying parameters.
7. They are usually more complex in architecture and are difficult to analyze than nonadaptive systems, but they offer substantially increased system performance with unknown or time varying input signal characteristics.

### 2.1.1 OPEN- AND CLOSED-LOOP ADAPTATION

The *open-loop* adaptive process involves making measurements of input or environmental characteristics, applying this information to a formula or a computational algorithm, and using the results to set the adjustments of the adaptive system.

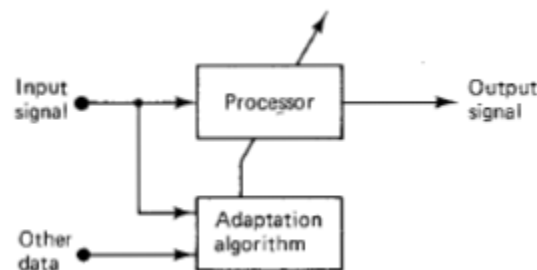
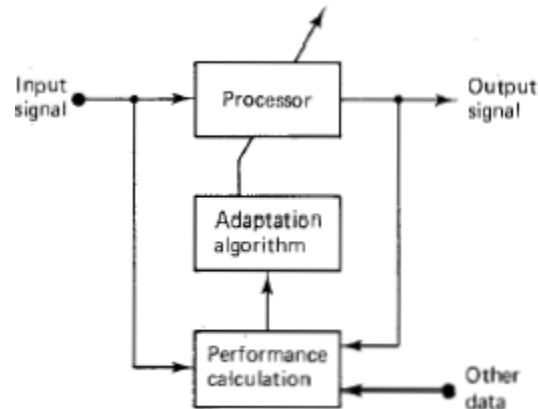


Figure 2.1: Open Loop Adaptive System

*Closed-loop* adaptation, on the other hand, involves automatic experimentation with these adjustments and knowledge of their outcome in order to optimize a measured system performance. The latter process may be called adaptation by “performance feedback”.

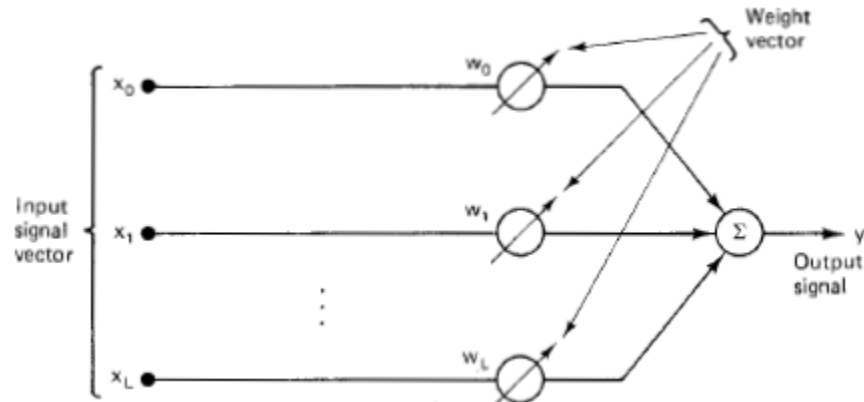


**Figure 2.2: Closed Loop Adaptive System**

## 2.2 THE ADAPTIVE LINEAR COMBINER

The adaptive linear combiner, or nonrecursive adaptive filter, is fundamental to adaptive signal processing and is the single most important element in “learning” systems and adaptive processes in general.

A general diagram of the adaptive linear combiner is shown in Figure 2.3.



**Figure 2.3: General Form of Adaptive Linear Combiner**

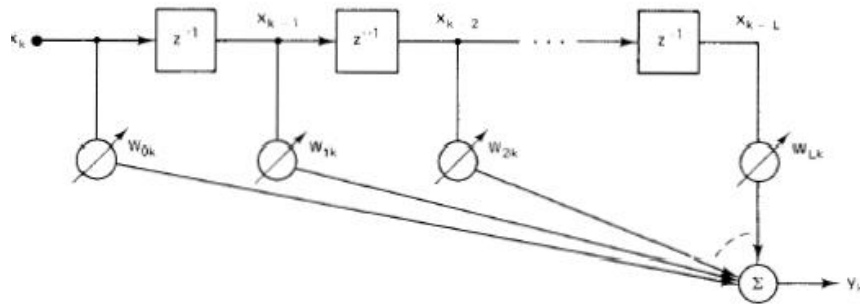
It consists of an input signal vector with elements  $x_0, x_1, \dots, x_L$ , a corresponding set of adjustable weights  $w_0, w_1, \dots, w_L$ , a summing unit, and a single output signal,  $y$ . The procedure for adjusting or adapting the weights is called a “weight adjustment”, “gain adjustment” or “adaptation procedure”. The combiner is called “linear” because for a fixed setting of the weights, its output is a linear combination of the input components.

### 2.2.1 INPUT SIGNAL AND WEIGHT VECTORS

The inputs may be obtained simultaneously from  $L+1$  different signal sources or they could be  $L+1$  sequential samples from the same signal source and their corresponding outputs are computed.

*Single Input: :*  $\mathbf{X}_k = [x_k, x_{k-1}, \dots, x_{k-L}]^T$

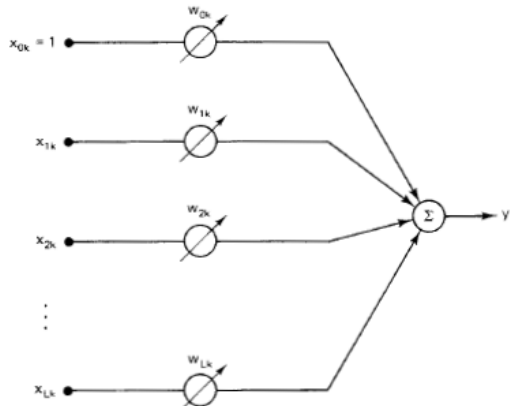
$$y_k = \sum_{l=0}^L w_{lk} x_{k-l}$$



**Figure 2.4: Adaptive Linear Combiner in the form of Single-input adaptive transversal filter**

*Multiple Inputs: :*  $\mathbf{X}_k = [x_{0k}, x_{1k}, \dots, x_{Lk}]^T$

$$y_k = \sum_{l=0}^L w_{lk} x_{lk}$$



**Figure 2.5: Multiple Input Adaptive Linear Combiner with bias weight  $w_{0k}$**

Correspondingly, there is a weight vector for both single and multiple inputs given by:

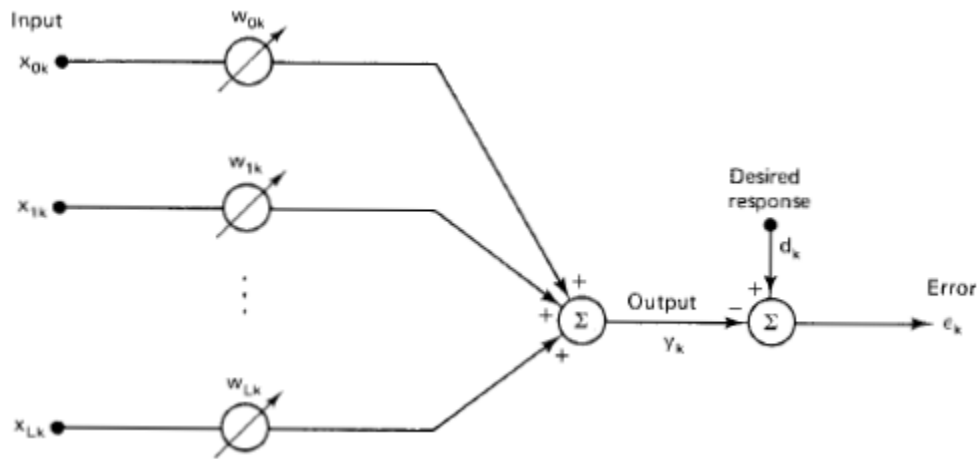
$$\mathbf{W}_k = [w_{0k} \quad w_{1k} \quad \dots \quad w_{Lk}]^T$$

Vectorially,

$$y_k = \mathbf{X}_k^T \mathbf{W}_k = \mathbf{W}_k^T \mathbf{X}_k$$

### 2.2.2 DESIRED RESPONSE AND ERROR

The method of deriving the error signal by means of the desired response input is shown in Figure 2.6.



**Figure 2.6: Multiple Input Adaptive Linear Combiner with desired response and error signals**

The output signal,  $y_k$ , is simply subtracted from the desired signal,  $d_k$ , to produce the error signal  $e_k$ . [10]

### 2.2.3 THE PERFORMANCE FUNCTION

The error signal with time index  $k$  is

$$\varepsilon_k = d_k - y_k$$

The Mean Square Error (MSE) is defined as

$$\xi = E[\varepsilon_k^2] = E[d_k^2] + \mathbf{W}^T \mathbf{R} \mathbf{W} - 2 \mathbf{P}^T \mathbf{W}$$

Assuming  $\mathbf{R}$  to be non-singular, the optimal weight vector  $\mathbf{W}^* = \mathbf{R}^{-1} \mathbf{P}$

The minimum mean square error is now obtained as:

$$\xi_{\min} = E[d_k^2] - \mathbf{P}^T \mathbf{W}^*$$

## 2.3 ADAPTIVE FILTER ALGORITHMS

These are the algorithms which work in nonstationary as well as stationary signal environments and are meant to “track” as well as “seek” the minimum point on the performance surface. The adaptive filter algorithms that have been used in this project include:

### 1. THE LMS ALGORITHM

The LMS algorithm, or Least Mean Squares algorithm, is an algorithm for descending on the performance surface. The LMS algorithm is important because of its simplicity and ease of computation, and because it does not require off-line gradient estimations or repetitions of data. If the adaptive system is an adaptive linear combiner, and if the input vector  $\mathbf{X}_k$  and the desired response  $d_k$  are available at each iteration, the LMS algorithm is usually the best choice for many different applications of adaptive signal processing.

#### DERIVATION OF THE LMS ALGORITHM

The adaptive linear combiner is applied in two basic ways, depending on whether the input is available in parallel (multiple inputs) or serial (single input) form.

In both cases, we have the combiner output,  $y_k$  as a linear combination of the input samples. We have

$$\epsilon_k = d_k - \mathbf{X}_k^T \mathbf{W}_k$$

where  $\mathbf{X}_k$  is the vector of input samples in either of the two configurations.

To develop an adaptive algorithm using the method, we would estimate the gradient of  $\zeta = E(\epsilon_k^2)$  by taking differences between the short term averages of  $\epsilon_k^2$ . Instead, to develop the LMS algorithm, we take  $\epsilon_k^2$  itself as an estimate of  $\epsilon_k$ . Then, at each iteration in the adaptive process, we have a gradient estimate of the form

$$\nabla'_k = -2 \epsilon_k \mathbf{X}_k$$

The derivatives of  $\epsilon_k$  with respect to the weights follow from the error signal equation.

With this simple estimate of the gradient, we can now specify a steepest – descent type of adaptive algorithm. Now, we have

$$\mathbf{W}_{k+1} = \mathbf{W}_k + 2\mu \epsilon_k \mathbf{X}_k$$

This is the LMS algorithm.

$\mu$  is the gain constant that regulates the speed and stability of adaptation. Since the weight changes at each iteration are based on imperfect gradient estimates, we would expect the adaptive process to be noisy; that is, it would not follow the true line of steepest descent on the performance surface.

$$1/\lambda_{\max} > \mu > 0 \text{ where } \lambda_{\max} \text{ is the largest eigen value. [10]}$$

## 2. THE RLS ALGORITHM

The Recursive Least Squares, or the RLS algorithm, is another adaptive filter algorithm which converges to the optimum filter values faster than the LMS algorithm.

**The update formula.** The simplest approach to updating  $\mathbf{W}_k^0$  is the following procedure:

- a) Update  $\mathbf{R}_{ss}$  via

$$\mathbf{R}_{ss,k+1} = \mathbf{R}_{ss,k} + \mathbf{X}(k) \mathbf{X}^t(k)$$

- b) Update  $\mathbf{P}_{ss}$  via

$$\mathbf{P}_{ss,k+1} = \mathbf{P}_{ss,k} - d(k) \mathbf{X}(k)$$

- c) Invert  $\mathbf{R}_{ss,k+1}$

- d) Compute  $\mathbf{W}_{k+1}^0$  via

$$\mathbf{W}_{k+1}^0 = \mathbf{R}_{ss,k+1}^{-1} \mathbf{P}_{ss,k+1}$$

The autocorrelation matrix and the cross correlation are updated and then used to compute  $\mathbf{W}_{k+1}^0$ . While direct, this technique is computationally wasteful. Approximately  $N^3 + 2N^2 + N$  multiplications is required at each update where  $N$  is the impulse response length, and of that  $N^3$  are required for the matrix inversion if done with the classical Gaussian elimination technique.

**The RLS algorithm.** We can write down step by step procedure for updating  $\mathbf{W}_k^0$ . This set of steps is efficient in the sense that no unneeded variable is computed and that no needed variable is computed twice. We do, however, need assurance that  $R_k^{-1}$  exists. The procedure then goes as follows:

- i. Accept new samples  $x(k)$ ,  $d(k)$ .
- ii. Form  $\mathbf{X}(k)$  by shifting  $x(k)$  into the information vector.
- iii. Compute the a priori output  $y_0(k)$ :  

$$y_0(k) = \mathbf{W}_k^{0t} \mathbf{X}(k).$$
- iv. Compute the a priori error  $e_0(k)$ :  

$$e_0(k) = d(k) - y_0(k).$$
- v. Compute the filtered information vector  $\mathbf{Z}_k$ :  

$$\mathbf{Z}_k = R_k^{-1} \mathbf{X}(k).$$
- vi. Compute the normalized error power  $q$ :  

$$q = \mathbf{X}^t(k) \mathbf{Z}_k.$$
- vii. Compute the gain constant  $v$ :  

$$v = 1 / (1 + q).$$
- viii. Compute the normalized filtered information vector  $\mathbf{Z}_k'$ :  

$$\mathbf{Z}_k' = v \cdot \mathbf{Z}_k.$$
- ix. Update the optimal weight vector  $\mathbf{W}_k^0$  to  $\mathbf{W}_{k+1}^0$ :  

$$\mathbf{W}_{k+1}^0 = \mathbf{W}_k^0 + e_0(k) \mathbf{Z}_k'.$$
- x. Update the inverse correlation matrix  $R_k^{-1}$  to  $R_{k+1}^{-1}$  in preparation for the next iteration:  

$$R_{k+1}^{-1} = R_k^{-1} - \mathbf{Z}_k' \mathbf{Z}_k^t.$$

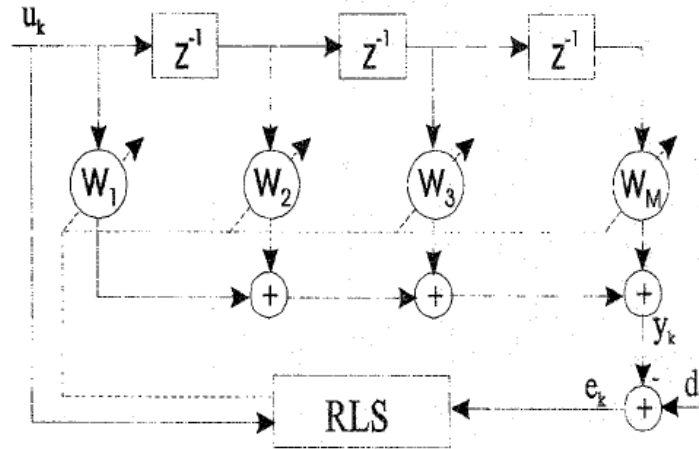
In most of the cases, a much simpler approach for  $R_k^{-1}$  is used.

$$R_{N-1}^{-1} = \eta \mathbf{I}_N$$

where  $\eta$  is a large positive constant and  $\mathbf{I}_N$  is the  $N$  by  $N$  identity matrix. Since  $R_{k+1}^{-1}$  almost certainly will not equal, this inaccuracy will influence the final estimate of  $R_k$  and  $\mathbf{W}_k$ . [4]

### 3. THE MODIFIED RLS ALGORITHM

In this design, there has been used an analogy with application of the RLS algorithm as an adaptation mechanism in transversal structure adaptive filters implementation, one such filter being presented in Figure 2.7.



**Figure 2.7: Adaptive transversal filter structure**

*It should be noted that the obtained sidelobe suppression filter is not adaptive, but just in its coefficients evaluation; an approach known from adaptive filter synthesis is being utilized.[25]*

Utilizing the standard RLS algorithm in a stationary scenario, an optimal filter is obtained in the sense of the mean square sidelobes level suppression. In order to improve the filter performance in the sense of the peak sidelobe values suppression, a modification of the standard RLS procedure has been performed by introducing a criterion:

$$|e_k| \geq TH$$

where TH represents the *threshold value* to which the instantaneous error value is being compared. If the error is greater than or equal to the threshold value, a correction of the estimated filter coefficients vector  $\mathbf{W}_k$  is being performed. If the error value is under the threshold, the correction is not being performed.

The Threshold is calculated as:

$$TH = \delta \text{ MAX\_ERR}_j, \text{ where}$$

$\text{MAX\_ERR}_j = \max(\text{err})$  and the constant  $\delta$  has a value close to 1 and it affects the convergence rate. [25]



## 2.4 SYSTEM IDENTIFICATION

**SYSTEM:** A collection of components which are coordinated together to perform a function. A system is a defined part of the real world. Interactions with the environment are described by inputs, outputs and disturbances.

The process of *System Identification* involves designing a filter by generating its coefficients using an algorithm (in this project, only adaptive filter algorithms are used) based on the given input and the desired output. [27]

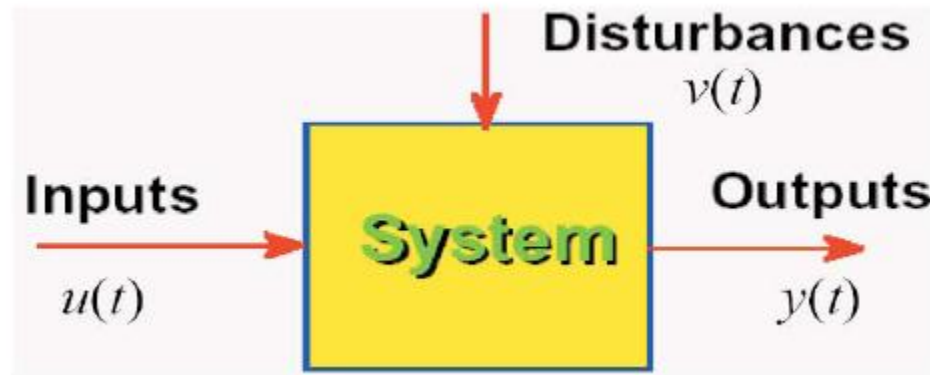
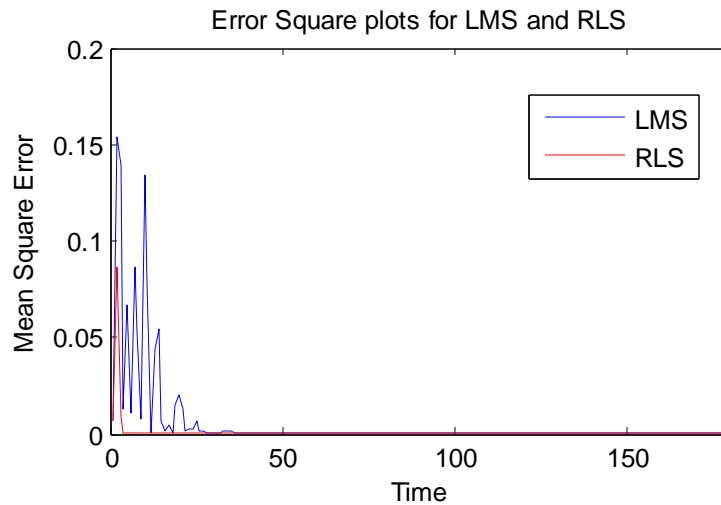


Figure 2.8: Block Diagram for System Identification

### 2.4.1 CONVERGENCE RATE COMPARISON OF THE LMS AND RLS ALGORITHMS IN SYSTEM IDENTIFICATION



**Figure 2.9: Simulation demonstrating faster convergence of RLS over LMS**

It can be clearly pointed out that the convergence rate of RLS algorithm is much higher than that of the LMS algorithm. System Identification, i.e., the convergence of the weight parameters of the system to optimum values or coefficients for appropriate input-output mapping, is achieved much more quickly in case of RLS than that of LMS.

## 2.5 RADAR PULSE COMPRESSION using ADAPTIVE FILTER ALGORITHMS

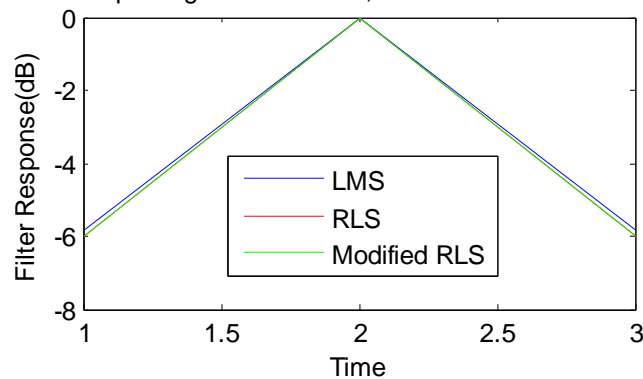
The algorithms are executed repeatedly for *1000 iterations*. The weights of all the layers are initialized to random values between  $\pm 0.1$  and the value of  $\mu$  is taken as 0.01.  $\delta$  for Modified RLS algorithm is taken to be 0.995.

**TABLE 2.1:**

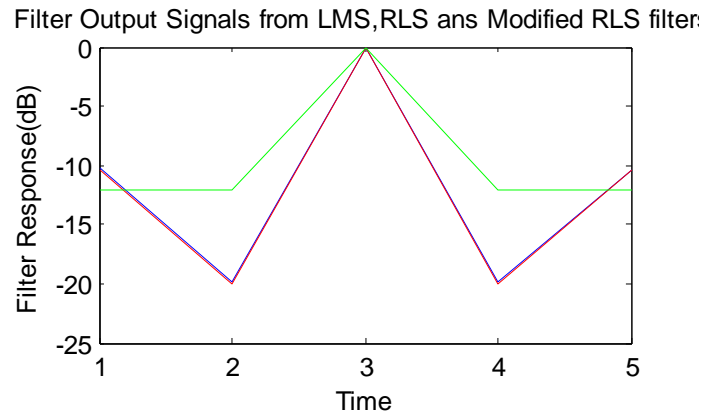
**Filter Comparison Parameters [Peak Side Lobe (PSR)] for the LMS, RLS and modified RLS method evaluation for Barker Codes of different lengths with filter lengths taken same as that of the code**

Length of Barker Code	Filter Length	PSR (dB) using LMS algorithm	PSR (dB) using RLS algorithm	PSR (dB) using modified RLS algorithm
2	2	-5.8451	-6.0134	-6.0149
3	3	-10.2940	-10.4536	-12.0224
4	4	-9.3423	-9.5414	-12.0169
5	5	-16.0997	-16.2561	-18.0383
7	7	-16.7238	-17.1423	-18.0397
11	11	-19.2555	-19.7760	-21.7645
13	13	-23.8592	-24.0037	-25.7404

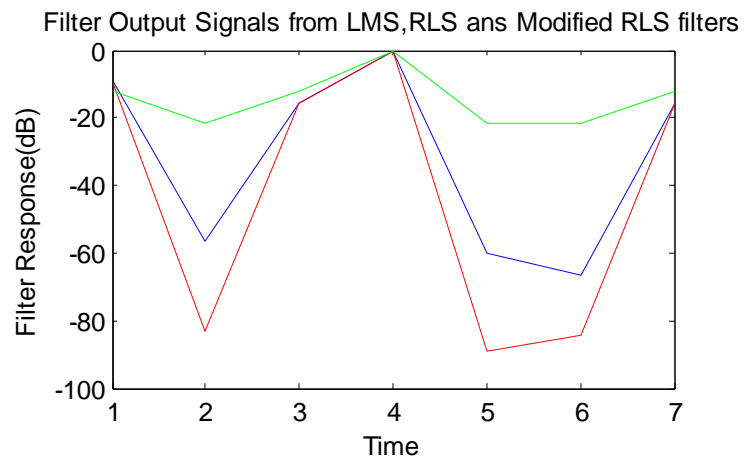
Filter Output Signals from LMS,RLS ans Modified RLS filters



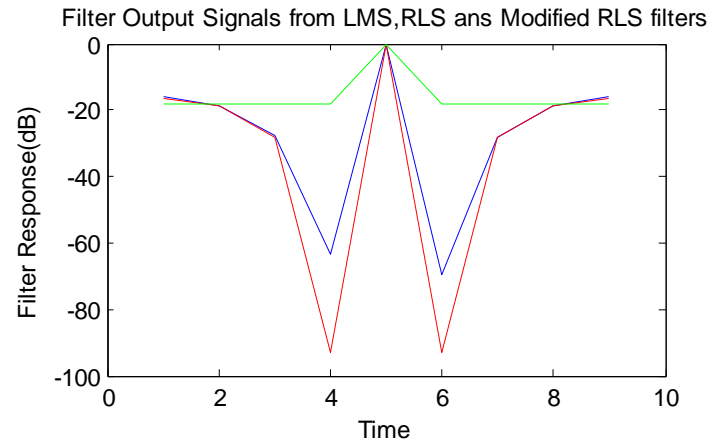
**Figure 2.10: Filter Output Signals for Barker Code of Length 2**



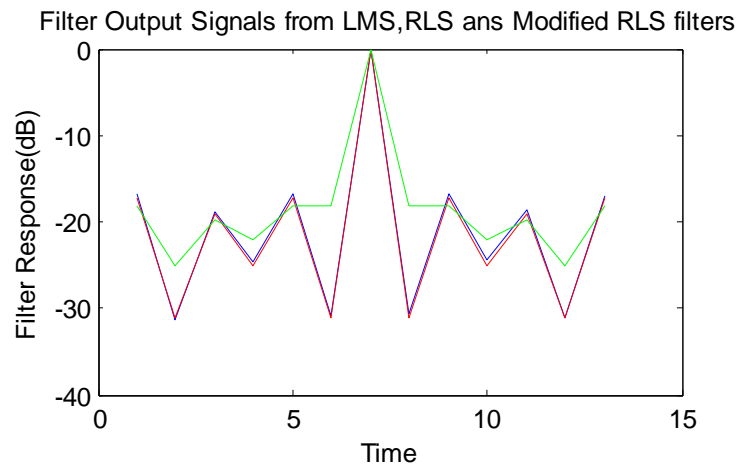
**Figure 2.11: Filter Output Signals for Barker Code of Length 3**



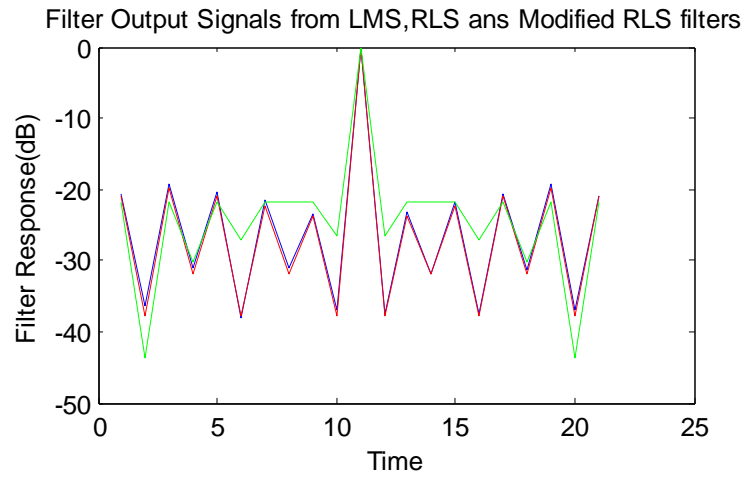
**Figure 2.12: Filter Output Signals for Barker Code of Length 4**



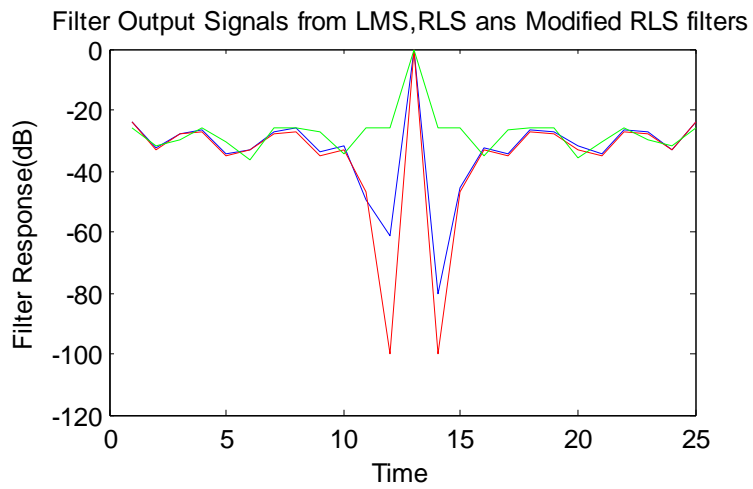
**Figure 2.13: Filter Output Signals for Barker Code of Length 5**



**Figure 2.14: Filter Output Signals for Barker Code of Length 7**



**Figure 2.15: Filter Output Signals for Barker Code of Length 11**



**Figure 2.16: Filter Output Signals for Barker Code of Length 13**

# Chapter 3

## Artificial Neural Networks and their application to RADAR Pulse Compression

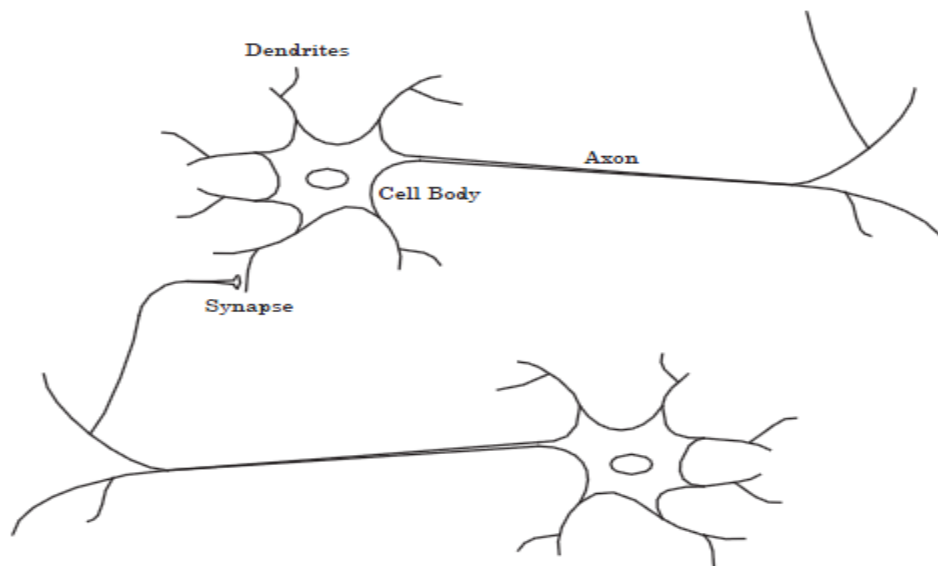
## 3.1 BIOLOGICAL INSPIRATION

The artificial neural networks are only remotely related to their biological counterparts. Here, we describe only those characteristics of brain function that have inspired the development of artificial neural networks.

The brain consists of a large number (approximately  $10^{11}$ ) of highly connected elements (approximately  $10^4$  connections per element) called neurons. These neurons have three principal components: the *dendrites*, the *cell body* and the *axon*.

- The *dendrites* are tree-like receptive networks of nerve fibers that carry electrical signals into the *cell body*.
- The *cell body* effectively sums and thresholds these incoming signals.
- The *axon* is a single long fiber that carries the signal from the cell body out to other neurons.
- The point of contact between an axon of one cell and a dendrite of another cell is called a *synapse*.

It is the arrangement of neurons and the strengths of the individual synapses, determined by a complex chemical process that establishes the function of the neural network.



**Figure 3.1: Schematic Diagram of Biological Neurons**

Some of the neural structure is defined at birth. Other parts are developed through learning, as new connections are made and others waste away. [29]



## 3.2 DEFINITION and PROPERTIES

*A neural network is a massively parallel distributed processor made up of simple processing units which has a natural tendency for storing experiential knowledge and making it available for use. It can be likened to the brain in two aspects:*

- 1. Knowledge is acquired by the network from its environment through a learning process.*
- 2. Interneuron connection strengths known as synaptic neurons are used to store the acquired knowledge. [28]*

Following are the useful *properties* and *capabilities* of neural networks:

### **1. Non-linearity**

A neural network, made up of an interconnection of nonlinear neurons, is itself nonlinear. Moreover, the nonlinearity is of a special kind in the sense that it is *distributed* throughout the network.

### **2. Input-Output Mapping**

The network is presented with a random example and the synaptic weights are modified to minimize the difference between the desired response and the actual response of the network. The training of the network is repeated for many examples till there are no further significant changes in the synaptic weights.

### **3. Adaptivity**

When a neural network is trained to operate in a specific environment, it can be easily retrained to deal with minor changes in the operating conditions. When it operates in a non-stationary environment, a neural network may be designed to change its synaptic weights in real time.

### **4. Evidential response**

A neural network not only helps in *pattern classification* but also provides information about the *confidence* in the decision made which can be used to reject ambiguous patterns.

### **5. Contextual information**

The structure and activation state of a neural network is representative of the knowledge contained in it. Each neuron in the network is potentially affected by the activity of its neighboring neurons.

### **6. Fault tolerance**

Damage of a neuron or its connecting links impair the quality of recall of a stored pattern. However, a neural network exhibits a graceful degradation in performance rather than catastrophic failure.

### 7. *VLSI implementability*

A neural network, being a massively parallel structure, can make fast computations. VLSI provides a means of capturing this behavior in a hierarchical fashion.

### 8. *Uniformity of analysis and design*

Neural networks enjoy universality as information processors, so the same notation is used in all domains involving the application of neural networks.

### 9. *Neuro-biological analogy*

The design of a neural network is motivated by analogy with the brain which is living proof that fault tolerant parallel processing is not only physically possible but also fast and powerful.

## 3.3 NEURON MODEL

### 3.3.1 Single-Input Neuron

A single-input neuron is shown in Figure 3.2. The scalar input  $p$  is multiplied by the scalar *weight*  $w$  to form  $wp$ , one of the terms that is sent to the summer. The other input, 1, is multiplied by a *bias*  $b$  and then passed to the summer. The summer output, often referred to as the *net input*, goes into a *transfer function*, which produces the scalar neuron output.

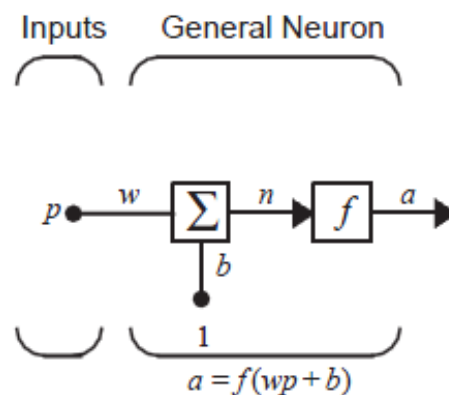











Figure 3.2: Single Input Neuron

Comparing this with the biological neuron discussed previously, the weight  $w$  corresponds to the strength of a synapse, the cell body is represented by the summation and the transfer function, and the neuron output  $a$  represents the signal on the axon.

The neuron output is calculated as:  $a = f(wp + b)$

### 3.3.2 Transfer Functions

The transfer function may be a linear or a nonlinear function of  $n$ . A particular transfer function is chosen to satisfy some specification of the problem that the neuron is attempting to solve. A variety of transfer functions are tabulated below.

Name	Input/Output Relation	Icon	MATLAB Function
Hard Limit	$a = 0 \quad n < 0$ $a = 1 \quad n \geq 0$		hardlim
Symmetrical Hard Limit	$a = -1 \quad n < 0$ $a = +1 \quad n \geq 0$		hardlims
Linear	$a = n$		purelin
Saturating Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \leq n \leq 1$ $a = 1 \quad n > 1$		satlin
Symmetric Saturating Linear	$a = -1 \quad n < -1$ $a = n \quad -1 \leq n \leq 1$ $a = 1 \quad n > 1$		satlins
Log-Sigmoid	$a = \frac{1}{1 + e^{-n}}$		logsig
Hyperbolic Tangent Sigmoid	$a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$		tansig
Positive Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \leq n$		poslin
Competitive	$a = 1 \quad \text{neuron with max } n$ $a = 0 \quad \text{all other neurons}$		compet

### 3.3.3 Multiple-Input Neuron

Typically, a neuron has more than one input. A neuron with  $R$  inputs is shown in Figure 3.3. The individual inputs  $p_1, p_2, \dots, p_R$  are each weighted by corresponding elements  $w_{1,1}, w_{1,2}, \dots, w_{1,R}$  of the *weight matrix*  $\mathbf{W}$ .

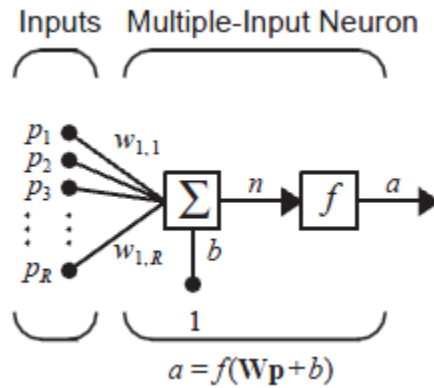


Figure 3.3: Multiple Input Neuron

The neuron has a bias  $b$ , which is summed with the weighted inputs to form the net input  $n$ :

$$n = w_{1,1}p_1 + w_{1,2}p_2 + \dots + w_{1,R}p_R + b$$

This expression can be written in matrix form:

$$n = \mathbf{W}\mathbf{p} + b$$

where the matrix  $\mathbf{W}$  for the single neuron case has only one row.

Now the neuron output can be written as

$$a = f(\mathbf{W}\mathbf{p} + b)$$

A multiple-input neuron using *abbreviated notation* is shown in Figure 3.4.

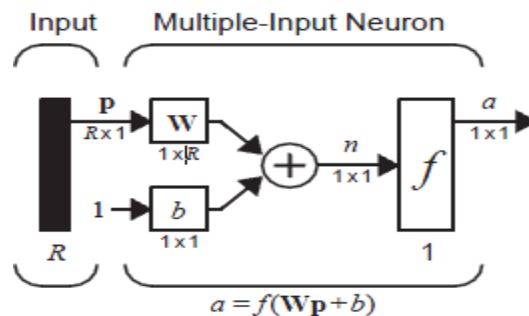


Figure 3.4: Multiple Input Neuron, Abbreviated Notation

## 3.4 NETWORK ARCHITECTURES

Commonly one neuron, even with many inputs, may not be sufficient. We might need five or ten, operating in parallel, in what we will call a “layer”. This concept of a layer is discussed below.

### 3.4.1 A Layer of Neurons

A single-layer network of  $S$  neurons is shown in Figure 3.5. Note that each of the  $R$  inputs is connected to each of the neurons and that the weight matrix now has  $S$  rows.

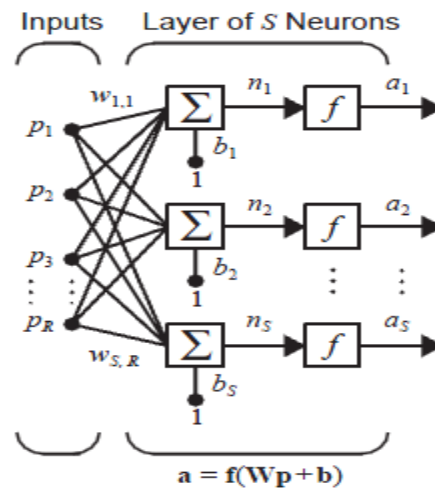


Figure 3.5: Layer of  $S$  Neurons

The layer includes the weight matrix, the summers, the bias vector  $\mathbf{b}$ , the transfer function boxes and the output vector  $\mathbf{a}$ .

Each element of the input vector is connected to each neuron through the weight matrix  $\mathbf{W}$ . Each neuron has a bias  $b_i$ , a summer, a transfer function  $f$  and an output  $a_i$ . Taken together, the outputs form the output vector  $\mathbf{a}$ .

The input vector elements enter the network through the weight matrix  $\mathbf{W}$ :

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,R} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,R} \\ \vdots & \vdots & & \vdots \\ w_{S,1} & w_{S,2} & \cdots & w_{S,R} \end{bmatrix}$$

The  $S$ -neuron,  $R$ -input, one-layer network also can be drawn in *abbreviated notation* in Figure 3.6.

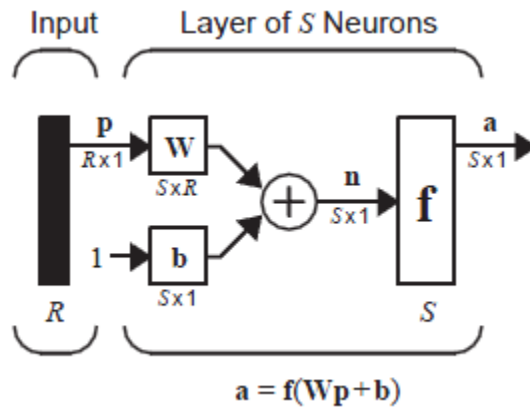


Figure 3.6: Layer of  $S$  Neurons, Abbreviated Notation

### 3.4.2 Multiple Layers of Neurons

In a network with several layers, each layer has its own weight matrix  $\mathbf{W}$ , its own bias vector  $\mathbf{b}$ , a net input vector  $\mathbf{n}$  and an output vector  $\mathbf{a}$ .

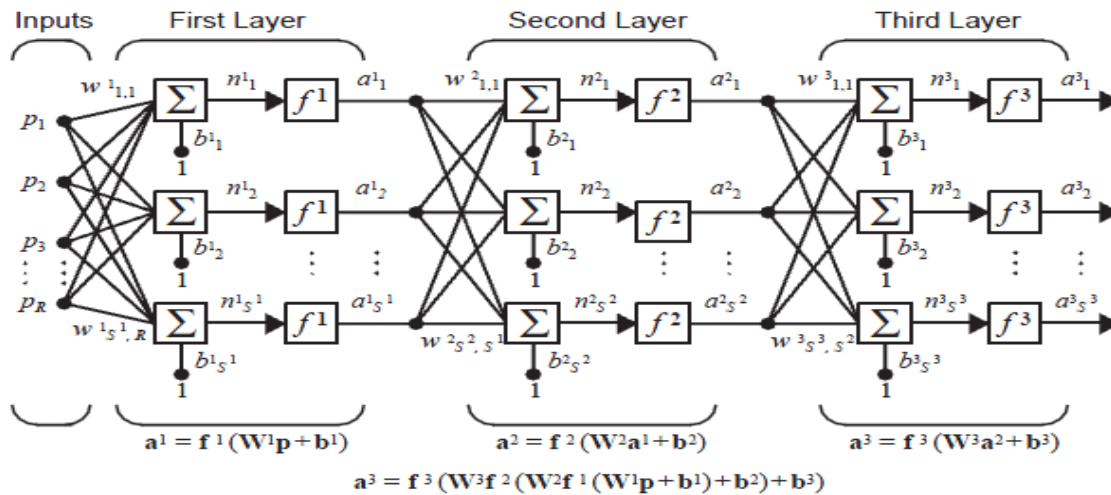


Figure 3.7: Three-Layer Network

As shown in Figure 3.7, there are  $R$  inputs,  $S^1$  neurons in the first layer,  $S^2$  neurons in the second layer, etc. As noted, different layers can have different numbers of neurons.

The outputs of layers one and two are the inputs for layers two and three. Thus layer 2 can be viewed as a one-layer network with  $R = S^1$  inputs,  $S = S^2$  neurons, and an  $S^1 \times S^2$  weight matrix  $\mathbf{W}^2$ . The input to layer 2 is  $\mathbf{a}^1$ , and the output is  $\mathbf{a}^2$ .

A layer whose output is the network output is called an *output layer*. The other layers are called *hidden layers*. The network shown above has an output layer (layer 3) and two hidden layers (layers 1 and 2).

The same three-layer network discussed previously also can be drawn using our abbreviated notation, as shown in Figure 3.8.

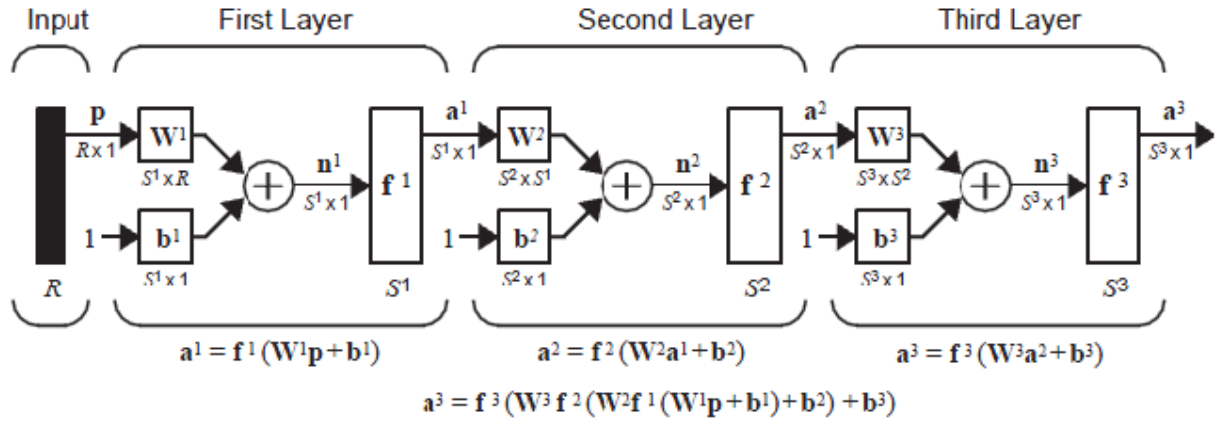


Figure 3.8: Three-Layer Network, Abbreviated Notation

## 3.5 PERCEPTRON LEARNING RULE

### 3.5.1 LEARNING RULES

By *learning rule* we mean a procedure for modifying the weights and biases of a network. The purpose of the learning rule is to train the network to perform some task. There are many types of neural network learning rules. They fall into three broad categories: supervised learning, unsupervised learning and reinforcement (or graded) learning.

In *supervised learning*, the learning rule is provided with a set of examples (the *training set*) of proper network behavior:

$$\{p_1, t_1\}, \{p_2, t_2\}, \dots, \{p_Q, t_Q\}$$

where  $p_q$  is an input to the network and  $t_q$  is the corresponding correct (*target*) output. As the inputs are applied to the network, the network outputs are compared to the targets. The learning rule is then used to adjust the weights and biases of the network in order to move the network outputs closer to the targets.

*Reinforcement learning* is similar to supervised learning, except that, instead of being provided with the correct output for each network input, the algorithm is only given a grade. The grade (or score) is a measure of the network performance over some sequence of inputs.

In *unsupervised learning*, the weights and biases are modified in response to network inputs only. There are no target outputs available. Most of these algorithms perform some kind of clustering operation. They learn to categorize the input patterns into a finite number of classes. [29]

### 3.5.2 PERCEPTRON ARCHITECTURE

The general perceptron network is shown in Figure 3.9.

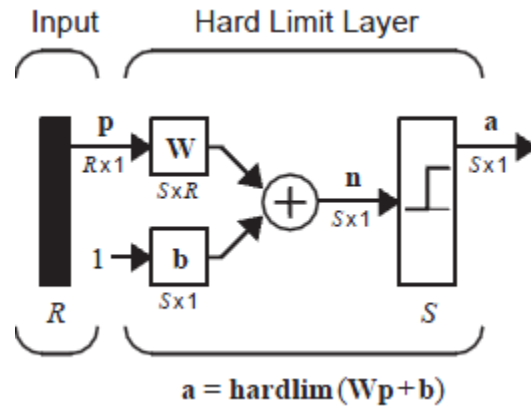


Figure 3.9: Perceptron Network

The output of the network is given by

$$\mathbf{a} = \text{hardlim}(\mathbf{W}\mathbf{p} + \mathbf{b})$$

A vector composed of the elements of the  $i$ th row of  $\mathbf{W}$ :

$${}_i\mathbf{W} = \begin{bmatrix} w_{i,1} \\ w_{i,2} \\ \vdots \\ w_{i,R} \end{bmatrix}$$

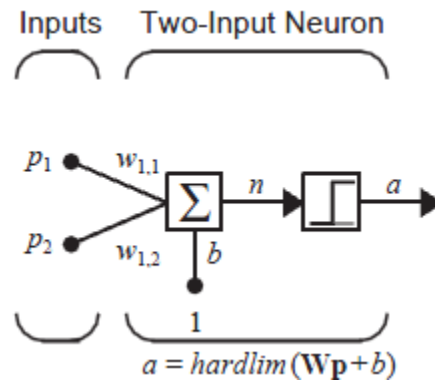
We can partition the weight matrix:



$$\mathbf{W} = \begin{bmatrix} {}_1\mathbf{w}^T \\ {}_2\mathbf{w}^T \\ \vdots \\ {}_S\mathbf{w}^T \end{bmatrix}$$

## Single-Neuron Perceptron

A two-input perceptron with one neuron is shown in Figure 3.10.



**Figure 3.10: Two-Input/Single Output Perceptron**

$$a = \text{hardlim}(n) = \text{hardlim}(\mathbf{W}\mathbf{p} + b) = \text{hardlim}(w_{1,1}p_1 + w_{1,2}p_2 + b)$$

The *decision boundary* is determined by the input vectors for which the net input  $n$  is zero:

$$n = w_{1,1}p_1 + w_{1,2}p_2 + b = 0$$

## Multiple-Neuron Perceptron

Note that for perceptrons with multiple neurons, as in Figure, there will be one decision boundary for each neuron. The decision boundary for neuron will be defined by

$$\mathbf{w}_i^T \mathbf{p} + b_i = 0$$

A single-neuron perceptron can classify input vectors into two categories, since its output can be either 0 or 1. A multiple-neuron perceptron can classify inputs into many categories. Each category is represented by a different output vector. Since each element of the output vector can be either 0 or 1, there are a total of  $2^S$  possible categories, where  $S$  is the number of neurons.

## 3.6 FUNCTION APPROXIMATION

Neural networks can also act as function approximators. In control systems, for example, the objective is to find an appropriate feedback function that maps from measured inputs to control outputs. In adaptive filtering, the objective is to find a function that maps from delayed values of an input signal to an appropriate output signal.

Consider, the two layer, 1-2-1 network shown in Figure 3.11. The transfer function for the first layer is log-sigmoid and the transfer function for the second layer is linear.

$$f^1(n) = 1/(1 + e^{-n}) \text{ and } f^2(n) = n$$

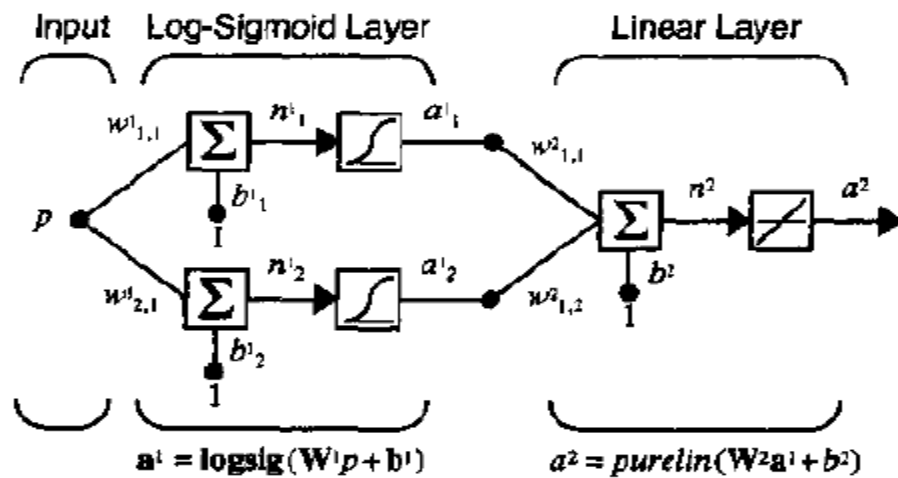


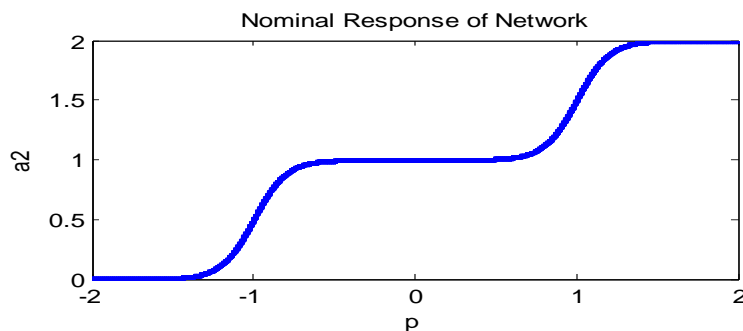
Figure 3.11: Example of Function Approximation Network

Let us take the nominal values of the weights and biases for this network as:

$$w_{1,1}^1 = 10, w_{2,1}^1 = 10, b_1^1 = -10, b_2^1 = -10,$$

$$w_{1,2}^2 = 1, w_{2,2}^2 = 1, b^2 = 0$$

The network response for the above parameters is:



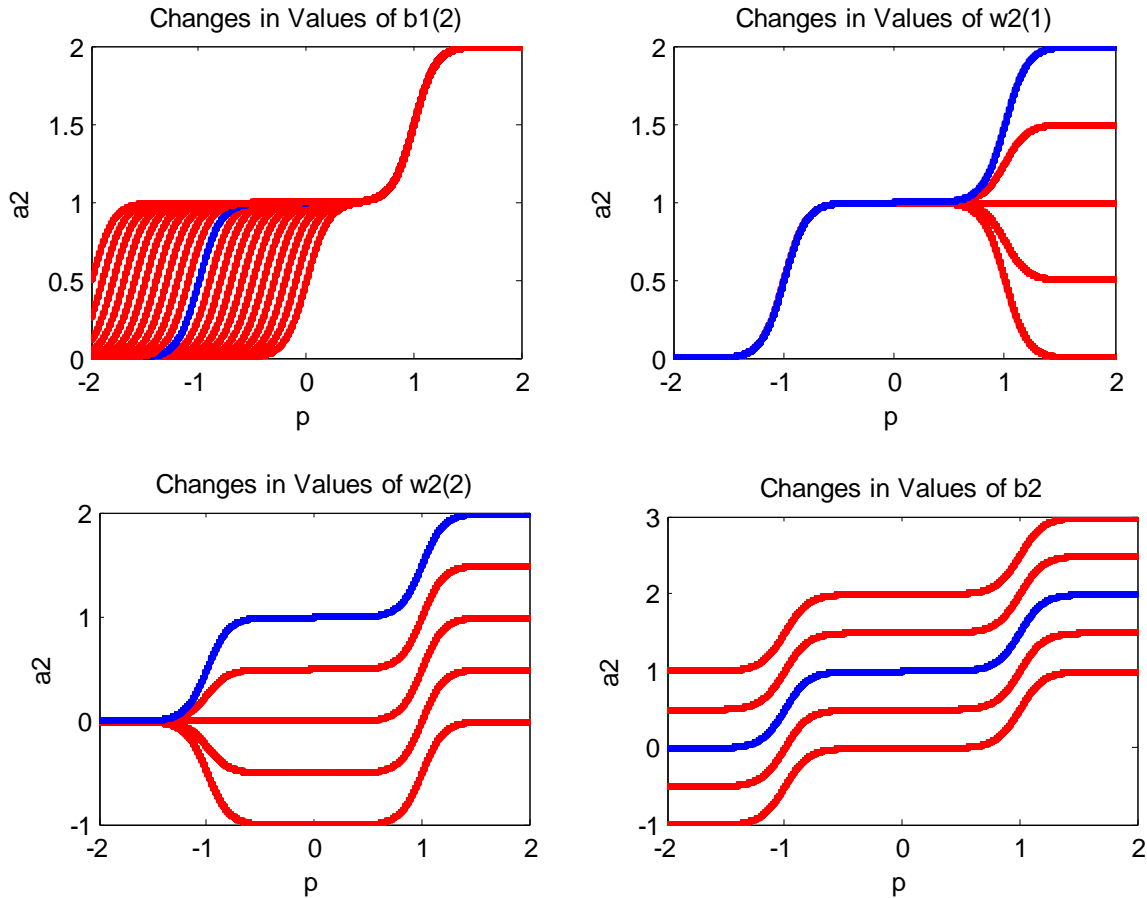
The centers of the steps occur where the net input to a neuron in the first layer is zero:

$$n_1^1 = w_{1,1}^1 p + b_1^1 = 0 \text{ or } p = -\frac{b_1^1}{w_{1,1}^1} = -\frac{-10}{10} = 1$$

$$n_2^1 = w_{2,1}^1 p + b_2^1 = 0 \text{ or } p = -\frac{b_2^1}{w_{2,1}^1} = -\frac{10}{10} = -1$$

Figure 3.12 illustrates the effects of parameter changes on the network response. The blue curve is the nominal response. The other curves correspond to the network response when one parameter at a time is varied over the following ranges:

$$-1 \leq w_{1,1}^2 \leq 1, -1 \leq w_{1,2}^2 \leq 1, 0 \leq b_2^1 \leq 20, -1 \leq b^2 \leq 1$$



**Figure 3.12: Effect of Parameter Changes on Network Response**

## 3.7 THE BACK-PROPAGATION ALGORITHM

The first step is to propagate the input forward through the network:

$$\begin{aligned} \mathbf{a}^0 &= \mathbf{p}, \\ \mathbf{a}^{m+1} &= \mathbf{f}^{m+1}(\mathbf{W}^{m+1}\mathbf{a}^m + \mathbf{b}^{m+1}) \text{ for } m=0,1,\dots,M-1 \\ \mathbf{a} &= \mathbf{a}^M \end{aligned}$$

The next step is to propagate the sensitivities backward through the network:

$$\begin{aligned} \mathbf{s}^M &= -2\mathbf{F}^M(\mathbf{n}^M)(\mathbf{t} - \mathbf{a}) \\ \mathbf{s}^m &= \mathbf{F}^m(\mathbf{n}^m)(\mathbf{W}^{m+1})^T \mathbf{s}^{m+1}, \text{ for } m=M-1, \dots, 2, 1. \end{aligned}$$

Finally, the weights and biases are updated using the approximate steepest descent rule:

$$\begin{aligned} \mathbf{W}^m(k+1) &= \mathbf{W}^m(k) - \alpha \mathbf{s}^m (\mathbf{a}^{m-1})^T \\ \mathbf{b}^m(k+1) &= \mathbf{b}^m(k) - \alpha \mathbf{s}^m \text{ [29]} \end{aligned}$$

### 3.7.1 RADAR PULSE COMPRESSION using Back-Propagation Algorithm in Multi-Layer Perceptron (MLP) for 13-bit BARKER CODE

In this method, *13-bit Barker Codes* are taken as *input* to the MLP Network. The training is performed for *1000 epochs*. The weights of all the layers are initialized to random values between  $\pm 0.1$  and the value of  $\eta$  is taken as 0.99.

The logistic function is used as activation function for both hidden and output neurons and is represented by,

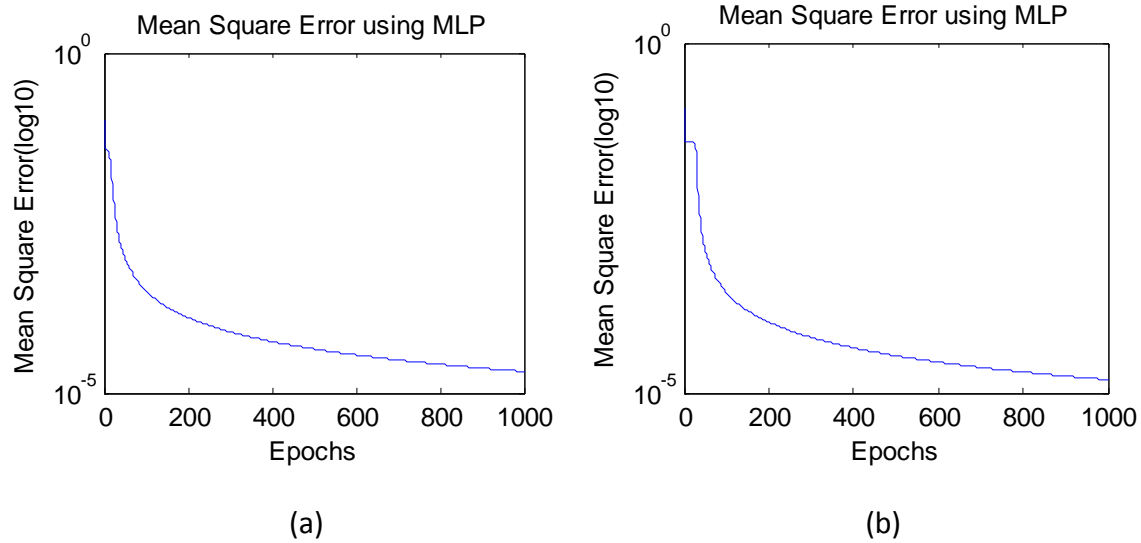
$$f(y) = \frac{1}{1 + e^{-y}}$$

The training of the network and hence, the updation of the weights and biases is done using the *Back-propagation Algorithm*.

The SSRs are then obtained by varying the *number of neurons in the hidden layers* and the *number of hidden layers*. They are then compared on the basis of the following criteria:

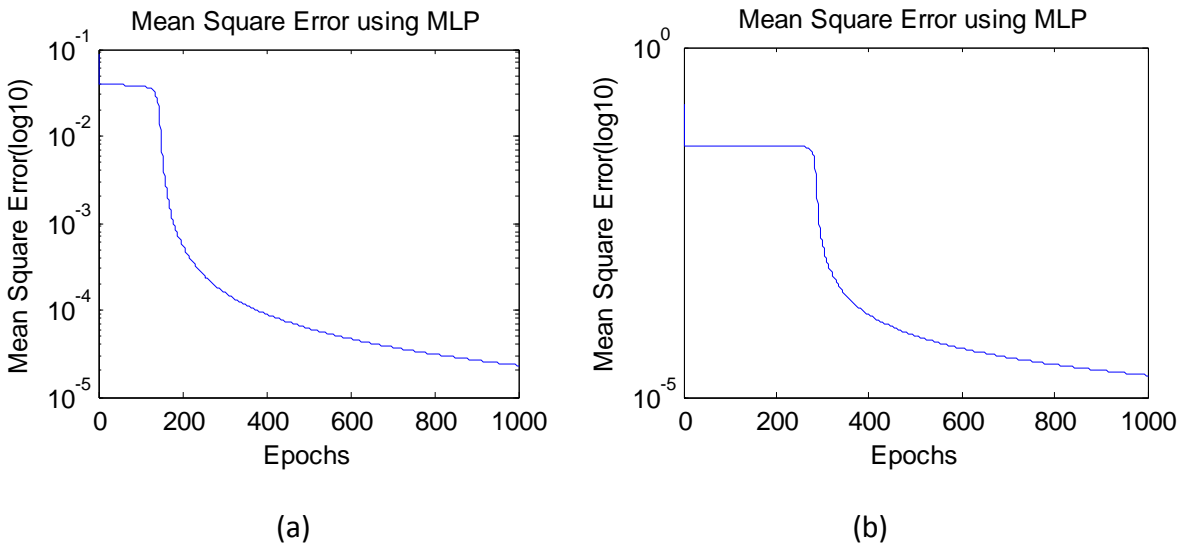
### A. Convergence Performance

The mean square error plots corresponding to different number of neurons in the hidden layer for MLP containing 1 hidden layer are shown in Figure 3.13.



**Figure 3.13: Mean Square Error plots for (a) 3 neurons and (b) 9 neurons in the hidden layer**

The mean square error plots corresponding to different number of neurons in the hidden layers for MLP containing 2 hidden layers are shown in Figure 3.14.



**Figure 3.14: Mean Square Error plots for (a) 3-5 and (b) 7- 9 neurons in MLP containing 2 Hidden Layers**

### B. SSR Performance

Signal-to-sidelobe Ratio (SSR) is the ratio of peak signal amplitude to maximum sidelobe amplitude.

**TABLE 3.1:**

**SSR Comparison in dB for different number of neurons in MLP containing 1 Hidden Layer**

Number of Neurons in the Hidden Layer	SSR (dB)
5	43.9851
9	47.5564
13	47.0187
19	48.5771

**TABLE 3.2:**

**SSR Comparison in dB for different number of neurons in MLP containing 2 Hidden Layer**

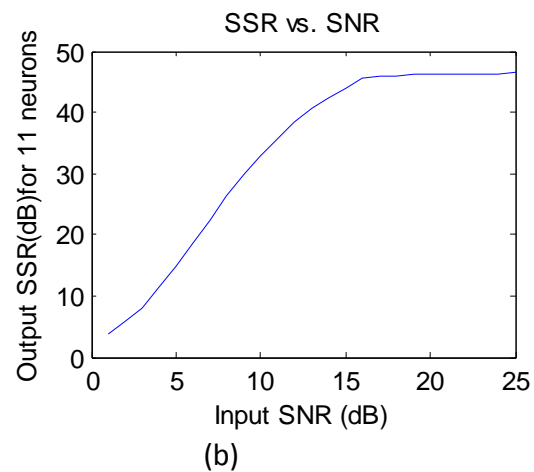
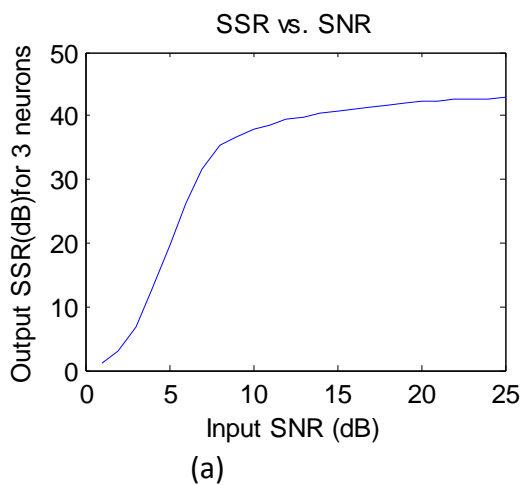
Number of Neurons in the 1 <sup>st</sup> Hidden Layer	Number of Neurons in the 2 <sup>nd</sup> Hidden Layer	SSR (dB)
3	7	47.8964
5	13	49.0722
7	15	47.1429
11	19	42.8630

### C. Noise Performance

The additive white Gaussian noise is added to input signal code. Consequently, the output is degraded and SSR is decreased gradually. The SSRs for different SNRs are tabulated and SSR vs. SNR graphs are plotted.

**TABLE 3.3:**  
**SSR Comparison in dB for different SNRs for different number of neurons in MLP containing 1 Hidden Layer**

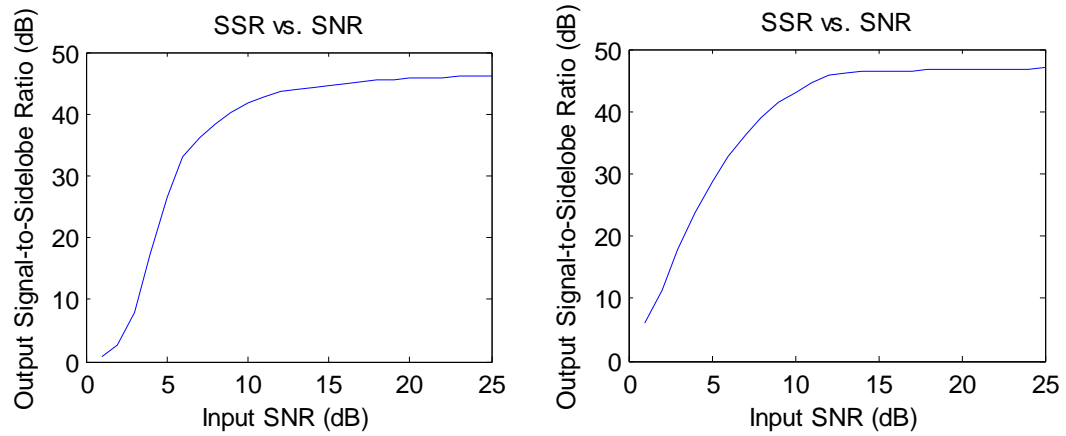
Number of neurons in the HIDDEN LAYER	SNR = 1 dB	SNR = 5 dB	SNR = 10 dB	SNR = 15 dB	SNR = 20 dB	SNR = 25 dB
1	0.2697	13.965	37.8197	41.6916	43.2368	43.5327
3	1.1668	19.8091	37.8114	40.8114	42.1449	42.8214
5	4.1712	20.1232	38.7173	41.4815	42.6685	43.2707
7	2.041	21.6259	39.8674	42.653	43.7414	44.2466
9	5.3084	27.8965	41.0523	44.582	45.9905	46.7106
11	3.5948	14.7718	32.9811	44.0575	46.1658	46.4402
13	9.0224	21.303	37.0488	43.7983	46.2642	46.7362
15	3.7224	15.8543	35.0372	44.1597	46.3064	46.5465
17	3.8924	16.8811	37.886	44.3849	45.8557	46.1104
19	5.6383	20.9294	36.2993	43.0907	45.8804	47.3765
20	4.0719	16.1363	33.6252	43.4301	46.2728	46.8659



**TABLE 3.4:**

**SSR Comparison in dB for different SNRs for different number of neurons in MLP containing 2 Hidden Layers**

Number of Neurons in the 1st Hidden Layer	Number of Neurons in the 2nd Hidden Layer	SNR = 1 dB	SNR = 5 dB	SNR = 10 dB	SNR = 15 dB	SNR = 20 dB	SNR = 25 dB
1	7	0.2528	19.9088	38.8952	44.43	46.5858	47.5633
1	9	0.2491	18.6388	38.3571	44.2721	46.6339	47.697
3	5	0.7931	26.7053	41.6806	44.6777	45.6661	46.13
3	7	0.8571	23.3123	39.2065	43.3445	45.4384	46.5497
3	11	1.0718	24.1717	41.3911	44.161	45.456	46.1385
3	13	1.1708	25.2417	41.7626	44.3733	45.644	46.3287
5	7	3.2947	31.4379	44.5597	46.6117	47.4279	47.8162
5	11	6.1551	30.6118	44.9987	47.3169	47.6638	47.8564
5	13	2.6059	28.2152	42.344	46.6218	48.3791	48.8482
7	5	4.0385	32.1655	44.0806	46.5317	46.7455	46.8645
7	9	6.0823	28.6948	43.1656	46.423	46.7295	46.901
7	11	4.9971	28.6378	43.6993	46.4641	46.7298	46.878
9	9	7.7688	31.5765	40.5437	42.1913	42.64	42.8459
9	11	1.6952	29.5374	38.532	41.5926	42.7712	42.7854
9	13	10.1014	36.188	41.4107	42.9554	43.2066	43.3471
11	11	7.1214	27.6626	36.7747	40.1591	41.2757	41.6925
13	13	1.3079	20.3109	37.3078	39.6308	40.1491	40.2645



**Figure 3.15: SSR vs. SNR plots for (a) 3-5 and (b) 7- 9 neurons in MLP containing 2 Hidden Layers**



#### *D. Doppler Tolerance*

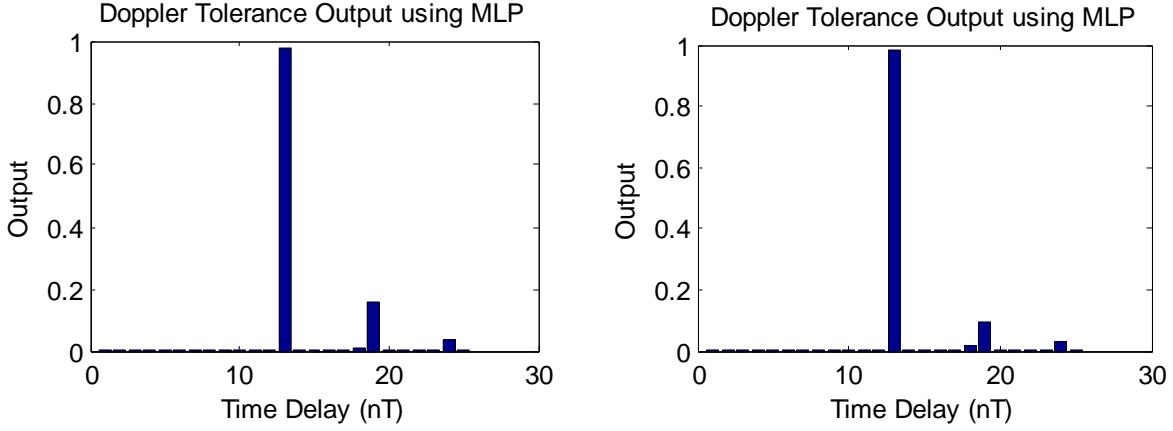
The Doppler tolerance of a system comes into picture when the frequency and/or phase of individual elements of the phase code are shifted.

In the extreme case, the code word is no longer matched with the replica, if the last element is shifted by 180 degree. For 13-bit barker code, the code is changed from (1,1,1,1,1,-1,-1,1,1,-1,1,-1,1) to (-1,1,1,1,1,-1,-1,1,1,-1,1,-1,1) and is fed to the networks. The SSR is then calculated for different number of neurons in MLP containing 1 and 2 Hidden Layers.

**TABLE 3.5:**

**Doppler Shift Performance in dB for MLP containing 1 Hidden Layer**

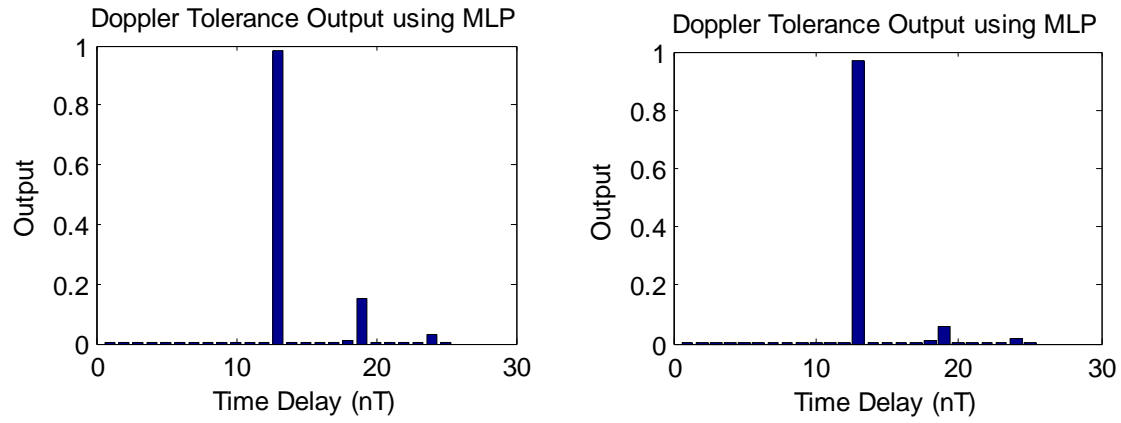
<b>Number of Neurons in the Hidden Layer</b>	<b>SSR (dB) <i>without</i> Doppler Shift</b>	<b><i>Doppler Shift</i> Performance (dB)</b>
1	43.5978	16.428
3	43.6296	15.6874
5	43.9851	19.9215
7	44.795	14.472
9	47.5564	14.9114
11	46.7855	20.5069
13	47.0187	14.4894
15	46.8489	14.7814
17	46.4311	14.1242
19	48.5771	11.2608
20	47.1648	15.0015



**Figure 3.16: Compressed Waveforms for Doppler Tolerance taking (a) 3 neurons and (b) 11 neurons in the hidden layer**

**TABLE 3.6: Doppler Shift Performance in dB for MLP containing 2 Hidden Layers**

Number of Neurons in the 1st Hidden Layer	Number of Neurons in the 2nd Hidden Layer	SSR (dB) without Doppler Shift	Doppler Shift Performance (dB)
1	1	43.7655	15.3624
1	9	47.7636	15.9011
3	7	47.8964	18.2616
3	11	46.9781	16.3934
5	13	49.0722	14.8157
5	17	48.0935	19.6598
5	19	48.0697	19.7777
7	1	43.9423	24.8851
7	3	45.9211	21.2999
7	7	46.5009	22.1625
7	15	47.1429	21.6014
7	19	46.6	20.5478
9	13	43.5131	24.2562
11	11	41.745	17.6705
11	19	42.863	24.5693
13	9	39.4495	21.0713
13	13	40.3331	23.2751
13	15	44.5892	27.0364



**Figure 3.17: Compressed Waveforms for Doppler Tolerance taking (a) 3-11 and (b) 7-1 neurons in the 2 hidden layers**

# Chapter 4

## CONCLUSION AND FUTURE WORK

## CONCLUSION:

From the analysis of the PSRs obtained in case of Radar Pulse Compression using Adaptive Filter Algorithms, the following points can be inferred:

- Given any adaptive algorithm, increasing the length of the Barker Code improves the PSR level.
- For a particular length of the Barker Code, the *Modified RLS filter* provides the minimum PSR (dB) and demonstrates the fastest convergence of the three adaptive filters.

In case of Artificial Neural Networks used for Radar Pulse compression:

- Increasing the SNR level in an MLP at first leads to a corresponding non-linear increase in SSR. Beyond a certain SNR level, the SSR tends to attain a steady state value.
- For 13-bit Barker Code, MLP containing a hidden layer of 11 neurons has the maximum Doppler Tolerance of 20.5069 dB.
- For 13-bit Barker Code, MLP containing 11 and 19 neurons in the 1<sup>st</sup> and 2<sup>nd</sup> hidden layers respectively has the maximum Doppler Tolerance of 24.5693 dB.

## FUTURE WORK:

The scope of Future Work is outlined below:

1. Varying filter lengths for different Barker Codes can be taken and a corresponding comparison of PSRs using LMS, RLS and Modified RLS algorithms can be analytically performed.
2. Other Adaptive Filter Algorithms, viz., IRLS, DIRLS algorithms etc., can be used for obtaining the matched filter coefficients in RADAR Pulse Compression.
3. Other codes, viz., *Ternary*, *Combined Barker Code* or *Complementary Sequences*, can be used for Radar Pulse Compression.
4. Filters such as *Mismatched Filter*, *Inverse Filter*, *Nonmatched Filter*, *TSSWA*, can be used in cascade with matched filter for better Pulse Compression results.
5. SSRs corresponding to MLPs containing more than 2 hidden layers of neurons can be obtained and compared.
6. Other neural network models such as *Radial Basis Function Networks*, *Recurrent Networks*, etc., can be implemented with various activation functions for training and determination of SSRs.

## Bibliography

- [1] Wikipedia. [Online]. [http://en.wikipedia.org/wiki/System\\_identification](http://en.wikipedia.org/wiki/System_identification)
- [2] Wikipedia. [Online]. [http://en.wikipedia.org/wiki/RLS\\_algorithm](http://en.wikipedia.org/wiki/RLS_algorithm)
- [3] Hayes M.H., *Statistical Digital Signal Processing and Modeling.*: Wiley, 1996.
- [4] Haykin S., *Adaptive Filter Theory.*: Prentice Hall, 2002.
- [5] Haykin S., *Least-Mean-Square Adaptive Filters*, Widrow B., Ed.: Wiley, 2003.
- [6] Wkipedia. [Online]. [http://en.wikipedia.org/wiki/Artificial\\_neural\\_network](http://en.wikipedia.org/wiki/Artificial_neural_network)
- [7] Wikipedia. [Online]. [http://en.wikipedia.org/wiki/Adaptive\\_filter](http://en.wikipedia.org/wiki/Adaptive_filter)
- [8] Wikipedia. [Online]. [http://en.wikipedia.org/wiki/Soft\\_computing](http://en.wikipedia.org/wiki/Soft_computing)
- [9] Wkipedia. Wkipedia. [Online]. [http://en.wikipedia.org/wiki/Least\\_mean\\_squares\\_filter](http://en.wikipedia.org/wiki/Least_mean_squares_filter)
- [10] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*. New Jersey: Prentice Hall, 1985.
- [11] Principe J. and Haykin S. Liu W., *Kernel Adaptive Filtering: A Comprehensive Introduction.*: John Wiley, 2010.
- [12] Graham C. and Payne, Robert L. Goodwin, *Dynamic System Identification: Experiment Design and Data Analysis.*: Academic Press, 1977.
- [13] Graupe D., *Identification of Systems*. New York: Van Nostrand Reinhold, 1972.
- [14] Juang J.N, *Applied System Identification*. Upper Saddle River, N.J.: Prentice Hall, 1994.
- [15] Ljung L., *System Identification — Theory for the User*. Upper Saddle River, N.J.: PTR Prentice Hall, 1999.
- [16] Harold J. and Yin, George G. Kushner, *Stochastic Approximation and Recursive Algorithms and Applications (Second ed.)*: Springer, 2003.
- [17] Nelles O., *Nonlinear System Identification.*: Springer, 2001.
- [18] Stoica P. Söderström T., *System Identification*. Upper Saddle River, N.J.: Prentice Hall, 1989.

- [19] Schoukens J. Pintelon R., *System Identification: A Frequency Domain Approach*. New York: IEEE Press, 2001.
- [20] Éric and Pronzato, Luc Walter, *Identification of Parametric Models from Experimental Data.*: Springer, 1997.
- [21] Lotfi A Zadeh, ""Fuzzy Logic, Neural Networks, and Soft Computing", " *Communications of the ACM*, pp. 77-84, 1994.
- [22] Monson H. Hayes, ""9.4: Recursive Least Squares", " in *Statistical Digital Signal Processing and Modeling.*: Wiley, 1996, p. 541.
- [23] Vinter R.B. Davis M.H.A., *Stochastic Modelling and Control.*: Springer, 1985.
- [24] Allen C., "Radar Pulse Compression," pp. 1-19, June 17, 2004.
- [25] Zejak A., Petrovic A., Simic I. Zrnic B., "Range Sidelobe Suppression for Pulse Compression Radars using Modified RLS algorithm," *IEEE papers*, pp. 1008-1011 , 1998.
- [26] Darwich T., "High Resolution Detection Systems using Low Sidelobe Pulse Compression Techniques," pp. 1-58, 2007.
- [27] Simani S., "System Identification and Data Analysis ," pp. 1-3.
- [28] Haykin S., *Neural Networks – A Comprehensive Foundation.*: Prentice Hall, 2005.
- [29] Demuth Howard B., Beale M. Hagan Martin T., *Neural Network Design*. New Delhi: Cengage Learning India Private Limited, 2009.
- [30] H. K. Kwan and C. K. Lee, ""A neural network approach to pulse radar detection", " *IEEE Trans. Aerosp. Electron. Syst.*, pp. vol. 29, pp. 9-21, Jan.1993.
- [31] Sahoo A.K., Panda G., Baghel V. Sailaja A., "A Recurrent Neural Network Approach to Pulse Radar Detection," *Dspace @ NITR*, pp. 1-5, 2009.
- [32] Merrill I. Skolnik, *Introduction to radar systems.*: McGraw Hill Inc., 2002.
- [33] and Xin Wu J.S. Fu, ""Sidelobe suppression using adaptive filtering techniques", " in *Proc. CIE International Conference on Radar*, Oct. 2001, pp. 788-791.
- [34] Mikael Boden. (2001, Nov ) "A guide to recurrent neural networks and backpropagation". [Online]. [www.itee.uq.edu.au/~mikaelpapers/rn\\_dallas.pdf](http://www.itee.uq.edu.au/~mikaelpapers/rn_dallas.pdf)



- [35] Meshulach D., and Silberberg Y. Yelin D., "Adaptive femtosecond pulse compression," *OPTICS LETTERS*, pp. Vol. 22, pp. 1793-1795, 1997.
- [36] Blunt S.D. and Gerlach K., "Adaptive Radar Pulse Compression, simulation, computing and modeling," *NRL Review*, pp. 215-217, 2005.
- [37] Ducoff M.R. Tietjen B.W., "Pulse Compression Radar," in *RADAR Handbook.*, 2008, pp. 8.1-8.3.
- [38] Owen J., "Simulation of a Pulse Compression RADAR Transmitter Receiver," *Agilent EEs of EDA*, pp. 1-5, 2003.
- [39] Yang J. and Sarkar T.K., "Doppler-invariant pulse compression for targets moving in an arbitrary direction," *MICROWAVE AND OPTICAL TECHNOLOGY LETTERS*, pp. Vol. 49, pp. 1449-1453, 2007.
- [40] Bharadwaj N., Chandrasekar V. George J., "PULSE COMPRESSION WITH WEATHER RADARS," p. 1.
- [41] Cohen M.N. Baden J.M., "Optical Peak Sidelobe Filters for Bi-phase Pulse Compression," *IEEE RADAR CONFERENCE*, pp. 249-252.
- [42] P. B. Chilson, B. L. Cheong, R. D. Palmer, M. Xue T. A. Alberts, "EVALUATION OF BINARY PHASE CODED PULSE COMPRESSION SCHEMES USING AND TIME-SERIES WEATHER RADAR SIMULATOR," pp. 1-6.
- [43] Daheleh, Munther A., System Identification, Lecture1, pp.1-20
- [44] Rao N.A.H.K., "Investigation of a Pulse Compression Technique for Medical Ultrasound: a simulation study," *Medical & Bio Engineering & Computing*, pp. 181-188, 1994.
- [45] Wikipedia. [Online]. [http://en.wikipedia.org/wiki/Pulse\\_compression](http://en.wikipedia.org/wiki/Pulse_compression)
- [46] Nathanson F.E., *Radar Design Principles.*: Prentice-Hall, 1999.