MIMO Channel Modeling using a Class of Soft-Computational Techniques

A Thesis submitted for the award of degree of
DOCTOR OF PHILOSOPHY

by

Kandarpa Kumar Sarma

Department of Electronics and Electrical Engineering
Indian Institute of Technology Guwahati
Guwahati-781039, Assam, India.
December, 2011

MIMO Channel Modeling using a Class of Soft-Computational Techniques

A Thesis submitted in partial fulfillment for the award of degree of
DOCTOR OF PHILOSOPHY

Kandarpa Kumar Sarma
Roll no. 09610211

Department of Electronics and Electrical Engineering
Indian Institute of Technology Guwahati
Guwahati-781039, Assam, India.
December, 2011.

Declaration

The work which is being presented in this thesis titled "MIMO Channel Modeling using a Class of Soft-Computational Techniques" in partial fulfillment for the award of the degree of PhD carried out by the undersigned during the period July, 2009 to December, 2011 at the Department of Electronics and Electrical Engineering, Indian Institute of Technology Guwahati, Guwahati-781039, Assam, India under the supervision of Dr. Abhijit Mitra, Associate Professor, is an authentic record for the same degree and has not been submitted elsewhere for a degree.

Kandarpa Kumar Sarma
December 2011
Department of Electronics and Electrical Engineering
Indian Institute of Technology Guwahati

Abstract

The key challenge faced by up-coming wireless communication systems is to provide high-data-rate wireless access at better quality of service (QoS). In such a situation, Multiple-Input Multiple-Output (MIMO) wireless technology seems to be able to meet these demands by offering increased spectral efficiency. A very common form of uncertainty and stochastic behaviour is observed in MIMO wireless communication due to interference and correlation among channel coefficients which makes channel estimation a challenging area. There are several statistical methods of channel estimation that have provided satisfactory performance while modeling the MIMO wireless systems. Soft-computational approaches are recent additions to the list of channel estimation methods of which most of the works have primarily focused on the training-learning aspects of ANN, fuzzy systems etc. Till now, no recorded efforts have been observed regarding expansion of the abilities of such architectures beyond the training-testing realm which includes certain architectural challenges. These challenges include: (i) incorporating temporal behaviour in the Multi Layer Perceptron (MLP)- a feedforward ANN enabling it to track time-variations in the input signal, (ii) ensuring stability to the system by appending a feedback path along with the usual feedforward structure of the ANN, (iii) retaining only the contextual portion of the information with the above structure, (iv) properly capturing the fast time-varying nature of the channels, (v) combining ANN and fuzzy based systems to obtain the capability of expert-level decision making while modeling uncertainty observed in the MIMO channel and (vi) realization of a suitable system with lower implementation and time complexity. Taking these challenges into consideration, a class of soft-computational tools based on ANN in feedforward layout called MLP and feedback form called Recurrent Neural Network (RNN) and fuzzy-based composite systems are explored with stress on architectural expansion so as to improve performance and precision than conventional methods while modeling the stochastic nature of the MIMO channels. Among the proposed techniques of this thesis, a fuzzy based approach, named Fuzzified Time Delay Fully RNN (FTDFRNN) establishes its superiority while modeling even a deeply faded MIMO channel in important practical applications like VOIP based transmissions. In such a framework of MIMO communication, the fuzzy based approach turns out to be the most suitable one for adaptive receiver designs optimized for high data rate wireless communication.

Acknowledgment

I would like to express my deep sense of gratitude to **Dr. Abhijit Mitra**, for his invaluable help and guidance during the course of work. I am highly indebted to him for constantly encouraging me by giving his critics on the work which ensured that the work reach its present form. I am grateful to him for having given me the support and confidence. The various values that I tried to learn from him shall remain a source of inspiration for me forever.

I also would like to express my thanks and gratitude to the members of the doctoral committee for their suggestions and insights. The positive environment of the Department of Electronics and Electrical Engineering (EEE), Indian Institute of Technology Guwahati (IITG) have also contributed significantly towards completion of the work. In this regards, I would like to convey special regards to Ramesh Mishra, Sayantan Hazra and Mukesh Singh, research scholars, EEE for their help and support at different times.

Special thanks and regards to my parents, wife and son for their support and encouragement in their respective ways all throughout.

Finally, I would like to thank the Almighty for all I have been given.

Kandarpa Kumar Sarma
December, 2011
Department of Electronics and Electrical Engineering
Indian Institute of Technology Guwahati

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The proliferation of mobile communication networks over the last decade has increased the use of the wireless spectrum in exponential terms. Increase in number of users in a limited spectrum have raised the levels of intersymbol interference (ISI) and also have increased the possibility of degraded quality of service (QoS). The key challenge faced by upcoming wireless communication systems is to provide high-data-rate wireless access at better QoS. Also, destructive addition of multipath components within the fast shrinking spectrum available for wireless communication have necessitated the development of methods to increase spectral efficiency and explore innovative solutions. Additionally there is a constant demand for higher bandwidth, increased data rates, lower cost, greater coverage etc for which the mobile networks are creating congestion in the available spectrum. Multiple-Input Multiple-Output (MIMO) wireless technology is a viable option in such a situation and is likely to be able to meet the demands of these ever-expanding mobile networks. With increased spectral efficiency, MIMO architectures are useful for combined transmit-receive diversity. For its parallel mode of transmission, MIMO systems offer high data rates in a narrow bandwidth. MIMO systems, characterized by multiple antenna elements at the transmitter and receiver, have demonstrated the potential for increased capacity in rich multipath environments.

In such a propagation environment, modeling MIMO channels is considered to be one of the challenging areas and many researchers have explored this field over a considerable period of time. But channel estimation is still a challenging area which is worth exploring. Two common practices of channel estimation in MIMO systems are blind and non-blind methods. Blind estimation techniques don't require training sequences but are extremely computationally intensive [1] [2]. Among non- blind methods, pilot- carrier based estimation techniques are common. These use least-squares (LS), minimum mean-

square error (MMSE) and linear MMSE (LMMSE) estimation. The pilot-based channel estimation, by requiring additional symbol bits known as pilot carriers to be inserted as training sequence along with signal-symbol blocks, causes waste of bandwidth. Therefore, alternative methods should be explored to prevent waste of bandwidth while modeling MIMO channels and thereby improving overall performance of such systems.

1.2 Motivation

Statistical methods of channel estimation are capable of properly modeling the MIMO wireless systems as well as recovering the transmitted signal (symbols) [3]-[7]. MIMO systems and related aspects including channel modeling and symbol recovery are treated extensively in [8] [9]. Soft-computational approaches, however, can be appended to these available methods for providing innovative solutions to channel modeling. This is primarily due to the fact that these tools are better placed to use channel side information (CSI) for improved performance. One of the viable means of such innovative channel estimation is the use of the Artificial Neural Network (ANN) [3]. ANN based systems have already received attention as an optional tool for equalization and other such applications in wireless communication. Such systems have also been considered for use with Single-Input-Single-Output (SISO) and Single-Input-Multiple-Output (SIMO) systems [10]-[13]. These applications are related mostly to channel equalization and noise cancelation. Over and above all these applications, some other areas include user detection and interference mitigation [10]-[33]. The most preferable aspects of the ANN in these applications have been parallelism, adaptive processing, self-organization, universal approximation and ability of tackling highly nonlinear problems. Also, as the ANN can learn complex patterns, it can act as a reliable estimator and hence can be used for modeling MIMO channels. The advantage of these schemes is that no pilot symbol bits are required to be inserted with the MIMO transmissions which can contribute towards preserving bandwidth and increasing spectral efficiency. The system can also be extended to symbol recovery in Orthogonal Frequency Division Multiplexing (OFDM) scheme and user detection as part of high data rate communication using MIMO-OFDM arrangement.

Such a work dealing with static and slowly varying MIMO channels has already been reported in [18] for MIMO-OFDM channel estimation. It uses a variable step-size recursive LS (RLS) method for nonlinear principle component analysis (PCA) to train a two-layered ANN for the purpose. However, all these works focus mainly on the training-learning aspect of the ANN and its capacity to deal with MIMO channel estimation with practically little/no consideration to time-varying characteristics of the wireless channels. Till now, no recorded efforts have been observed regarding expansion of the abilities of

such architectures beyond the training-testing realm which includes certain architectural challenges. These challenges include: (i) incorporating temporal behaviour in the Multi Layer Perceptron (MLP)− a feedforward ANN with suitable architectural modifications enabling it to track time-variations in the input signal, (ii) ensuring stability to the system when a feedback path is appended along with the usual feedforward structure of the ANN, (iii) retaining only the contextual portion of the information with the above structure, (iv) properly capturing the fast time-varying nature of the channels, (v) combining ANN and fuzzy based systems to obtain the capability of expert-level decision making while modeling uncertainty observed in the MIMO channel and (vi) realization of a suitable system for adaptive receiver design with lower implementation and time complexity. Taking these challenges into consideration, a few appropriate alternatives to the ANN approach derived are the temporal-MLP and the Recurrent Neural Network (RNN)-based designs. Both these approaches have the capacity to deal with time-varying inputs due to the presence of at least one feedback loop despite these being essentially feedforward structures [34]. The temporal-MLP has greater implementation complexity and maybe replaced by RNN-based designs. Although RNN has already been used to predict a MIMO channel in [22], the focus has again been to develop a hybrid Particle Swarm Optimization (PSO)-Evolutionary Algorithm (EA)-Differential Evolution PSO (DEPSO) (PSO-EA-DEPSO) training algorithm. No novelty on the architectural part of RNN has been covered in this work. Similar is the case with a few related works as well. Hence, the exploration of the architectural expansion of these RNN based systems for time-varying MIMO channel modeling becomes pertinent.

The complexity associated with MIMO channel modeling is considerable with time-varying properties which generate co-channel interference (CCI) and disturbance resulting in uncertainty in estimating coefficients, recovery of transmitted symbols and phase-time delays. Also, in certain practical cases like Voice Over IP (VOIP) based transmission, the performance of RNN-based methods needs to be improved further especially the accuracy of the recovered content. In such situations, the natural option that emerges as the suitable addition along with the RNN is either fuzzy system or genetic algorithm (GA). Due to certain limitations involving computational complexity, GA approach is less preferable. Hence, the fuzzy based system integrated with RNN block turns out to be the only viable option to provide the desired accuracy. All the reported works on such soft-computational applications for MIMO channel modeling can be regarded to be entirely training-learning paradigm dependent solutions with little or no stress on architectural expansion of MLP, RNN and fuzzy-based systems.

The primary motivation of our work is to explore whether ANN and fuzzy-based systems, as important components of soft-computation, can be specially configured to model the MIMO channel and mitigate some of the deficiencies observed. These schemes not only

can contribute towards increasing spectral efficiency, but are also resilient to noise and similar disturbances and has an inherent property of providing equalization to the received signal. Further such systems with their ability to learn are in a better position to use transmitter side information (TSI), channel side information (CSI) and receiver side information (RSI) for design of adaptive communication methods. The work focuses on certain architectural challenges of configuring MLP, RNN and composite fuzzy-neural system (FNS) unlike the known works reported in contemporary literature so that processing time and design complexity is considerably reduced with enhanced performances.

1.3 Literature Survey

ANNs have already received attention as an optional tool for equalization and other such applications in wireless communication. In the following, we divide our literature survey in several subsections so that our problem definition becomes apparent to the readers.

1.3.1 Application of feedforward ANN and MLP in wireless communication

For SISO and SIMO set-up, MLPs have been extensively used. As an extension to the application of SISO and SIMO, MIMO systems also have attracted considerable employment of ANN for a range of applications like channel equalization, interference cancelation, identification and estimation. A few such works are included below:
A three layer ANN along with feedback is used for MIMO channel estimation and equalization and is reported in [10]. The work uses a Kalman filter and a feedforward ANN to perform MIMO channel estimation. Another work cited in [11] reports the application of ANN for location estimation and CCI suppression in cellular networks. A work related to blind equalization of a noisy channel by linear ANN is reported in [12]. Another work of similar nature is available as cited in [13] where blind channel equalization and estimation is performed using ANN. This work discusses application of ANN mainly with a time invariant SISO channel. Along with CCI cancelation and equalization, estimation of MIMO channels have also received attention with regards to application of ANN and related tools. A method based on iterative estimation of MIMO channels using Support Vector Machine (SVD) is reported in [14]. Another work [15] reports the application of SVD based adaptive channel estimation for MIMO-OFDM systems. SVD has also been used for subchannel CCI cancelation in a MlMO system as described in the work cited in [16]. Like SVD, ANN has always been a preferred tool for developing applications in high data rate systems like MIMO-OFDM systems. Following these applications, identi-

fication of nonlinear MIMO channels using ANNs has also been reported. One such work is [17]. Works of similar nature that deals with static and slowly varying MIMO channels has already been reported. A work [18], reports the use of ANN for MIMO-OFDM channel estimation. The work uses pilots to estimate the channel impulse response based on LS criteria. To improve the estimation performance, an ANN approach is applied to track the variations of the channel using a variable step-size RLS. Application of Multi-ADAptive LINear Element (MADALINE) which is a feedforward ANN, for parameter estimation of Linear Time Invariant (LTI) MIMO systems is reported in [19]. An ANN can be specially configured to mitigate some of the deficiencies of multi-user transmission. The advantage of these schemes is that no pilot symbol bits are required to be inserted with the MIMO transmissions which can contribute towards preserving bandwidth and increasing spectral efficiency. However, in all these cases, be it SISO or SIMO or MIMO, our observation is that the reported works are mainly confined for slowly time varying channels and they portray the training-testing modifications.

1.3.2 Application of Recurrent Neural Network (RNN) in wireless communication

RNNs have also been used for a diverse range of applications in wireless communication. Some of the relevant literature are cited between [20]-[33]. Blind equalization has been the most common area in which RNNs have been applied. A work cited in [20] uses a RNN for blind equalization which proves to be effective. A more extensive application of the RNN is observed in case of a system developed for MIMO channel prediction using a PSO-EA-DEPSO off-line training algorithm. This predictor is shown to be robust to varying channel scenarios [21]. The work only concentrates on MIMO channels and is not directed towards recovery of data symbols transmitted through the channel. Another related work cited in [22] improves the effort reported in [21] by using an on-line approach. A new hybrid PSO-EA-DEPSO algorithm is presented for training a RNN for MIMO channel prediction. This algorithm is shown to outperform RNN predictors trained off-line by PSO, EA, and DEPSO as well as a linear predictor. This work also is not directed towards recovery of transmitted signal content. Further the work doesn't specify if real and complex signals are considered in split form or in coupled form. A work [23] provides a new adaptive neural predictor for GPS jamming suppression applications designed using the efficient square-root extended Kalman filter (SREKF) algorithm to adjust the synaptic weights in a RNN architecture and thereby estimate the stationary and non-stationary narrowband / FM waveforms. A novel approach to adaptive channel equalization with RNN for a QSPK signal constellation is given in [24]. The work deals with wireless communications in non linear channels for M-PSK and M-QAM modulation schemes.

RNNs with extended Kalman filter (EKF) algorithm has also been used for nonlinear equalization in satellite communication. This is reported by a work as indicated by [25]. Another work cited in [26] applied Complex Real Time Recurrent Learning fully RNN extended Kalman filter trained (CRTRLEKF) in adaptive equalization for cellular communications. Results illustrate the strength of the method in Wide Sense Stationary -Uncorrelated Scattering (WSS-US) channel model. Accurate and timely estimation of CSI will guarantee the Quality of Service (QoS) by admission control, inter and intra network handovers in NLOS channels. For such a problem, BER is predicted by two different RNN architectures such as Recurrent Radial Basis Function Network (RRBFN) and Echo State Network (ESN) [27]. RNNs trained with gradient-based algorithms such as Real-Time Recurrent Learning (RTRL) or Back-Propagation Through Time (BPTT) have a drawback of slow convergence rate. A derivative-free Kalman filter, also called the unscented Kalman filter (UKF), for training a fully connected RNN is presented for non-linear equalization in [28] [29]. A work on the use of Kalman filter-trained RNN equalizers for time-varying channels is reported in [30]. A decision feedback RNN based equalization with fast convergence rate for time-varying channels is described in [31]. In the work described in [32], the application of fully connected RNNs (FCRNNs) is investigated in the context of narrow-band channel prediction using three different algorithms, namely the RTRL, the global extended Kalman filter (GEKF) and the decoupled extended Kalman filter (DEKF). The system is designed for training the RNN - based channel predictor. The work shows that GEKF and DEKF training are faster than the RTRL based learning. A new method for pruning the Complex Bilinear Recurrent Neural Network (CBLRNN) using genetic algorithm is proposed in [33] and is applied to equalization of wireless Asynchronous Transfer Mode (ATM) channels.

The most preferable aspects of the RNN in these applications have been parallelism, adaptive processing, self-organization, universal approximation and ability of tracking highly nonlinear problems. Here too, the reported works of application of RNN for MIMO channel modeling have not crossed the traditional limits of experimenting with the training−testing realm. In particular, no reported works have dealt with architectural expansion of the RNN which would have been a natural modification over traditional RNN structures.

1.3.3 Application of Fuzzy, Fuzzy-Neural and Neuro-Fuzzy Systems in Wireless Communication

Fuzzy and related hybrid systems namely Fuzzy-Neural (FN) and Neuro-Fuzzy (NF) Systems provide adaptive expert level decision making capacity, hence is suitable for a wide range of applications. Fuzzy based systems are efficient tools to be utilized in problems

for which either information or knowledge of all factors is insufficient or impossible to obtain. Fuzzy and related hybrid systems have also received attention for application in wireless communication. A work cited in [35] is a must read description for all fuzzy related implementation. The description provides an exhaustive survey of neuro-fuzzy rule generation algorithms together under a unified soft computing framework. The work includes some important works related to rule generation of NFS and relates them to certain real world applications. Another work of similar nature is [36]. Some nobel efforts have been put into a publication cited in [37], which contains a survey of fuzzy logic applications and principles in wireless communications. It is reported with the aim of highlighting successful usage of fuzzy logic techniques in telecommunications and signal processing. The authors claimed this is to be the first such study of its kind. This paper focuses firstly on discerning prevalent fuzzy logic or fuzzy-hybrid approaches in the areas of channel estimation, equalization and decoding, and secondly outlining what conditions and situations for which fuzzy logic techniques are most suited for these approaches. A detailed account of some applications of fuzzy systems in communication is provided in this report. One of the earliest reported applications of fuzzy systems in wireless communication in [38]. It reports the use of an RLS fuzzy adaptive filter for non-linear channel equalization. A work of similar nature that can also be considered to be among the few earliest reported is [39] and it deals in some detail with a fuzzy based channel equalization problem. Another contemporary work is [40] which shows the use of fuzzy systems to carry out channel estimation. A similar survey paper is [41]. Another work [42] reports the use of fuzzy systems to perform channel estimation in CDMA based wireless communication. A simple method reported in [43] shows that data sequence and estimates of the channel condition can be carried out at the same time using the Viterbi algorithm and fuzzy logic for the convolutional code. After a fixed number of decoding steps, the fuzzy logic unit reads the branch metric value of the survivor and the difference between maximum and minimum survivor path metric values at the Viterbi decoder and estimates the channel condition with the signal-to-noise ratio (SNR). The proposed method enables the channel estimation regardless of what kinds of modulator and demodulator are used. Another work refereed in [44] presents the equalization of channel distortion by using NF network. The structure and learning algorithm of NF network have been described. Using learning algorithm of NF network an adaptive equalizer have been developed. The developed equalizer recovers transmitted signal efficiently. The use of NF equalizer in digital signal transmission allows to decrease training time of parameters and the complexity of network. A work cited in [45] discusses about a TSK fuzzy approach to channel estimation for OFDM systems. Fuzzy systems can be used to update LMS algorithm for OFDM channel estimation in time-variant mobile channels. Such a work is reported in [46]. The workers of the publication cited in [45] extends the work further

using a TSK fuzzy approach to channel estimation for MIMO-OFDM systems [47]. An adaptive NF inference for OFDM channel estimation is reported in [48]. Use of fuzzy logic as the core of the reasoning engine to determine different parameters used by the WiMAX system is reported in [49]. This work focuses on one of the main functions of the reasoning engine i.e. determination of the channel type and the number of pilots used for channel estimation. Anther work introduces an Adaptive Neural Fuzzy Channel Equalizer (ANFCE) based on Adaptive Neural Fuzzy Filter (ANFF) [50]. The ANFF is a five layer ANN which is able to use the expert knowledge in its structure. The structure and parameters of this network are adjusted according to the training data and the available expert knowledge. A work cited in [51] propose a computationally efficient NF system based equalizer for use in communication channels. This equalizer performs close to the optimum maximum a-posteriori probability (MAP) equalizer with a substantial reduction in computational complexity and can be trained with a supervised scalar clustering algorithm. The work [52] proposes an adaptive NF inference system (ANFIS) for channel estimation in OFDM systems. To evaluate the performance of this estimator, the authors compare the ANFIS with LS algorithm, MMSE algorithm by using bit error rate (BER) and mean square error (MSE) criteria.
Here, we noticed that the major literature have been restricted to the popular NFS or FNS learning and decision making arena with no focus on performance variations that maybe derived from the application of fuzzy-based hybrid estimators and architectural expansion of such systems for time-varying MIMO channel modeling.

1.4 Problem Definition

After a detailed survey of the available main literature, a few open literature problems are found which yield enough implementational challenges for our interest. These are summarized below:

- We found in Section 1.3.1 that MIMO channels are predicted using an ANN method [10] and the procedure is a direct extension of Kalman filtering while using the MLP as a classifier. This, however, prove to be effective mainly for slowly time varying channels. Moreover, symbol recovery has not been reported using ANN. Therefore, our primary focus is to devise a proper form of ANN which can perform both channel estimation and symbol recovery. Our next focus is to incorporate suitable temporal behaviour in the MLP with architectural modifications so that it can model fast time varying MIMO channels too.

- Temporal MLP provides certain architectural constraints which can be efficiently replaced by using RNN. In major literature as given in Section 1.3.2, we found

the examples of usage of RNNs for MIMO channel prediction [22] where hybrid evolutionary computation is deployed to handle MIMO channels. However, the works are confined mainly to training−testing realm and these provide no specific information regarding expansion of the RNN into composite architectures. Our efforts are mainly focused on architectural challenges, i.e., to explore the complete range of capability of the RNN and use it in combinations with Self Organizing Map (SOM) to model time varying characteristics of MIMO wireless channels.

- The outcome of MIMO channel modeling using RNN architectures leave a scope for lower computational complexity and higher reliability. One such viable component is the fuzzy system which has the capacity to model uncertainty. No reported works are available till now, as given in Section 1.3.3, to the best of our knowledge, regarding the use of fuzzy-based hybrid estimators for time-varying MIMO channels. As FN systems are oriented more towards real situations, an attempt is made to configure a Fuzzy-Neural System (FNS) to deal with time varying inputs in fuzzified form to model MIMO channels.

All these defined problems are elaborated in the next section as well as are discussed at length in Chapters 3 to 5.

1.5 Main Contribution of the Thesis

Keeping a close eye on the challenges associated with the MIMO channel modeling and defined problems, the main contribution of the thesis can be summarized as follows:

1. **MIMO channel modeling using MLP and its temporal form:** MLP architectures are designed, trained and tested to tackle static and slowly varying multipath fading channels with and without LOS components. The work demonstrates higher success rates in symbol recovery and achieving lower BER than LS and MMSE estimators. As the MLPs are suitable for static and slowly varying cases, the work is further modified to enable the system acquire a temporal characteristic with which it could model time-varying MIMO channels. The temporal MLP structures proposed for this purpose are:

   (a) Finite Impulse Response (FIR) MLP (FIR-MLP) with six sub-classes due to the variation in positions of placement of the FIR synapse. The FIR synapses are placed in the input, output and hidden layers which provide temporal characteristics to the MLP.

   (b) Infinite Impulse Response (IIR) MLP (IIR-MLP) also with six sub-classes as the FIR-MLP.

The success rates achieved are higher than conventional MLP based system but the primary constraints observed is higher design complexity and intricacies associated with the configuration.

2. **RNN and SOM based hybrid architectures for modeling MIMO Channels**: The design complexities demonstrated by the temporal-MLP estimators could be reduced considerably by the use of the RNN-based designs. This is due to the fact that RNNs naturally possess the ability to track time variations in input samples. This attribute was the primary motivation behind the exploration of RNN-based MIMO estimators. The following RNN based architectures are proposed which show improved capacity with faster processing speed while handling time-dependent MIMO channels. These are (*i*) Complex Time Delay Fully Recurrent Neural Network (CTDFRNN) with time averaging, (*ii*) CTDFRNN with SOM optimization, (*iii*) CTDFRNN - Cluster optimized by SOM and (*iv*) CTDFRNN - Modular Network SOM (CTDFRNN-MNSOM). From experimental results, we show that CTDFRNN-MNSOM provides the best results among the proposed frameworks.

3. **Fuzzy Neural System (FNS) based MIMO channel modeling**: The outcome of MIMO channel modeling using RNN architectures leaves a scope for lower computational complexity and higher reliability for which fuzzy-based hybrid systems are considered. Fuzzy systems provide expert level knowledge for control and decision making while ANNs are non- parametric prediction tools that have the ability to replicate biological cognitive behaviour. These attributes of fuzzy based systems in combination with ANNs are explored and configured for MIMO channel modeling. No reported works are available as per our knowledge regarding the use of FNS to model MIMO channels with time-varying characteristics. A fuzzy based approach is attempted here where the estimation mechanism is formed by an inference engine which is a multi-layered set-up designed by following two approaches based on (*i*) FNS with Fuzzified MLP (FMLP) and (*ii*) FNS with Fuzzified TDFRNN (FTDFRNN). These two fuzzy blocks lay the foundation of the FNS used for MIMO channel modeling.

1.6 Organization of the Thesis

The thesis is organized as below:

1. Chapter 1 discusses the background of the present work and survey of the related major literature. It also defines the problem as well as outlines the major contribution of the next chapters.

2. Since the principal objective of this thesis is to develop MIMO channel modeling methods by ANN, a variant of the ANN and composite FNS; conceptual backbone is needed to discuss these with underlying mathematics. The principles behind these systems and the relevant details are therefore discussed in Chapter 2.

3. Chapter 3 introduces the first proposal of this thesis, i.e, MIMO channel modeling by MLP and its temporal forms. In particular, two different structures, FIR-MLP and IIR-MLP are proposed in this chapter to model MIMO channels.

4. Temporal MLP provides certain architectural constraints which can be efficiently replaced by using RNN. A class of RNN architectures are proposed in Chapter 4 for MIMO channel modeling. It is also shown experimentally that these structures perform better than the proposal of Chapter 3.

5. The outcome of MIMO channel modeling using RNN architectures leaves a scope for lower computational complexity and higher reliability. One such viable option is to consider fuzzy-based hybrid systems. Such systems are better suited for adaptive receiver design. These considerations are formulated in Chapter 5.

6. Chapter 6 includes a summary of the work contained in the thesis and provides certain future directions by pointing out the open ended treatments of the present proposals.

Chapter 2

Fundamentals of MIMO Wireless Channels and Soft Computational Techniques

2.1 Introduction

It is mentioned in Section 1.2 that apart from conventional methods, MIMO channels can also be modeled by soft-computational approaches providing innovative solutions. Since the principal objective of this thesis is to develop the same treatment, i.e. modeling MIMO channels using a suitable form of ANN, RNN and composite FNS, it is necessary to explain the fundamental aspects of the above notions so as to develop the conceptual backbone of the next chapters. It is also required to introduce the notations of the mathematical deductions which are frequently used in the following chapters. This is the primary aim of this chapter. Towards this, in the next section, we explain the basic characteristics of the MIMO channels and the conventional statistical channel estimation methods. Then, the fundamental considerations of an ANN and a few of its architectural variants are given. Finally, the basics of a fuzzy system is dealt with which can be efficiently used with a neural network as a composite system.

2.2 Characteristics of Wireless Channel

Wireless communication utilizes modulation of electromagnetic (radio) waves with a carrier frequency varying from a few hundred MHz to several GHz depending on the system. Therefore, the behavior of the wireless channel is a function of the radio propagation effects of the environment. In such an environment the following may happen [53] [54]: (a) multiple delayed receptions of the transmitted signals due to the reflections of build-

ings, hills, cars and other obstacles, etc., (b) absence of a line-of-sight (LOS) path and prominence of non-LOS (NLOS) components, (c) varying attenuation, time delay, phase shift etc in each path and (d) constructive and destructive addition of the constituent paths due to multiple phase shifts resulting in fluctuations in the signal strength. These factors act together to give rise to a phenomenon known as fading. The stochastic nature observed in wireless channels can be described either by a Rayleigh fading or a Rician fading model. Sometimes, these two models are generalized by another model, called Nakagami model [53]. Using the knowledge of standard fading models, the capacity of a SISO channel is expressed as

$$C = log(1 + \frac{\bar{P}}{N_0 w}) \qquad bits/s/Hz; \tag{2.1}$$

where w is the number of samples per second, $\bar{P}$ is the average energy per transmit symbol and $N(0, N_0)$ is additive white Gaussian noise (AWGN). When we extend this formula for SIMO or MISO channel, the capacity of such a system turns out to be

$$C = log(1 + \frac{P \parallel \underline{h} \parallel^2}{N_0}) \qquad bits/s/Hz; \tag{2.2}$$

where $\underline{h}$ is a vector of channel transfer function coefficients at the receiver or transmitter side respectively. However, for a MIMO channel, due to the presence of transmit-receive diversity, the channel modeling is relatively difficult. This is discussed next.

2.2.1 MIMO channel model

MIMO architectures are useful for combined transmit receive diversity. For its parallel mode of transmission, MIMO systems offer high data rates in a narrow bandwidth. MIMO systems, characterized by multiple antenna elements at the transmitter and receiver, have demonstrated the potential for increased capacity in rich multipath environments [10]. A generic MIMO system maybe shown as in Figure 2.1. MIMO approach is to transmit and receive two or more data streams through a single radio channel. This means the system can deliver two or more times the data rate per channel. By allowing for simultaneous transmission of multiple data streams (Figure 2.1), MIMO multiplies wireless data capacity without using additional frequency spectrum. Peak throughput in MIMO systems increases by a factor equal to the number of signal streams transmitted in the radio channel. Because there are multiple signals, each of which is transmitted from a different radio and antenna, MIMO signals are sometimes called "multidimensional" signals. Conventional radio signals are referred to as "one-dimensional signals" because they only transmit one data stream over the radio channel even if multiple antennas are

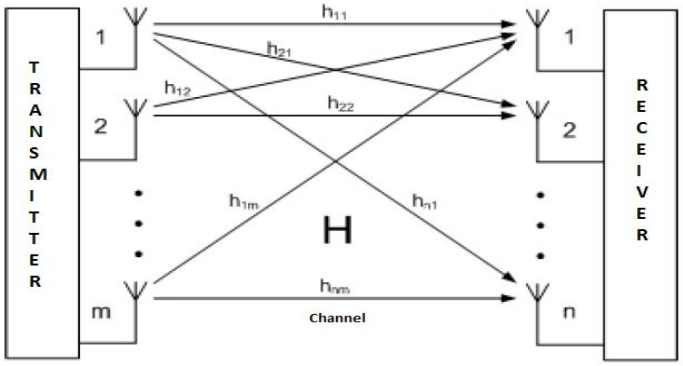


Figure 2.1: A generic MIMO scheme

used [8] [9]. A generic MIMO system maybe represented as below:
A $m \times n$ linear MIMO system with outputs $x_1, x_2, \cdots, x_m$ and inputs $s_1, s_2, \cdots, s_m$ can be expressed as: $x_1 = f_1(s_1, s_2, \cdots, s_m)$, $x_2 = f_2(s_1, s_2, \cdots, s_m), \cdots, x_n = f_n(s_1, s_2, \cdots, s_m)$ where $f_i(.)$ represents a transformation function at the i^{th} receiver end. The MIMO channel input-output relationship may be written as [3]

$$\mathbf{x}(n) = \mathbf{H}(n)\mathbf{s}(n) + \mathbf{v}(n) \tag{2.3}$$

where $\mathbf{x(n)}$ is an $M \times 1$ vector with $x_i(n)$, i= 1, $\cdots$, M as the elements, $\mathbf{s}(n)$ is an $N \times 1$ vector of the input symbols, $\mathbf{v}(n)$ is an additive noise vector, $\mathbf{H}(n)$ is an $M \times N$ channel matrix with elements $h_{ij}(n)$, $i = 1, \cdots, M$, $j = 1, 2, \cdots, N$ denoting the transfer function between the j^{th} transmit and i^{th} receive antennas respectively. Channel related variations modeled by Rayleigh and Rician distribution can also be included in the channel matrix **H**.

Unlike traditional means of increasing throughput, MIMO systems do not increase bandwidth in order to increase throughput. They simply exploit the spatial dimension by increasing the number of unique spatial paths between the transmitter and receiver. However, to ensure that the channel matrix is invertible, MIMO systems require an environment rich in multipath. This has significant ramifications as it suggests that operators can provide broadband services within the current spectrum. Staying at the current carrier frequencies implies that [8] [9]: (a) signals can propagate further thus reducing the cost of overall network deployment and (b) RF subsystems can be built using today's well understood and inexpensive processes.

For fuzzy based MIMO channel modeling described in Chapter 5, the above considerations are a bit modified. Using an OFDM signal for a 4 × 4 MIMO set-up the input-output

relation maybe written as

$$x_i(k+1) = [s_i(k) + s_i(k-\tau)]H(i,k) + v(k) \tag{2.4}$$

where $x_i(k+1)$ is the received signal at time $k+1$, $s_i(k)$ is the transmitted signal state at time k, $s_i(k-\tau)$ is the transmitted signal at time $k-\tau$, $F(.)$ is a mapping function representing the transformation in the transmission, τ is the delay associated with multipath fading and the channel coefficients for i^{th} input and k^{th} time constitutes the channel matrix $H(i,k)$. The expression (2.4) shows the contribution of direct and delayed versions of the signal components while propagating through the MIMO channel. The mapping function $F(.)$ also relates present and delayed versions of the signal **s** along with the channel matrix **H** for formulating a generation mechanism of the received signal sequences which also includes considerable amount of CCI.
In general, the received signal state at time $k+n$ can be expressed as

$$x_i(k+n) = F[s_i(k,\tau), s_i(k), H(i,k)] + v(k). \tag{2.5}$$

In another form, expression of the received signal in a MIMO set-up maybe expressed showing dependence of delayed outputs and other parameters as below

$$\tilde{y}(n) = FM_G[\tilde{y}(n-1), \cdots, \tilde{x}(n-1), \cdots, [\tilde{H}], \cdots] \tag{2.6}$$

where FM_G is a fuzzy mapping process to be generated by an ANN-based process. The expression $\tilde{y}(n)$ becomes the target pattern which should be given to a soft-computational tool during training. The requirement is to design another ANN-based process such that its output is given by

$$\tilde{y_G}(n) = FM_G[\tilde{y}(n-1), \cdots, \tilde{x}(n-1), [\tilde{H}], [\tilde{W}], [\tilde{V}], \tilde{\theta}] \tag{2.7}$$

so that $\tilde{y}_G(n)$ tends to $\tilde{y}(n)$ in terms of a cost function. Here $[\tilde{W}]$, $[\tilde{V}]$ and $\tilde{\theta}$ are forward and backward connectionist weights and biases respectively. Therefore, the training of a fuzzy process is related to the minimization of a cost function expressed as

$$CF = \frac{1}{TN \times VD} \sum \sum d(y_{di}, y_{ai}) \tag{2.8}$$

where TN is the number of training samples, VD is the dimension of the samples, $d(.)$ is a distance measure, y_{di} is the desired output and y_{ai} is the actual output. The channel matrix **H** represents the wireless medium through which the propagation takes place. This channel is considered to be a time variant system with an impulse response given

by [53]:

$$h(t,\tau) = \sum_{g=0}^{N_p-1} \alpha_g(t,\tau) \exp(j(2\pi f_c \tau_g(t) + \phi(t,\tau)))\delta(\tau - \tau_g(t)); \tag{2.9}$$

where N_p is the number of multipath components, $\alpha_g(t,\tau)$ is the amplitude component and $\tau_g(t)$ is the excess delay component caused by the g^{th} multipath component at time t and δ is the delta function. The inverse dynamics allows a definition

$$[\hat{H}(i,k)] = G[x_i(k), x_i(k-\tau)] \tag{2.10}$$

and

$$[\hat{s}_i(k)] = G[x_i(k), x_i(k-\tau)]. \tag{2.11}$$

The received signal states $x_i(k)$ and $x_i(k-\tau)$ can be obtained from expressions (2.4) and (2.5). The mapping $G[.]$ represents a trained soft-computational tool which can apply its learning to estimate the channel coefficients and recover the signal symbols. It signifies a modeling mechanism related to the MIMO channel from which the significant and CCI coefficients can be determined from the received and transmitted signal components. The inverse dynamics resembles a control process the outcome of which, denoted by $\mathbf{H}$, is regulated by factors like the received signal $x_i(k)$, transmitted signal $s_i(k)$ and pathdelay τ. It indicates that some *apriori* knowledge about the transmitted content is required which enables the system to learn the patterns and then use the learning for estimating the channel coefficients and recovering the symbol bits from the received content. The channel matrix $\mathbf{H}$ can be determined from the inverse dynamics $G[.]$ obtained from the following sets of data:

$$[x_i(k), s_i(k), s_i(k-\tau)]. \tag{2.12}$$

The learning of the soft-computational network is oriented to achieve the following objectives:

$$\|e_1(i,k)\|^2 = \|H(i,k) - \hat{H}(i,k)\| \tag{2.13}$$

$$\|e_2(i,k)\|^2 = \|s_i(k) - \hat{s}_i(k)\| \tag{2.14}$$

from which

$$\|e_1(i,k)\|^2 = \|H(i,k) - G[x_i(k), x_i(k,\tau)]\| \tag{2.15}$$

$$\|e_2(i,k)\|^2 = \|s_i(k) - G[x_i(k), x_i(k,\tau)]\|. \tag{2.16}$$

These considerations are followed while designing the MIMO channel modeling approaches described in the subsequent chapters.

2.2.2 Performance gains in a MIMO channel

The capacity of an n_t x n_r i.i.d. Rayleigh fading MIMO channel H with receiver CSI is:

$$C_{nn}(SNR) = E\ [log\ det\,(I_{n_t} + \frac{SNR}{n_t} HH^*)]. \tag{2.17}$$

In an $n \times n$ MIMO channel, the capacity increases linearly with n over the entire SNR range. With channel knowledge at the transmitter, an additional n_t/n_r-fold transmit beamforming gain can be realized with an additional power gain from temporal-spatial waterfilling at low SNR [96].

2.2.3 MIMO channel estimation methods

Channel estimation is an integral part of the receiver designs [55]. For conventional coherent receivers, the effect of the channel on the transmitted signal must be estimated to recover the transmitted information [56]. As long as the receiver can accurately estimate the modification in the transmitted signal, it can recover the content of the signal. Channel estimation can be avoided by using differential modulation techniques, however, such systems result in low data rate and there is a penalty for 34 dB SNR [57] [58]. In some cases, channel estimation at user side is not required when the base station performs the channel estimation and sends a pre-distorted signal [59]. However, for fast varying channels, the pre-distorted signal is unlikely to convey the current state of channel distortion which will reduce reliability of the process. Hence, systems with a channel estimation block are needed for the future high data rate systems [60]. There are two main types of channel estimation for most of the practical systems- blind and non-blind for MIMO systems. Out of all the present channel estimation techniques, training based algorithms have low computational complexity and good robustness in noisy environments [61]. So, these are popular and the most dominant channel estimation techniques. Blind channel estimation techniques have high computational complexity, hence have not been considered for this work. The non-blind channel estimation can be studied under two main groups: decision directed channel estimation and data aided. In decision directed channel estimation (DDCE), for the data detection of the current estimation, channel estimation of a previous symbol is used. The newly detected data is then used for current channel estimation. In DDCE, since an outdated channel is used in the decoding process, so the estimates are less reliable as the channel can vary drastically from symbol to symbol. Hence additional information is usually incorporated in DDCE such as periodically sent training symbols, channel coding, interleaving, and iterative type approach for boosting its performance [60].

In block-type pilot channel estimation, pilot tones are inserted into all subcarriers (pilot

symbols) of signal symbols with a specific period in time. The block-type pilot is developed under the assumption of slow fading channel [60]. Two basic algorithms for block type pilot based channel estimation are LS and MMSE channel estimation. As we have referred to these methods in the work reported in the thesis, we shall briefly focus on them. Other methods are beyond the scope of this thesis.

1. **Least-Square (LS) Estimation**: In real time applications, a prior knowledge about the channel and noise statistics is not possible. It features simplicity, because no consideration of noise and ISI is needed for it. Without using any knowledge of the statistics of the channels, the LS estimator generates the signal samples with low complexity. But this method suffers from a high MSE. So, generally LS method is utilized to get initial estimates for many channel estimation techniques [60].

2. **Minimum Mean Square Error (MMSE) Estimation**:
   As the name suggests, MMSE estimator minimizes the MSE by employing the second order statistics of the channel conditions [62]. The MMSE estimation gives much better performance than LS estimator especially under low SNR scenario [60]. The MMSE estimation shows $10 - 15$ dB gain in SNR for the same MSE over LS estimation [59] [60]. But for systems with high SNR, the performance of MMSE estimation is the same as LS estimation.

Some relevant literature are [55]-[65].

2.2.4 Symbol recovery for MIMO channels

Symbol recovery enables a system to capture the relevant data bits from the received content which we have considered as part of a composite process along with channel estimation. Here, the received signal is taken to be a modified form of that given by equation 2.3. It is expressed as

$$\mathbf{x} = \mathbf{s} * \mathbf{H} + \mathbf{v} \tag{2.18}$$

where $*$ is a convolution operation of the signal $\mathbf{s}$, $\mathbf{H}$ is the channel matrix and $\mathbf{v}$ is the AWGN. The output of a trained classifier like ANN is the signal $\mathbf{x}_N$ which is compared with the received signal $\mathbf{x}$. The symbol recovery is carried out by using $\mathbf{x}_N/\mathbf{s}$ assuming that AWGN $\mathbf{v}$ can be ignored as the training makes the ANN resilient enough to minimize its ill-effects. An error signal $\mathbf{e}$ is obtained such that

$$\mathbf{e} = \mathbf{x} - \mathbf{x}_N. \tag{2.19}$$

Here $\mathbf{s}_N$ is the signal generated by the ANN such that

$$\mathbf{x}_N = \mathbf{s} * \mathbf{H}_N + \mathbf{v} \tag{2.20}$$

with $\mathbf{H}_N$ being the ANN estimated channel matrix. At the trained state $\mathbf{e} \longrightarrow 0$ such that $\mathbf{s}_N \longrightarrow \mathbf{s}$.

Statistical approaches are preferred for performing symbol recovery operation in MIMO systems [66]-[69]. This is because of the fact that these approaches are able to provide satisfactory discrimination capacity and have proven to be reliable with acceptable computational complexities. They use statistical or deterministic properties of the transmit and receive signals, properties such as cyclic and pilot-induced redundancy, cyclostationarity, finite alphabets, virtual carriers etc have been effectively exploited [70]-[72]. ANN approach can be an important addition to this list as it can effectively use its ability to learn and work as part of composite blocks to perform symbol recovery and channel estimation together which is attempted in the subsequent chapters.

2.3 Soft-Computational Tools

Soft computing is the mixture of processes and methodologies that are designed to model and generate solutions to handle real world problems [73]. Soft computing, also called the *Computational Theory of Perception*, is a term first introduced by Dr. Lotfi Zadeh– the father of Fuzzy Logic, in 1981, to mean systems that "exploit the tolerance for imprecision, uncertainty, and partial truth to achieve tractability, robustness, low solution cost and better rapport with reliability". Zadeh visualized the creation of new generation Artificial Intelligence (AI), known as Computational Intelligence (CI) which covered fuzzy logic, neural computing, evolutionary computing and probabilistic computing as main methodologies [73]-[81]. The applications of soft computing have provided two main advantages. First, it made solving nonlinear problems, in which mathematical models are not available, possible. Second, it introduced the human knowledge such as cognition, recognition, understanding, learning, retentivity and others into the fields of computing. This resulted in the possibility of constructing intelligent systems such as autonomous systems. Though soft computation includes several branches like fuzzy logic, neural computing, evolutionary computing, probabilistic computing etc, but here we are concerned only with ANN and fuzzy systems, since our treatment is based on these two techniques in the subsequent chapters.

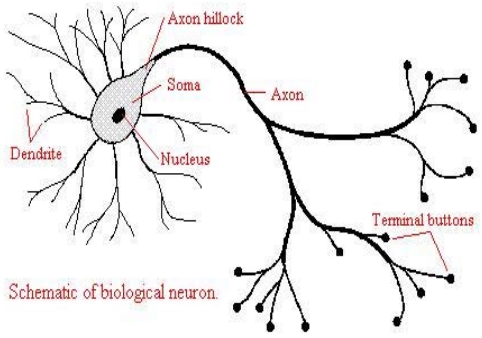


Figure 2.2: A generic biological neuron

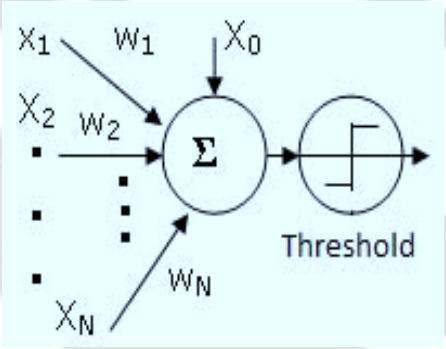


Figure 2.3: A model of an artificial neuron

2.3.1 Artificial Neural Network (ANN)

ANN is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure composed of a large number of highly interconnected processing elements (neurones) working in unison to solve specific problems [34], [82], [83].

1. **Basic Considerations of ANN**: An ANN has certain performance characteristics with some commonality with biological neural networks [83] formed by neurons. A generic biological neuron is shown in Figure 2.2 [34] [83]. A mathematical analogy named the McCulloch-Pitts Neuron (1943) [34] is the primary processing unit of the ANN. The block diagram of Figure 2.3 shows the model of an artificial neuron [34], which forms the basis of designing an ANN. From this model the interval activity of a neuron k can be shown to be

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{2.21}$$

and

$$y_k = \varphi(u_k + b_k) \tag{2.22}$$

where $x_1, x_2, \cdots, x_k$ are the input signals, $w_{k1}, w_{k2}, \cdots, w_{km}$ are the synaptic weights of neuron k, b_k is the bias, $\varphi(.)$ is the activation function and y_k is the output signal of the neuron. The activation function acts as a squashing function or a transfer function, such that the output of a neuron in a ANN is between certain values (usually 0 and 1, or -1 and 1). The basic types of activation functions are, namely, threshold function, linear function, sigmoid function, tan-sigmoid function etc [34]. The learning algorithms define an architectural-dependent procedure to encode pattern information into weights to generate these internal models. These information codes are retained by the adaptation and modification that takes place while the training continues. Learning proceeds by modifying connection strengths [82]. In artificial systems, learning changes the synaptic weights in the model [82]. Learning is data driven. The nature of capturing details of applied samples allows to demarcate learning algorithms into two categories:

(a) **Supervised Learning-** Supervised learning employs the *teaching input* D_k , the associates of X_k to reduce the error (D_k, S_k) in the response the system. Input-output sample pairs are employed to train the ANN through a simple form of error correction learning or gradient descent weight adaptation. These procedure are based on global error measures derived from the difference between the desired (D_k) and actual (S_k) output of the network.

(b) **Unsupervised Learning-** The alternate way of learning is to simply provide the system with an input X_k, and allow it to self-organize its parameters, which are the weight of the network to generate internal prototypes of sample vector. An unsupervised learning system attempt to represent the entire data set by employing a small number of prototypical vectors- enough to allow the system to retain a desired level of discrimination between samples. There is no teaching input. This kind of learning is similar to *adaptive vector quantization* and is essentially competitive in nature.

2. **A Few ANN Classes**: Depending on various learning rules, a few fundamental classes of ANN architectures are defined as mentioned below:

(a) **Single-Layer Feedforward Networks:** In a layered ANN, the neurons are organized in the form of layers. In this simplest form of layered network, there is an input layer of source nodes that projects onto an output layer of neurons, but not vice versa. In other words this network is strictly a feedforward or acyclic type. A single-layer network with R input elements and S neurons is

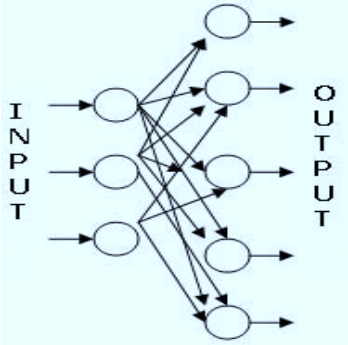


Figure 2.4: Single-Layer Feedforward Networks

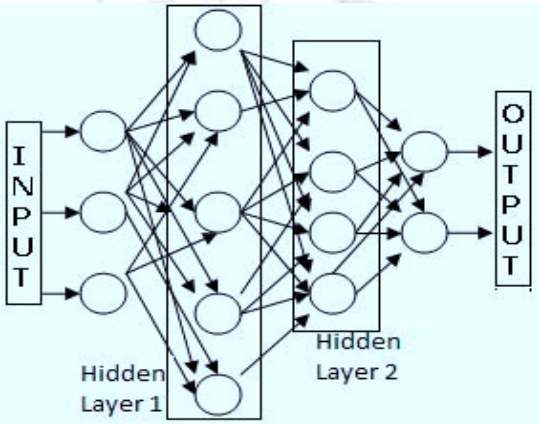


Figure 2.5: Multilayer Feedforward Networks

shown in Figure 2.4. Such a network is called single layer network, with the designation "single-layer" referring to the output layer of computation neurons [34].

(b) **Multilayer Feedforward Networks:** The second class of a feedforward ANN distinguishes itself by the presence of one or more hidden layers in between the input and output layers, whose computation nodes are correspondingly called hidden neurons or hidden units. The function of neurons is to intervene between the external input and the network output in some useful manner. By adding one or more hidden layers, the network is enabled to extract higher-order statistics. The ability of hidden neurons to extract higher-order statistics is particularly valuable when the size of the input layer is large [34]. The architectural graph in Figure 2.5 shows a representation of a simple 3-layer feedforward ANN with a few input, hidden and output nodes [34]. The MLP is the product of several researchers: Frank Rosenblatt (1958), H. D. Block (1962) and M. L. Minsky with S. A. Papart (1988). Backpropagation, the training algorithm, was discovered independently by several researchers (Rumelhart et. al.(1986) and also McClelland and Rumelhart (1988). A sim-

ple perceptron is a single McCulloch-Pitts neuron trained by the perceptron learning rule, given as:

$$O_x = g(([\mathbf{w}].[\mathbf{x}] + b) \tag{2.23}$$

where $\mathbf{x}$ is the input vector, $\mathbf{w}$ is the associated weight vector, $[\mathbf{w}].[\mathbf{x}]$ is the element wise product, b is a bias value and $g(.)$ is the activation function. Such a setup, namely the perceptron is able to classify only linearly separable data. A MLP, in contrast, consists of several layers of neurons. The equation for output in a MLP with one hidden layer is given as:

$$O_{jx} = \sum_{j=1}^{M} \sum_{i=1}^{N} f_{ji}\{[w]_{ji}.[x] + b_i\} \tag{2.24}$$

where f_{ji} represents the i^{th} activation function in the j^{th} layer, w_{ji} is the weight matrix for j^{th} layer value between the i^{th} hidden neuron and $\mathbf{x}$ input patterns with b_i bias. The process of adjusting the weights and biases of a perceptron or MLP is known as *training*. The perceptron algorithm (for training simple perceptrons consists of comparing the output of the perceptron with an associated target value. The most common training algorithm for MLPs is (error) backpropagation (BP). This algorithm entails a back propagation of the error correction through each neuron in the network. Classification by a ANN involves training it so as to provide an optimum decision rule for classifying all the outputs of the network. The training should minimize the risk functional [34] :

$$R = \frac{1}{2N} \sum \|d_j - F(x_j)\|^2 \tag{2.25}$$

where d_j is the desired output pattern for the prototype vector x_j, $\|.\|$ is the Euclidean norm of the enclosed vector, N is the total number of samples presented to the network in training and $F(.)$ is a mapping representing the output of the ANN. The decision rule can be given by the output of the network:

$$y_{kj} = F_k(x_j) \tag{2.26}$$

for the j^{th} input vector x_j applied to the network, the response of which is approximated by the function $F(.)$ for the k^{th} neuron. The MLPs provide global approximations to non-linear mapping from the input to the output layers. Training the ANN is done in two broad passes -one a forward pass and the other a backward calculation with error determination and connecting weight updating in between. Batch training method is adopted as it accelerates the speed of training and the rate of convergence of the MSE to the desired

value [34]. The steps are as below:

i. **Initialization and presentation of training samples**: Initialize weight matrix **W** with random values either between [-1,1] if or $[0, 1]$. Input is $p_m = [p_{m1}, p_{m2}, \cdots, p_{mL}]$. The desired output is d_m=$[d_{m1}, d_{m2}, \cdots, d_{mL}]$.

- Compute the values of the hidden nodes as:

$$net^h_{mj} = \sum_{i=1}^{L} w^h_{ji} p^{mi} + b^h_j \tag{2.27}$$

where w^h_{ji} represent the weight matrix at j^{th} hidden layer for i^{th} neuron receiving p^{mi} patterns.

- Calculate the output from the hidden layer as

$$o^h_{mj} = f^h_j(net^h_{mj}) \tag{2.28}$$

where the value of net^h_{mj} is obtained from the expression (2.27). The function maybe $f(x) = \frac{1}{e^x}$ or $f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ depending upon the activation considered.

- Calculate the values of the output node as:

$$o^o_{mk} = f^o_k(\sum_j (o^h_{mj} w^o_{kj})) \tag{2.29}$$

where the value of o^h_{mj} is obtained from expression (2.28) and w^o_{kj} is the set of weight values for k^{th} previous layer and j^{th} neuron.

ii. **Forward Computation**: Compute the errors:

$$e_{jn} = d_{jn} - o_{jn} \tag{2.30}$$

where d_{jn} is the desired response for the j^{th} neuron for n^{th} sample.
Calculate the MSE as:

$$MSE = \frac{\sum_{j=1}^{M} \sum_{n=1}^{L} e^2_{jn}}{2M}. \tag{2.31}$$

Error terms for the output layer is:

$$\delta^o_{mk} = o^o_{mk}(1 - o^o_{mk}) e_{mk} \tag{2.32}$$

where the value of o^h_{mj} is obtained from expression (2.28) and for k^{th} sample, the term e_{mk} is evaluated using the equation (2.30). Error terms

for the hidden layer:

$$\delta^h_{mk} = o^h_{mk}(1 - o^h_{mk}) \sum_j \sum_k \delta^o_{mj} w^o_{jk} \tag{2.33}$$

where δ^o_{mj} is obtained from expression (2.32) and w^o_{jk} is the j^{th} layer weight for k^{th} neuron in the output layer.

iii. **Weight Update**:

- Between the output and hidden layers

$$w^o_{kj}(t+1) = w^o_{kj}(t) + \eta \delta^o_{mk} o^o_{mj} \tag{2.34}$$

where δ^o_{mk} is obtained from expression (2.33), o^o_{mj} is derived from expression (2.29) for the j^{th} neuron and η is the learning rate ($0<\eta<1$).

- Between the hidden layer and input layer:

$$w^h_{ji}(n+1) = w^h_{ji}(n) + \eta \delta^h_{mj} p^{mi}. \tag{2.35}$$

A momentum term, α, helps in better convergence of the training which is added as:

$$(1-\alpha) w^h_{ji}(n+1) = w^h_{ji}(t) + \eta \delta^h_{mj} p_i - \alpha w_{ji}(n). \tag{2.36}$$

One cycle through the complete training set forms one epoch. The above is repeated till MSE meets the performance criteria. While repeating the above the number of epoch elapsed is counted. A few methods used for MLP training includes: (a) Gradient Descent (GDBP), (b) Gradient Descent with Momentum BP (GDMBP), (c) Gradient Descent with Adaptive Learning Rate BP (GDALRBP) and (d) Gradient Descent with Adaptive Learning Rate and Momentum BP (GDALMBP).

2.3.2 Recurrent Neural Network (RNN)

A RNN is another approach which distinguish itself from a feedforward ANN in that it has at least one feedback loop [34]. A RNN may consist of a single layer of neurons with each or a few of these processing units feeding complete or part of the output signal back to the input side, as illustrated in the architectural graph of Figure 2.6. The RNN may have feedback from the output neurons to the input layer or to the hidden layer. Another possible form of the feed back is from the output to the hidden and next to the input layer from the hidden [34]. The feedback path provides the RNN storage capability

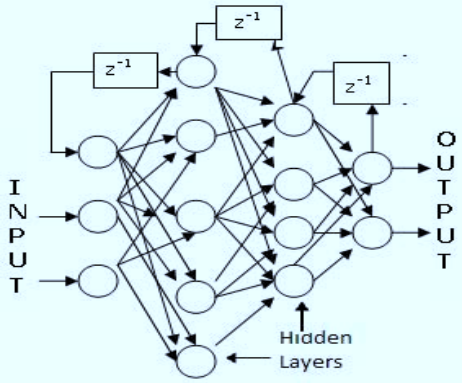


Figure 2.6: Recurrent Neural Network

which makes it dynamic enabling it to track time variation in the input signal. It also facilitates accumulation of learning at each instant of time for which the training is faster and takes lesser number of epochs than the MLP.

- **Learning Algorithms** The basic algorithms for RNNs are summarized below-

  1. **Back-Propagation Through Time:** The back propagation through time (BPTT) algorithm for training of a recurrent network is an extension of the standard back-propagation algorithm. It may be derived by unfolding the temporal of the network into a layered feedforward network, the topology of which grows by one layer at every time step.
  2. **Real-Time Recurrent Learning:** Another learning method, referred to as real-time recurrent network (RTRL), derives its name from the fact that adjustments are made to the synaptic weights of a fully connected RNN in real time, i.e. while the network continues to perform its signal processing function (Williams and Zipser, 1989). The network has two distinct layers: a concatenated input-feedback layer and a processing layer of computation nodes. Correspondingly, the synaptic connections of the network are made up of feedforward and feedback connection.
  3. **Decoupled Extended Kalman Filter (DEKF) and RNN Learning:** Despite its relative simplicity (as compared to recurrent back-propagation), the problem of vanishing gradient makes the RNN training using direct gradient descent algorithm to be a slow process to converge. The solution to this problem has been derived by considering the training of RNN to be an optimum filtering problem similar to that shown by the Kalman filter. Here only previous data requires storage. This algorithm is derived from the Extended Kalman Filter (EKF) algorithm. The EKF is a state estimation technique for nonlinear systems derived from the Kalman filter algorithm considering

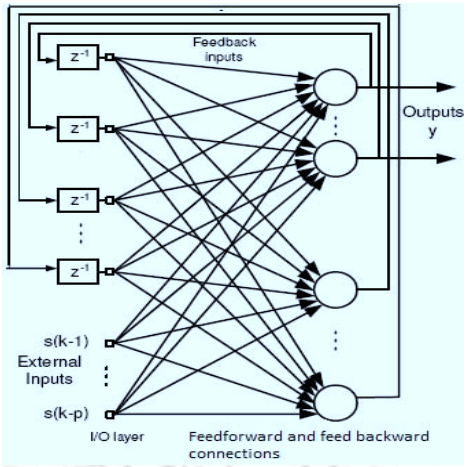


Figure 2.7: Fully connected RNN

the current state estimate. Similarly, by configuring the network with decoupled subnetworks, related learning is carried out with considerable reduction in computations [84] which results in saving training time. This gives rise to the Decoupled Extended Kalman Filter (DEKF) algorithm [34].

4. **Complex - Valued RTRL Algorithm:** Figure 2.7 shows a CFRNN block, which consists of N neurons with p external inputs. The network has two distinct layers consisting of the external input-feedback layer and a layer of processing elements. Let $y_l(k)$ denote the complex valued output of each neuron, $l = 1, \cdots, N$ at time index k and $s(k)$ the $(1 \times p)$ external complex-valued input vector. The overall input to the network $I(k)$ represents the concatenation of vectors $y(k)$, $s(k)$ and the bias input $(1 + j)$, and is given by

$$I(k) = \begin{pmatrix} [s(k-1), \cdots, s(k-p), 1+j, y_1(k-1), \cdots, y_N(k-1)]^T \\ I_n^r(k) + jI_n^i(k), n = 1, \cdots, p+N+1. \end{pmatrix} \quad (2.37)$$

where $j = \sqrt{-1}$, $(.)^T$ denotes the vector transpose operator and the superscripts $(.)^r$ and $(.)^p$ denote, respectively,the real and imaginary parts of a complex number. A complex valued weight matrix of the network is denoted by W, where for the i^{th} neuron, its weights form a $(p+F+1) \times 1$ dimensional weight vector $W_l = [w_{l,1,\cdots}, w_{l,p+F+1}]^T$ where F is the number of feedback connections. The feedback connections represent the delayed output signals of the CFRNN. The output of each neuron can be expressed as

$$y_l(k) = \phi(net_l(k)), l = 1, \cdots, N. \quad (2.38)$$

Table 2.1: Computational complexity of RNN training algorithms

Sl Num	Algorithm	Computational Complexity
1	BPTT	Time and storage, O(WL+SL)
2	RTRL	Time = $O(WS^2L)$, Storage=O(WS)
3	DEKF	Time=$O(p^2W + p\sum_{i=1}^{g} k_i^2)$ Storage=$O(\sum_{i=1}^{g} k_i^2)$ p is the no. of outputs, g is the no. of groups k_i is the no. of neurons in group k_i
4	CRTL	Time complexity= $O(N^4)$ Storage complexity= $O(N^3)$

where

$$net_l(k) = \sum_{n=1}^{p+N+1} w_{l,n}(k) I_n(k) \tag{2.39}$$

is the net input to l^{th} node at time index k. For simplicity, we state that

$$y_l(k) = \phi^r(net_l(k)) + j\phi^i(net_l(k)) = u1(k) + jv_l(k) \tag{2.40}$$

$$net_l(k) = \sigma_l(k) + j\tau_l(k) \tag{2.41}$$

where ϕ is a complex nonlinear activation function [85].
The output error consists of its real and imaginary parts and is defined as

$$e_l(k) = d(k) - y_l(k) = e_l^r(k) + je_l^i(k) \tag{2.42}$$

$$e_l^r(k) = d^r(k) - u_l(k), e_l^i(k) = d^i(k) - v_l(k), \tag{2.43}$$

where $d(k)$ is the teaching signal. For real time applications the cost function of the RNN is given by (Widrow et al., 1975) [85],

$$E(k) = \frac{1}{2}\sum_{l=1}^{N} \mid e_l(k) \mid^2 = \frac{1}{2}\sum_{l=1}^{N} e_l(k)e_l^*(k) = \frac{1}{2}\sum_{l=1}^{N}[(e_l^r)^2 + (e_l^i)^2], \tag{2.44}$$

where $(.)^*$ denotes the complex conjugate.

5. **Computational Complexity of BPTT, RTRL, DEKF and CRTRL Algorithms:** Let S be the number of states, W be the number of synaptic weights and L be the length of the training sequence. The computational complexity [34] of these algorithms maybe summarized as in Table 2.1.

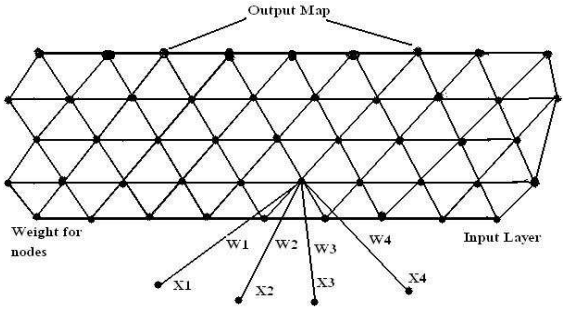


Figure 2.8: Self Organizing Map

2.3.3 Self Organizing Map (SOM)

Self organizing map (SOM) is a type of ANN developed by Kohonen, (Figure 2.8) preferable for reducing dimensions. It is done by producing a map of usually one or two dimensions that plots the similarities of the data by grouping the closely matching data items together. The SOM uses an unsupervised learning method to map high dimensional data into a $1-D$, $2-D$ or $3-D$ data space, subject to a topological ordering constraint. A major advantage is that the clustering produced by the SOM retains the underlying structure of the input space, while the dimensionality of the space is reduced [86]. The following are among some of methods used for training and weight updating of SOMs as part of competitive learning [82] [86]: (a) Inner Product and (b) Euclidean Distance Based Competition. A detailed account of the two methods are given in [34] [82]. A SOM follows competitive learning which is based upon a philosophy called 'winners take all'. Competitive learning requires that the weight vector of the winning neuron be made to correlate more with the input vector. This is done by perturbation of only the winning weight vector $W_J = (w_{1J}, \cdots, w_{nJ})^T$ towards the input vector. The scalar implementation of this learning law in difference form is presented below:

$$w_{iJ}^{k+1} = w_{iJ}^{k} + \eta x_i^k \qquad i = 1, \cdots, n. \tag{2.45}$$

The weight grows without bound for which some form of normalization is appended as

$$w_{iJ}^{k+1} = w_{iJ}^{k} + \eta(\frac{x_i^k}{\sum_j x_j^k} - w_{iJ}^{k}) \qquad i = 1, \cdots, n. \tag{2.46}$$

This expression leads to total weight normalization so that the total weight update is given as:

$$\tilde{W}_{J(k+1)} = \tilde{W}_{J(k)} + \eta(1 - \tilde{W}_{J(k)}) \tag{2.47}$$

where $\tilde{W}_{J(k)} = \sum_{i=1}^{n} w_{iJ}^{k}$. If the inputs are already normalized then normalization is not required and weight update is direct:

$$w_{iJ}^{k+1} = w_{iJ}^{k} + \eta(x_i^k - w_{iJ}^{k}) \qquad i = 1, \cdots, n. \tag{2.48}$$

A standard competitive form of the competitive learning in discrete time is given as:

$$w_{ij}^{k+1} = w_{ij}^{k} + \eta s_j^k (x_i^k - w_{ij}^{k}) \qquad i = 1, \cdots, n \quad j = 1, \cdots, m \tag{2.49}$$

where $s_j^k = 1$ only for $j = J$ and is zero otherwise, for a hard competitive field. This represents the most suitable weight value obtained from the competitive learning mechanism which makes it appropriate for an optimization process. The optimization thus generated is fast and reliable. These aspects of the SOM are applied for designing certain modeling mechanism of the time-varying MIMO channels using RNN and fuzzy systems which are described in detail in Chapter 4 and 5.

2.3.4 Fuzzy systems

Fuzzy systems were derived from the work in the field of Fuzzy Logic carried out by Zadeh (1965). The invention or proposition of *Fuzzy Set*s was motivated by the need to capture and represent the real world with its fuzzy data due to uncertainty. Uncertainty can be caused by imprecision in measurement due to several factors and other reasons. Zadeh showed the limitations of *Crisp Set Theory* in representing such descriptions and classifications. Unlike crisp sets, fuzzy systems generate intermediate steps or stages defined using membership functions. Crisp sets can be useful for many situations but cannot quantify situations like 'very young, young, not so old' related to age or 'desirable city to live in' etc. Hence, crisp sets are a special case or a subset, of fuzzy sets, where elements are allowed a membership degree of 100% or 0% but nothing in between [77]. Some of the advantages of fuzzy logic can be summarized as (a) formulation of conceptually easy set of rules for input processing and decision making, (b) flexibility, (c) tolerance to imprecise data, (c) ability to model nonlinear functions of unknown complexity, (d) capability to relate previous experience and knowledge for deriving decisions, (e) work in concert with conventional techniques and improve performance and (f) formulation of fuzzy systems using basic principles of natural language. These attributes make the fuzzy systems suitable for modeling of MIMO channels as described in Chapter 5. Fuzzy sets are characterized by membership functions which assigns to each element x in a fuzzy set a number, $A(x)$, in the closed unit interval $[0, 1]$. The number, $A(x)$, represents the degree of membership of x in A. In the case of crisp sets, the members of a set are either out of the set, with membership degree of zero, or in the set, with the value one

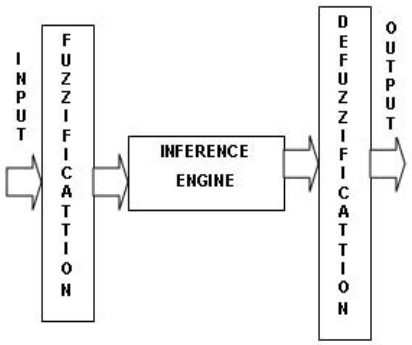


Figure 2.9: Components of a fuzzy system

being the degree of membership. The primary components of a fuzzy-based system are fuzzification, inference and defuzzification stages which is shown in Figure 2.9. But there are problems associated with fuzzy systems as well. Some of them can be: (i) Stability: No theoretical guarantee can be assumed regarding the fact that a general fuzzy system remains stable. (ii) Learning capability: Fuzzy systems lack capabilities of learning and have no memory. It necessitates hybrid systems, particularly neuro-fuzzy system (NFS) and fuzzy-neural system (FNS). (iii) Membership functions and fuzzy rules: Defining appropriate membership functions and fuzzy rules are not always easy. It is always a complicated task to determine membership functions and related rules. Fuzzy systems can be broadly categorized into two families. The first includes linguistic models based on collections of $IF - THEN$ rules, whose antecedents and consequents utilize fuzzy values. It uses fuzzy reasoning and the system behavior described in natural terms. The Mamdani model [78] falls in this group. The second category, based on Sugeno-type systems [79], uses a rule structure that has fuzzy antecedent and functional consequent parts. The approach approximates a nonlinear system representing both qualitative and quantitative information and allow relatively easier application of powerful learning techniques for their identification from data [79]. They are capable of approximating any continuous real-valued function on a compact set to any degree of accuracy [79] [35].

2.3.5 Fuzzy neural systems

The need for developing hybrid systems that combine fuzzy technology and ANN was motivated by the shortcomings and the complementary nature of each of the two methodologies. The performance of ANN becomes degraded and less robust when the inputs are not well defined, i.e., fuzzy inputs. On the other hand, fuzzy systems are not capable

of learning. Moreover, the fuzzy rules that define the input/output relationship must be known in advance. Therefore, combining the two technologies to create hybrid systems that fill the gabs of one paradigm by means of the other is highly motivated [90].
Both fuzzy systems and ANN's are soft computing approaches formulated to deal with uncertainty and provide decision with expert-level knowledge and behavior. The goal is to mimic the actions of an expert who solves complex problems [35]. Both ANN and fuzzy system are dynamic, parallel processing systems that estimate input-output functions. They estimate a function without any mathematical model and learn from experience with sample data. A fuzzy system adaptively infers and modifies its fuzzy associations from representative numerical samples. ANNs, on the other hand, can blindly generate and refine fuzzy rules from training data [73]. Fuzzy sets are considered to be advantageous in the logical field, and in handling higher order processing easily. The higher flexibility is a characteristic feature of ANNs produced by learning and, hence, this suits data-driven processing better [74] [35].
Fuzzy logic and ANNs are suited for contrasting application requirements. For example, fuzzy systems are appropriate if sufficient expert knowledge about the process is available, while ANNs are useful if sufficient process data are available or measurable. ANNs are treated in a numeric quantitative manner, whereas fuzzy systems are handled in a symbolic qualitative manner. Fuzzy systems, however, exhibit both symbolic and numeric features. Therefore, neural and fuzzy systems together creates a frame work which leads to a symbiotic relationship in which fuzzy systems provide a powerful framework for expert knowledge representation, while ANNs provide learning capabilities and exceptional suitability for computationally efficient hardware implementations. The significance of this integration becomes even more apparent by considering their disparities. ANNs don't provide a strong scheme for knowledge representation, while fuzzy logic controllers do not possess capabilities for automated learning [35].

2.4 Conclusion

In this chapter, initially we have dealt with basic characteristics of the MIMO channels and the conventional statistical channel estimation methods and symbol recovery techniques. Next, we covered briefly the fundamental considerations of the ANN and some of its special versions used for the work included in this thesis. Next we included the basics of a fuzzy system which can be efficiently combined with a neural network for modeling MIMO channels. Since the principal objective of the next three chapters is to model efficiently a MIMO channel by means of different kind of soft-computing tools, the basic concepts presented here in this chapter would help readers understand the mathematical deductions in the sequel. Apart from ANN and RNN, fuzzy based composite systems are

preferred in our work over GA approaches for reasons as explained later in Chapter 5 and thus the introductory notion of fuzzy systems have also been taken up in this chapter.

Chapter 3

MIMO Channel Modeling using MLP and its Temporal Form

3.1 Introduction

In general, wireless channels, including MIMO channels, are modeled by statistical means. Estimation of channels, and symbol recovery as a consequence, however, can be done better if the transmitter, channel and receiver side information (TCRSI) can be utilized in proper manner as is done in adaptive systems. Soft-computational tools too, due to their ability to learn, can use TCRSI and contribute towards better performance [101] to estimate MIMO channels. They, however, prove to be effective mainly for slowly varying channels as is suggested by most of the literature [11]-[17]. It should also be noted that although statistical approaches are preferred for performing symbol recovery operation [66]-[69], an ANN can also accomplish the same. Nevertheless, symbol recovery in MIMO wireless systems using ANN has not yet been reported to the best of our knowledge due of the fact that in the rudimentary form, the ANN fails to make discrimination between finely varying input patterns which is a frequently observed event in the communication arena. Therefore, the primary challenge in this work has been to devise a proper form of ANN so that not only the channel estimation can be done, but the task of symbol recovery can be achieved as well. The next challenge has been to incorporate the temporal behaviour in the MLP [102]-[104] with suitable architectural modifications enabling it to track the time-varying nature of the input signals properly.

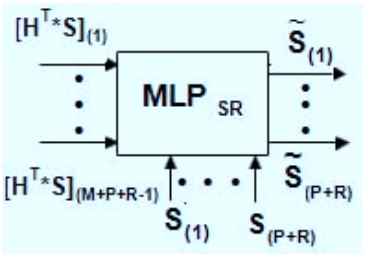
In this chapter, we propose a proper ANN based modeling of the MIMO channels under time-varying multipath fading environment. In particular, the first challenge is treated with two different MLP structures for training, validation and testing purpose, one for channel estimation and the other for symbol recovery. The second challenge is met by replacing the synaptic weights shared by the artificial neurons in input, hidden and output

layers by the well known Finite Impulse Response (FIR) and Infinite Impulse Response (IIR) units. Here, in specific, two modified architectures of MLP are proposed that can effectively deal with practical time variations of a MIMO channel and can be used both for channel estimation and symbol recovery. In the basic form, the connectionist weights between the neurons of the MLP are replaced by FIR and IIR links [102]-[104]. These links are used in several forms and combinations which facilitate the formation of a few MLP structures that demonstrates better ability to track time-varying properties of signals propagating through a MIMO channel. The work is also extended at the end to include OFDM transmissions since the combination of MIMO-OFDM together provides greater spatial multiplexing gain [91]-[100].

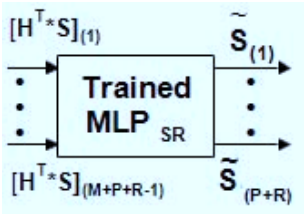
3.2 Training, Validation and Testing of ANN

Accuracy of a prediction and estimation process depends, to a large extent, on enhanced accuracy of classification. In case of ANN, this is dependent on the configuration and the training [34]. The specific ANN type considered here in this work is a MLP which is a class of feedforward structure. The application of the MLPs for MIMO channel modeling considers multiple aspects. The first is the training, next is the validation and finally checking of the robustness of the MLP under a range of channel conditions. The training is carried out by using signal inputs from the transmitter section. The set-up is so constituted that it can tackle any estimation problem despite the presence of irregularities with no parametric dependence. The (error) back-propagation (BP) algorithm used for training often suffers from more than one problem leading to difficulties in MSE convergence. The training continues till the MSE convergence attains the desired level. Here, two simple MLP structures are considered, one named MLP_{SR} intended to perform symbol recovery and the other named MLP_{CE} trained to carry out channel estimation. In general, if we consider a received signal with a channel matrix $\mathbf{H}$ of dimension $M \times N$, as given in equation (2.3), where N is the number of transmitting antennas and M is the number of receiving antennas as before and an input symbol $\mathbf{s}$ of dimension $(R + P) \times 1$ where R is the number of received bits and P denotes the number of parity bits, then for each row of $\mathbf{H}$, the length of the computed convolution operation $[\mathbf{H}^T * \mathbf{s}]$ would be $(M + P + R - 1)$. Therefore, the symbol recovery of the order $(P + R) \times 1$ and the estimation of prime channel coefficient of the order $(Q \times N)$ can be achieved using this convolution operation as shown in Figure 3.1. A shown in Figure 3.1 (a) and Figure 3.1 (c), the MLP_{SR} and MLP_{CE}, during the training phase, are provided with the respective symbol sequences and prime channel coefficients and put into the validation stage only after attaining a desired MSE during the training.

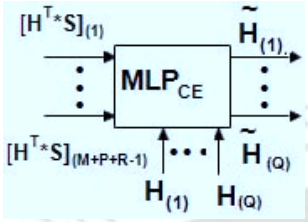
For simulation purposes, we use two different forms of Clarke-Gans channel model [53]



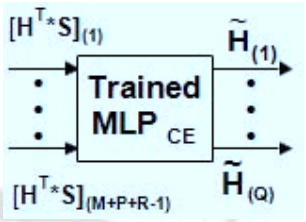
(a) Training for symbol recovery



(b) Validation for symbol recovery



(c) Training for channel estimation



(d) Validation of channel estimation

Figure 3.1: Arrangement of ANN to perform symbol recovery and channel estimation. The operations shown are row wise.

each generated using different AWGN inputs viz. ±3dB, ±5 dB and ±10 dB for each of the Gaussian, Rayleigh and Rician faded channels. In particular, for a 64-bit received sequence with 7-cyclic prefixes (as done with OFDM) and a twenty tap channel vector in a 4-path MIMO set-up, the received signal has a length 90 with four rows of which 90 is the number of neurons in the input layer. For the symbol recovery part, the output layer has a size of 71. For the MLP performing channel estimation, the channel vector has a length of 20 of which only four significant coefficients are considered.

3.3 Some Results Derived using the Basic ANN Configuration

[1] Using the basic ANN configurations as shown in the last section, a discrete time multipath fading channel considered in the modeling process is simulated using the Clarke and Gans model with a pass-band of 0.2 GHz to 8 GHz to represent a frequency range suitable for mobile communication. Four channels are used to avoid complications to begin with which can be extended further. The experiments, though are carried out with flat response generated using Butterworth filter, can include equi-ripple behavior in the

[1]Please note that in this chapter and subsequent chapters, the experiments are usually carried out in MATLAB with some specialized toolboxes such as Signal Processing Toolbox, Neural Network Toolbox, Fuzzy Toolbox and the Communication Toolbox on a Dual Core machine @2Ghz with 1GB DDR RAM.

Table 3.1: Parameters used for simulating channel using Clarke-Gans model

Sl Num	Parameter	Value
1	Freq., f_c	$900MHz$
2	ω_c	$2\ \pi\ f_c$
3	Mobile Speed, V	3 kmph
4	No. of paths	8
5	Wavelength, λ	$\frac{3\times10^8}{f_c}$
6	Doppler shift, f_m	$\frac{V}{\lambda}$
7	Sampling Freq., f_s	$8 \times f_m$
8	No of samples, N	10000
9	Paths	16
10	Sampling Period, T_s	$\frac{1}{f_s}$

channel response which is a condition that offers greater variation in the patterns for the ANN to go through the learning process and capture the relevant portion. Another set of characteristics of the channel model developed using the Clarke-Gans formulation considers the parameters as given in Table 3.1. The OFDM signals generated for the work has a center frequency at 100 MHz. The training is carried out by using signal inputs from the transmitter section. The signal inputs used for ANN training are OFDM symbols transmitted through the MIMO channel. An illustrative case maybe a sample set of 6400 bits with OFDM block size of 64, cyclic prefix length of 7, carriers numbering 64 at intervals of 4 with a maximum delay of 10 μs taking channel length of 20, a Doppler shift of maximum 100 Hz and re-transmissions of atleast 10 times. At a given point of time, for this case 71 bit blocks are transmitted through the channel and the training performed. The channel conditions are altered between Rayleigh and Rician with certain SNR variations. The signal symbols and channel-coefficient combined samples used for training helps the ANN to act as a combined channel estimator and OFDM symbol recovery system. The arrangement for training the MLPs for symbol recovery and channel estimation is shown in Figure 3.1 (a)-(d). Several configurations of the MLP can be utilize for training. The ANN configurations used have one input layer, one hidden layer and one output layer. A single hidden layered MLP is found to be computationally efficient for the work as 2-hidden layered or a 3-hidden layered MLPs are found to be showing no significant performance improvement at the cost of slowing down training. The choice of the length of the hidden layers have been fixed by not following any definite reasoning but by using trial and error method. For this case several sizes of the hidden layer have been considered. Table 3.2 shows the performance obtained during training by varying the size of the hidden layer. The case where the size of the hidden layer taken to be 1.5 times to that of the input layer is found to be computationally efficient. Its MSE convergence rate and learning ability is found to be superior to the rest of the cases.

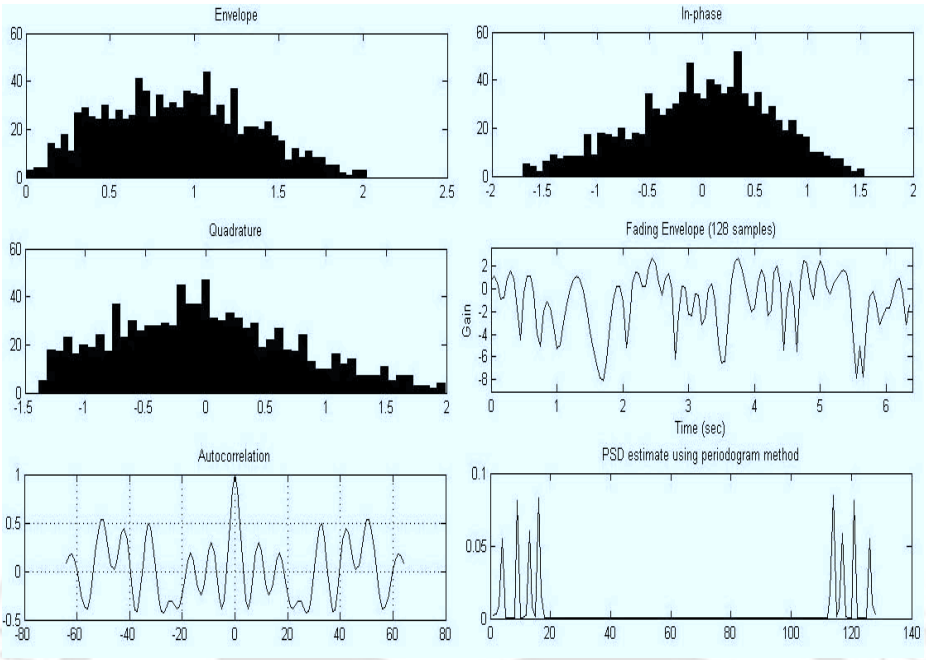


Figure 3.2: Fading channel simulated using Clarke and Gans model

Table 3.2: Performance variation after 1000 epochs during training of an ANN with variation of size of the hidden layer

Case	Size of hidden layer (x input layer)	MSE Attained	Precision attained in %
1	0.75	1.2 x 10^{-3}	87.1
2	1.0	0.56 x 10^{-3}	87.8
3	1.25	0.8 x 10^{-4}	87.1
4	1.5	0.3 x 10^{-4}	90.1
5	1.75	0.6 x 10^{-4}	89.2
6	2	0.7 x 10^{-4}	89.8

Table 3.3: Effect on average MSE convergence after 1000 epochs with variation of activation functions at input, hidden and output layers

Case	Input layer	Hidden Layer	Output Layer	MSE x 10^{-4}
1	log-sigmoid	log-sigmoid	log-sigmoid	1.45
2	tan-sigmoid	tan-sigmoid	tan-sigmoid	1.32
3	tan-sigmoid	log-sigmoid	tan-sigmoid	1.01
4	log-sigmoid	tan-sigmoid	log-sigmoid	1.08
5	log-sigmoid	log-sigmoid	tan-sigmoid	1.15
6	log-sigmoid	tan-sigmoid	log-sigmoid	1.19

Hence, the size of the hidden layer of the ANNs considered is 1.5 times to that of the input layer. The size of the input layer depends upon the length of the input vector and the output layer represents the number of coefficients retained. The channel length is taken to be of length 20 for which the four path MIMO set-up creates a 4×20 matrix which is convolved with the transmitted signal determines the size of the output layer. Noise free and noised data were used for the training.

The training of the ANN for the MIMO channel modeling considers four ANN training methods to ascertain the best configuration for testing. These are Gradient Descent (GDBP), Gradient Descent with Momentum BP (GDMBP), Gradient Descent with Adaptive Learning Rate BP (GDALRBP) and Gradient Descent with Adaptive Learning Rate and Momentum BP (GDALMBP). The selection of the activation functions of the input, hidden and output layers plays another important part in the performance of the system. A common practice can be to use a similar type of activation function in all layers. But certain combinations and alterations of activation function types carried out during training provide improved learning and show a way to attain better performance. Two types of MLP configurations are considered- the first type constituted by a set of similar activation functions in all layers of the ANNs and the other with a varied combination of activation functions in different layers. Both these two configurations are trained with GDMALBP as a measure of training performance standardization. A set of results derived while performing a series of trials for fixing activation functions of the MLPs are included in Table 3.3.

Immediately after the training component is over, the performance is validated using a set of arrangements shown in Figure 3.1 (a)-(d). A data set formed from the training samples with $\pm 10\%$ variation is used for the purpose. At the end of the validation stage, the ANNs are ready to perform both symbol recovery and channel estimation together. The simulations carried out are designed with components providing it with similarities with certain practical situations. The training considerations include time and frequency domain approaches of generation of signal samples in different block sizes like 64, 128,

Table 3.4: Capacity (b/s/Hz) achieved by MIMO-channels

T_X-R_X	SNR=12 dB	SNR=24 dB
(1, 1)	0.3	1.8
(2, 2)	2.2	7.2
(3, 3)	5.8	10.8
(4, 4)	7.5	26.2
(5, 5)	12.5	28.5

6400 and 64,000 bit iterated upto 10 times between four transmitters and receivers in the MIMO set-up with Rayleigh and Rician fading under varying SNR conditions. It creates considerable amount of uncertainty which the systems are trained to handle. Moreover the experiments are repeated with multiple block size pilots with the OFDM transmissions to enable pattern-learning by the MLP blocks for proper identification and recovery of the transmitted data. The MLP configurations are tested for VOIP based broadcast under severe fading. The first VOIP transmissions involve 64 and 128-bit size OFDM data blocks of around fifty while the second set of broadcasts involve over a thousand 6400 and 64,000-bit block transmissions. The training is faster with the first set for the VOIP application compared to the case when samples of 6400 and 64,000-bit blocks are used. The above is repeated for Gaussian, Rayleigh and Rician channels and average BER values noted for SNR values between -10 to 10 dB.

3.3.1 Performance of the MIMO channel

MIMO is an important means to use the available bandwidth effectively. Unlike traditional means of increasing throughput, MIMO systems do not increase bandwidth in order to increase throughput. It simply exploits the spatial dimension by increasing the number of unique spatial paths between the transmitter and receiver. The application of MIMO-principles increase the effective utilization of the channels. Table 3.4 shows that compared to a $(1, 1)$ T_X-R_X block a $(5, 5)$ pair achieves over 40 times more capacity utilization.

3.3.2 ANN based symbol recovery

The MLPs are tested to check their ability to perform symbol recovery under fluctuating conditions shown by the MIMO channels. The arrangement shown in Figure 3.1 (a)-(b) makes the MLP ready to perform symbol recovery during testing. The experiments are performed with data blocks ranging between 64 to 6400 bit sizes. For slowly varying cases, the results are satisfactory. The average performance is around 94%. The MLPs, however,

Table 3.5: Truncated data set used to train an ANN for four different path delays and four different frequency selective paths of a Rayleigh faded channel

Case	1	2	3	4
Delays	1 x 10^{-5}	1.5 x 10^{-5}	2 x 10^{-5}	2.5 x 10^{-5}
Path gain 1	0.467	0.353	0.555	0.036
Path gain 2	0.281	0.031	0.815	0.732
Path gain 3	0.367	0.893	0.462	0.381
Path gain 4	0.212	0.691	0.021	0.811

Table 3.6: Truncated data set used to train an ANN for channel estimation under Rician fading

Case	1	2	3	4
Delays	1 x 10^{-5}	1.5 x 10^{-5}	2 x 10^{-5}	2.5 x 10^{-5}
Path gain 1	0.136	0.446	0.125	0.162
Path gain 2	0.087	0.306	0.595	0.433
Path gain 3	0.761	0.239	0.064	0.213
Path gain 4	0.572	0.196	0.351	0.087

show an inability to deal with the time-varying nature of the MIMO channels which is reflected during symbol recovery. When certain training data with minor variations in content are repeated for certain intervals of time, the MLP responds with a specific set of values. Despite repeated variations in the content, this response shows nearly no change. It indicates that the MLPs find it difficult to generalize and have retained or memorized a fixed set of patterns. The training data, therefore, is again applied with time-varying channel conditions and the signal symbols repeated. The system shows no improvement. The MLPs, when face nearly similar patterns being applied to it within certain interval of time, shows lower performance in tracking the differences between samples applied at different intervals of it.

3.3.3 ANN based channel estimation

Principles governing training of the ANN have been included in Section 2. Tables 3.5 and 3.6 show data sets used to train an ANN for four different path delays and four different frequency selective paths of Rayleigh and Rician faded channels generated using Clarke-Gans model [53]. The size of the data involved in the training is not very large, hence a careful selection of training sessions is an important criteria to prevent over training of the ANN and thereby ensure optimum performance. If the ANN is over-trained it will fail to model the channel in general. Rather, it might memorize a few coefficients with higher probability. The number of training epochs required, therefore, are limited to a few hundreds only depending upon the size of the data considered. For

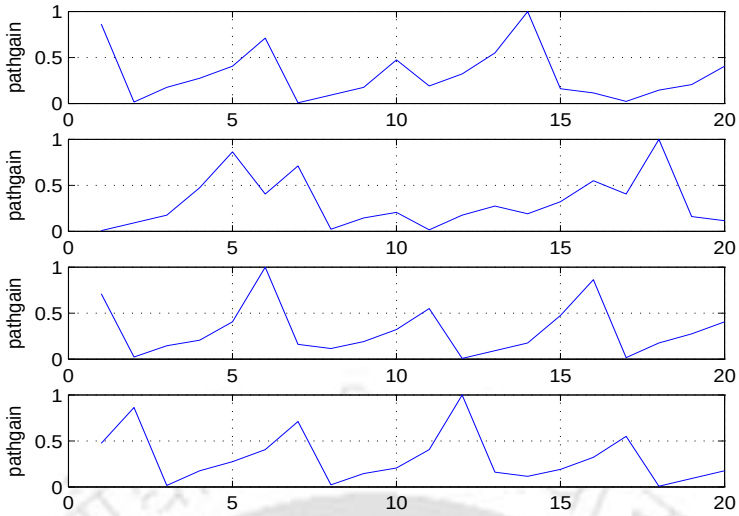


Figure 3.3: A few samples of channel path gains used for training MLP

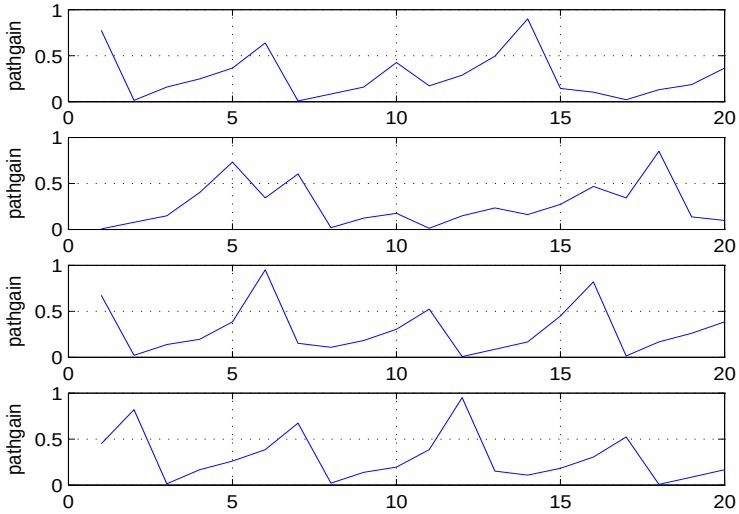


Figure 3.4: Channel path gains generated by MLP after a few hundred training sessions when used with 64-bit data blocks

64-bit data blocks, training epochs are within a few hundred. But for larger size of data, the training is confined to a few thousands to enable the ANN learn the patterns appropriately. Figure 3.3 shows a set of four channel path gain samples used with signals for training the MLP. At the end of around 150-200 sessions (average of ten trails), the MLP estimates a set of channel coefficients that have path gains as shown in Figure 3.4. These patterns show that with training upto a few hundred, the MLP is able to track the channel coefficients properly. But when the training is extended to a few thousands (1000-1300 for ten trails), the proper tracking which the MLP showed in the previous

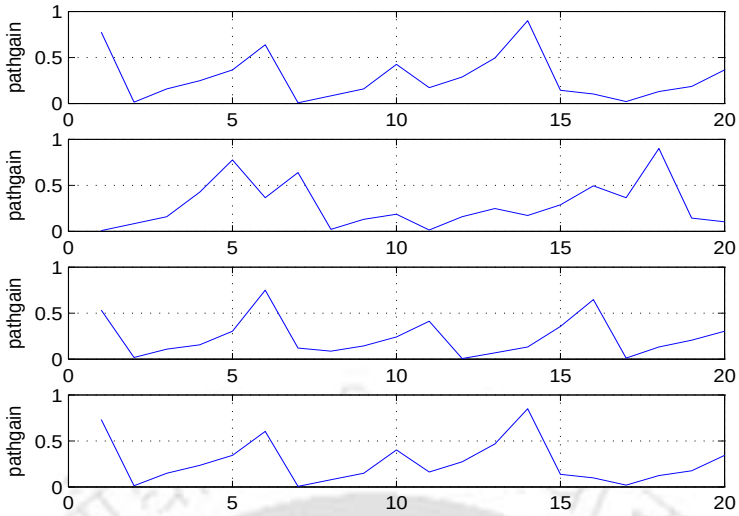


Figure 3.5: Channel path gains generated by MLP with extended training upto a few thousand sessions when used with 64-bit data blocks

case is not observed. Instead, the MLP captures a few variations with greater probability, memorizes and generates them in response to a set of inputs applied to it (Figure 3.5). It is a case of over training and the MLP failing to generalize.

The performance achieved during these epochs are noted as follows: the ANN training considers the MSE convergence and precision generated in channel estimation, symbol recovery and the BER calculation. If the MSE has converged to the fixed target value, the precision levels and the associated BER values are determined. If the values fall within the desired levels, the training is extended to include more number of samples which represent varying channel conditions. The iterations are confined to only to a few hundreds during which a minimum MSE convergence of 0.006 x 10^{-2} has been attained using the GDALRMBP training method. The ANN trained by following these considerations is taken for performing the channel estimation. The MSE values attained using different methods of during training is shown in the Table 3.7. The MSE obtained using GDALRBP is the lowest within the given training sessions but such values generated by other training methods are comparable too. Table 3.8 shows the precision performance of a 1-hidden layered MLP trained with the four mentioned training methods. The highest precision is around 92.5 % which is attained by the GDMALRBP based training. After the training aspects are properly addressed, a validation process is carried out which is followed by testing. Out of the total data, about 30% are used for training, 15% for validation and rest for testing. The greatest strength of the ANN in such applications is related to the opportunity that the system derives in extending the performance domain by adopting better configuration and allowing increased number of sessions to continue

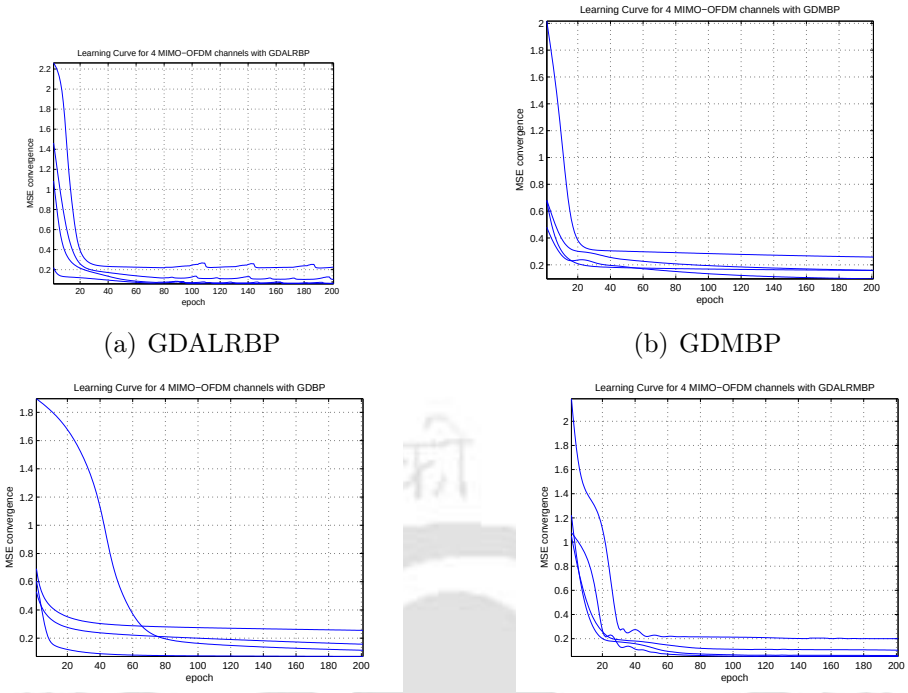
Table 3.7: MSE attained during training by a one-hidden layered MLP with a learning rate of 0.4

Sessions	MSE attained with four different training methods			
	GDBP	GDMBP	GDALBP	GDMALRBP
100	4.1×10^{-2}	3.1×10^{-2}	2.6×10^{-2}	1.23×10^{-2}
200	0.42×10^{-2}	0.31×10^{-2}	0.21×10^{-2}	0.12×10^{-2}
300	0.21×10^{-2}	0.07×10^{-2}	0.04×10^{-2}	0.03×10^{-2}
400	0.02×10^{-2}	0.01×10^{-2}	0.01×10^{-2}	0.006×10^{-2}

Table 3.8: Precision performance in % of channel estimation during training by a 1-hidden layered MLP trained with four different raining methods with a learning rate of 0.4

Sessions	Precision % for different training methods			
	GDBP	GDMBP	GDALBP	GDMALRBP
1000	76.1	79.3	83.3	84.2
2000	77.9	81.2	84.2	85.7
3000	79.1	82.3	86.6	89.4
4000	81.9	84.4	88.6	92.5

the learning till the desired performance levels are attained. The learning patterns and thereby performance of ANN varies with training method. Faster the learning, greater is the chance of the ANN falling a local trap where the convergence curve oscillates around a local minima. No such problems have been observed in the present case with all the four training methods. A set of learning curves of the MLPs during training are shown in Figure 3.6 (a)-(d). A BER plot (Figure 3.7) is generated using training and and testing samples to ascertain the performance of the system designed to perform channel estimation. The MLP trained for a few hundred sessions, track the variations in the input samples. The performance difference between the training and the testing samples is less than 10% despite the fact that the sample variation is around 15-20%. Another set of test samples are used to generate a BER plot (Figure 3.8) which is compared with an estimator with perfect channel state information (CSI). The perfect CSI state is given to an estimator by training it with a set of BER values which are generated for the considered MIMO set-up and fading conditions with *apriori* knowledge derived from the theoretical limits. A few results are derived for the statistical channel estimation methods as well. The results shown are derived from about fifty trials with a host of variations including SNR and background noise. The MLP based system is found to be resilient enough in dealing with multipath variations observed in MIMO channels. Figure 3.9 shows a comparative plot between LS, MMSE and MLP based methods of BER value



(a) GDALRBP (b) GDMBP

(c) GDBP (d) GDALRMBP

Figure 3.6: ANN learning curves for four channels with

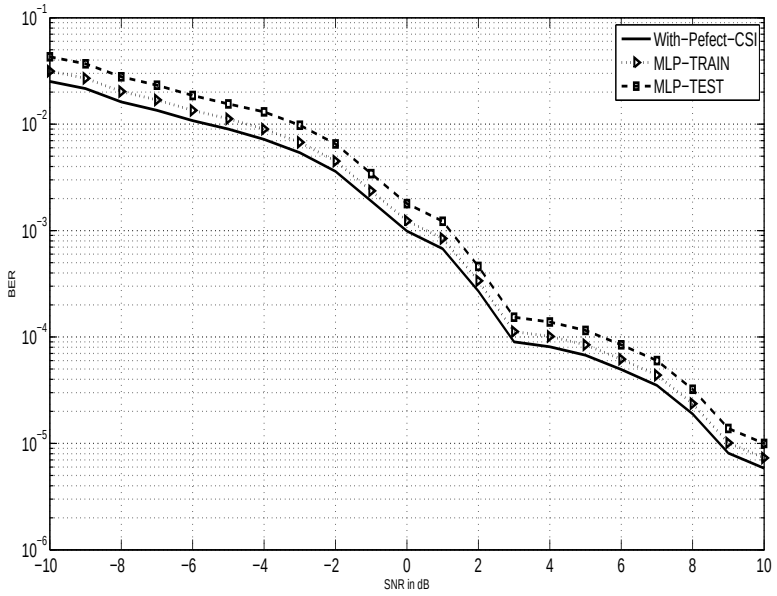


Figure 3.7: BER vs SNR plot of estimated channel (4×4 MIMO using OFDM) obtained with MLP during training and testing compared to an estimator with perfect CSI

calculation related to MIMO- channel estimation. The MLP-based values show better results which can be attributed to the fact that such a system has used TCRSI better

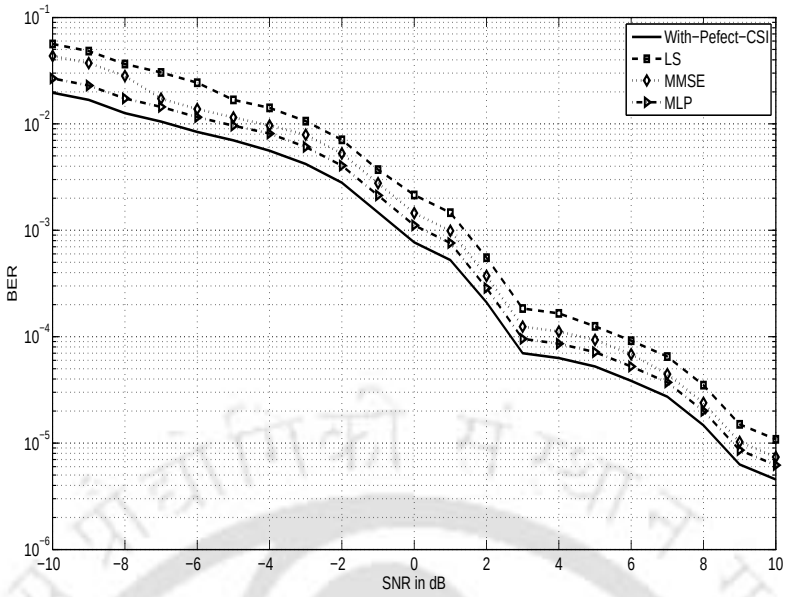


Figure 3.8: BER vs SNR plot of estimated channel (4 x 4 MIMO using OFDM) obtained with LS, MMSE, MLP methods compared to an estimator with perfect CSI

than conventional methods. In all the cases of simulation, results are shown considering the channel to be Rayleigh. Since the results derived from the Rician channel are close to that obtained from the Rayleigh case, those are not shown.

3.3.4 Limitation of MLP based symbol recovery and channel estimation

The average results derived from the considerations described in the Section 3.2 above show that the MLP in its rudimentary form recovers the symbols well and provides channel estimation. But it fails to track time variation and thereby provides sets BER values which needs improvement. It also takes longer training time for high input correlation. Additionally, if number of training sessions are increased to improve learning, the MLP based estimation tends to memorize instead of generalizing for which performance suffers. Similar is the situation when patterns at different time intervals are applied to the system and then repeated. These issues are taken into consideration for designing innovative MLP based architectures for performance improvement. Such constrained performance illustrates that basic MLP shows fluctuating performance with changing conditions while modeling time-varying properties of the MIMO channel. It is attributed to the fact that the MLP in this form fails to capture time-dependent variation in the input samples with a distinct dependence on data size, presentation rate and channel conditions. Hence, certain modifications are needed which are the main proposals of this chapter and carried

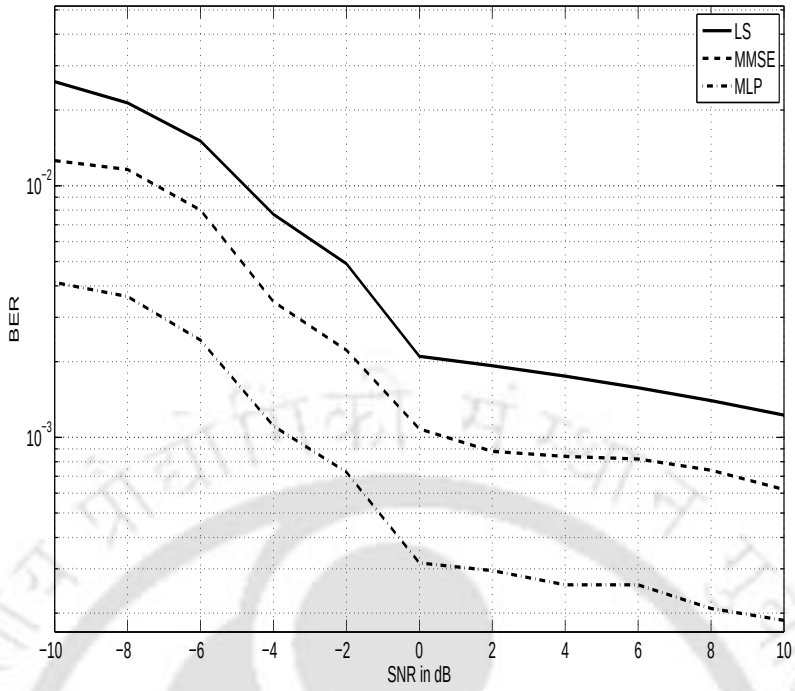


Figure 3.9: BER values generated by LS, MMSE and MLP methods for different SNRs

out as described in the Section 3.4 below.

3.4 Temporal-MLP Architecture for MIMO Channel Modeling

Though the MLP based systems provide the ability to perform channel estimation and recover signal symbols, it fails to make appropriate discrimination between samples that are presented to it at different time intervals. It is attributed to the fact that it fails to capture fastly time-varying patterns of the input data. This limitation can be removed considerably by incorporating temporal characteristics [102]-[104] into the basic MLP structure. Temporal characteristics in a basic MLP are developed by building memory blocks into an ANN [34]. There are two basic methods which can be used to introduce memory into an ANN. The first one is to introduce time delays in the ANN and to adjust parameters during learning phase. The second way is to use feedback which makes the ANN recurrent [34].

Here, we propose a technique to build short-term memory into the MLP to make the network dynamic. In doing so, we consider both FIR and IIR structures for better learning. In particular, we deploy FIR and IIR sections in the input, output and hidden

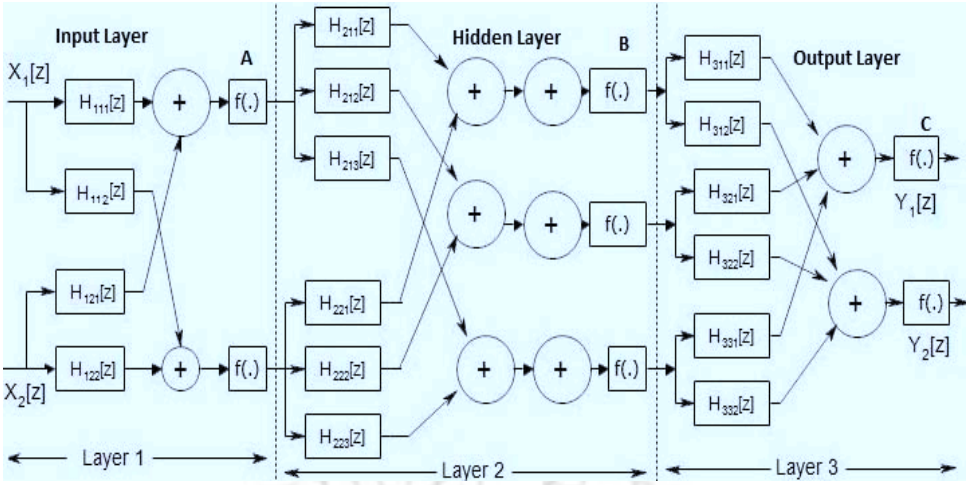


Figure 3.10: Proposed temporal MLP model

layers of the MLP which enables it to acquire dynamic characteristics. The use of these short-term memory blocks makes the output of the MLP time dependent and modifies it to a non-linear dynamical system. The static network provides the non-linear characteristics while the memory accounts for time-dependent behavior. As said, the developed structures (A) FIR-MLP and (B) IIR-MLP are further subdivided into (i) input-sided (IS), (ii) output-sided (OS) and (iii) both-sided (BS) forms. For all these forms, hidden layers with conventional neurons and FIR/IIR-linked neurons are considered as well. A generalized temporal-MLP architecture is shown in Figure 3.10 which is used to configure all the above stated systems for MIMO modeling. Considering the synapses to be formed by either FIR or IIR components, let for a two input case the applied signals be $x_1[n]$ and $x_2[n]$. The FIR synapses maybe given as

$$H_{ijl}[z] = \sum_{k=1}^{2} w_{ijk} z^{-k} \tag{3.1}$$

where w_{ijk} is the connectionist weight linked with the FIR synapses with k determining the number of delays considered in the FIR segment, j=1, 2 denoting the number of connections between perceptrons within a layer and i representing the number of layers which in this case is three. Similarly, the IIR synapses maybe given as

$$H_{ijl}[z] = \frac{\sum_{k=1}^{2} a_{ijk} z^{-k}}{1 - \sum_{k=1}^{2} b_{ijk} z^{-k}} \tag{3.2}$$

with a_{ijk} being the filter coefficients in the moving average (MA) arm while b_{ijk} represents the link values in the auto-regressive (AR) section. With these considerations, the input output relations at points A, B and C in Figure 3.10 are expressed as below:

- **At point A**:

$$Y_{Al}[z] = f[H_{111}[z] * X_1[z] + H_{112}[z] * X_2[z]] \tag{3.3}$$

$$Y_{A2}[z] = f[H_{121}[z] * X_1[z] + H_{122}[z] * X_2[z]] \tag{3.4}$$

- **At point B**:

$$Y_{Bl}[z] = f[H_{211}[z] * Y_{A1}[z] + H_{212}[z] * Y_{A1}[z]] \tag{3.5}$$

$$Y_{B2}[z] = f[H_{221}[z] * Y_{A1}[z] + H_{222}[z] * Y_{A2}[z]] \tag{3.6}$$

$$Y_{B3}[z] = f[H_{231}[z] * Y_{A1}[z] + H_{232}[z] * Y_{A2}[z]] \tag{3.7}$$

- **At point C**:

$$Y_{Cl}[z] = f[H_{311}[z] * Y_{Bl}[z] + H_{312}[z] * Y_{B2}[z] + H_{313}[z] * Y_{B3}[z]] \tag{3.8}$$

$$Y_{C2}[z] = f[H_{321}[z] * Y_{Bl}[z] + H_{322}[z] * Y_{B2}[z] + H_{323}[z] * Y_{B3}[z]]. \tag{3.9}$$

Using discrete time domain notations, let $d_k[n]$ be the desired result and $y_k[n]$ be the actual result. The instantaneous performance criteria is defined as

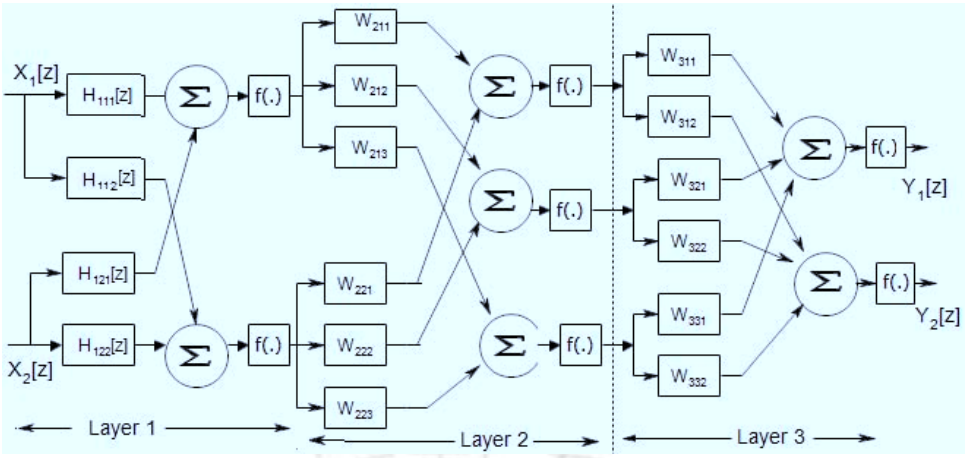
$$\xi[n] = \frac{1}{2}\sum_{k=1}^{M}(d_k[n] - y_k[n])^2. \tag{3.10}$$

The total error or cost function is given by summing the instantaneous error over all time T in the training sequence:

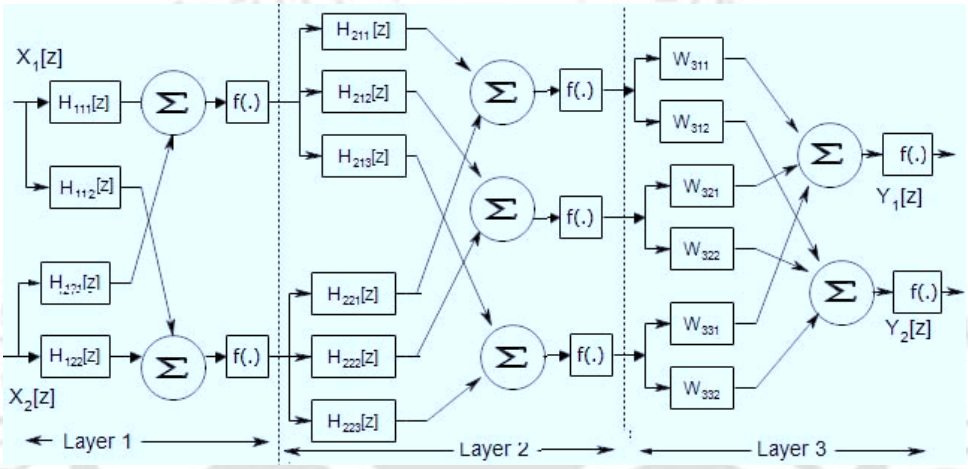
$$\xi_T = \sum_{n=0}^{T}\xi[n]. \tag{3.11}$$

3.4.1 FIR-MLP architecture

Although the introduction of the time-delays increases the computational complexity of the ANN, it is however necessary for better learning as said earlier. Therefore, we start with a simple FIR-MLP structure initially. The temporal attributes of the FIR-MLP set-up is derived by incorporating FIR synapses in the input, output and hidden layers. These connections allow the formation of the following architectures: (i) Input Sided FIR MLP with Conventional Hidden Layer (IS-FIR-MLP-CHL), (ii) Input Sided FIR MLP with Temporal Hidden Layer (IS-FIR-MLP-THL), (iii) Output Sided FIR MLP with Conventional Hidden Layer (OS-FIR-MLP-CHL), (iv) Output Sided FIR MLP with Temporal Hidden Layer (OS-FIR-MLP-THL), (v) Both Sided FIR MLP with Conventional Hidden Layer (BS-FIR-MLP-CHL) and (vi) Both Sided FIR MLP with Temporal Hidden Layer (BS-FIR-MLP-THL). The input output expression and the related considerations of all



(a) IS-FIR-MLP-CHL



(b) IS-FIR-MLP-THL

Figure 3.11: Temporal MLP architectures with FIR links at input side

these architectures are discussed below with reference to the Figure 3.10.

1. **IS-FIR-MLP-CHL:** In this form, the temporal MLP is constructed by placing FIR synapses in the input side only. The MLP uses conventional weights in the hidden and output layers. So, the input layer weights have the form

$$H_{1jl}[z] = \sum_{k=1}^{2} w_{1jk} z^{-k} \tag{3.12}$$

where j takes values of 1 and 2 for two numbers of inter-layer connections. The hidden layer and the output layers simply have conventional weights $[w_{2j}]$ and $[w_{3j}]$ (Figure 3.11 (a)). This architecture possess improved temporal characteristics compared to the conventional MLP but as the FIR synapses are confined only to the input, the set-up has limited memory and hence can use only constrained

contextual information for learning.

2. **IS-FIR-MLP-THL:** This MLP block uses conventional weights only in the output layer. So, the input and hidden layer weights have the forms given by equation (3.12) and

$$H_{2jl}[z] = \sum_{k=1}^{2} w_{2jk} z^{-k} \tag{3.13}$$

respectively where j takes values of 1, 2 and 3 for three numbers of inter-layer connections. The output layer simply have conventional weight $[w_{3j}]$ Figure 3.11 (b). This set-up has FIR synapses at the hidden as well along with the input layers for which the temporal characteristics improve considerably compared to the conventional MLP and the IS-FIR-MLP-CHL structure. The FIR synapses at the hidden layer extend its storage capability for which temporal differential processing improves. It, therefore, shows better ability to track time-variation in the applied patterns.

3. **OS-FIR-MLP-CHL:** In the output sided FIR connection with the MLP, the input and hidden layers have conventional weights given as $[w_{1j}]$ and $[w_{3j}]$. The output layer has FIR synapses given as (Figure 3.12 (a))

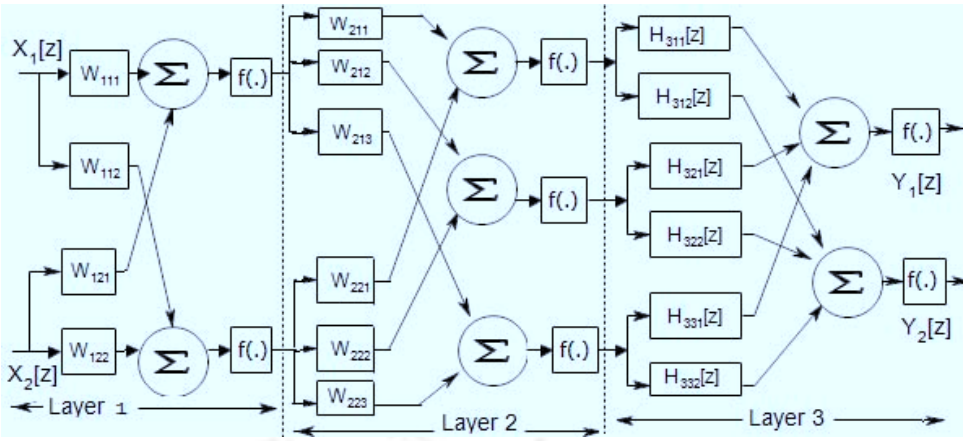
$$H_{3jl}[z] = \sum_{k=1}^{2} w_{2jk} z^{-k}. \tag{3.14}$$

FIR links in the output layer provides limited temporal ability and hence shows poor performance in tracking time-varying inputs.

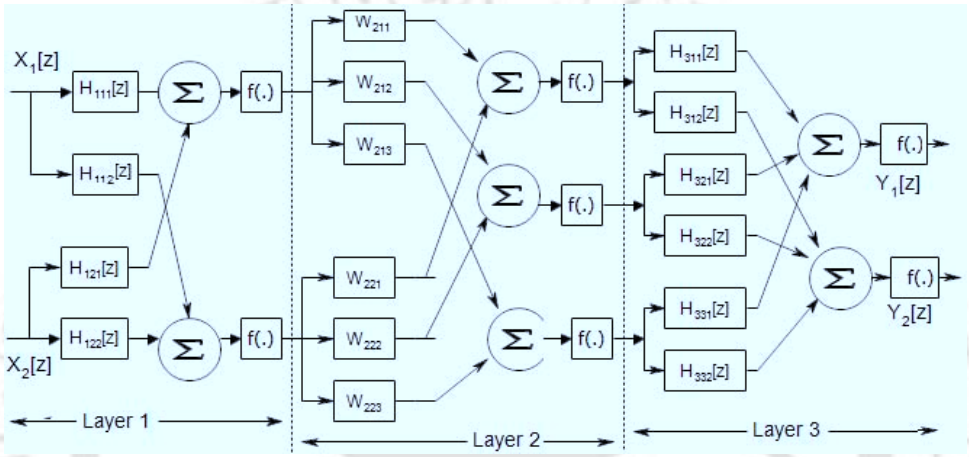
4. **OS-FIR-MLP-THL:** When the FIR link is placed in the hidden and output layers, the weight expressions as similar to that given by equations (3.13) and (3.14).

5. **BS-FIR-MLP-CHL:** In the both sided configuration, the FIR synapses are placed in the input and output layers. The weight expressions as given by equations (3.12) and (3.14). The hidden layer, in this case, is formed by conventional weights only (Figure 3.12 (b)). With FIR links at output and hidden layers, this configuration has extended temporal ability for which shows superior MSE convergence.

6. **BS-FIR-MLP-THL:** In the last configuration with FIR synapses, the temporal components are placed in input, hidden and output layers. The weight expressions for these layers are given respectively by equations (3.12) - (3.14) (Figure 3.10). Among all the configured FIR-MLP architectures, this set-up shows the best MSE convergence. This is because of the fact that the presence of FIR links at input, output and hidden layers provide it extended tapped delay line sections which can



(a) OS-FIR-MLP-CHL



(b) BS-FIR-MLP-CHL

Figure 3.12: Temporal MLP architectures with FIR links at output and both input-output sides

circulate contextual information better, and thereby sustain the learning process with better accuracy. This arrangement enables the mechanism to track the subtle variations shown by time-dependent input patterns. Of course, the complexity increases the most in this case.

The experimental details including MSE convergence are further covered in the Section 3.5.

3.4.2 IIR-MLP architecture

The primary disadvantage of the FIR-MLP architectures are related to certain learning capability with respect to time-varying inputs which can be extended further with additional modifications. Therefore, a large number of filter coefficients are required in order to retain the contextual information. Further, their ability of reduction of correlation

among samples is also constrained. Hence, feedback options are preferred [102]-[104] so as to enable a circulation of retained knowledge among involved segments of the network. This is achieved by replacing the FIR links with IIR synapses at the input, hidden and output layers for which the following IIR-MLP architectures result: (i) Input Sided IIR MLP with Conventional Hidden Layer (IS-IIR-MLP-CHL), (ii) Input Sided IIR MLP with Temporal Hidden Layer (IS-IIR-MLP-THL), (iii) Output Sided IIR MLP with Conventional Hidden Layer (OS-IIR-MLP-CHL), (iv) Output Sided IIR MLP with Temporal Hidden Layer (OS-IIR-MLP-THL), (v) Both Sided IIR MLP with Conventional Hidden Layer (BS-IIR-MLP-CHL) and (vi) Both Sided FIR MLP with Temporal Hidden Layer (BS-IIR-MLP-THL). The FIR links are replaced by IIR synapses, as given by equation (3.2). The primary objective behind all these IIR structures is to retain the same contextual information and improve performance with less number of filter coefficients as compared to FIR structures. In these cases, Figure 3.11 and Figure 3.12 would be replaced by respective IIR synapses and therefore are not repeated in this section.
The training of the FIR and IIR MLP structures are carried out using the temporal back propagation algorithm. The governing principles are based on the gradient calculations of the weight adaptation process followed by structures which in general is described in the next section as these are important in deriving certain cost function. The cost function is used to represent certain performance criteria of these proposed ANN architectures.

3.4.3 Gradient computation for the temporal architectures

The gradient computation steps are required to determine the cost function and corresponding weight adaptation associated with the training process. The training is based on temporal back-propagation where the actual response of the network is compared with the desired response at each instant of time [34]. The weight adaptation resulting out of the temporal back propagation can be written by a gradient method where weight W^l_{ikj} for j^{th} sample value associated with the k^{th} neuron in the i^{th} layer is updated with an expression:

$$W^l_{ikj}[n+1] = W^l_{ikj}[n] + \nabla W^l_{ikj}[n] \tag{3.15}$$

and synaptic gain C^{l+1}_{km} is updated as

$$C^l_{lk}[n+1] = C^l_{lk}[n] + \nabla C_i k^l[n] \tag{3.16}$$

such that

$$\nabla W^l_{ikj}[n] = -\eta \frac{\partial \xi[n]}{\partial W^l_{ikj}[n]} = -\eta \frac{\partial \xi[n]}{\partial x_k[n]} * \frac{\partial x_k[n]}{\partial W^l_{ikj}[n]} \tag{3.17}$$

and

$$\nabla C_{ik}^{l}[n] = -\eta \frac{\partial \xi[n]}{\partial C_{ik}^{l}[n]}. \tag{3.18}$$

Here η denotes the learning rate and $\xi[n]$ is the instantaneous performance criteria generated for each instant of time n. Also, $\xi[n]$ represents the error between actual and desired response at each instant of time as given by equation (3.10). Since we prefer to compute the gradient using a total cost function, the weight changes in equations (3.17) and (3.18) are here modified as [34], [102]-[104]

$$\nabla W_{ikj}^{l}[n] = -\eta \frac{\partial \xi_T}{\partial W_{ikj} l[n]} \tag{3.19}$$

and

$$\nabla C_{ik}^{l}[n] = -\eta \frac{\partial \xi_T}{\partial C_{ik} l[n]} \tag{3.20}$$

where ξ_T is the total error or cost function given by equation (3.11). The importance of such cost functions for the temporal back propagation algorithm is shown in Section 3.5.

3.5 Results and Discussion

In Section 3.2, we have shown channel estimation and symbol recovery results using the rudimentary form of MLP. Here, we show the experimental results when the proposed temporal components (FIR and IIR) are integrated to the system as given by equations (3.3)-(3.20). The proposed temporal-MLP architectures are trained and the performance is validated following modalities similar to that described in Section 3.2 and depicted by Figure 3.1. The training process, however, is somewhat different to that followed with a rudimentary MLP set-up. The training aspects are governed by the considerations described in Sections 3.4.1- 3.4.3. The summary of the training considerations followed are shown in Table 3.9. The temporal structures with filter lengths of 4-8 for FIR and 3-6 for IIR segments provide the best training results. These configurations when used with the MLP also provide better precision and ability to track subtle variations in the input samples. The BS-FIR-MLP-THL has the highest computational complexity among FIR-MLP while BS-IIR-MLP-THL shows maximum number of additions and multiplications among IIR-MLP architectures (Table 3.9). The IIR-MLP shows around 68 to 70% increase in computational complexity which contributes to improvement in performance which is outlined below. Another set of parameters required during the training is shown in Table 3.10. With the given training set (Table 3.10), the learning rates 0.8 and 0.1 provides a midway between number of sessions required and precision obtained. At 0.09, the precision is the best but learning time is atleast 21% higher, on

Table 3.9: Training configuration and computational complexity of the two temporal-MLP architectures where N denotes the filter length in general

Synapse	Model	Hidden layer size	Optimal filter length	Number of	
				addition	multiplication
FIR	IS-FIR-MLP-CHL	Tansig, 30	4-8	$2N+2$	$2N+2$
	IS-FIR-MLP-THL	Tansig, 30	4-8	$3N$	$2N+2$
	OS-FIR-MLP-CHL	Tansig, 30	4-8	$2N+2$	$2N+2$
	OS-FIR-MLP-THL	Tansig, 30	4-8	$3N$	$2N+2$
	BS-FIR-MLP-CHL	Logsig-Tansig, 30	4-8	$3N+2$	$4N$
	BS-FIR-MLP-THL	Logsig-Tansig, 30	4	$4N+2$	$6N$
IIR	IS-IIR-MLP-CHL	Tansig, 30	3-6	$3N+2$	$4N$
	IS-IIR-MLP-THL	Tansig, 30	3-6	$4N$	$5N$
	OS-IIR-MLP-CHL	Tansig, 30	3-6	$3N+2$	$4N$
	OS-IIR-MLP-THL	Tansig, 30	3-6	$4N$	$5N$
	BS-IIR-MLP-CHL	Logsig-Tansig, 30	3-6	$5N+2$	$7N$
	BS-IIR-MLP-THL	Logsig-Tansig, 30	3	$7N+2$	$10N$

Table 3.10: Training parameters for the two temporal MLP architectures

Sl. No.	Item	Value / size
1	Training set size	500
2	Testing Set Size	500
3	Learning rate, μ	0.8, 0.1
4	Momentum Coeff., α	0.2
5	Randomization	10 times
6	Epochs (max.)	500
7	Input normalized to	[0-1] and [-1 1]

an average, that that obtained with 0.8 and 0.1. Also, a momentum, α with a value of 0.2 helps to accelerate the learning. At the end of the training and validation stages a channel estimation is performed by the temporal-MLP blocks. The MSE convergence plots give a measure of the extent of learning which the architectures have acquired during training. A set of average MSE plots of LS, MMSE, MLP, FIR-MLP and IIR-MLP is depicted in Figure 3.13. The IIR-MLP structures with local recurrent loops prove to be better learners and thereby show better precision in tracking the time-varying inputs. Figure 3.14 shows an average set of four Rayleigh faded paths generated using a BS-IIR-MLP-THL with AWGN of $\pm$ 3dB from ten sets of trials carried out using 64-block OFDM symbols. The results show a precision around 92-94 % range which is atleast 3% better than the statistical and 1.2% superior to the MLP based approaches. Table 3.11 shows a set of results derived for a VOIP-based wireless transmission for a 4×4 MIMO with 64-bit data blocks. The results are compared with the work reported

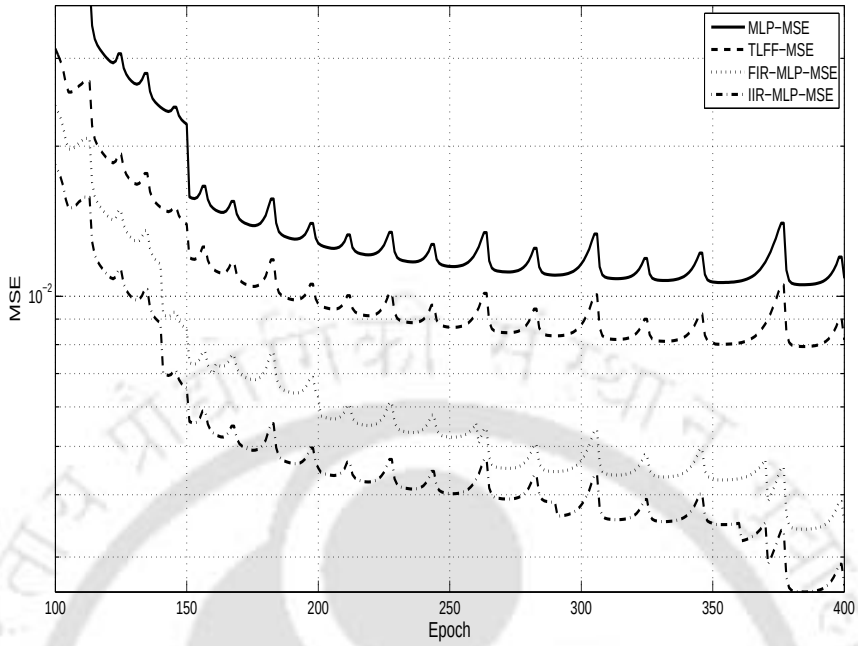


Figure 3.13: MSE convergence plots of MLPs in conventional and temporal forms

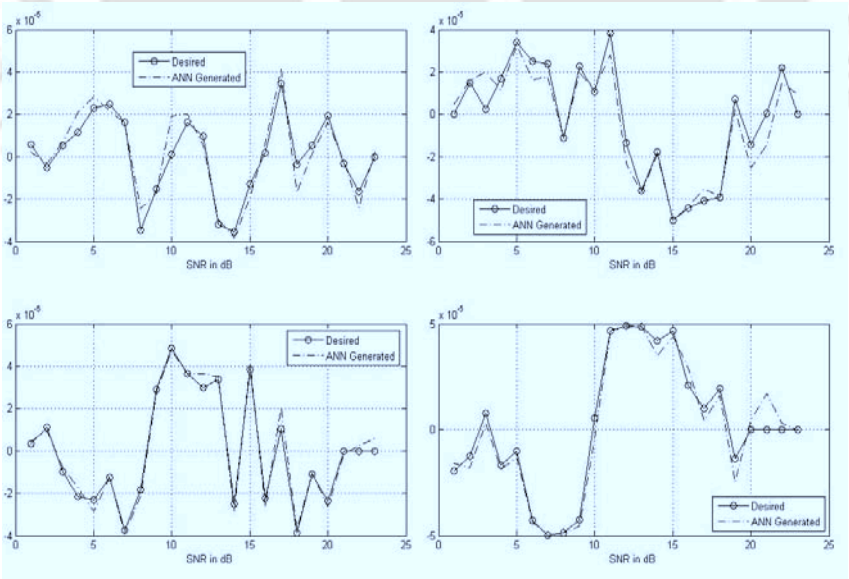


Figure 3.14: Four path Rayleigh faded channel coefficients generated using average response from a trained temporal-MLP

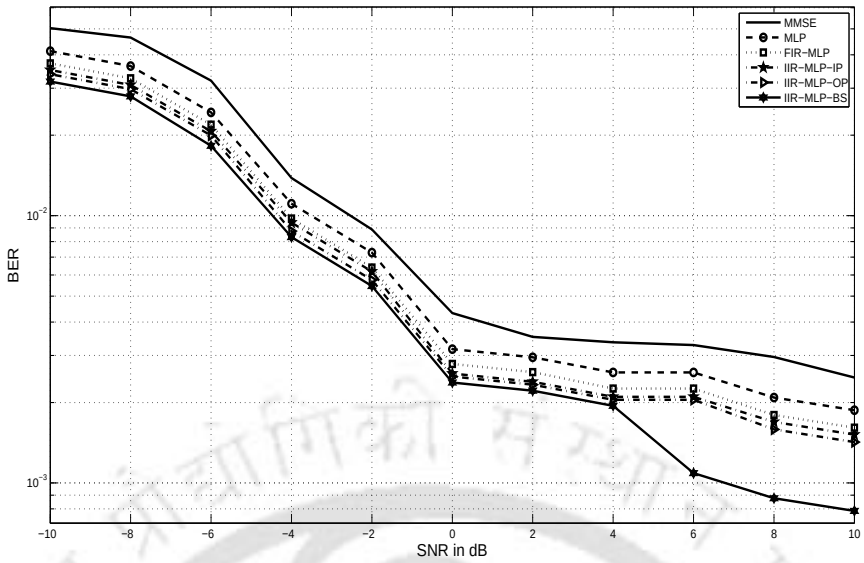


Figure 3.15: Average BER values generated by LS, MMSE, MLP and temporal-MLP methods

Table 3.11: Average values derived during channel modeling using LS, MMSE, MLP and temporal-MLP architectures

Method	Iterations	Average Precision (%)
LS	52	91
MMSE	33	92
MLP	25-29	94
3L-FF	23-28	94.5
Temporal-MLP	23-27	95

in [10] in which a three layered feedforward ANN (3L-FF in Table 3.11) is used. The performance of the conventional MLP is marginally lower than the 3L-FF structure. The temporal-MLP, however, shows better results compared to the 3L-FF consistently with lower number of training sessions and better performance. It shows the effectiveness of the temporal-MLP compared to the conventional MLP in tackling time-varying MIMO channels. Figure 3.16 shows a set of average BER values of Rayleigh channels generated by a conventional MLP and temporal-MLPs derived from 6400-bit data blocks involving VOIP communication with channel SNR varying between $\pm 3dB$. The plot also shows a theoretical limit which the temporal-MLP approaches with better precision. A similar set of results are also derived for the Rician channel. As the values are comparable, we have not shown them. A set of experiments are also performed for symbol recovery. Figure 3.17 shows a set of results of symbol recovery derived using an BS-IIR-MLP-THL. It is an average of fifty sets of data applied to the temporal-MLP for a 2×2 MIMO using OFDM

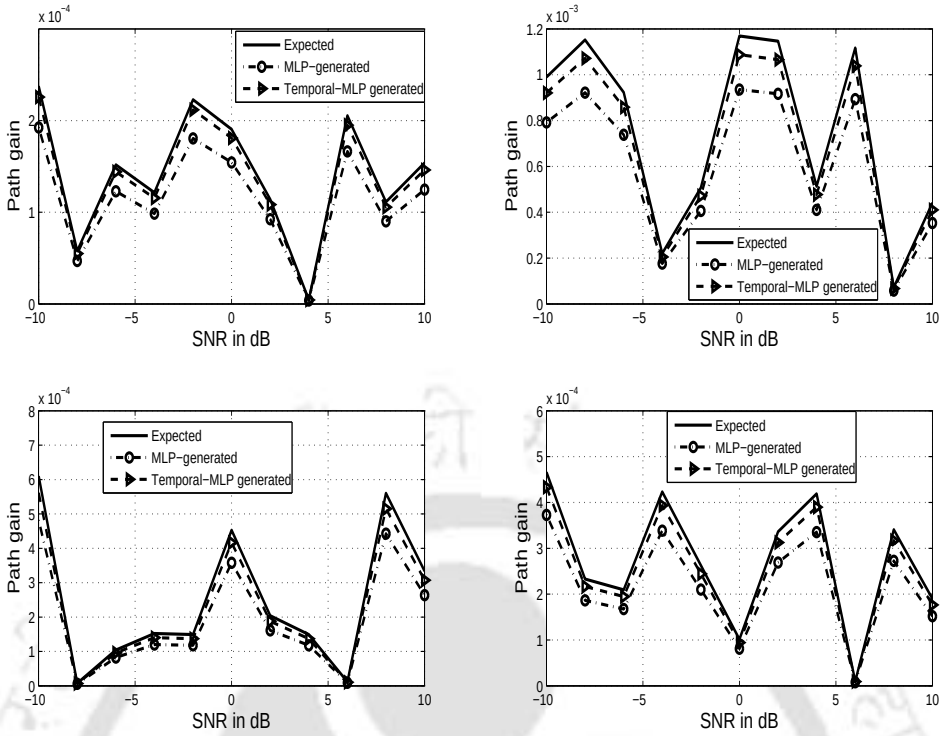


Figure 3.16: Four path Rayleigh faded channel coefficients generated using ANN and temporal-MLP approaches compared to expected results

signals having 64 sub-carriers. The 2×2 MIMO is considered to reduce the training time and simplify design considerations. The results show improvement in performance compared to the conventional MLP by about 2% which is significant for a symbol recovery process. Moreover, the temporal-MLP provides better generalization capability than the conventional MLP which is required to model the time-varying MIMO channels. A set of average total cost function measures for both the temporal-MLP architectures at different training epochs are calculated using the expression (3.20). The values generated are shown in Table 3.12. The cost function indicates the improvement in adaptation and learning shown by the proposed MLP structures per instant of time. The temporal-MLP structures show an improvement of 1.4-6.3% as compared to the conventional MLP which indicates better learning. Table 3.13 show a set of average results derived from about fifty trials. The IIR-MLP structure show superior temporal attributes which is established by the best MSE convergence that it generates which is atleast twenty times better than the corresponding value provided by a conventional MLP. Further, the precision shown in symbol recovery by the temporal MLPs is atleast 2% better than that shown by the conventional MLP. The experimental results in terms of channel estimation represented by BER plots, symbol recovery quantified by precision values and percentage processing time establish the superiority of the temporal-MLP over the statistical and rudimental MLP

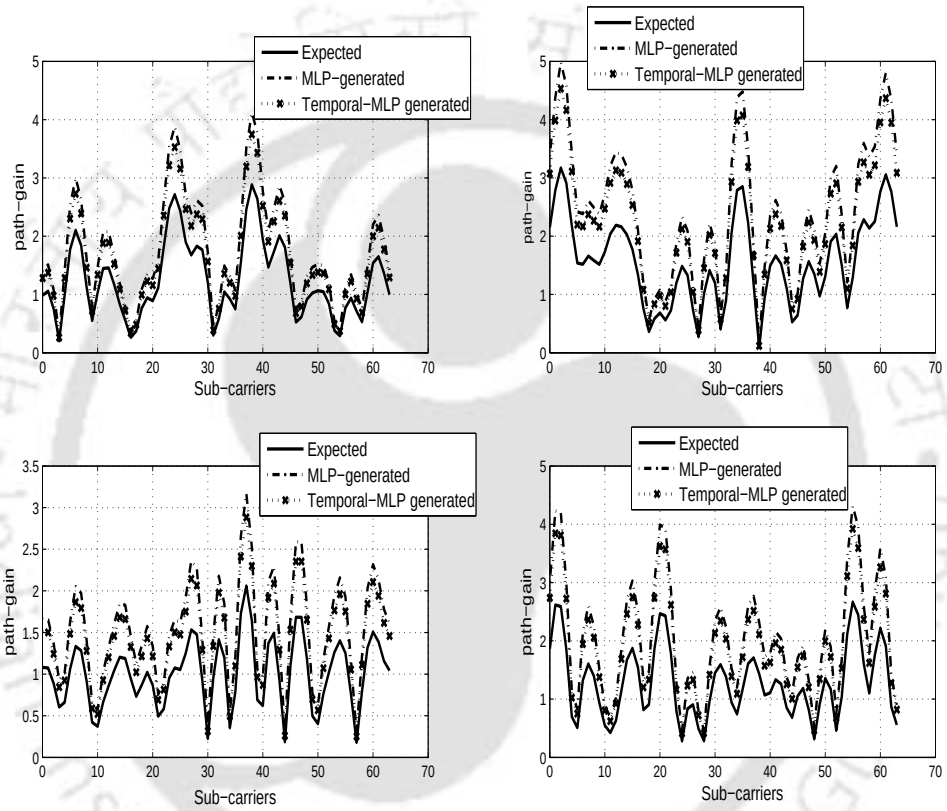


Figure 3.17: Illustrative values of symbol recovery for a 2 x 2 MIMO using OFDM signals using 64 sub-carriers in a severely faded Rayleigh channel

Table 3.12: Average % variation of normalized total cost-function with training epochs of the conventional-MLP and temporal-MLP architectures upto 500 epochs

Sl. No.	Epochs	MLP	Temporal MLP
1	100	86.1	87.3
2	200	86.9	88.9
3	300	87.8	90.8
4	400	88.3	92.7
5	500	89.9	95.5

Table 3.13: MSE convergence upto 500 epochs and precision generated by the two temporal-MLP architectures

Synapse	Case	Model	MSE $\times 10^{-4}$	% Precision
FIR	1	IS-FIR-MLP-CHL	0.06	94.1
	2	IS-FIR-MLP-THL	0.049	94.2
	3	OS-FIR-MLP-CHL	0.051	94.1
	4	OS-FIR-MLP-THL	0.047	94.3
	5	BS-FIR-MLP-CHL	0.035	94.3
	6	BS-FIR-MLP-THL	0.029	94.4
IIR	7	IS-IIR-MLP-CHL	0.045	94.3
	8	IS-IIR-MLP-THL	0.042	94.4
	9	OS-IIR-MLP-CHL	0.033	94.7
	10	OS-IIR-MLP-THL	0.029	95.1
	11	BS-IIR-MLP-CHL	0.027	95.1
	12	BS-IIR-MLP-THL	0.021	95.3

based methods. The results derived demonstrate consistent improvement with temporal-MLP based modeling of time-varying MIMO channels applied for VOIP transmission under severe fading and presence of CCI. Moreover, from the Table 3.13 it is seen that the IIR-MLP structures show improved performance in lesser iterations (Table 3.11) despite a marginal rise in computational complexity (Table 3.9) and hence are better suited for the purpose than the FIR-MLP structures.

3.6 Conclusion

In this chapter, we initially propose two different MLP structures for training, validation and testing phase for performing both estimation of a MIMO channel and recovery of signal symbols. However, these structures are found to be inefficient to track time-variation of the MIMO channels. Therefore, a set of modifications are carried out to replace the synaptic links of the MLP. In particular, two temporal architectures of the

MLP using FIR and IIR links have been proposed. The temporal-MLP architectures provide better learning ability to make discrimination between time-varying samples. However, design complexities make such configurations a bit difficult to implement for practical purposes. The logical replacement is to use a feedback path which makes the ANN recurrent, as mentioned in Section 3.4. A detailed treatment of this special class of ANN architectures to model the time-varying MIMO channel is proposed in the next chapter.

Chapter 4

Modeling MIMO Channels using a Class of Complex Recurrent Neural Network Architectures

4.1 Introduction

In Chapter 3, our main focus was on the development of time-varying MIMO channel modeling using two temporal-MLP architectures. The results show that temporal-MLP architectures are capable of making discrimination between time-varying samples, but design complexities impose certain limitations primarily with respect to implementation. The natural alternative choice is the formulation of RNN based approaches due to the fact that these soft-computational tools have the in-built capability to deal with time-varying signals. It stems from the fact that RNN has at least one built-in global/local feedback loop despite generally being a feedforward structure [34]. It also has a faster learning process than the temporal MLP with lower implementation complexity. Most of the reported works on RNN based applications in wireless communication including channel modeling (Section 1.3) have focused mainly on the training-learning aspect and the capacity to deal with channel estimation with practically little/no consideration on time-varying characteristics of the wireless propagation medium. Till now, no recorded efforts have been observed regarding expansion of the abilities of ANN and RNN architectures beyond the training-testing realm which considers certain architectural challenges. These challenges include: (i) ensuring stability to the system when a feedback path is appended along with the usual feedforward structure of ANN, (ii) retaining only the contextual portion of the information with the above structure and (iii) properly capturing the fast time-varying nature of the channels.

In this chapter, we propose a class of novel RNN architectures while addressing all the

above mentioned threefold challenges, with the help of certain basic training algorithms, a split in-phase and quadrature components of input signal for speedup considerations and a Self Organizing Map (SOM) based optimization. In particular, several RNN architectures are explored to formulate an appropriate mechanism for estimation of MIMO channels with time dependent characteristics. In the most rudimentary form, the RNN can be modeled to use split in-phase and quadrature components for training with delayed inputs and feedback loops. These are called Complex Time Delay Fully RNN (CTDFRNN) blocks. The output from these blocks can be combined using time averaging. In this chapter, we are dealing with the application of multiple CTDFRNN blocks while modeling MIMO channels, which, to the best of our knowledge, has not been reported in the literature till now. Although such an application has the advantage of certain diversity gain to support high data rate transmissions, it also poses the architectural challenge of cooperative learning of the blocks. We take care of this challenge with a framework supporting expendability of block level CTDFRNN units with responses trimmed by SOM optimization with reduced time delays and BER values. Multiple CTDFRNN units trained in a cooperative environment improves learning and the ability to tackle inputs with time-variation. An attempt is made here to show that the RNN in CTDFRNN form with SOM optimization can be configured to provide certain new possibilities. CTDFRNN clusters optimized with SOM with diversity gains is found to be useful for low signal to noise ratio (SNR) environments. Further, the Modular Network SOM (MNSOM), which is considered to resemble closely the biological computation with an inherent reinforced modular learning, is also proposed and formulated using CTDFRNN blocks for application in MIMO channel estimation. The results, when applied to a wireless VOIP platform, show improvement yet provide the scope for further exploration for devising innovative methods of MIMO channel estimation. It is also found that CTDFRNN-MNSOM emerges out to be a viable alternative to the conventional stochastic estimation methods with an average 60% saving of processing time.

4.2 Configuring the RNN for MIMO Channel Modeling

The constraints associated with suitably modified temporal-MLP structures as explained in Chapter 3 necessitate exploring the RNN as an option to model time-varying MIMO channels. The RNN maybe considered to be a MLP having one or more local or global feedback in a variety of forms [34]. Figure 4.1 shows an RNN-block known as Time Delay Fully Recurrent Neural Network (TDFRNN). The present value of the model input is denoted by $s(n)$, and the corresponding value of the model output is denoted by $y(n+1)$;

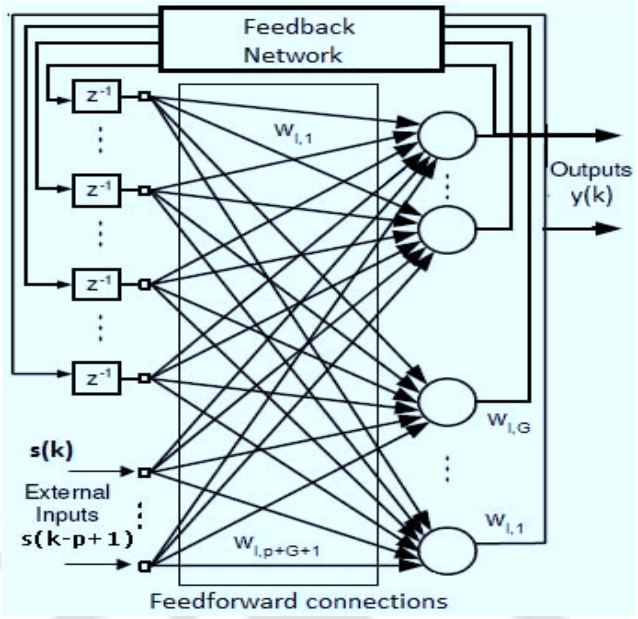


Figure 4.1: Time Delay Fully Recurrent Neural Network

that is, the output is ahead of the input by one time unit [34]. The dynamic behavior of the the RNN model is described by

$$y(n+1) = F(y(n), \cdots, y(n-p+1), s(n), \cdots, s(n-p+1)) \tag{4.1}$$

where $F(.)$ is a nonlinear function of its arguments. The use of the feedback loop in an RNN enhances its ability to learn over the given period of time by connecting contextual knowledge with inputs and responses which are circulated among adaptively updating weighted links and stored between layers. This leads to an accumulation of knowledge which, due to the multiple delays, generates a set-up for learning, comparing and retaining relevant information only. Moreover, a subtractive process rejects common data and holds back only relevant information, hence correlation among circulating samples inside the RNN layers are very less.

In the present work we have adopted the split-activation Complex TDFRNN (CTDFRNN) architecture with individual real and imaginary inputs fed to segregated blocks. The TDFRNN converted to CTDFRNN in several versions proposed here contributes towards better learning, faster convergence and lower error margins. The lowering of error values is explained in Section 4.3.1. The basic algorithms for RNN training are [34] Back Propagation Through Time (BPTT), Real Time Recurrent Learning (RTRL), Decoupled Extended Kalman Filtering (DEKF) [84] and Complex Valued RTRL (CVRTRL). All these algorithms are used in the learning phase of the TDFRNN blocks and the one that gives the best training time performance is used with the CTDFRNN blocks that constitute the MIMO channel estimation system. The details of the RNN and related training

methods have been included in Section 2.3.2.

A TDFRNN block (Figure 4.1) has a few distinct layers consisting of the external input-feedback, an output and a hidden layer each having recurrent artificial neurons as processing elements. Taking such a network as reference, let $y_l(k)$ denote the complex valued output of each neuron, $l = 1, \cdots, N$ at time index k and $s(k)$ be the $(1 \times p)$ external complex-valued input vector. The overall input to the network $I(k)$ represents the concatenation of vectors $y(k)$, $s(k)$ and the bias input $(1 + j)$. A complex valued weight matrix of the network is denoted by W, where for the i^{th} neuron, its weights form a $(p + G + 1) \times l$ dimensional weight vector w_l=$[w_{l,1,\cdots}, w_{l,p+G+1}]^T$ where G is the number of feedback connections. The feedback connections represent the delayed output signals of the TDFRNN. The output of each neuron can be expressed as

$$y_l(k) = \phi(\Psi_l(k)), l = 1, \cdots, N. \tag{4.2}$$

where

$$\Psi_l(k) = \sum_{j=1}^{p+N+1} w_{l,j}(k) I_j(k) \tag{4.3}$$

is the net input to l^{th} node at time index k and ϕ is a complex nonlinear activation function [85]. The RNNs have the capacity to deal with time varying inputs due to the presence of atleast one feed-backward loop despite it being essentially a feedforward structure [34]. The feedback in singular or multiple delayed forms from the output to the hidden layer and then to the input layer store information for one time step at a time which cumulates over the given time cycles. After the cumulation process in the hidden layer is over, the information propagates to the input layer such that at each step dynamic updating of the synaptic weights continues as per the norms of the adopted method of learning or training. This backward transmission of information enables the input, hidden and output layer neurons to learn over an extended period of time by obtaining records of prior activation or responses and share the updates with the synaptic links and the recurrent units constituting the processing system. Moreover, the RNN architecture, by using the state vector relates the present and the past context and transitions in-between the layers to describe future response of the system thereby dynamically learning temporal behaviour of input samples [34].

Signals encountered in wireless communication demonstrate variations in terms of their complex valued nature. Application of ANNs for channel estimation therefore must take into account how this aspect is tackled. Learning and nature of activation functions of ANNs must have the ability to deal with signals that have strong coupling between magnitude and phase components of signals. One way to ensure that ANNs learn magnitude and phase components separately is to use split-complex activation functions. The

practice is to provide real and imaginary components in splitted form to RNN blocks and allow them to carry on the training as mutually exclusive events [107]. However, for cases where in-phase and quadrature components are tightly coupled the split-complex activation approach yields poor performance. Communication requirements necessitate that the signals be tightly coupled. Therefore, channel estimation set-up with split RNNs trained to handle real and complex components separately shall demonstrate lower performance [107]. Two contrasting pictures, hence, emerge: on one hand, the real and complex components in split form aid the RNN learning and on the other, such RNN set-up cannot provide satisfactory performance for signals where real and imaginary components are tightly coupled. This trade off between the learning and activation functions of RNNs optimized for actual applications requiring real and imaginary components separately and the necessity of better performance for signals with tightly coupled in-phase and quadrature components needs to be sorted out. An attempt is made here to formulate a few RNN structures designed to reduce this trade off. Split-complex activation is adopted to allow the RNNs to learn the real and imaginary parts separately and the output of these exclusive blocks are combined to obtain a time average (Figure 4.2) for the inputs applied between a defined time window. By allowing the RNN blocks to train with real and imaginary components as exclusive events, the error surfaces are aligned appropriately to form classification boundaries for segregation of inputs from composite segments. This at times though is undesirable but reduces computational complexity. The RNN in this configuration can be termed as CTDRNN with separate sub-sections dealing with in-phase and quadrature components. The output from the real and imaginary blocks are combined and optimized using the time-averaging. The results obtained are better than that obtained with temporal-MLP (Section 3.4). However, the results can be further improved if time averaging is replaced by SOM based optimization. The exploration of the RNN in CTDRNN form configured in a cluster provides further opportunity to improve results and provides certain diversity gains. The RNN in CTDRNN form is also arranged to constitute a MNSOM which provides another direction to improve the performance of the system.

4.3 MIMO Channel Modeling with RNN and SOM

The signal content observed in real-world situations with associated stochastic attributes are handled satisfactorily by CTDRNN models. The abilities demonstrated by these models however is expanded by applying multiple CTDFRNN blocks in clusters for estimation and modeling of MIMO channels. Responses of multiple CTDRNN units in a cluster are optimized by SOM which improves learning and the ability to tackle inputs with time-variation. The betterment in learning is due to the fact that individual

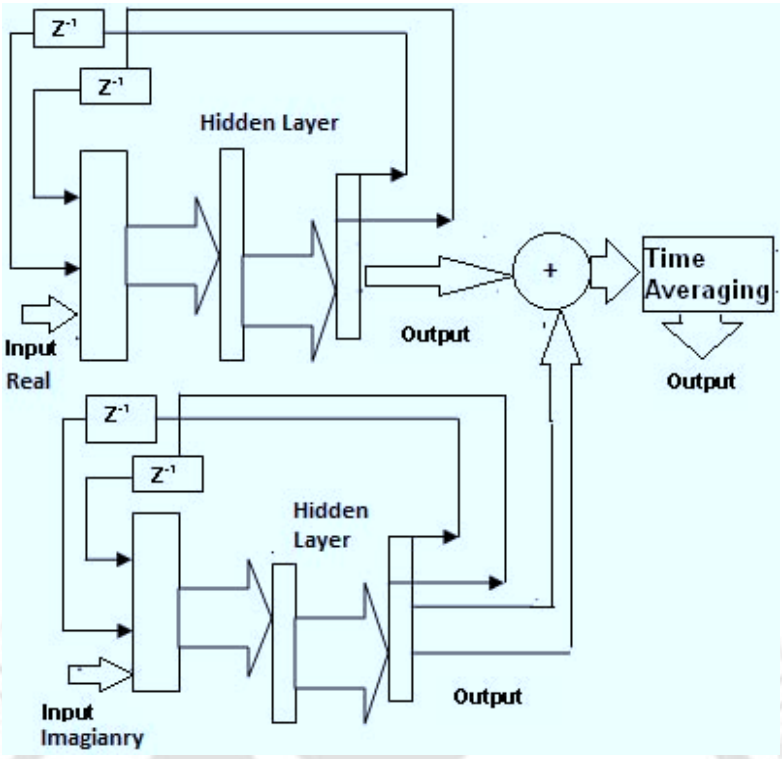


Figure 4.2: CTDFRNN Estimator with time averaging

CTDFRNN blocks receive simultaneous sample inputs and execute a race to attain the optimally trained state. The learning thus acquired by each of the blocks are trimmed and optimized further by the SOM to provide the most probable response. The CTDFRNN blocks follow a time-varying supervised learning while the SOMs manage their processing by resorting to unsupervised training. The combination together, due to their difference of learning methods, strives to achieve an appropriately trained state by following a supervised-method driven unsupervised approach of generating the required result. Homogeneous blocks of ANNs in form of modular architectures demonstrate certain bio-inspired characteristics. MNSOM, with close resemble to biological computation, provides certain attributes making it suitable for MIMO channel estimation. This is due to the ability of the MNSOM formed by CTDFRNN blocks to generate a competitive approximation of the response constituted by complex RTRL algorithm. The following sections provide a detailed treatment of the functional forms of the above mentioned RNN-based approaches to MIMO channel modeling covering essentially both estimation of channel coefficients and recovery of transmitted symbols.

4.3.1 CTDFRNN with SOM optimization

In this proposal, the CTDFRNNs are allowed to learn the real and complex parts separately and adapt as per requirement. However, to preserve tightly coupled complex

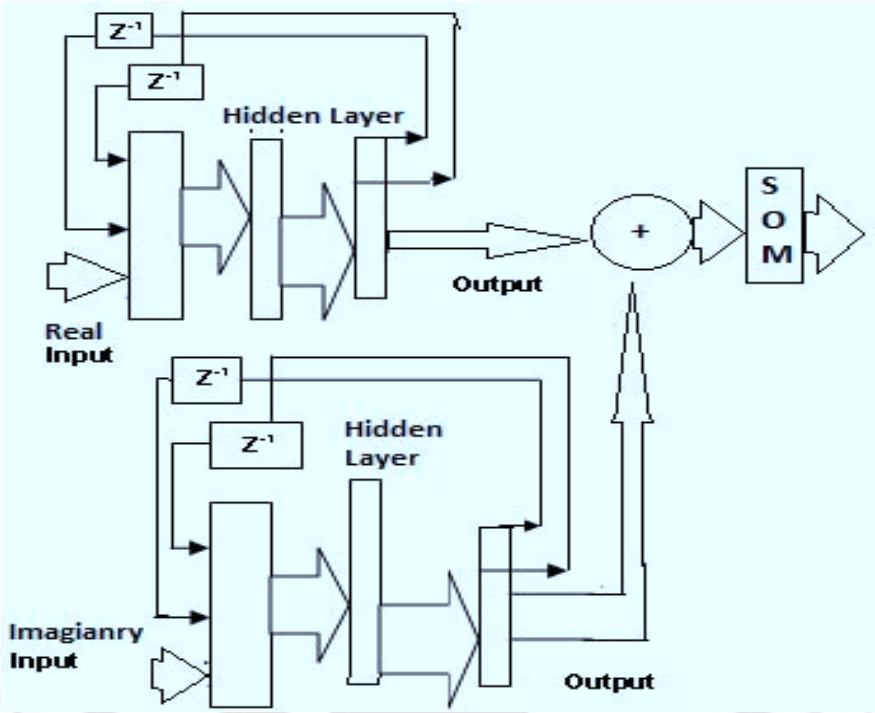


Figure 4.3: CTDFRNN Estimator with SOM Optimization

characteristics, the output of these exclusive blocks are combined and optimized with SOMs (Figure 4.3). Let $u_{Ri}[n]$ be a sequence of real components of the input which is a MIMO-OFDM signal. Also let $x[n]$ be the response of the output layer at time n to be fed-back as state vectors to the input layer. The expression of the input to the hidden layer is given as

$$y_{R1}[n] = \sum_i f_i(w_{11}[n]x[n-1] + w_{12}[n]x[n-2] + w_{13}[n]u_{Ri}[n]) \tag{4.4}$$

where $f_i(.)$ is the i^{th} activation function and $w[.]$ are synaptic links. The expression for the output of the hidden layer is given as

$$y_{R2}[n] = \sum_j f_j(\sum_i y_{1i}[n]w_{2i}[n]). \tag{4.5}$$

where $y_{1i}[n]w_{2i}[n]$ represents the i^{th} weighted inputs (expression (4.4)) to the hidden layer. The expression for the response of the output layer is given as

$$y_{R3}[n] = \sum_k f_k(y_{R2k}[n]w_{3k}[n]). \tag{4.6}$$

A similar expression for the quadrature component can be obtained as

$$y_{Q3}[n] = \sum_k f_k(y_{Q2k}[n]w_{3k}[n]). \tag{4.7}$$

Let d_r and d_q be the desired outputs for the RNN modules respectively. The error signals can then be obtained as

$$e_R[n] = \frac{1}{2}\sum_{k=1}^{N}(y_{R3k}[n] - d_{rk}[n]]^2 \tag{4.8}$$

$$e_Q[n] = \frac{1}{2}\sum_{k=1}^{N}(y_{Q3k}[n] - d_{qk}[n]]^2. \tag{4.9}$$

and the total complex error can be expressed as

$$e_{RNN}[n] = e_R[n] + je_Q[n]. \tag{4.10}$$

In case of a feedforward structure, the total error is defined as [34]

$$e_{FF} = \sum_k (d_k - y_k)^2. \tag{4.11}$$

Comparing this with (4.8), (4.9) and (4.10), it can be shown that

$$|e_{RNN}[n]| = 0.707|e_{FF}|. \tag{4.12}$$

Therefore, it can be inferred that the proposed RNN structures yield better error performance than any feedforward ANN structure. The SOM block is used to carry out an optimization of the outputs generated by the RNN modules in a given time of say n seconds. Let the input to the SOM block be a combined signal $y_0[n]$ formed by $y_{R0}[n]$ and $y_{Q0}[n]$. The output of the SOM block is given by

$$y_j = y_{0j}^T W_j \qquad j = 1, \cdots, m \tag{4.13}$$

where W_j is the random weight vector of the competitive layer of the SOM. The 'winners take all' competition starts in this layer such that the winning neuron index, J, satisfies

$$y_J = max_j\{y_{0j}^T, W_j\}. \tag{4.14}$$

A detailed treatment of competitive learning of the SOM is given in Section 2.3.3.
A SOM is used to receive the outputs of the TDFRNN blocks to perform the task of optimization. This is done because of the fact that the SOM is fast and it guarantees

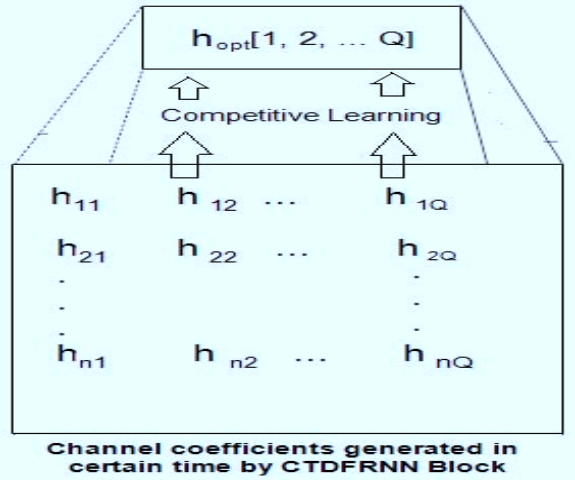


Figure 4.4: Channel coefficient vector generated using SOM Optimization

convergence in an optimization problem. It works with a spatial neighbourhood which shrinks after training leading to enhanced performance as an optimizer compared to time-averaging and other conventional approaches. Using self-organization governed by the principles of competitive learning, a SOM plays the role of a dimension reducer. A SOM has one input and one output layer. Further, a SOM works with vectors and at the end of training, generates the most appropriate set of values from the ones presented to it. The selections after each iteration are updated which at the end of the training attains the optimized state. This working principle governs the use of the SOM to select the most appropriate set of channel coefficients $(1-D)$ from the composite cluster $(2-D)$ generated by the CTDFRNN block after n training times. The process of optimization of the channel coefficients from a matrix block (Q numbers of coefficients generated in n different training slots) to the most appropriate vector form (in this case) is depicted in Figure 4.4.

4.3.2 CTDFRNN-Cluster (CTDFRNN-C) optimized by SOM

Estimation of MIMO channels can also be carried out by a cluster of N CTDFRNN blocks each of which are formed by two TDFRNN units. This mainly helps in improved error performance and a diversity gain. From a given input, the split in-phase and quadrature components are fed separately to the TDFRNN units and the outputs derived from a combination of responses from the two units. The individual outputs of all the N blocks are fed to a SOM for optimization which after a given time interval of a few seconds generates the most likely estimate of the channel coefficient. The layout of the CTDFRNN-Cluster (CTDFRNN-C) is depicted in Figure 4.5. Here a TDFRNN unit with inputs having time-delays acts as a processing unit and contributes towards the response produced by

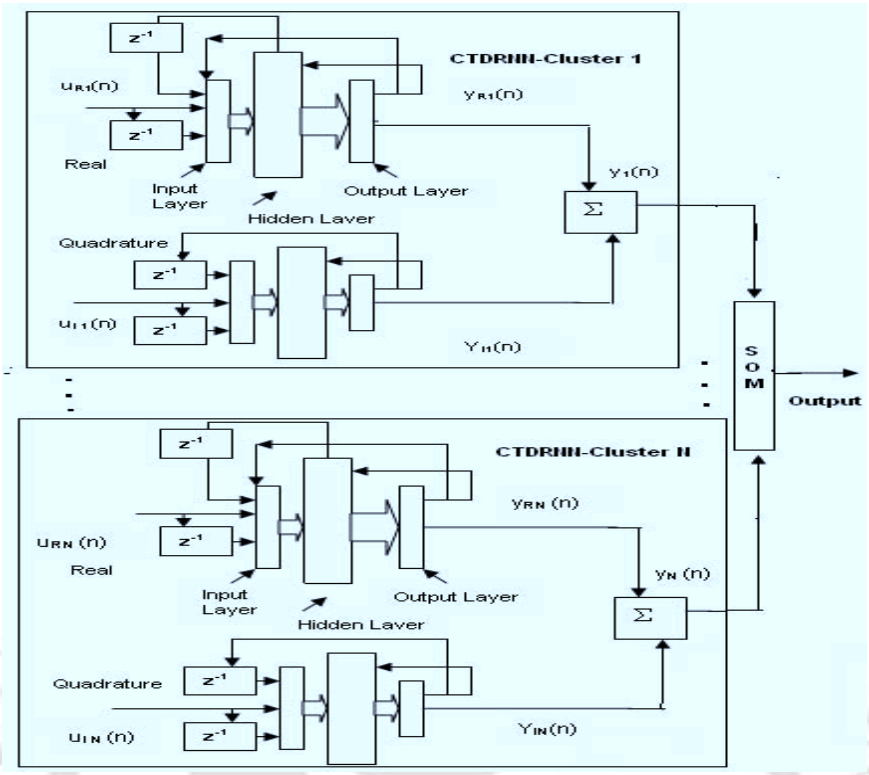


Figure 4.5: CTDFRNN-Cluster optimized by SOM

the cluster formed by N such blocks meant for tightly coupled signal components. The arrangement mimics a diversity transmission for which the response generated includes some improvement. Though the inputs are applied in split component format to enable the RNN blocks to learn the patterns and capture the temporal characteristics, the combined output provides the response for tightly coupled signal components optimize by a SOM block. For a real input $u_{R1}(n)$, the contribution to the output is given as

$$y_{R1}[n] = \sum_{k} f_k[\sum_{i,j} u_{R1}[n-1-i]w_{1i}[n] + x[n-j]w_{fbj}[n]] \tag{4.15}$$

where w_{fbj} is the j^{th} value of the weight vector in the feedback path. The output expression is finally found to be

$$y_{R31}[n] = \sum_{j} f_j[\sum_{i} f\{y_{R1ij}[n]w_{2i}[n]\}w_{3j}[n]]. \tag{4.16}$$

A similar expression maybe obtained for the quadrature component:

$$y_{Q31}[n] = \sum_{j} f_j[\sum_{i} f\{y_{Q1ij}[n]w_{2i}[n]\}w_{3j}[n]]. \tag{4.17}$$

The combined output for the first cluster is given by

$$y_1[n] = y_{R31}[n] + jy_{Q31}[n] \tag{4.18}$$

where $j = \sqrt{-1}$. For N such clusters, the output is obtained as

$$y_i[n] = \sum_{i=1}^{N} [y_{R3i}[n] + jy_{Q3i}[n]]. \tag{4.19}$$

This combined output is applied to the SOM optimizer such that the best set of channel coefficients are generated following the process

$$y_{opt}[n] = \sum_{i=1}^{N} [y_i[n] W_{ij}] \tag{4.20}$$

where W_{ij} is the weight matrix associated with the target neighbourhood of the SOM optimizer with weight update process controlled by the considerations explained in Section 2.3.3. The composite error as given for a single pair of CTDFRNN blocks as explained in Section 4.3.1 and given by (4.8) is further scaled down by an order of $1/N$ than the errors given by a CTDFRNN block. Earlier it was shown that the CTDFRNN has better error performance than feedforward ANNs for a similar application. The combined error response of an N-block CTDFRNN cluster is given as

$$e_T[n] = \frac{1}{N} \sum_{i=1}^{N} [e_i[n]] \tag{4.21}$$

which provides considerable improvement in error performance compared to a feedforward ANN. For smaller values of N the performance enhancement achieved is significant ($\geq$ 25% as found experimentally). Apart from this error performance, it also yields certain diversity gain which is discussed below.

4.3.2.1 Diversity considerations

At the receiver end, the trained cluster formed by N blocks of CTDFRNNs constitute a receive arrangement as depicted by Figure 4.6. Each of the received individual branch signals are given as

$$x_n(t) = A(t) g_n + \varepsilon_n \tag{4.22}$$

where *A(t)* is the complex envelop containing the OFDM signals, g_n is the channel gain and ε_n is the additive Gaussian noise. The set - up generates a diversity reception

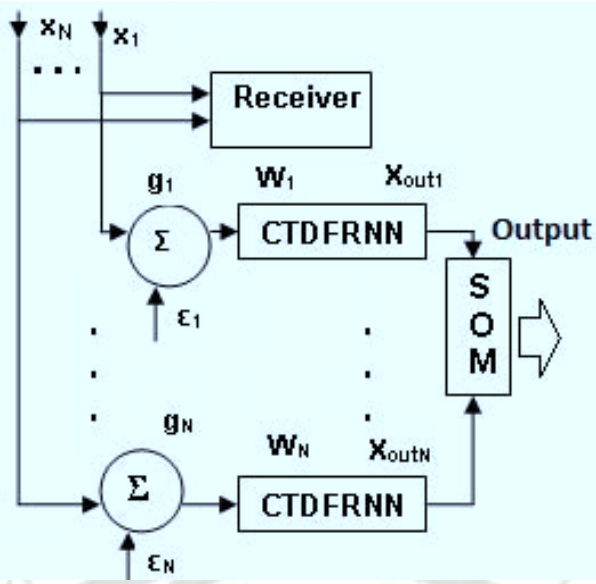


Figure 4.6: CTDFRNN - Cluster based reception arrangement

expressed as

$$X_{out}(t) = \sum_n W_n x_n(t) = \sum_n [A(t)g_n + \varepsilon_n]W_n \tag{4.23}$$

where W_n is the branch grain associated with each antenna which can be related to the CTDFRNN units. The output SNR is expressed as

$$SNR_{out} = \frac{P_s}{P_\varepsilon} = \frac{1}{2} \mid A(t) \mid^2 \frac{|\sum_n g_n|^2}{\sum_n \sigma_n^2}. \tag{4.24}$$

This results in

$$\gamma_{max} = \sum_n \gamma_n \tag{4.25}$$

where branch SNR is expressed as

$$\gamma_n = \frac{|A|^2}{2\sigma_n^2}. \tag{4.26}$$

With the complex envelop *A(t)*, the output signal is given by

$$r(t) = A(t)\sqrt{\sum_{i=1}^{N} f_i^2} + \sum_{i=1}^{N} \frac{f_i n_i(t)}{\sqrt{f_i^2}} \tag{4.27}$$

where f_i is the fading envelop and $n_i(t)$ is the additive noise of the i^{th} path. This can be optimized by the SOM with an expression

$$y_{opt}[n] = \sum_{i=1}^{N} [y_i[n] \times W_{ij}] \tag{4.28}$$

which gives

$$\sum_{i=1}^{N}[y_i[n] \times W_{ij}] = A(t)\sqrt{\sum_{i=1}^{N} f_i^2} + \sum_{i=1}^{N} \frac{f_i n_i(t)}{\sqrt{f_i^2}}. \tag{4.29}$$

At the trained state, with optimum weights $[W_{ij}]$

$$\hat{A(t)} = \sum_{i=1}^{N} \frac{[y_i[n] \times W_{ij}]}{f_i} + \sum_{i=1}^{N} \frac{n_i(t)}{f_i}. \tag{4.30}$$

The channel gains obtained using (4.23) maybe given as

$$\hat{g_n} = \sum_{n} \frac{X_{out}(t)}{\hat{A(t)} \times W_n} - \frac{\varepsilon_n}{\hat{A(t)}}. \tag{4.31}$$

This expression can be modified to obtain

$$\hat{g_n} = \sum_{n} [\frac{X_{out}(t)}{W_n} - \varepsilon_n] \sum_{i} \frac{f_i}{y_i[n] \times W_{ij} + n_i(t)}. \tag{4.32}$$

The channel gain helps the CTDFRNN-C configuration to provide performance improvement which is outlined in Section 4.4.

4.3.3 CTDFRNN - Modular Network SOM (CTDFRNN-MNSOM)

Modular Network SOM (MNSOM) is an assembly of information processing blocks that can dynamically process data and can select the best set of coefficients. Every reference vector unit of a conventional SOM is replaced by a trainable functional module such as an RNN while preserving the basic layout of the SOM. The trainable RNN units contribute towards the composite response generated by the SOM. Each of the RNN units is provided with input and reference signals for individual training. At the end of the training, each RNN unit works in unison to generate the SOM behaviour. The SOM behaviour is approximated by a Euclidean distance which forms the basis of carrying out a competitive learning to closely resemble the required response. In this case, the architecture of the MNSOM is so configured that at a given instant of time, the best set of channel coefficients are obtained through self-organization carried out among individual RNN units. The process is sustained by the trained RNNs such that a fast adaptation generates the best set of channel coefficients. The learning and the adaptation process considers two aspect. First, the training of the individual RNN units with corresponding learning algorithm which in this case is DEKF. The trained RNN units provide the functional MNSOM layer which acquires a dynamic pattern optimization capability. Second, a concurrent generation a feature map is facilitated which enables the MNSOM to distinguish required

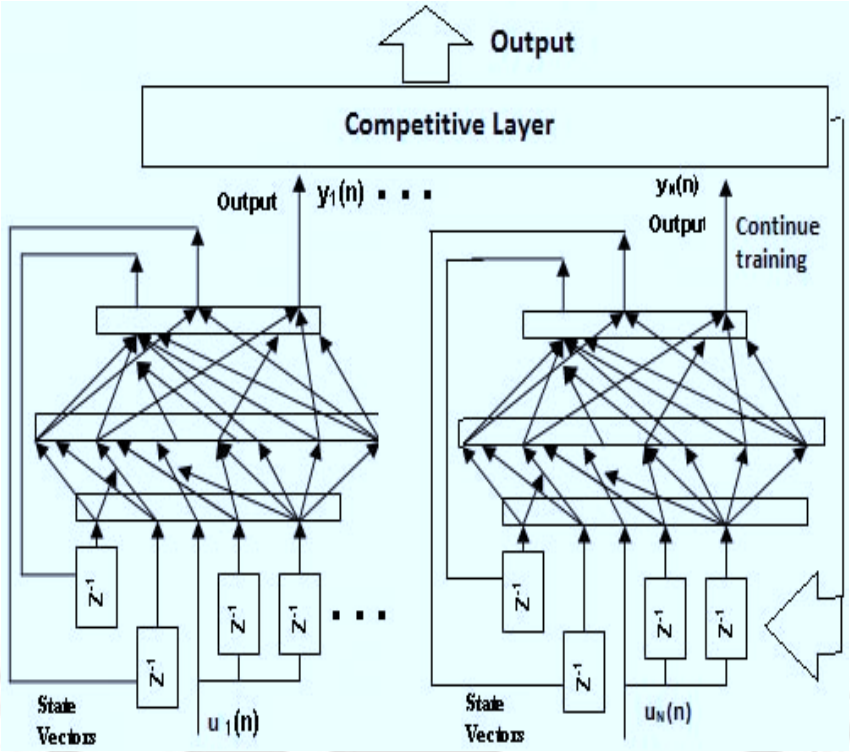


Figure 4.7: CTDFRNN - MNSOM

channel coefficients from a cluster of inputs by considering the most viable similarities. At the end of the training, the RNN units constitute a dynamic mechanism which self-organizes to provide the best approximation of the channel coefficients from a given cluster of inputs. Though the approach adds considerably to the design complexity, the performance improves considerably as shown in Section 4.4.

A generalized SOM approach is adopted in case of a MNSOM which is somewhat different to that of a conventional SOM [108]. This can be observed while mapping a set of objects, $O = O_1, O_2, \cdots, O_I$. In a conventional SOM, each data vector is a mapping object where as in a MNSOM each object corresponds to a non-linear function similar to the input - output relation that exists in an ANN like the RNN. In this case, a CTDFRNN is considered to be the processing module of the MNSOM (Figure 4.7). Let $D_i = r_{i,1}, \cdots, r_{i,j}$ be a data set observed from the i^{th} object O_i such that $r_{i,j} = (x_{i,j}, y_{i,j})$ are input - output vectors and the mapping objects are systems or functions implemented by an ANN. Let the MNSOM have k functional modules $M^1, M^2, \cdots, M^k$ such that each module can approximate an object O_i after training by D_i. Let the property of the k-functional modules be determined by a parameter set θ_k which represents the weight vector of the k^{th} module each of which is formed by a CTDFRNN. While placed in the SOM layout, each RNN is assigned a coordinate. The MNSOM needs to define a distance measure $L^2(O_i, M^k)$ such that a weight update algorithm [108] maybe obtained for the

MNSOM as below:

$$\theta^k(t+1) = arg_\theta min \sum_{i=1}^{L} \phi_i^k(t) L^2(\hat{O}(D_i), M(\theta)) \tag{4.33}$$

where $\phi_i^k(t)$ is the neighbourhood in which the SOM maps its outputs given as:

$$\phi_i^k(t) = exp[-\frac{\|\xi_i^* - \xi_i^k\|^2}{\sigma(t)}]. \tag{4.34}$$

Here ξ_i^* denotes the coordinate of the winner module of the i^{th} object. For a CTDFRNN-MNSOM the distance measure can be expressed as

$$L^2(O_i, M^k) = \int \|f_i(x) - g^k(x)\|^2 p(x) dx \tag{4.35}$$

with $f_i(x)$ being the i^{th} object or non-linear function and $g^k(x)$ is the function representing the RNN module. Here $f_i(x)$ is assumed to be unknown and observed input - output data set to be given. A mean square error (MSE) term maybe obtained as

$$e_{i,j}^k = \|g^k(x_{i,j}) - y_{i,j}\|. \tag{4.36}$$

The MSE module of D_i is concluded to be the winner neuron as per SOM training.

4.4 Results and Discussion

The sample size considered for training includes two different forms of Clarke - Gans channel model each generated using three different AWGN values viz.$-$3dB, 1dB and 3dB for the Gaussian, Rayleigh, Rician and Nakagami faded channels. This way several sample sets with different SNRs are obtained. OFDM signals are included with the MIMO to verify whether the system is able to perform symbol recovery. The samples are accumulated for a 4×4 MIMO - OFDM set-up for each of the channel types considered. The testing includes a range of signal conditions with SNR values ranging from -10 to 10 dB. The testing carried out with inputs from the receiver side calculates channel coefficients and compares them to the theoretically generated values for a frequency range of $0.2 - 8$ GHz. The size of the data involved in the training is not very large, hence a careful selection of training sessions is an important criteria to prevent over training of the RNN and thereby ensure optimum performance. The performance achieved during the training epochs are noted. The RNN training considers the MSE convergence and precision generated in channel estimation and BER calculation. If the MSE has converged to the fixed target value, the precision levels and the associated BER values are calculated.

Table 4.1: Average MSE convergence after 1000 epochs of training for the RNN- architectures trained with four different methods

Architecture	BPTT	RTRL	DEKF	CRTRL
CTDFRNN	6×10^{-5}	0.5×10^{-5}	0.35×10^{-5}	0.35×10^{-5}
CTDFRNNC	4×10^{-5}	0.25×10^{-5}	0.19×10^{-5}	0.19×10^{-5}
CTDFRNN-MNSOM	0.37×10^{-6}	0.35×10^{-6}	0.25×10^{-6}	0.26×10^{-6}

If the values fall within the desired levels, the training is extended to include more number of samples which represent varying channel conditions. The initial experimental part considers the generation of the training and testing samples. The RNN in the mentioned forms are configured and trained. The MSE values attained using four different training methods are summarized in Table 4.1.

The CTDFRNN architecture with SOM optimization generates better performance in terms of time required to reach an MSE value while generating an average success rate of around 96.2 % which is note-worthy in comparison to LS, MMSE, MLP, temporal-MLP and CTDFRNN techniques which reflect their capacity to deal with time varying signals. Yet there is scope of further improvement as demonstrated by the CTDFRNN-MNSOM architecture. Another set of trials are performed with this architecture as well. The role played by the SOM blocks in CTDFRNN is paramount due to the fact that the result generated is an optimized one. For a given window of N-sec.s several sets of output are generated. The optimization process carried out by following the competitive learning algorithm selects the best set of results at the end of training iterations. After these iterations, the result is found to generate the best approximation using which the BER values are generated. Initially all the neurons in the SOM block are given some coordinates and the connecting wights some random values. As the training continues the updating process of the weights continue upto, on an average, 50 sessions after which the generated value is taken to be the result of the RNN- estimator. For a 20 tap channel filter, a training session of the SOM consists of 50 filter coefficient vectors from which the most suitable set is extracted at the end of around 55 to 60 sessions of training of the SOM. The results extracted are average readings of about 25 runs per sample set. The MSE values shown are those obtained during training of the different estimators with 64 bit OFDM blocks in a 4×4 MIMO set-up with atleast 10 iterations between the transmitter and the receiver handling twenty five sets of data of varying length. The MSE plots of LS, MMSE, MLP, temporal-MLP and RNN architectures are depicted in Figure 4.10. The results shown for the CTDFRNN case is derived from values obtained from real and imaginary blocks of the estimators time-averaged over the given time interval. The results are marginally lower compared to the performance shown by the CTDFRNN-SOM which is primarily due to the presence of the SOM block. the SOM proves its ability as

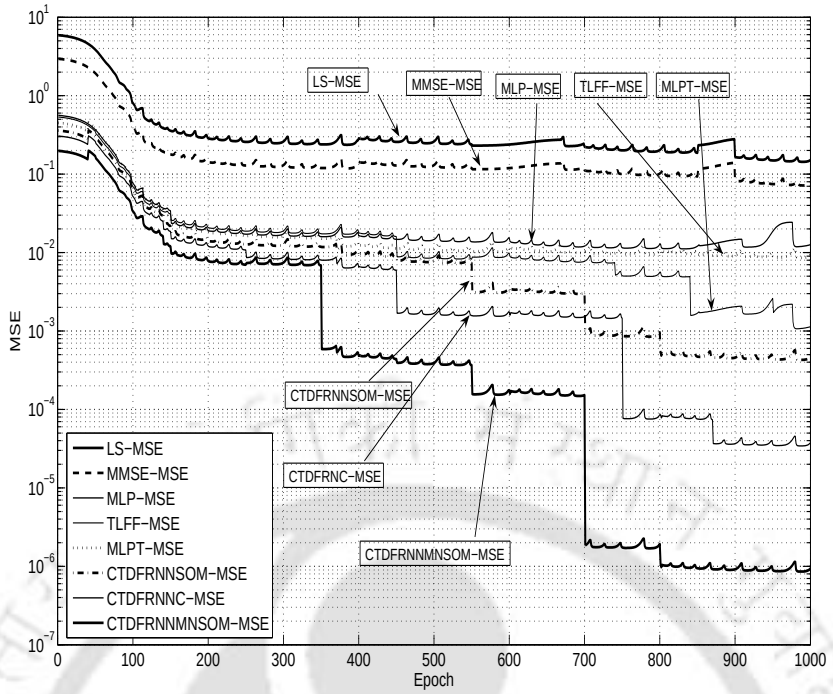


Figure 4.8: MSE convergence plots of statistical methods, ANN based methods and and proposed RNN based methods

an excellent approximator and produces an output which is the best set of values from a number of samples produced over the given time interval. The SOM generated values provide better response than time-averaging where the smoothing operation removes certain vital information content which a learning mechanism like RNN-SOM can use effectively for MIMO channel estimation. The training considerations also include time and frequency domain approaches of generation of signal samples in different block sizes like 64, 128, 6400 and 64,000 bit forms iterated upto 10 times between four transmitters and receivers in MIMO set-up with Rayleigh, Rician and Nakagami fading under varying SNR conditions. It creates considerable amount of uncertainty which the systems are trained to tackle. Moreover the experiments are repeated with multiple block size pilots with the OFDM transmissions to enable pattern-learning by the RNN blocks for proper identification and recovery of the transmitted data. The MIMO set-up is varied in pairs of $2 \times 2, 2 \times 3, 3 \times 3, 3 \times 4, 4 \times 4, 4 \times 5, 5 \times 4$ and 5×5. This is done to see the effect of varying antenna numbers in the MIMO set-up. Some of the results derived are shown in Figure 4.9 which depicts performance comparisons of some MIMO set-ups. The performance difference in terms of BER due to the variation of MIMO transmitter and receiver numbers as stated above is summarized in Table 4.9. A set of results generated by the channel estimation process for a Rayleigh faded case using the RNN architectures

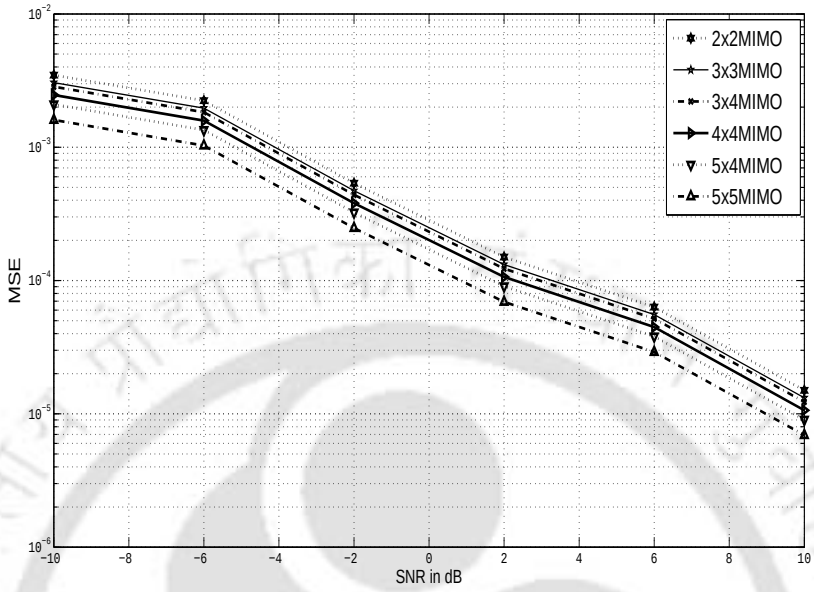


Figure 4.9: BER performance variation of different MIMO set-ups for Rayleigh channel

Table 4.2: BER performance variation of different MIMO blocks compared to a 4×4 set-up

MIMO Set-up	Average difference (%)
2×2	-12.1
3×3	-10.4
3×4	-5.5
5×4	+9.5
5×5	+25.3

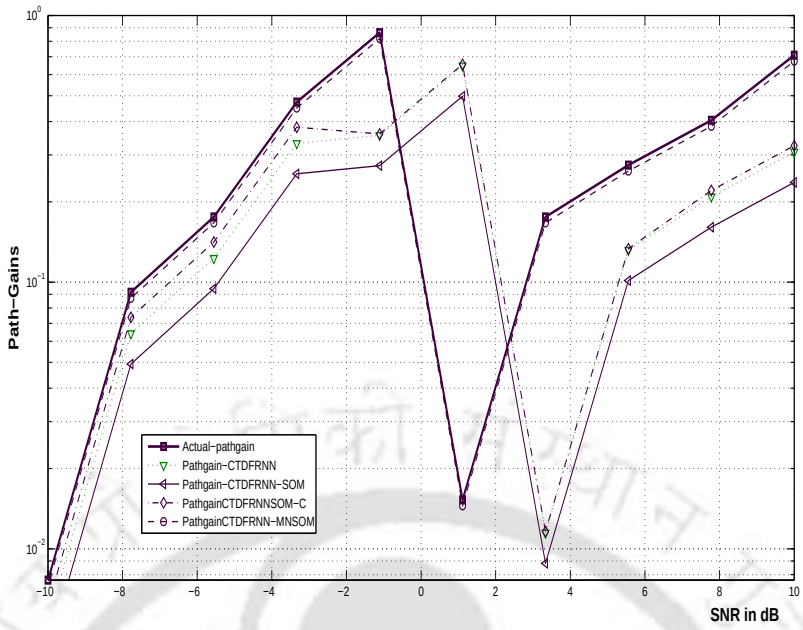


Figure 4.10: Channel coefficients generated by the RNN architectures

is depicted in Figure 4.10. For a specific case involving 64 to 64,000-bit size data blocks, the training of the RNN units continue separately with real and complex components. The samples are created with a maximum of 100 Hz Doppler frequency and continued for a few seconds. The output of the RNN units are accumulated and fed to the SOM block which performs an optimization and produced a response which is an approximation of the expected output. The channel conditions are varied and all the considered types are simulated through which the signal samples are passed. The received signals are used as the inputs to the RNN blocks at the receiver which generates instantaneous approximations of the signals transmitted from which OFDM symbols are recovered.

The results generated for a Nakagami channel are also identical and prove the effectiveness of the RNN architectures configured for MIMO channel modeling. Another set of results are derived for VOIP transmissions where the MIMO set-ups are trained with 8000 bit blocks with twenty five sets of data between 50,000 and 5,00,000. These examples show that the training data size variation plays an important role in determining the convergence parameters of the different RNN architectures proposed and the performance derived. Processing speed, however, is hardware dependent and hence a relative plot for each of the methods compared to an estimator with perfect CSI is shown in Figure 4.11. In terms of number of iterations required, the RNN blocks show significant improvement. There are marked rise of precision levels as well. The CTDFRNN-MNSOM architecture generates the best performance in terms of time required to reach an MSE value in the $10^{-4} - 10^{-6}$ category while generating an average success rate of around 96.7 %. The

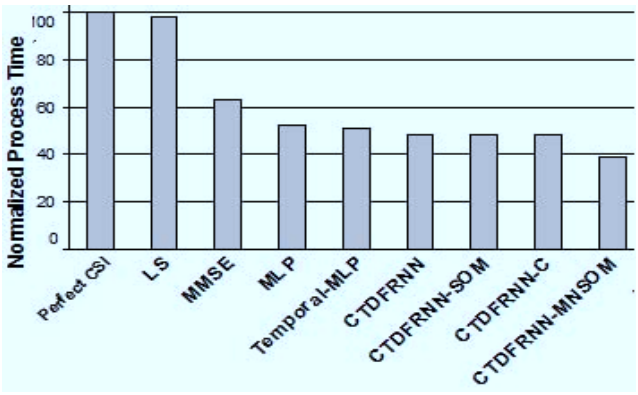


Figure 4.11: Relative number of iterations taken by estimation process with (i) statistical methods (LS, MMSE), (ii) ANN based standard methods (MLP, Section 3.2), 3L-FF [10], Temporal-MLP (Section 3.4), CTDFRNN (Section 4.3.1) and (iii) the CTDFRNN-SOM architectures (CTDFRNN-SOM, CTDFRNN-C, CTDFRNN-MNSOM) (Sections 4.3.2-4.3.3)

enhanced performance is significant in comparison to LS, MMSE, MLP and temporal-MLP techniques. This improvement is also reflected in the BER values generated by taking SNR values between -10 and 10 dB. The composite plot is depicted in Figure 4.12. This plot while showing the BER graphs of the different methods, depicts the performance of an estimator assumed to possess perfect CSI. The performance provided by the CTDFRNN-MNSOM nearly approaches the estimator with perfect CSI knowledge. The improvement in BER vlaues as shown by the RNN architectures reflect their capacity to deal with time varying signals. Thus these are found to be suitable for MIMO channel estimation and OFDM symbol recovery.

CTDFRNN units learn over the given period of time by connecting contextual knowledge to different layers and allowing subsequent samples to use these stored information as update reference. This leads to an accumulation of the knowledge which, due to the multiple delays, generates a set-up for learning, comparing and retaining relevant information only. Moreover, a subtractive process rejects common data and holds back only relevant information, hence correlation among circulating samples inside the CTDFRNN layers are less. As a result, the convergence process to the optimal level is fast and no problems of 'local-minima', as observed in MLPs, are seen. This further helps the RNN based architectures to avoid the curse of dimensionality. Moreover, CTDFRNN in clusters creates a cooperative environment which further rejects common elements of input, feedback and state samples. In basic split activation form, the error is reduced considerably due to the architectural contributions for which the higher order clusters decrease further with optimum resource utilization. Thus, RNN based architectures, for deploying the split-activation, feedback, delayed inputs and cumulative knowledge circulation carried out in a real-time cooperative learning environment, help in tracking time-varying inputs

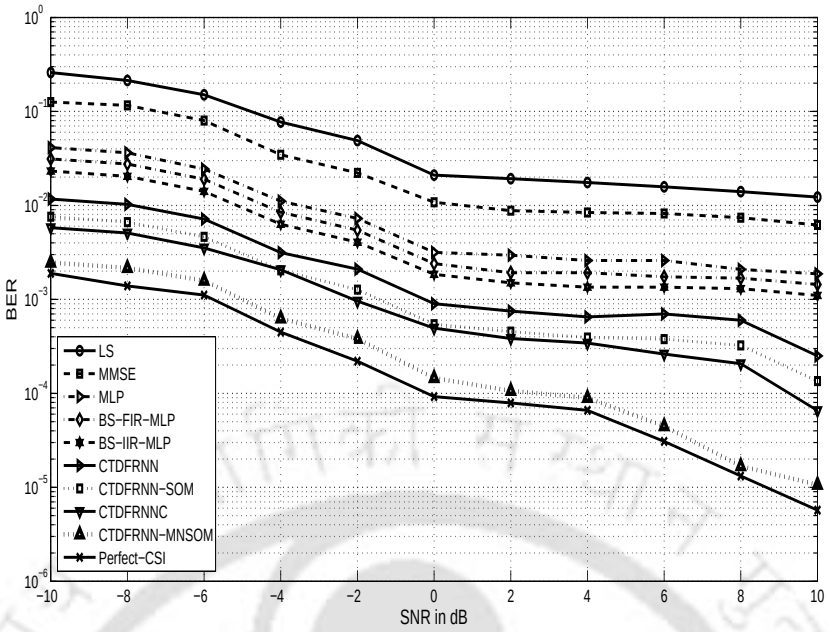


Figure 4.12: BER plots of statistical methods, ANN based standard methods and proposed RNN based methods

Table 4.3: Computational complexity of RNN-based architectures compared to two temporal-MLP architectures with N length filter (for temporal MLP structures) or delay blocks (for RNN-based structures), n number of parallel structures, l size competitive layer and $(P+R)$ length of the signal

Model	Number of	
	additions	multiplications
FIR-MLP	$4N+2$	$6N$
IIR-MLP	$7N+2$	$10N$
CTDFRNN	$7N$	$8N$
CTDFRNN-SOM	$7N+1+l$	$7N+2l$
CTDFRNN-C SOM	$7N^n$+2+l	6^n+2l
CTDFRNN-MNSOM	$(P+R)7N+1$+l	$(P+R)7N+1$+2l

for which it effectively handles a highly complicated process like MIMO channel estimation. A set of experiments are also performed to carry out symbol recovery. The results obtained show marginal improvement compared to that provided by the temporal-MLP and hence have not been included here.

Considering l to be the length of a competitive layer, n to be the number of parallel structures and $(P+R)$ to be the length of the signal, the computational complexity of the RNN-based architectures are shown in Table 4.3. It should be noted that the proposed RNN structures, namely, CTDFRNN-SOM, CTDFRNN-C and CTDFRNN-MNSOM, in-

Table 4.4: Time complexity and precision for LS, MMSE, MLP, temporal - MLP and RNN architectures

Method	Iterations	Average Precision (%)
LS	52	91
MMSE	33	92
MLP	25	94
3L-FF	23	94.5
Temporal-MLP	23	95
CTDFRNN	23	95.7
CTDFRNN-SOM	23	96.2
CTDFRNN-C	18	96.5
CTDFRNN-MNSOM	15	97.4

creases the computational complexity in the order due to the introduction of more level of parallel processing. Table 4.3 clearly shows that a basic CTDFRNN-SOM has around $(P+R)$ times lesser complexity than the CTDFRNN-MNSOM where $(P+R)$ is the length of the signal. However, as shown in Figure 4.11, the time-complexity of CTDFRNN-MNSOM is around 23% less than that of CTDFRNN-SOM. Moreover, Table 4.4 shows the average time complexity and precision of the RNN based architectures with respect to LS, MMSE, MLP, temporal-MLP configurations and the three layered feedforward ANN with feedback (3L-FF) referred to in [10]. This table depicts that the precision of the CTDFRNN-MNSOM is atleast 23.1% better than the basic CTRFRNN-SOM. Thus, the CTDFRNN-MNSOM is preferred for operational purposes. However, for a 4×4 MIMO set-up trained with 64 bits under Rayleigh channel conditions in an indoor environment for a wireless VoIP application, we found that the precision performance of the systems used for MIMO channel modeling plays a critical role. This can be more emphatically established by considering a severely faded channel in a VOIP based transmission infested with considerable CCI. Here, for an average increase of 0.9% in precision of performance provided by CTDFRNN-MNSOM compared to CTDFRNN-C, the BER values show a fall of 50-66% as shown by Figure 4.13. It necessitates the importance of design of systems for MIMO channel modeling with additional aids which, along with increase in precision by even a small fraction, shall eventually improve BER performance significantly as well as can yield lesser computational complexity than CTDFRNN-MNSOM.

4.5 Conclusion

In this chapter we have proposed a new set of complex RNN architectures with SOM optimization for MIMO channel modeling. The basic challenge in doing so has been to

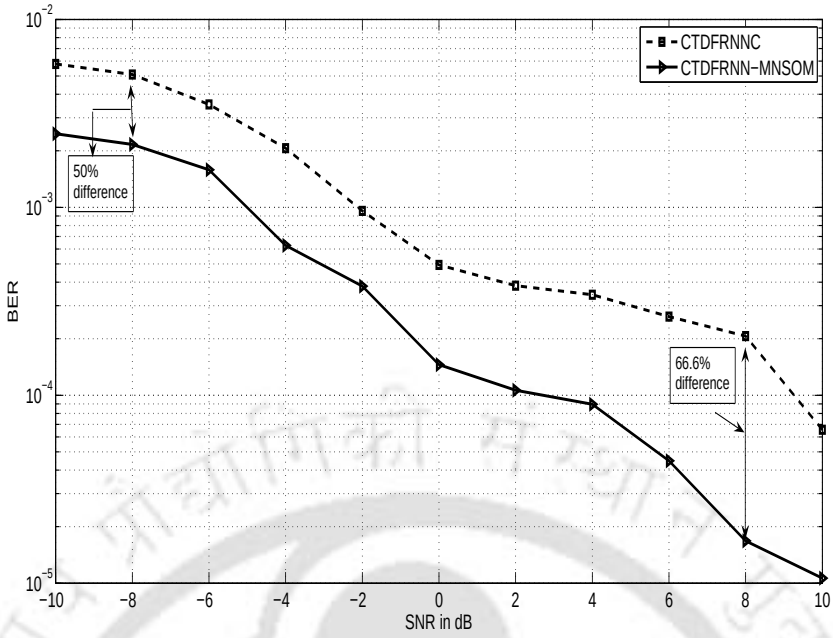


Figure 4.13: BER performance gain obtained with CTDFRNN-MNSOM compared to CTDFRNN-C architecture

keep the architecture as effective and as simple as possible while dealing with signals with in-phase and quadrature components. The objectives have been to improve performance in terms of lower BER values, reduced processing time and enhanced precision. In particular, we have proposed CTDFRNN estimator with SOM optimization, CTDFRNN-C and CTDFRNN-MNSOM architectures for MIMO channel modeling which provide continuously improving BER values and lower processing delays. It should also be noted from Table 4.4 that the CTDFRNN-MNSOM has less time-complexity than either FIR-MLP or IIR-MLP. Therefore, from all these results, it can be concluded that the CTDFRNN-MNSOM provides the best results among the proposed frameworks. However, in some applications such as short length wireless VOIP based transmission while modeling MIMO channels, the precision plays a crucial role as is apparent from Figure 4.13. It also suggests that in such severely faded channels with certain application, CTDFRNN-MNSOM may not provide the required precision level within a certain limit of computational complexity. In such cases, a simpler RNN structure along with a decision driven tool such as fuzzy system may provide better design solution. Such a proposal is given in the next chapter.

Chapter 5

MIMO Channel Estimation using Fuzzy Based System

5.1 Introduction

In Chapter 4, we showed how a class of RNN architectures in combination with SOM units provide satisfactory performance in terms of lower BER values, reduced processing time and enhanced precision while modeling time-varying MIMO channels. We also showed how the betterment of precision by just 0.9% enabled the CTDFRNN-MNSOM to provide lesser BER values by atleast 50% compared to CTDFRNN-C which is significant. In fact, in certain practical cases like a VOIP based transmission in a severely faded channel, the performance of RNN-based methods needs to be improved further especially in connection with the accuracy of the recovered content associated with such precision oriented communication. In such situations, the natural option that emerges as the suitable addition along with a simpler RNN, to enhance the system precision as well as to reduce the computational complexity, is either fuzzy system or the genetic algorithm (GA). The primary limitation associated with GA, however, is the underlying computational complexity and therefore the time constraint. Hence, the fuzzy based system integrated with RNN block turns out to be the only viable option to attain the desired precision performance. Further, fuzzy systems possess an integral ability to deal with uncertainty, can discern minute variations, attach probabilistic linkages to such finer changes [35] [110] and facilitate the formation of a mechanism for near error-free decision making using ANN. Also, the uniqueness associated with these soft-computational tools enables a smooth blending of fuzzy system with ANN in a supplementary-complementary arrangement which leads to enhancement in performance in terms of processing time, lower system complexity, increase in precision, better adaptability and an unmatched capability to track and model uncertain behaviour of stochastic MIMO channels. The

end result is a mechanism which shares a symbiotic association with each other that enhances respective and cumulative performances. The ANN provides process data extracted from non-parametric sources executing the task with model-free processing such that the fuzzy system uses the neural response to provide expert-level decision with the aim to achieve better overall performance. This is in addition to the stability of the set-up and the capability to process minutely varying contextual and relevant information due to the presence of feedback loops in the RNN [34] components encapsulated in the system with fuzzy-related modifications. In terms of implementation aspects, the advantage of such a composite architecture is a set-up which can make better discrimination between the significant and correlated coefficients of time-varying channels such that the correct class-wise grouping of the system improves and misclassification reduces with less processing time and lower design complexity. For its inherent capability to derive expert level knowledge and decision making [35] [110], fuzzy based systems have already been used in wireless communication in areas like equalization etc. A few important works are cited in [41] -[52]. All of these works emphasize applications of fuzzy-based systems for wireless communication areas with nearly no stress on architectural challenges and benefits that can be derived using such systems with modifications. Certain challenges can be identified as (i) combining ANN and fuzzy based systems to obtain the capability of expert-level decision making while modeling uncertainty observed in the MIMO channel and (ii) realization of a suitable system for adaptive receiver design with lower implementation and time complexity.

In this chapter, we propose the formulation of certain fuzzy-based systems using TDFRNN with the ability to deal with time-varying inputs in fuzzified form to meet the above two challenges while modeling time-varying MIMO channels. The fuzzified TDFRNN blocks are designed to generate a set of input conditioning norms and inference logic which captures finer variations in the signals transmitted by the MIMO set-up. The fuzzification process is automated using a ANN-aided approach while two distinct TDFRNN blocks generate the classification boundaries in terms of in-phase and quadrature components combined and optimized using a Self Organizing Map (SOM) unit. The training process is accelerated by constantly changing fuzzy samples generated by dissimilar processing blocks. The results show significant improvement in processing time and accuracy as compared to ANN based approaches. The fuzzy systems, on an average, provide atleast 5% improvement in accuracy as compared to the RNN-based estimation which has already been established to be a better alternative to statistical and ANN based approaches as stated in Chapter 4. This is in addition to the processing time advantage that the fuzzy systems provide.

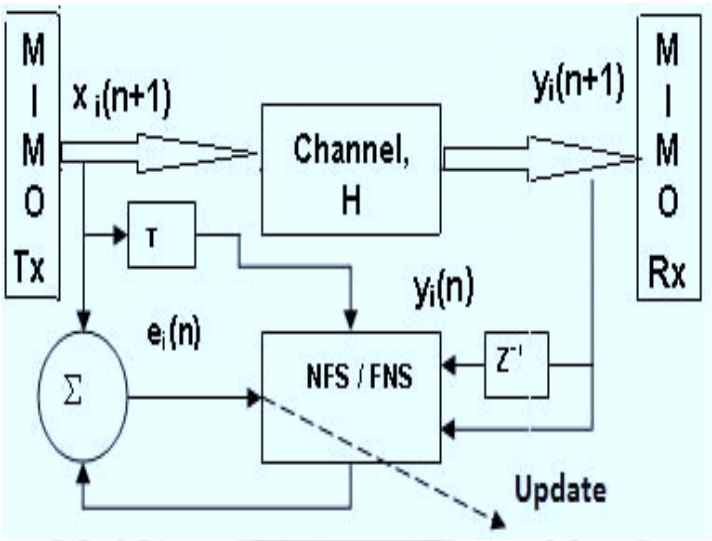


Figure 5.1: Fuzzy system based channel estimation

5.2 MIMO Channel Estimation using Fuzzy Based Systems

Fuzzy systems are rule driven tools that provide expert-level knowledge for decision making [35] [110]. ANNs, on the other hand, are adaptive, robust and non-parametric prediction techniques that demonstrate cognitive behaviour [34]. Fuzzy systems alone, however, find it difficult to tackle uncertainty hence require support from other tools like ANN. Integration of a fuzzy system with ANN enables the former to acquire capabilities of implementing a framework of processing modalities like signal conditioning rules, inference norms etc so that the composite systems acquire the ability of better learning, retentivity of knowledge and expert-level decision making with the ability to capture finite variations in the input. Hence, hybrid forms, like Fuzzy-Neural System (FNS) or Neuro-Fuzzy System (NFS) are popular in a diverse range of applications. A generic framework considering such a hybrid system is depicted in Figure 5.1. However, experimental results derived for VOIP based transmissions show that the FNS approach is better suited for time-varying MIMO channels and hence is adopted in our work for such a highly volatile propagation environment.

Therefore, before we go to our proposal, we briefly discuss the merits and demerits of FNS and NFS in the following section.

5.2.1 Fuzzy-MIMO channel estimation: Neuro-Fuzzy or Fuzzy Neural?

As discussed earlier, hybrid systems formed by combinations of fuzzy and ANN methods have adaptability, parallelism, non-linear processing, robustness and learning in data rich environment and modeling uncertainty. A fuzzy based system is formed by a fuzzification layer, an inference engine and a de-fuzzification stage. Available literature reports the combination of fuzzy systems and ANNs in neuro-fuzzy (NF) or fuzzy-neural (FN) forms that demonstrate adaptability and robustness while handling unknown process or situations. Each of these systems has their advantages and disadvantages which are considered while designing applications involving them. The NFS processes numerical relationships and uses ANNs to derive the parameters of a fuzzy system. It performs crisp input to output mapping using ANNs. There can be several layers, the first being an ANN which maps process data (crisp) into fuzzy form. These fuzzy samples are used by the inference engine to make decisions. The de-fuzzification layer again converts the fuzzified form into real world values. NFS systems don't have the ability to combine perception based information and are used to derive solutions involving the ANN's advantages of model-free learning, appropriate generalization and powerful non-linear mapping capabilities [35]. The approach followed in the FNS is somewhat different. The FNS is an implementation of a fuzzy system within ANN architecture. In this approach, both numerical (measurement based) information and perception based information represented as fuzzy numbers are handled. Therefore, FNS capture more relevant content from an input, hence is better suited for real world situations [111]. The FNS provides a fuzzy input to output mapping unlike the NFS which relates a crisp input to a corresponding output. As a result, the FNS has greater ability to keep track of finer contextual variations unlike the NFS. Also, FNS allows automation of generation of fuzzy rules and has the ability to perform combined learning of numerical data as well as expert-knowledge expressed as fuzzy if-then-else rules. Moreover, FNS have smaller networks and faster processing time compared to ANNs and NFS [111], hence are more suitable for applications like adaptive receiver designs for high data rate mobile communication.
In the following sections, we provide our proposal along with experimental results. Finally, in Section 5.4, the NFS approach is also covered along with a brief comparison of this method vis-a-vis our proposed FNS approach.

5.2.2 FNS MIMO channel estimation

The FNS deviates considerably from the traditional ANN and has the following attributes [109]: (i) fuzzy inputs, (ii) fuzzy outputs, (iii) fuzzy weights and (iv) fuzzy aggregation operation instead of summation as observed in ANNs. For the present work a FNS MIMO

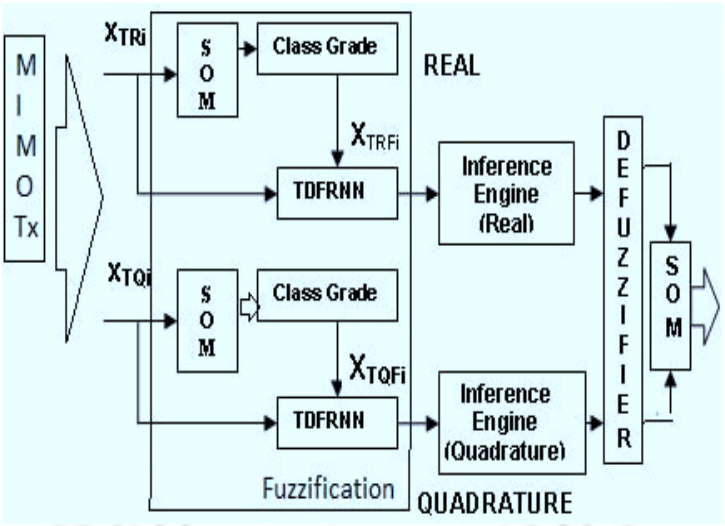


Figure 5.2: Proposed FNS MIMO channel estimation

Table 5.1: Linguistic steps used to condition the inputs

Sl. no.	State	Notation
1	Negative Large	NL
2	Negative Medium	NM
3	Negative Small	NS
4	Zero	Z
5	Positive Large	PL
6	Positive Medium	PM
7	Positive Small	PS

channel estimator is formulated and is depicted in Figure 5.2. The primary input to the system comes from a MIMO transmitter during training. The inputs in in-phase and quadrature forms are fuzzified and membership grades assigned. The fuzzy inputs next go to the inference engine which decides upon the class decisions as per the assigned inference logic. the output is de-fuzzified and obtained from real and imaginary sections and combined. The outputs are trimmed using a Self Organizing Map (SOM) optimizer. The details of each of these sections are discussed in the following.

5.2.2.1 Inputs and fuzzification

The MIMO channel input-output relationship is already given in Section 2.2.1. The training samples are normalized and confined within a few linguistic steps [110] as shown in Table 5.1. The inputs are also constrained by the following norms:

$$f_1(x) = \begin{pmatrix} NL, -0.99 \leq x < -0.66; \\ NM, -0.66 \leq x < -0.33; \\ NS, -0.33 \leq x < 0; \\ Z, \quad x = 0; \\ PL, \quad 0.66 \leq x < 0.99; \\ PM, \quad 0.33 \leq x < 0.66; \\ PS, \quad 0 < x < 0.33; \end{pmatrix}. \tag{5.1}$$

The performance of fuzzy based systems depend on the norms adopted for conditioning inputs and decision making. The set of norms adopted to see the effect of inclusion of greater number of conditioning steps in the overall performance of the system is formulated as a: new set as:

$$f_2(x) = \begin{pmatrix} Minus9, -0.99 \leq x < -0.88; \\ Minus8, -0.88 \leq x < -0.77; \\ Minus7, -0.77 \leq x < -0.66; \\ Minus6, -0.66 \leq x < -0.55; \\ Minus5, -0.55 \leq x < -0.44; \\ Minus4, -0.44 \leq x < -0.33; \\ Minus3, -0.33 \leq x < -0.22; \\ Minus2, -0.22 \leq x < -0.11; \\ Minus1, -0.11 \leq x < -0.01; \\ Z, \quad x = 0; \\ Plus1, \quad 0.01 \leq x < 0.11; \\ Plus2, \quad 0.11 \leq x < 0.22; \\ Plus3, \quad 0.22 \leq x < 0.33; \\ Plus4, \quad 0.33 \leq x < 0.44; \\ Plus5, \quad 0.44 \leq x < 0.55; \\ Plus6, \quad 0.55 \leq x < 0.66; \\ Plus7, \quad 0.66 \leq x < 0.77; \\ Plus8, \quad 0.77 \leq x < 0.88; \\ Plus9, \quad 0.88 \leq x < 0.99; \end{pmatrix}. \tag{5.2}$$

To ascertain the variation in performance, two sets of input conditioning methods (equations 5.1 and 5.2) and inference rules are derived and variations in outcomes noted while dealing with time-varying MIMO channels invested with multiple interferences. The inputs are divided into in-phase and quadrature components to allow the FN-system to learn the individual signal segments separately. The fuzzy sets of the respective inputs are generated following two ways:

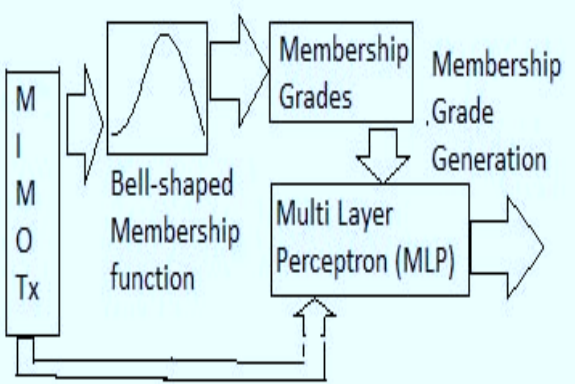


Figure 5.3: Membership grade generation using Bell-shape function and trained MLP

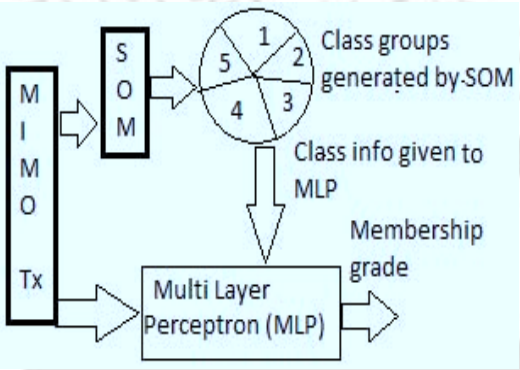


Figure 5.4: Membership grade generation using SOM and trained MLP

1. In the first method, a Bell-shaped membership [110] function is used to generate the member-grades of each input. A one-to-one correspondence is established between input and fuzzy sets. This association is taken to train an MLP to generate the membership function. At the end of the training, the MLP becomes an automatic membership generator (Figure 5.3).

2. The second method is to use a SOM to create clusters for each of the samples forming the input matrix. This clustering is used to train a MLP to act as an automated membership generator (Figure 5.4).

5.2.3 Experimental considerations

The FNS structure deviates considerably from the traditional ANN and has the following attributes [110]: fuzzy inputs, weights, outputs and aggregation operation instead of summation as observed in ANNs. For the present work a FNS MIMO channel estimator is formulated and is shown in Figure 5.2. Let x_T be the input to the MLP (Figure 5.3) while y_{EMG} is the expected set of membership grades given by the Bell-shaped function.

The actual output of the 3-layered MLP in this case is given as

$$y_{AMG} = \sum_k f_k[\sum_i f_i[\sum_j f_j(x_{Tj}W_{jn} + b_j)W_{in}]] \tag{5.3}$$

where for n^{th} sample in j^{th}, i^{th} and k^{th} layers $f(.)$s are activation functions, $W[.,.]$'s are connectionist weights and b's biases related to specific layers of the MLP that produce the response for given inputs. The instantaneous error for p^{th} pattern is given as

$$E_p = \sum_k [y^p_{EMG_k} - y^p_{AMG_k}]^2 \tag{5.4}$$

such that the weight adaptation can be continued till the goal is reached. The weight adaptation process is related to a gradient given as

$$\Delta W_{kj} = -\eta \frac{\partial E_p}{\partial W_{kj}} \tag{5.5}$$

for use in the training process. Here η is the learning rate.
In case of the SOM-MLP approach, the membership grades are generated as:

$$y_{EMG} = x_T[W_j] \tag{5.6}$$

where W_j is the random weights of the competitive layer of the SOM. The 'winners take all' competition starts [34] in this layer such that the winning neuron index, J, satisfies

$$y_J = max_j\{y^T_{0j}, W_j\}. \tag{5.7}$$

For a 4×4 MIMO set-up, four training signals are transmitted with three different AWGN values viz.−3 dB, 1 dB and 3 dB for a Gaussian, Rayleigh and Rician faded channels. The channel has h_{11}, h_{22}, h_{33} and h_{44} as primary direct channel impulse responses while $h_{12}, h_{21}, h_{31}, h_{41}, \cdots$ etc are cross-channel impulse responses between the specific transmitter and the receiver. The input training sequences x_1, x_2, x_3, x_4 are transmitted in data blocks with each block constituted by N_I information and N_T training symbols. For different time slots the training sequences provide

$$h_{i1} = \frac{Y_{Ti,1}}{x_i}, h_{i2} = \frac{Y_{Ti,2}}{x_i}, h_{i3} = \frac{Y_{Ti,3}}{x_i}, h_{i4} = \frac{Y_{Ti,4}}{x_i} \tag{5.8}$$

where $i = 1, 2, \cdots, 4$. Let the above knowledge be considered to be *apriori*. So if during training $[\tilde{h}_{11}, \tilde{h}_{12}, \cdots, \tilde{h}_{21}, \tilde{h}_{22}, \cdots, \tilde{h}_{41}, \cdots, \tilde{h}_{44}]$ are the estimates of channel coefficients, the fuzzy inference can be developed from the risk functional $d(h_{ij}, \tilde{h}_{ij})$ where $d(.)$ represents the distance measure between actual and expected responses. The fuzzified

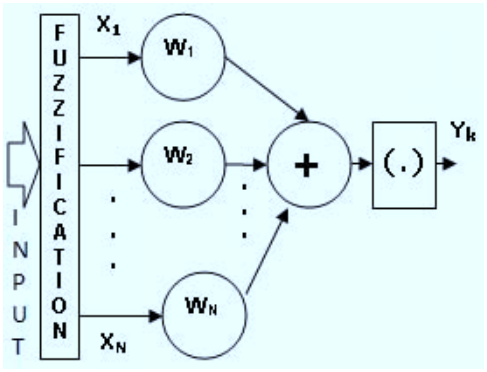


Figure 5.5: Fuzzy artificial perceptron

inputs are generated by the process as described in Section 5.2.2.1. The fuzzified inputs thus obtained are next passed on to the inference engine which is a multi-layered set-up designed by following two approaches based on

1. Fuzzy Multi Layer Perceptron (FMLP) and
2. Fuzzy Time Delay Fully Recurrent Neural Network (FTDFRNN).

5.2.3.1 Fuzzy MLP (FMLP) based inference engine

The FMLP is formed by multiple layers of fuzzy neurons (Figure 5.5). The inputs (X_{kj}), weights (W_j) and outputs are all fuzzified [109]. The output of such a set of neurons can be expressed as

$$Y_k = \sum_k S_{\beta,k}(\sum_{j=0}^{n} W_j X_{kj}) \tag{5.9}$$

where S_β is a sigmoid function for certain steepness parameter β. The calculation covered by the summation sign in (5.9) is done using principles of fuzzy arithmetic. The output thus obtained is next passed on to the hidden and output layers such that the final response is expressed as

$$Y_m = \sum_m S_{\beta,m}\{(\sum_l S_{\beta,l}(\sum_k Y_k W_{kl})W_{lm})\}. \tag{5.10}$$

For separate real and imaginary inputs the respective outputs are combined so that the final response is found as

$$Y_{mF} = \sum_m S_{\beta,m}\{(\sum_l S_{\beta,l}(\sum_k Y_{kR} W_{kl}))\} + j\sum_m S_{\beta,m}\{(\sum_l S_{\beta,l}(\sum_k Y_{kI} W_{kl}))\}. \tag{5.11}$$

For determining the channel coefficients $[\tilde{h}_{11}, \tilde{h}_{12}, \cdots, \tilde{h}_{21}, \tilde{h}_{22}, \cdots, \tilde{h}_{41}, \cdots, \tilde{h}_{44}]$, the expression given by (5.11) is used after completing the training using back-propagation

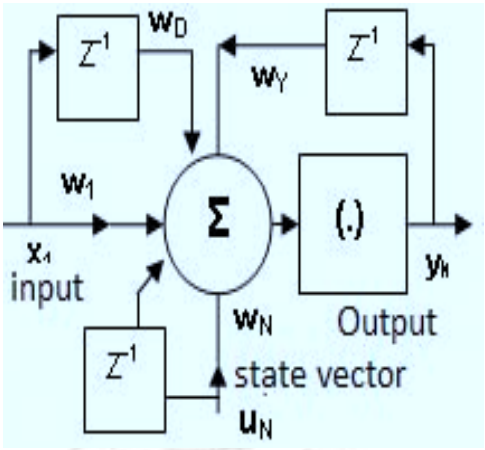


Figure 5.6: Fuzzy recurrent neuron

algorithm.

5.2.3.2 Fuzzy TDFRNN (FTDFRNN) based inference engine

The FMLP based inference system is effective in case of static, slowly varying channels. The training and estimation time required, however, is much less than the pure ANN based systems as described in Section 5.3. It can also tackle time-varying channels but there is always a scope for further improvement. However, as RNNs are known to be suitable for time-varying inputs, hence an option emerges based on these ANNs. Such a combination called Fuzzy RNN (FRNN) is specially configured as a fuzzy inference system. The core of a FRNN is a Fuzzy Recurrent Neuron (FRN) shown in Figure 5.6. The MLP block of a FMLP is replaced by a TDFRNN with split-activation so that the combination can use de-coupled in-phase and quadrature components of input signals and use a SOM to combine the outputs in an optimized form. The output of such a set of neurons for a set of inputs X_j and state vectors $u_{N,j}$ is expressed as

$$Y_{rk}(n) = \sum_k S_{\beta,k}(\sum_{j=0}^{n}(W_j X_j + W_{j-1}X_{j-1} + W_{j,y}Y_{k-1} + W_{jN}u_{N,j} + W_{jN}u_{N-1,j})). \quad (5.12)$$

After the input propagates through the hidden and the output layers, for a real input X_{Rj}, the response shall be

$$Y_{rmR} = \sum_m S_{\beta,m}\{(\sum_l S_{\beta,l}(\sum_k Y_{rkR}W_{kl} + u_{M-i,j}W_{jM})W_{lm})\}. \quad (5.13)$$

A similar expression for imaginary inputs can also be obtained as

$$Y_{rmI} = \sum_m S_{\beta,m}\{(\sum_l S_{\beta,l}(\sum_k Y_{rkI}W_{kl} + u_{M-i,j}W_{jM})W_{lm})\}. \quad (5.14)$$

Table 5.2: Inference rule sets for decision making in a FNS based MIMO channel estimator

Set	Y_{imax}	Y_{imin}	Y_T inference	$\hat{h}$ inference	Remark
1	1.0	0.001	$y_T = 0.99\sum y_i h_{ii} + 0.01\sum y_i h_{ij}$	$\hat{h} = 0.99\sum h_{ii} + 0.01\sum h_{ij}$	$\hat{h}_{ii} \uparrow$
	0.9	0.1	$y_T = 0.99\sum y_i h_{ii} + 0.01\sum y_i h_{ij}$	$\hat{h} = 0.99\sum h_{ii} + 0.01\sum h_{ij}$	$\hat{h}_{ii} \uparrow$
	0.7	0.3	$y_T = 0.7\sum y_i h_{ii} + 0.3\sum y_i h_{ij}$	$\hat{h} = 0.7\sum h_{ii} + 0.3\sum h_{ij}$	$\hat{h}_{ii} \uparrow$
	0.5	0.5	$y_T = 0.5\sum y_i h_{ii} + 0.5\sum y_i h_{ij}$	$\hat{h} = 0.5\sum h_{ii} + 0.5\sum h_{ij}$	$\hat{h}_{ii} \uparrow, \hat{h}_{ij} \uparrow$
	0.3	0.5	$y_T = 0.3\sum y_i h_{ii} + 0.5\sum y_i h_{ij}$	$\hat{h} = 0.3\sum h_{ii} + 0.5\sum h_{ij}$	$\hat{h}_{ii} \downarrow, \hat{h}_{ij} \uparrow$
	0.3	0.7	$y_T = 0.3\sum y_i h_{ii} + 0.7\sum y_i h_{ij}$	$\hat{h} = 0.3\sum h_{ii} + 0.7\sum h_{ij}$	$\hat{h}_{ii} \downarrow, \hat{h}_{ij} \uparrow$
2	0.99	0.88	$y_T = 0.9\sum y_i h_{ii} + 0.1\sum y_i h_{ij}$	$\hat{h} = 0.9\sum h_{ii} + 0.1\sum h_{ij}$	$\hat{h}_{ii} \uparrow, \hat{h}_{ij} \downarrow$
	0.87	0.77	$y_T = 0.87\sum y_i h_{ii} + 0.13\sum y_i h_{ij}$	$\hat{h} = 0.87\sum h_{ii} + 0.13\sum h_{ij}$	$\hat{h}_{ii} \uparrow, \hat{h}_{ij} \downarrow$
	0.76	0.66	$y_T = 0.76\sum y_i h_{ii} + 0.24\sum y_i h_{ij}$	$\hat{h} = 0.76\sum h_{ii} + 0.24\sum h_{ij}$	$\hat{h}_{ii} \uparrow, \hat{h}_{ij} \downarrow$
	0.65	0.55	$y_T = 0.65\sum y_i h_{ii} + 0.35\sum y_i h_{ij}$	$\hat{h} = 0.65\sum h_{ii} + 0.35\sum h_{ij}$	$\hat{h}_{ii} \uparrow, \hat{h}_{ij} \downarrow$
	0.54	0.44	$y_T = 0.54\sum y_i h_{ii} + 0.44\sum y_i h_{ij}$	$\hat{h} = 0.55\sum h_{ii} + 0.44\sum h_{ij}$	$\hat{h}_{ii} \downarrow, \hat{h}_{ij} \uparrow$
	0.43	0.33	$y_T = 0.43\sum y_i h_{ii} + 0.57\sum y_i h_{ij}$	$\hat{h} = 0.43\sum h_{ii} + 0.57\sum h_{ij}$	$\hat{h}_{ii} \downarrow, \hat{h}_{ij} \uparrow$
	0.22	0.11	$y_T = 0.22\sum y_i h_{ii} + 0.78\sum y_i h_{ij}$	$\hat{h} = 0.22\sum h_{ii} + 0.78\sum h_{ij}$	$\hat{h}_{ii} \downarrow, \hat{h}_{ij} \uparrow$
	0.1	0.09	$y_T = 0.1\sum y_i h_{ii} + 0.9\sum y_i h_{ij}$	$\hat{h} = 0.1\sum h_{ii} + 0.9\sum h_{ij}$	$\hat{h}_{ii} \downarrow, \hat{h}_{ij} \uparrow$
	0.08	-	$y_T = 0.08\sum y_i h_{ii} + 0.92\sum y_i h_{ij}$	$\hat{h} = 0.08\sum h_{ii} + 0.92\sum h_{ij}$	$\hat{h}_{ii} \downarrow, \hat{h}_{ij} \uparrow$

For a duration of n seconds if training samples are presented to the inference engine, the output matrix generated is optimized by a SOM such that the final result obtained has a form given by

$$Y_{rF} = \varnothing\{Y_{rmR}(n) + jY_{rmI}(n)\} \tag{5.15}$$

where $\varnothing\{.\}$ is an optimization process carried out by following 'winners take all' competition such that the winning neuron index, J, satisfies

$$Y_J(n) = max_j\{Y_{rFj}^T, W_j\} \tag{5.16}$$

A detailed treatment of competitive learning of the SOM is given in Section 2.3.3.

5.2.4 Inference rule set

Two sets of inference rules are formulated as shown in Table 5.2. The first set has less number of intermediate states while the second one contains more steps to facilitate finer classification and thereby generates better precision. The set-up is constituted to handle the inputs and rely on an inference mechanism for fast decision making with greater precision. The objective is to determine the optimal combination of input norms and inference rules so that the decision making process in the FNS MIMO estimator provides better precision and faster processing speed compared to ANN based approaches. Appropriate encoding schemes are generated for implementing the inference rules which

are executed by FTDFRNN blocks with split activation for in-phase and quadrature components. The output decisions in fuzzy form are optimized by a SOM. In this form these blocks are called Fuzzy Complex TDFRNN (FCTDFRNN) with FRNs as basic building units. The FCTDFRNN block shows excellent capacity to deal with time variation observed in MIMO transmissions.

5.2.5 Defuzzification

After the fuzzified outputs are generated, a mapping is performed to convert each of the conclusion into a single real number. This mapping process provides the required estimation of the channel coefficients obtained from a fuzzy inference. There are several defuzzification methods but the 'Center of Arc' or 'Centroid Method' proposed by Sugeno (1985) and Lee (1990) is the most acceptable of all [109]. For a discrete case, if C represents a finite universal set $\{z_1, z_2, \cdots, z_n\}$, the defuzzification method is expressed as

$$d_{CA}(C) = \frac{\sum_{k=1}^{n} C(z_k) z_k}{\sum_{k=1}^{n} C(z_k)}. \tag{5.17}$$

5.3 Results and Discussion

The performance of the FNS based MIMO channel estimation method is dependent on several aspects. The complete system performance is affected by the responses of the fuzzification, inference and defuzzification stages. Section 5.3.1 discusses about the results obtained from the two membership generation approaches. Section 5.3.2 includes the performance of the fuzzy inference engines designed using FMLP and FTDFRNN approaches. The summarized results of the fuzzy-based MIMO channel estimation is included in Section 5.3.3.

5.3.1 Performance of the two membership generation approaches

A comparative depiction of the performance of the above two membership grade generation methods are shown in Figure 5.7. While the SOM-MLP method is less accurate on an average by about 4% but it is faster by atleast 21%, hence it is more acceptable for speed critical applications but for situations where precision is of greater importance the Bell-MLP method is preferable. This method of extracting membership grades of the inputs works well with static and slowly varying signal samples. For time-varying cases including fast fading, a better option is the use of Time Delay Fully Recurrent Neural Network (TDFRNN) blocks in place of the MLP. The primary motivation is that the RNN has the capacity to deal with time varying inputs due to the presence of atleast one feed-backward loop for which learning accumulates over an extended period of time.

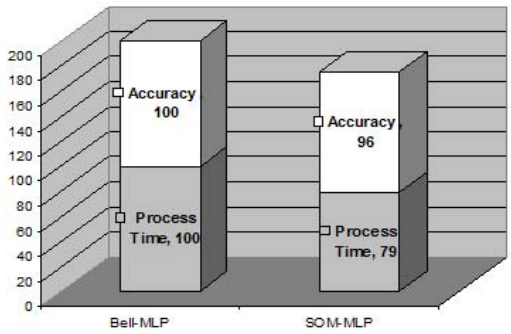


Figure 5.7: Comparative % performance of membership grade generation methods

Table 5.3: Truncated data set of four different delays and four different frequency selective paths of a Rayleigh faded channel

Case	1	2	3	4
Delays (s)	1 x 10^{-5}	1.5 x 10^{-5}	2 x 10^{-5}	2.5 x 10^{-5}
Path gain 1	0.467	0.353	0.555	0.036
Path gain 2	0.281	0.031	0.815	0.732
Path gain 3	0.367	0.893	0.462	0.381
Path gain 4	0.212	0.691	0.021	0.811

Moreover, the RNN is faster compared to the MLP, hence takes less process time, in this case it shows atleast 23% improvement. Therefore the functional membership grade block is constituted by a SOM and a TDFRNN block. The TDFRNN blocks uses split activation and are trained with Decoupled Extended Kalman Filter (DEKF) algorithm. Segregated real and imaginary inputs are applied to different TRFRNN blocks with the output going to respective sections as shown in Figure 5.2.

5.3.2 Performance of the FMLP and FTDFRNN approaches of fuzzy inference

The samples are accumulated for a 4×4 MIMO - OFDM set-up for three different AWGN values viz.$-$3dB, 1dB and 3dB for the Gaussian, Rayleigh and Rician faded channels. The testing includes a range of signal conditions with SNR values ranging from -10 to 10 dB. The testing carried out with inputs from the receiver side calculates channel coefficients and compares them to the theoretically generated values for a frequency range of 0.2Ghz to 8GHz. Some of the values taken for training the system to deal with Rayleigh and Rician channels are generated using Clarke-Gans model. Two such sets are shown in Tables 5.3 and 5.4.

Table 5.4: Truncated data set used for training to carry out estimation under Rician fading

Case	1	2	3	4
Delays	1 x 10^{-5}	1.5 x 10^{-5}	2 x 10^{-5}	2.5 x 10^{-5}
Path gain 1	0.136	0.446	0.125	0.162
Path gain 2	0.087	0.306	0.595	0.433
Path gain 3	0.761	0.239	0.064	0.213
Path gain 4	0.572	0.196	0.351	0.087

Table 5.5: Parameters used for generating the OFDM signal

Sl Num	Parameter	Specification
1	Baseband modulation	16-QAM, BPSK
2	FFT length	512
3	Number of carriers	128
4	Cyclic Prefix	16

These channel coefficients are combined with OFDM symbols generated using the parameters shown in Table 5.5. The MIMO channel matrix (MCM) is formed using a representation given as

$$H_{i,j}(z) = \sum_{i,j} \{a_{i,j} z^{-j}\} \tag{5.18}$$

and the co-channel interference (CCI) responses are expressed as

$$C_{m,n}(z) = \sum_{m,n} \{c_{m,n} z^{-n}\} \tag{5.19}$$

so that the complete signal content at the receiver is expressed as

$$Y_R(z) = X_{i,j}[\sum_{i,j} \{a_{i,j} z^{-j}\} + \sum_{m,n} \{c_{m,n} z^{-n}\}] + N \tag{5.20}$$

where X is the OFDM matrix generated using the parameters given in Table 5.5. Before applying these symbols to the fuzzy system, they are all fuzzified. The FMLP is formed by three layers, with the input layer having a length equal to the symbol length, the hidden layer is 1.5 times to that of the input samples while the output layer has a length of twenty. Combinations of *tan-sigmoid* and *log-sigmoid* functions are used as activation function and a back-propagation algorithm with adaptive learning rate is used for training the set-up. Also Levenberg-Marquardt (LM) optimization is used to speed-up the process of training.

In case of the FTDFRNN structures, split-activation technique is adopted to train real and imaginary sections separately with specific signal segments. The FTDFRNN struc-

Table 5.6: Training parameters of FMLP and FTDFRNN with respect to TDFRNN-MNSOM

Case	Item	TDFRNN-MNSOM	FMLP	FTDFRNN
1	MSE Goal	10^{-3}	10^{-3}	10^{-3}
2	Epochs required	15	15	13
3	Data size	500-1000	500-1000	500-1000
4	Training method	DEKF	Back Propagation with LM optimization, with fuzzification	DEKF with fuzzy consideration
5	Precision in %	97.4	98.4	98.5

tures are trained with DEKF algorithm. A training window of n seconds is given to the two FTDFRNN structures during which several estimated of the signal samples are generated. The SOM placed at the end of the two FTDFRNN structures are used to combine the outputs and provide an optimized estimate of the response. From the estimates of the signal the channels are derived. Training performance is judged using mean square error (MSE) convergence rate and the precision derived. A comparative depictive of the training performance of the FMLP and the FTDFRNN structures with respect to conventional TDFRNN architectures is provided in Table 5.6. The fuzzy based methods clearly show an advantage.

5.3.3 Performance of fuzzy-based MIMO channel modeling

Figure 5.8 presents the normalized processing time of the two fuzzy methods against LS, MMSE, MLP, temporal-MLP, 3L-FF and CTDFRNN-MNSOM techniques. The fuzzy approaches generate minimum $2-6$ % difference in processing time while providing better precision performance consistently over atleast fifteen trails. The MCM has four numbers of channels with twenty taps of which four are significant for the estimation process. Figure 5.9 shows the average BER vs SNR provides by FMLP and FTDFRNN systems compared to an estimator with perfect CSI. The fuzzy systems, on an average differ by about $3-8\%$ with the BER values generated by the estimator with perfect CSI. This is atleast 5% improvement in accuracy compared to the CTDFRNN-MNSOM based estimation of which the BER values are shown as well. This is in addition to the processing time advantage that the fuzzy systems provide. The average pathgains of four faded channels generated by FN systems compared to those provided by the CTDFRNN-MNSOM based estimation (Section 4.3.3, Figure 4.12). Like the CTDFRNN-MNSOM based estimation, the FN-estimators closely approximate the actual channel path but at

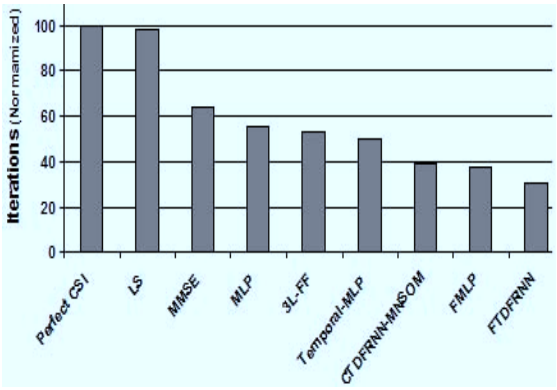


Figure 5.8: Relative time taken by estimation process carried out with (i) statistical methods (LS, MMSE), (ii) ANN based standard methods (Section 3.2), 3L-FF [10], Temporal-MLP (Section 3.4), (iii) RNN based approaches (CTDFRNN-MNSOM) (Section 4.3.3) and (iv) the proposed architecture (FTDFRNN)

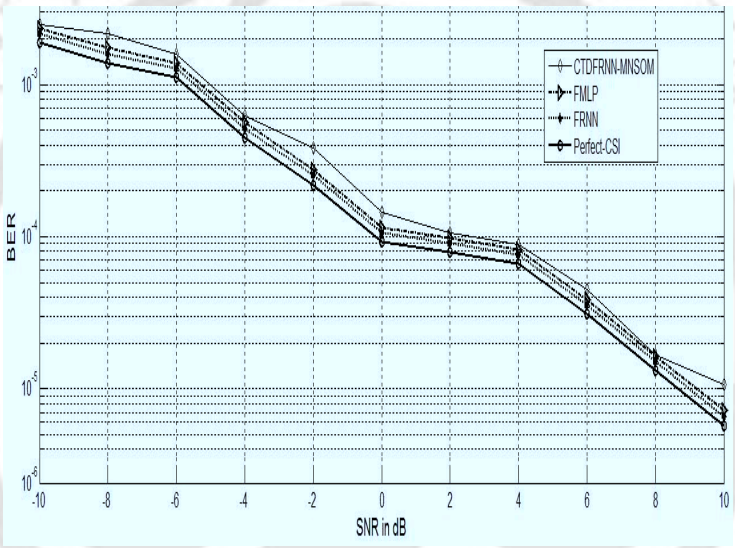


Figure 5.9: BER generated by FMLP and FTDFRNN in comparison to an estimator with perfect CSI

less processing time.

The performance of the system is dependent on the inference rule set adopted for carnying out the estimation process. A standard set of six rules gives optimal performance but experiments are carried out to see the effect of variation of the inference stage. Table 5.7 summarizes the results obtained. The implementation of the inference engine using six set of rule shows a dependence on the network structure adopted for the purpose. Table 5.8 summarizes the variation in performance of the FNS with change in the network structure adopted for the implementation of the inference engine. The inference network structure is considered to be formed by an input, two hidden and one output layer. Four different FNS structures are used to implement the inference rules. While all the four networks using six inference rules generate a MSE convergence between 0.24×10^{-5} and

Table 5.7: Effect of variation of number of inference rules

No. of inference rules	MSE attained $\times 10^{-5}$	Precision in %	Epochs
4	0.5	93	34
6	0.28	97	31
8	0.15	97	43
9	0.09	97	58
10	0.08	97	75
12	0.03	98	91

Table 5.8: Effect in performance due to variation in network structure adopted for implementation of inference engine

Case	Network Size	No. of rules	MSE $\times 10^{-5}$	Epochs	Precision in %
1	20-24-12-4 ($N1$)	6 9	0.3 0.06	34 102	96.5 97.3
2	20-30-15-4 ($N2$)	6 9	0.26 0.045	34 96	96.8 97.0
3	20-40-20-4 ($N3$)	6 9	0.33 0.05	39 101	96.0 97.0
4	20-50-25-4 ($N4$)	6 9	0.24 0.038	44 101	96.0 97.0
5	20-50-30-4 ($N5$)	6 9	0.25 0.039	47 103	95.4 96.1

0.33×10^{-5}, the value comes down significantly to a range of 0.038×10^{-5} to 0.05×10^{-5} with nine inference rules. It indicates that the MSE convergence rate improve and falls to lower limits with more inference stages. With more inference rules, the networks learn better and approach the level of optimality with greater closeness. It amounts to an improvement between 84 to 87% in MSE convergence rates. But this happens at the cost of greater processing time. Number of epochs increases by over 1.5 times when nine inference rules are used compared to case when the system is designed with a set of six such sets. But this increase in processing time and lowering of the MSE values result in an improvement of precision marginally between 1 to 1.5%. This marginal improvement is further compounded by increase in design complexity as shown by the network size (Table 5.8). Hence, the six inference rule format is adopted to carry out the FNS based MIMO channel estimation. The four networks are trained separately and the MSE convergence plots are shown in Figure 5.10. The network named $N4$ with a structure of $20 - 50 - 25 - 4$ gives the best performance. MSE value converges, on an average, to about 0.24×10^{-5} during the stipulated training slots below 50 epochs with

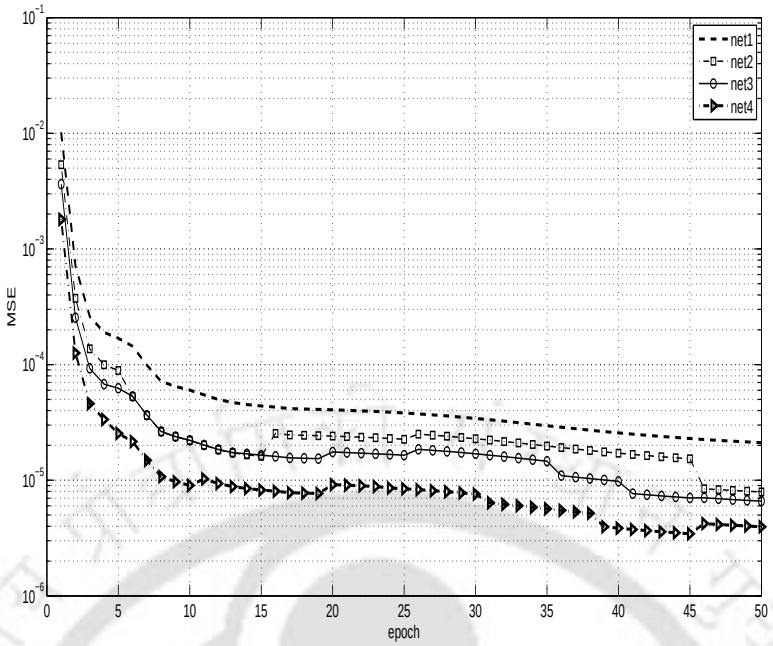


Figure 5.10: MSE plot of FNS estimator training

the data set taken. This network has an input layer of 20 FRNs in the input layer. This value is derived from the fact that the signal block is formed by 8-data bits and three parity prefixes while a channel length of ten. After convolution between the signal block and the channel coefficients, the composite sample input has a sequence length of twenty. Similarly, there are 50 and 25 FRNs in the two hidden layers. The output layer has four FRNs as it needs to retain only four sets of data for a 4×4 MIMO set-up designed for the purpose. The significant part of the data set are retained, interpretation derived after de-fuzzification and BER values calculated. The above results are generated for the two sets of inference rules considered. The two configurations are tested for a set representing VOIP based broadcast respectively. The first set used in the VOIP transmission involve 128-bit size OFDM data blocks of around fifty while the second broadcast involves over a thousand 6400-bit transmissions. These variations of sample sizes are taken to ascertain the effect of data on training and performance of the system. The implementation of the inference engine using six set of rule shows a dependence on the network structure adopted for the purpose. Table 5.8 summarizes the variation in performance of the FNS with change in the network structure adopted for the implementation of the inference engine. The inference network structure is considered to be formed by an input, two hidden and one output layer. Four different FNS structures are used to implement the inference rules. While all the four networks using six inference rules generate a MSE convergence between 0.24×10^{-5} and 0.33×10^{-5}, the value comes down significantly to a

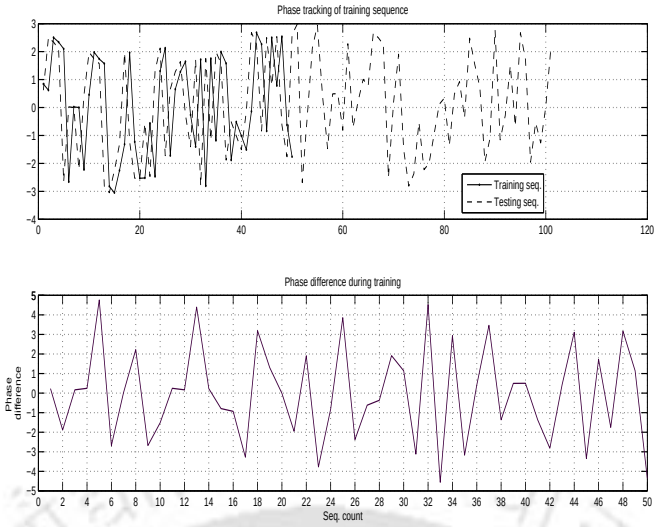


Figure 5.11: Phase tracking by FNS based MIMO estimator during training

range of 0.03×10^{-5} to 0.05×10^{-5} with twelve inference rules. It indicates that the MSE convergence rate falls with more inference stages. With more inference rules, the networks learn better and approach the level of optimality with greater closeness. It amounts to an improvement between 84 to 87% in MSE convergence rates. But this happens at the cost of greater processing time. Number of epochs increases by about 1.32 to 1.7 times when twelve inference rules are used compared to case when the system is designed with a set of six such sets. But this increase in processing time and lowering of the MSE values result in an improvement of precision marginally between 1.04 to 1.5%. This marginal improvement is further compounded by increase in design complexity. Hence, the six inference rule format is adopted to carry out the FNS based MIMO channel estimation. The four networks are trained separately and the MSE convergence plots are shown in Figure 5.10. The network named N4 with a structure of $20 - 50 - 25 - 4$ gives the best performance. MSE value converges, on an average to about 0.24×10^{-5} during the stipulated training slots below 50 epochs with the data set taken. This network has an input layer of 20 FRNs in the input layer. Similarly, there are 50 and 25 FRNs in the two hidden layers. The output layer has four FRNs as it needs to retain only four sets of data for a 4×4 MIMO set-up designed for the purpose. The significant part of the data set are retained, interpretation derived after de-fuzzification and BER values calculated.

A worst case result of phase tracking generated with VOIP based wireless transmission in a Rayleigh faded channel with a Doppler shift of 100 Hz is shown in Figure 5.11. Though the result shows a phase difference of around ± 4.5 deg., it improves with higher SNR. The frequency responses of the two channel types is depicted in Figure 5.12. The frequency response shows clearly the importance of precision.

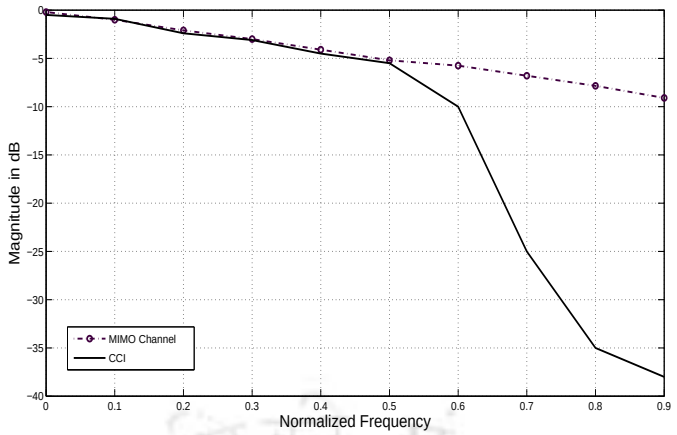


Figure 5.12: Frequency response of the MIMO channel and the CCI against a normalized frequency band

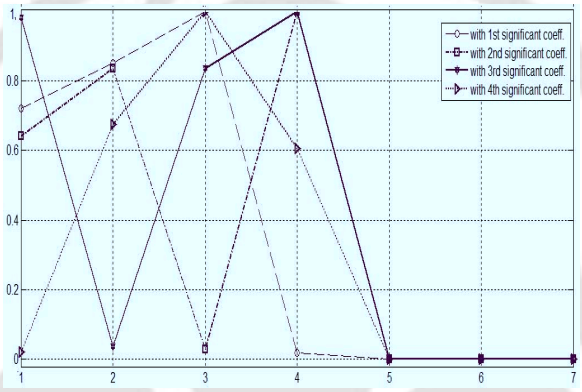


Figure 5.13: CCI plot generated by channel coefficients with estimated path gains

A co-channel interference pattern of the channels is also generated and is depicted in Figure 5.13. The plot shows the normalized correlation of the CCI on significant estimated channel coefficients. It indicates though the significant channel coefficients are estimated with adequate path gains, yet significant interference exists. If the CCI is higher, the frequency response suffers indicating a fall in spectral efficiency. The FNS provides better precision in making discrimination between CCI and significant channel coefficients (Figure 5.13), hence demonstrates greater spectral efficiency.

A set of experiments are also perform to see the effectiveness of the fuzzy MIMO estimator in NFS and FNS forms. The two configurations are tested for a VOIP based broadcast using a 4×4 MIMO wireless set-up infested with Rayleigh fading. The first transmissions involve 128-bit size OFDM data blocks of around fifty while the second broadcast involves over a thousand 6400-bit transmissions. These data blocks are used to train the NFS and FNS MIMO estimators. The design, experimental and results derived from the NFS based MIMO channel modeling is covered in Section 5.4.

Table 5.9: Computational complexity of FNS based design compared to temporal-MLP and RNN-based architectures with N length filter (for temporal MLP structures) or delay blocks (for RNN-based structures), n number of parallel structures, l size competitive layer and $(P + R)$ length of the signal

Model	Number of	
	additions	multiplications
IIR-MLP	$7N + 2$	$10N$
CTDFRNN-MNSOM	$(P + R)34+2+l$	$(P + R)24+2l$
FTDFRNN	$12N+2+l$	$11N+2+l$

The fuzzy approaches generate minimum 2 to 6 % difference in processing time while providing better precision performance consistently over atleast fifteen trails. Similarly there is enhancement of BER performance as well over the entire SNR spectrum for all the multipath fading channels considered. The fuzzy-based methods clearly provide advantages of faster processing time, lower BERs and better precision while carrying out symbol recovery. In this performance enhancement, the membership generation and the inference mechanism plays a decisive role. The choice of the specific membership norms and inference rule set is critical in resolving the speed-precision stand-off. Table 5.9 shows the computational complexity associated with the FNS based MIMO channel modeling approach. Considering l to be the length of a competitive layer, n to be the number of parallel structures and $(P + R)$ to be the length of the signal, the computational complexity of the RNN-based architectures are shown in this table. Here, IIR-MLP is taken as the representative of the temporal-MLP family as it provides better results. Similarly, CTDFRNN-MNSOM is taken as the comparative model since it generates the best performance in terms of BER and precision among the proposed RNN-based architectures. The FNS approach has a RNN-based structure followed by a competitive layer which for each input generates sizable amount of computation which is fast and converges to the desired level within a few number of training epochs. The increase in performance is in addition to the fall in computational complexity compared to CTDFRNN-MNSOM as outlined in Section 5.3.3. Figure 5.14 shows a comparative plot generated by IIR-MLP, CTDFRNN-MNSOM and RNN-FNS approaches in a severely faded channel for a VOIP transmission. The FNS based method of MIMO channel modeling shows a clear advantage.

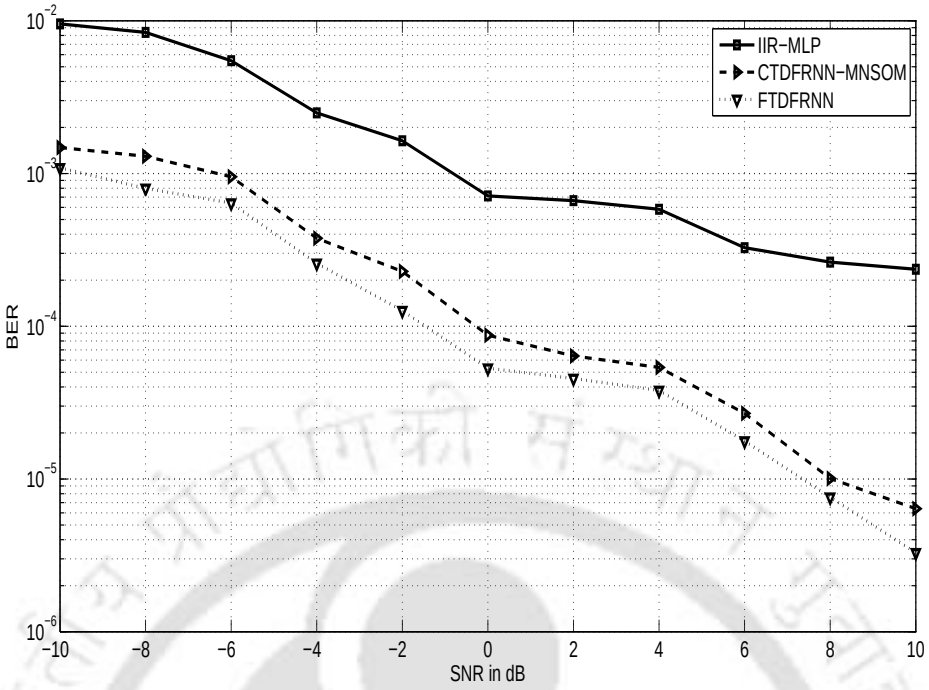


Figure 5.14: Comparative BER plots generated by IIR-MLP, CTDFRNN-MNSOM and FTDFRNN approaches in a severely faded channel

5.4 FTDFRNN based NFS for MIMO Channel Modeling

[112] In order to show the effectiveness of our proposed technique in Section 5.2, we are providing the basic mathematical treatments of an NFS in this section so as to prove the effectiveness of our proposed technique through the comparison with NFS. The NFS based method is implemented in an Adaptive Neuro-Fuzzy Inference System (ANFIS) of which the fuzzy rules can be either based on the Takagi-Sugeno or the Mamdani model. The Mamdani model is used in this case as it is found to be better suited for practical applications [36]. The inference rules are derived using the Mamdani model [112]. If $\overline{y_j}$ is the center of gravity (CG) of the fuzzy set B^j, then the total output y of the fuzzy system is given by the following equation:

$$y = f(x) = \frac{\sum_{j=1}^{M} w_j \overline{y_j}}{\sum_{j=1}^{M} w_j} = \frac{g}{h} \tag{5.21}$$

with an input $X = [x_1, x_2, \cdots, x_N]^T$, where,

$$w_j = \prod_{i=1}^{N} \mu_{A_i^j}(x_i) \tag{5.22}$$

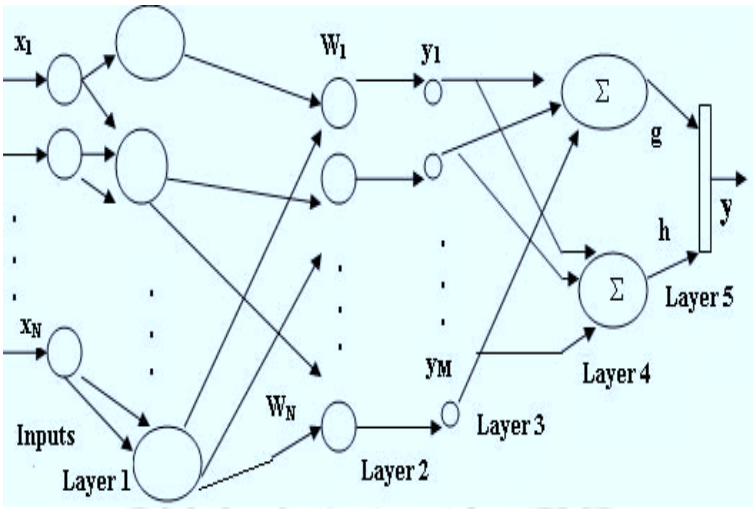


Figure 5.15: ANFIS block for the Mamdani model used for the MIMO modeling

if the logic multiplication rule is used, and

$$w_j = \min\{\mu_{A_1^j}(x_1), \mu_{A_2^j}(x_2), \cdots, \mu_{A_N^j}(x_N),\} \tag{5.23}$$

if the minimum rule is used. The form of the membership function of A_i^j is Gaussian:

$$\mu_{A_i^j}(x_i) = \alpha_i^j \exp[-(\frac{x_i - m_i^j}{\sigma_i^j})^2] \tag{5.24}$$

where α_i^j is the amplitude, m_i^j is the mean and σ_i^j is the standard deviation and defines the function shape.

A multiplier forward feedback type ANN of five layers formed by FRN blocks, as shown in Figure 5.15 is used. The first layer receives different fuzzy inputs. The second layer facilitates the implementation of the system's rules and ensures adaptive behaviour. The triggering ability of the network is ensured by the third layer. The inference process is carried out in the fourth layer and the defuzzification process takes place in the fifth layer. The ANN has to adjust the parameters $\overline{y_j}, m_i^j$ and σ_i^j so that the square error is minimized:

$$e(t) = \frac{1}{2}[f(x^p) - y_d]^2 \tag{5.25}$$

where p is the input pattern indicator x^p and y_d is the desired output. The training (updating) of $\overline{y_j}$ is performed using following rule:

$$\overline{y_j}(t+1) = \overline{y_j}(t) - \gamma \frac{\partial e(t)}{\partial y_j}. \tag{5.26}$$

Table 5.10: NFS and FNS estimator performance comparison

Case	Parameter	NFS	FNS
1^{st} set,	Network Size	20-50-59-61-25-4	20-50-25-4
50 no.s of	Precision (%)	95.2	96.1
128-	MSE attained	0.8×10^{-5}	0.24×10^{-5}
bit	Inference rules	6 to 10	6
blocks	Epochs	42	31
	% Processing time	68	61
2^{nd}	Network Size	20-50-59-61-25-4	20-50-25-4
set,	Precision (%)	95.1	96
100	MSE attained	0.88×10^{-5}	$0.26x10^{-5}$
no.s of	Inference rules	6 to 10	6
6400-bit	Epochs	1534	1338
blocks	% Processing time	73	65

Updating of m_i^j is done by using the learning rule:

$$m_i^j(t+1) = m_i^j(t) - \gamma \frac{\partial e(t)}{\partial m_i^j} \tag{5.27}$$

for $i = 1, 2, \cdots, M$ and $j = 1, 2, \cdots, M$. Likewise, the learning rule for σ_i^j, can be given as

$$\sigma_i^j(t+1) = \sigma_i^j(t) - \gamma \frac{\partial e(t)}{\partial \sigma_i^j} = \sigma_i^j(t) - \gamma(f - y_d) \frac{\partial f}{\partial w_j} \frac{\partial w_j}{\partial \sigma_i^j} \tag{5.28}$$

where

$$\frac{\partial f}{\partial w_j} = \frac{\overline{y_j} - f}{h} \tag{5.29}$$

and

$$\frac{\partial w_j}{\partial \sigma_i^j} = 2w_j \frac{(x_i^p - m_i^j(t))^2}{(\partial_i^j(t))^3}. \tag{5.30}$$

The processing blocks in the ANFIS block are formed by FRN units which can tackle time-varying input with fuzzified forms of patterns and targets. The training of the ANFIS system is carried out in two phases: first a forward pass and then a backward cycle during which the weights and other network parameters are updated. A set of experiments are also perform to see the effectiveness of the fuzzy MIMO estimator in NFS and FNS forms. The two configurations are tested for a VOIP based broadcast using a 4×4 MIMO wireless set-up infested with Rayleigh fading. The VOIP transmissions involve 128-bit size OFDM data blocks of around fifty while the another set of broadcast involves over a thousand 6400-bit transmissions. These data blocks are used to train the NFS and FNS MIMO estimators. The results derived are summarized in Table 5.10. Figure 5.16 shows a MSE convergence plot of six situations of NFS-estimators trained with highly

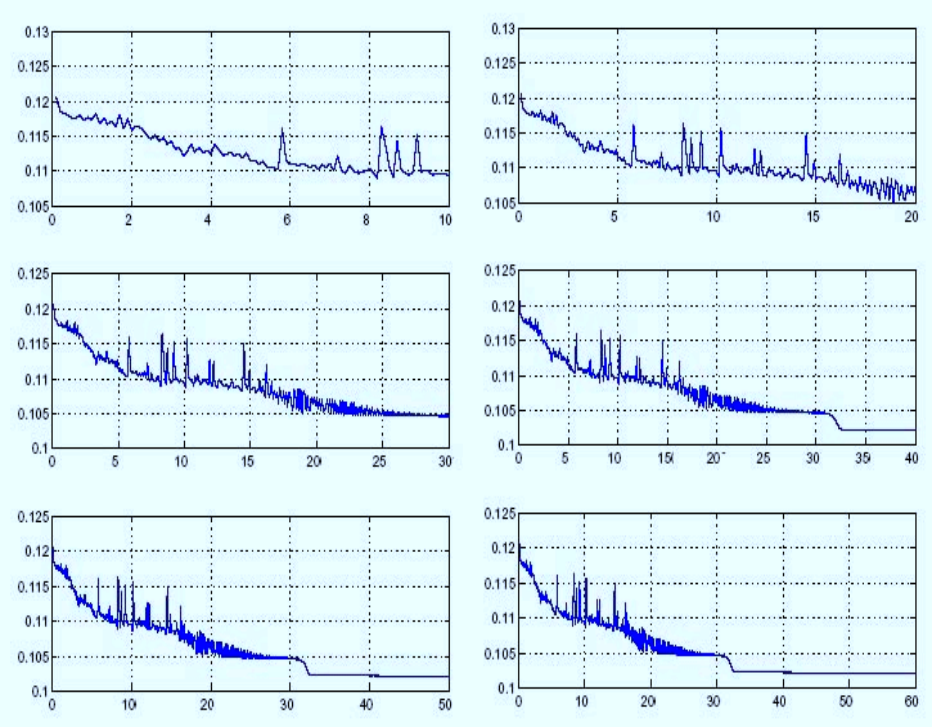


Figure 5.16: MSE plot of NFS estimator training

correlated data samples. The epochs fixed are within sixty which generates a MSE convergence of about 0.105×10^{-3}. This is considerably higher than the corresponding FNS system which reaches 10^{-5} limit. The MSE convergence of the NFS model improves with noise corrupted ($\simeq -3dB$) data but precision suffers when the same block is tested with samples filled with severe disturbance ($\simeq -10dB$). The FNS, however, shows better precision in a identical situations. The results clearly indicate that the FNS design is better and is more preferable for adaptive receiver designs.

5.5 Conclusion

In this chapter, we have shown that fuzzy based composite architectures provide better processing speed and precision while modeling time-varying MIMO channels. This improvement is due to its ability to track microscopic variation in the given macroscopic space which is useful for wireless and mobile communication applications. Fuzzy systems in two forms, namely FNS and NFS, have been explored. A careful examination of these two available hybrid fuzzy systems shows that the FNS is more suitable for real world situations. There are certain issues which determine such improved performance. The first factor is membership generation. The membership generation process is dependent on the method adopted for implementation. We found that a SOM-MLP based membership generation method is a compromise between processing speed and precision generated.

The second issue is the inference engine. A six rule inference block formulated using a 20-30-15-4 network structure of FRNs give the best performance in terms of precision and processing speed. Experimental results show that the FNS approach is more suitable than the NFS method for MIMO channel modeling. The FNS is atleast 9% faster, needs around 23% less training sessions and shows 1.2% superior precision in symbol recovery than the NFS. Using these consideration the FNS based system give a set of results which show a minimum of 2 to 6 % difference in processing time compared to statistical and ANN based approaches. The FNS also provides better precision performance consistently by about 2.7% compared to any average RNN based architecture while dealing with VOIP based broadcast using a 4×4 MIMO wireless set-up. It should also be noted that the proposed FTDFRNN system yields around 46% lesser computational complexity, on an average, than the corresponding CTDFRNN-MNSOM structure as shown in Table 5.9. Moreover, as depicted in Figure 5.14, among the proposed techniques of this thesis, namely, IIR-MLP, CTDFRNN-MNSOM and FTDFRNN, the last technique shows its superiority while modeling even a deeply faded MIMO channel in important practical applications like VOIP based transmissions. Therefore, it can be concluded that the proposed FTDFRNN technique is better suited than the IIR-MLP or CTDFRNN-MNSOM in this framework of high data rate MIMO communication.

Chapter 6

Conclusion

6.1 Summary of the Present Work

A common form of uncertainty and stochastic behaviour is observed in time-varying Multi Input Multi Output (MIMO) wireless communication due to interference and correlation among channel coefficients. Soft-computational techniques can be added to the list of methods already available for modeling MIMO channels to provide innovative solutions primarily due to the fact that these tools are better placed to use channel side information (CSI) for improved performance. In this thesis, we have made an attempt to model MIMO channels with a class of soft-computational tools based on ANN in feedforward form, known as MLP, and feedback form, called RNN, as well as fuzzy based composite systems. Such an aspect of modeling MIMO channels has so far not been carried out and reported. Thus the ideas presented in this thesis are, to the best of our knowledge, new, original and not built-up around any pre-existing schemes. In the following, we provide a chapter wise break-up of the main contributions of this thesis.

In Chapter 3, a few MLP architectures are designed, trained, validated and tested to model static and slowly varying multipath fading channels with and without Line of Sight (LOS) components. The work demonstrates higher success rates in channel estimation and symbol recovery achieving lower BER than LS and MMSE estimators. But the MLPs prove to be suitable for static and slowly varying cases and show limited capability to capture fastly time-varying patterns of the input data. This limitation has been removed considerably by incorporating temporal characteristics into the basic MLP structure enabling it to model time-varying MIMO channels. We have proposed a technique to build short-term memory into the MLP to make the network dynamic. In doing so, we consider both FIR and IIR structures for better learning. In particular, we deploy FIR and IIR sections in the input, output and hidden layers of the MLP which enables it to acquire dynamic characteristics. These facilitate the creation of six sub classes each

of FIR-MLP and IIR-MLP. The use of these short-term memory blocks makes the MLP time dependent and converts it a non-linear dynamical system. The static MLP network provides the non-linear characteristics while the memory accounts for time-dependent behavior. The success rate achieved are higher than conventional MLP based system but the drawback observed is higher design complexity and intricacies associated with the configuration.

In Chapter 4, a solution to the computationally expensive temporal-MLP method of modeling MIMO channels is proposed. We deal with the RNN based approaches which is a natural alternative choice to the temporal-MLP due to the fact that these soft-computational tools have the in-built capability to deal with time-varying signals. It stems from the fact that RNN has at least one feedback loop despite generally being a feedforward structure. In this chapter, we proposed a class of novel RNN architectures configured with certain basic training algorithms, split input and activation for speedup considerations and a SOM based optimization. In particular, several RNN frameworks supporting expendability of block level CTDFRNN units with responses trimmed by SOM optimization, multiple CTDFRNN blocks trained in a cooperative environment with improved learning and the ability to tackle inputs with time-variation have been proposed. An attempt has also been to show that CTDFRNN clusters optimized with SOM with diversity gains is found to be useful for low signal to noise ratio (SNR) environments. Further, the Modular Network SOM (MNSOM), which is considered to closely resemble biological computation with an inherent reinforced modular learning, has also been proposed and formulated using CTDFRNN blocks for application in MIMO channel modeling. Experimental results show that CTDFRNN-MNSOM emerges out to be a viable alternative to the conventional stochastic estimation methods with an average 60% saving of processing time. The CTDFRNN-MNSOM has less time-complexity than either FIR-MLP or IIR-MLP. However, in some applications such as short length wireless VOIP based transmission while modeling MIMO channels, the precision plays a crucial role. In certain severely faded channels configured for VOIP based transmission, for an average increase of 0.9% in precision of performance provided by CTDFRNN-MNSOM compared to CTDFRNN-C, the BER values show a fall of 50-66%. It also suggests that in such severely faded channels with certain application, CTDFRNN-MNSOM may not provide the required precision level within a certain limit of computational complexity. In such cases, a simpler RNN structure along with a decision driven tool such as fuzzy system may provide better design solution.

Chapter 5 deals with a hybrid solution where the performance of RNN-based methods needs to be improved further especially in connection with the accuracy of the recovered content. For a deeply faded MIMO channel modeling, fuzzy based systems integrated with RNN block prove to be a viable option to attain the desired precision performance.

Further, fuzzy systems possess an integral ability to deal with uncertainty, can discern minute variations, attach probabilistic linkages to such finer changes and facilitate the formation of a mechanism for near error-free decision making using ANN. Also, the uniqueness associated with these soft-computational tools enables a smooth blending of fuzzy system with ANN in a supplementary-complementary arrangement which leads to enhancement in performance in terms of processing time, lower computational complexity, increase in precision, better adaptability and an unmatched capability to track and model uncertain behaviour of stochastic MIMO channels. In this chapter, in particular, we have proposed the formulation of certain fuzzy-based systems using TDFRNN with the ability to deal with time-varying inputs in fuzzified form enabling a better modeling of time-varying MIMO channels. The fuzzified TDFRNN blocks are designed to generate a set of input conditioning norms and inference logic which captures finer variations in the signals transmitted by the MIMO set-up. The results show significant improvement in processing time and accuracy as compared to ANN based approaches. The fuzzy systems, on an average, provide atleast 5% improvement in accuracy as compared to the RNN-based estimation which has already been established to be a better alternative to statistical and ANN based approaches as stated in Chapter 4. This is in addition to the processing time advantage that the fuzzy systems provide.

6.2 Scope of Future Work

In this thesis we have explored the effectiveness of modeling a MIMO channel using ANN and its variants like RNN and fuzzy based composite systems. It is, however, felt that there may exist considerable scope for extending the work reported here along various directions for future investigations. Some of these are briefly outlined below:

1. The work included in Chapter 3 considers the results derived without taking into account the performance difference that maybe observed by using source and channel coding. Though these will provide certain performance improvements, but will also add to system complexity. Another aspect that maybe considered includes a cluster arrangement for FIR-MLP and IIR-MLP configurations which will definitely provide certain diversity gains. Coding and diversity gain together will enable the proposed temporal-MLP architectures to model the MIMO channel better. However, the associated time complexity will determine which structure would emerge as the optimal one.

2. The RNN-based architectures formulated in Chapter 4 suffers from a drawback that certain time is lost in training the networks (this is true for all our proposals in general). This loss can be prevented by carrying out a sequence by sequence learning

with training reference derived from a Kalman filter predicted future sample from the received signal. It will make the systems faster and when combined with source and channel coding, might enable better modeling of the MIMO channels. Here too, the computational complexity and the faster learning rate of the proposed modules will determine the optimal solution.

3. The work included in Chapter 5 can be extended further to incorporate GA and Particle Swarm Optimization (PSO) assisted FNS-based MIMO estimators trained with evolutionary approaches. The close resemblance to biological computations should be the key factor in such proposals. Such a system can further be implemented as a System-on-Chip (SOC) package and made a part of adaptive receivers for upcoming MIMO wireless networks.

References

[1] S. Colieri, M. Ergen, A. Puri and A. Bahai: "A Study of Channel Estimation in OFDM Systems", in Proceedings of *IEEE* 56^{th} *Vehicular Technology Conference*, vol. 2, pp.894-898, Vancouver, Canada, Sept., 2002.

[2] H. Gacanin, S. Takaoka and F. Adachi: "A Study of Channel Estimation in OFDM Systems- Pilot-assisted Channel Estimation for OFDM/TDM with Frequency-domain Equalization", in Proceedings of 62^{nd} *Vehicular Technology Conference*, pp. 554-558, Dallas, USA, Sept., 2005.

[3] M. Jiang and L. Hanzo: "Multiuser MIMO-OFDM for Next-Generation Wireless Systems", *Proceedings of the IEEE*, vol. 95, no. 7, pp. 1430-1469, Jul., 2007.

[4] H. Ozcelik, N. Czink and E. Bonek: "What Makes a Good MIMO Channel Model?", in Proceedings of 61^{st} *IEEE Vehicular Technology Conference,* pp. 1-5, Stockholm, Sweden, 2005.

[5] W. Wang, Z. Zhengdao, and G. B. Giannakis: "Wireless Multicarrier Communications", *IEEE Signal Processing Magazine*, pp. 29-48, May, 2000.

[6] L. Hanzo, L. L. Yang, E. L. Kuan, and K. Yen: "Single-and Multi-Carrier DS-CDMA: Multi-User Detection, Space-Time Spreading, Synchronisation and Standards", IEEE Press/Wiley, Piscataway, NJ, 2003.

[7] L. Hanzo, B. J. Choi, and M. Munster: "A Stroll along Multi-Carrier Boulevard Towards Next-Generation Plaza - Space-Time Coded Adaptive OFDM and MC-CDMA comparison ", *IEEE Veh. Technol. Soc. Newslett,* vol. 51, pp. 10-19, Nov., 2004.

[8] T. M. Duman and A. Ghrayeb: *Coding for MIMO Communication Systems*, John Wiley and Sons Ltd, England, 2007.

[9] C. Oestges and B. Clerck: *MIMO Wireless Communications- From Real-World Propagation to Space-Time Code Design*, Academic Press, Elsevier Ltd., 1^{st}*ed.*, London, 2007.

[10] Z. Ling and Z. Xianda: "MIMO Channel Estimation and Equalization Using Three-Layer Neural Networks with Feedback", *Tsinghua Science and Technology Journal*, vol. 12, no. 6, pp. 658-661, Dec., 2007.

[11] J. Muhammad: "Artificial Neural Networks for Location Estimation and Co-Channel Interference Suppression in Cellular Networks", *Master of Philosophy Thesis submitted to University of Stirling,* Feb., 2007.

[12] Y. Fang and T. W. S. Chow: "Blind Equalization of a Noisy Channel by Linear Neural Network," in Proceedings of *IEEE Transaction on Neural Networks,* vol. 10, no. 4, pp. 918-924, Jul., 1999.

[13] A. Naveed, I. M. Qureshi, T. A. Cheema and A. Jalil: "Blind Equalization and Estimation of Channel using Artificial Neural Networks," in Proceedings of *INMIC,* pp. 184-190, Lahore, Pakistan, 2004.

[14] H. Zamiri-Jafarian and G. Gulak: "Iterative MIMO Channel SVD Estimation," in Proceedings of *IEEE International Conference on Communications,* pp. 1-5, Seoul, Korea, 2005.

[15] H. Zamiri-Jafarian and G. Gulak: "Adaptive Channel SVD Estimation for MIMO-OFDM Systems," in Proceedings of 61^{st} *IEEE Semiannual Vehicular Technology Conference,* pp. 240-245, Stockholm, Sweden, 2005.

[16] W. Ping, L. Lhua and Z. Ping: "Subchannel Interference Cancelation in SVD-based MlMO System", *Journal of China Universities of Posts and Telecommunications*, vol. 14, issue 3, Sept., 2007

[17] M. Ibnkahla: "Neural Network Modeling and Identification of Nonlinear MIMO Channels", in Proceedings of 9^{th} *International Symposium on Signal Processing and Its Applications, (ISSPA 2007)*, pp. 1-4, UAE, 2007.

[18] C. Hua and Z. Xiao-Hui: "MIMO-OFDM Channel Estimation Based on Neural Network", in Proceedings of *International Conf. Wireless Communications Networking and Mobile Computing (WiCOM)*, no. 6, pp. 1-4, Chengdu, China, 2010.

[19] Z. Wenle: "MADALINE Neural Network for Parameter Estimation of LTI MIMO Systems", in Proceedings of 29^{th} *Chinese Control Conference*, pp. 1346-1351, Beijing, China, 2010.

[20] J. R. Paul and T. Vladimirova: "Blind Equalization with Recurrent Neural Networks Using Natural Gradient", in Proceedings of *ISCCSP 2008,* pp. 178-183, Malta, Mar., 2008.

[21] C. G. Potter: "Multiple Input Multiple-Output wireless communications with imperfect channel knowledge", *Missouri University of Science and Technology, USA, document available at: http:// proquest.umi.com*, 2008.

[22] C. G. Potter, G. K. Venayagamoorthy and K. Kosbar: "RNN based MIMO Channel Prediction", *Elsevier Journal of Signal Processing*, vol. 90, issue 2, pp. 440-450, Feb., 2010.

[23] W. Mao: "Novel SREKF-based Recurrent Neural Predictor for Narrowband / FM Interference Rejection in GPS", *Elsevier International Journal of Electronics and Communication*, vol. 62, pp. 216-222, 2008.

[24] J. R. Paul and T. Vladimirova: "Blind Equalization with Recurrent Neural Networks Using Natural Gradient", in Proceedings of *ISCCSP 2008*, pp. 178-183, Malta, Mar., 2008.

[25] Y. Li and Y. Deng: "Nonlinear Equalization With Known Channel State Information in Satellite Communication", in Proceedings of *ICCS 2008*, pp 1081 - 1085, Guangzhou, China, 2008.

[26] P. H. G. Coelho: "Adaptive Channel Equalization Using EKF-CRTRL Neural Networks", in Proceedings of *International Joint Conference on Neural Networks*, pp. 1195-1199, Honolulu, Hawaii, 2002.

[27] Gowrishankar and P. S. Satyanarayana: "Recurrent Neural Network based BER Prediction for NLOS Channels", in Proceedings of 4^{th} *International Conference on Mobile Technology, Applications and Systems*, pp. 410-416, Singapore, 2007.

[28] J. Choi, M. Bouchard, T. H. Yeap and O. Kwon: "A Derivative-Free Kalman Filter for Parameter Estimation of Recurrent Neural Networks and its Applications to Nonlinear Channel Equalization", in Proceedings of 4^{th} *International ICSC Symposium on Engineering of Intelligent Systems (EIS 2004)*, pp. 1-7, Madeira, Portugal, 2004.

[29] J. Choi, K. Ko, and I. Hong: "Equalization Techniques using a Simplified Bilinear Recursive Polynomial Perceptron with Decision Feedback", in Proceedings of *International Joint Conference on Neural Networks*, vol. 4, pp. 2883-2888, Washington DC,USA, 2001.

[30] J. Choi, A. C. Lima and S. Haykin: "Kalman Filter-trained Recurrent Neural Equalizers for Time-varying Channels", *IEEE Transactions on Communications*, vol.53, pp. 472-480, Mar., 2005.

[31] J. Choi, M. Bouchard and T. H. Yeap: "Decision Feedback Recurrent Neural Equalization with Fast Convergence Rate", *IEEE Transactions on Neural Networks*, vol.16, pp. 699-708, May 2005.

[32] W. Liu, L. Yang and L. Hanzo: "Recurrent Neural Network Based Narrowband Channel Prediction", in Proceedings of 63^{rd} *IEEE Vehicular Technology Conference (VTC 2006)*, pp. 2173-2177, Melbourne, Australia, 2006.

[33] D. Park: "Equalization for a Wireless ATM Channel with a Recurrent Neural Network Pruned by Genetic Algorithm", in Proceedings of 9^{th} *ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, pp. 670-674, Phuket, Thailand, 2008.

[34] S. Haykin: *Neural Networks A Comprehensive Foundation*, Pearson Education, 2^{nd} ed., 2003

[35] S. Mitra and Y. Hayashi: "Neuro-Fuzzy Rule Generation: Survey in Soft Computing Framework", *IEEE Transactions on Neural Networks*, vol. II, no. 3, pp. 748-768, 2000.

[36] S. Mitra, K. M. Konwar and S. K. Pal: "Fuzzy Decision Tree, Linguistic Rules and Fuzzy Knowledge-Based Network: Generation and Evaluation", *IEEE Transactions on Systems, Man and Cybernatics- Part C: Applications and Reviews*, vol. 32, no. 4, pp. 748-768, 2002.

[37] M. Erman, A. Mohammed and E. Rakus-Andersson: "Fuzzy Logic Applications in Wireless Communications", in Proceedings of *IFSA-EUSFLAT*, pp. 763-767, Lisbon, Portugal, 2009.

[38] L-X Wang and J. M. Mendel: "An RLS Fuzzy Adaptive Filter with Applications to Nonlinear Channel Equalization", in Proceedings of 2^{nd} *IEEE International Conference on Fuzzy Systems*, vol. 2, pp. 895-900, San Francisco, USA, 1993.

[39] P. Sarwal and M. D. Srinath: "A Fuzzy Logic System for Channel Equalization", *IEEE Transactions on Fuzzy Systems*, vol. 3, no. 2, pp. 246-249, 1995.

[40] F. Arnai, P. Coulton and B. Honary: "Real Time Channel Estimation Based on Fuzzy Logic", in Proceedings of *IEEE International symposium on Information Theory*, pp. 289-298, St.-Petersburg, Russia, 1995.

[41] S. Ghosh, Q. Razouqi, H. Jerry Schumacher and A. Celmins: "A Survey of Recent Advances in Fuzzy Logic in Telecommunications Networks and New Challenges", *IEEE Transactions on Fuzzy Systems*, vol. 6, no. 3, pp. 443-447, 1998.

[42] A. Niemi, J. Joutsensalo and T. Ristaniemi: "Fuzzy Channel Estimation in Multipath Fading CDMA Channel", in Proceedings of 11^{th} *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, vol. 2, pp. 1131-1135, London, UK, 2000.

[43] B. Young-Bae, Y. Takama and K. Hirota: "Combined Channel Estimation and Data Decoding Based on Fuzzy Logic", *IEEE Transactions on Instrumentation and Measurement*, vol. 51, no. 2, pp. 342-346, 2002.

[44] R. H. Abiyev and T. Al-shanableh: "Neuro-Fuzzy Network for Adaptive Channel Equalization", as a chapter of *LNCS Advances in Neural Networks-ISNN 2007*, vol. 4492, pp. 241-250, 2007.

[45] J. Zhang, Z. M. He, X. Wang and Y.Huang: "A TSK fuzzy approach to Channel Estimation for OFDM Systems", *Journal of Electronic Science and Technology in China*, vol. 4, no. 2, 2006.

[46] J. Wen, C. Chang, G. Lee and C. Huang: "OFDM Channel Prediction using Fuzzy Update LMS Algorithm in Time-Variant Mobile Channels", in Proceedings of *IEEE* 64^{th} *Vehicular Technology Conference*, pp.1-5, Montreal, Canada, 2006.

[47] J. Zhang, Z. He, X. Wang and Y. Huang: "TSK Fuzzy approach to Channel Estimation for MIMO-OFDM Systems", *IEEE Signal Processing Letters*, vol. 14, no. 6, 2007.

[48] M. Nuri Seyman and N. Taspinar: "Channel Estimation based on Adaptive Neuro-Fuzzy Inference System in OFDM", *IEICE Trans. Commun.*, vol. E91-B, no. 7, pp. 2426-2430, 2008.

[49] H. Shatila, M. Khedr and J.H. Reed: "Channel Estimation for WiMaX Systems using Fuzzy Logic Cognitive Radio", in Proceedings of *IFIP International Conference on Wireless and Optical Communications Networks (WOCN '09)*, pp. 1 - 6, Cairo, Egypt, 2009.

[50] F. Gharibi, J. R. Jamjah, F. Akhlaghian and B. Z. Azami: "An improved Adaptive Neural Fuzzy Channel Equalizer", in Proceedings of 18^{th} *Iranian Conference on Electrical Engineering (ICEE)*, pp. 326-330, Isfahan, Iran, 2010.

[51] P. K. Sahu, S.K. Patra, S.P. Panigrahi: "Non-linear Channel Equalization using Computationally efficient Neuro-Fuzzy Channel Equalizer," in Proceedings of 18^{th} *Iranian Conference on Electrical Engineering (ICEE)*, pp. 326-330, Isfahan, Iran, 2010.

[52] M. Nuri, Seyman, N. Taspinar: "Channel Estimation Based on Adaptive Neuro-Fuzzy Inference System in OFDM", *IEICE Transactions on Communications*, vol. E91.B, no. 7, pp. 2426-2430, 2010.

[53] T. S. Rappaport: *Wireless Communications: Principles and Practice*, Pearson Education, 2^{nd} ed., 7^{th} Indian Reprint, 2004

[54] J. G. Proakis: *Digital Communications*, McGraw-Hill Publication, 4^{th} ed., 2001

[55] Y. Li, L. J. Cimini and N. R. Sollenberger: "Robust Channel Estimation for OFDM Systems with Rapid Dispersive Fading Channels", *IEEE Transaction on Communication*, vol. 46, no. 7, pp. 902-15, Jul., 1998.

[56] H. Arslan and G. E. Bottomley: "Channel Estimation in Narrowband Wireless Communication Systems", *Wireless Commun. and Mobile Comp.*, vol. 1, no. 2, pp. 201-19, Apr. 2001.

[57] T. Himsoon, S. Weifeng, and K. J. R. Liu: "Single-Block Differential Transmit Scheme for Broadband Wireless MIMO-OFDM Systems", *IEEE Trans. Signal Processing*, vol. 54, no. 9, pp. 3305-14, Sept. 2006.

[58] W. Hashim et al.: "Performance Comparison of Differential Space-Time Signalling Schemes for OFDM Systems", in Proceedings of *IEEE Malaysia International Conference of Communication and Networks*, vol. 2, pp. 1-6, Kuala Lumpur, Malaysia, Nov., 2005.

[59] A. I. El-Arabawy and S. C. Gupta: "Reduced Mobile Complexity Scheme for Fast Fading Channel Estimation in OFDM-FDD Mobile Communication Systems", in Proceedings of *IEEE International Conference of Universal Personal Communication*, San Diego, CA, vol. 1, pp. 274-78, Oct. 1997.

[60] M. Ozdemir and H. Arslan: "Channel Estimation for Wireless OFDM Systems", *IEEE Communications Survey and Tutorials*, vol. 9, no. 2, pp. 60-74, 2007.

[61] M. Yalcin, and A. Akan: "Pilot Based 2D Channel Estimation for OFDM over Fast Fading Channels", *Journal of Electrical and Electronics Engineering,* vol. 9, no. 2, pp. 1029-1035, 2009.

[62] O. Edfors, M. Sandell, J. J. Beek, S. K. Wilson, and P. O. Brjession: "OFDM Channel Estimation by Singular Value Decomposition", *IEEE Transactions on Communications*, vol. 46, no. 7, pp. 931-939, Jul., 1998.

[63] S. Saleem, and Q. U. Islam: "On Comparison of DFT-Based and DCT-Based Channel Estimation for OFDM System", *IJCSI International Journal of Computer Science Issues*, vol. 8, no. 2, pp. 353-358, May, 2011.

[64] M. C. Necker, and G. L. Stber: "Totally Blind Channel Estimation for OFDM on Fast Varying Mobile Radio Channels", *IEEE Transactions on Wireless Communications*, vol. 3, no. 5, pp. 1514-1525, Sept., 2004.

[65] T. Cui, and C. Tellambura: "Semiblind Channel Estimation and Data Detection for OFDM Systems with Optimal Pilot Design", *IEEE Transactions on Communications*, vol. 55, no. 5, pp. 1-11, May, 2007.

[66] F. Gini and G. B. Giannakis: "Frequency Offset and Symbol Timing Recovery in Flat-Fading Channels: A Cyclostationary Approach", *IEEE Transcactions on Communications*, vol. 46, no. 3, pp. 399-410, 1998.

[67] A. Scaglione and A. Vosoughi: "Turbo Estimation of Channel and Symbols in Precoded MIMO Systems", in Proceedings of *ICASSP*, vol. IV, pp. 413-417, Montreal, Canada, 2004.

[68] A. Scaglione and A. Vosoughi: "On Superimposed Training for MIMO Channel Estimation and Symbol Detection", *IEEE Transaction on Signal Processing,* vol. 55, no. 6, pp. 3007-3012, 2007.

[69] A. Scaglione and A. Vosoughi: "Blind Channel Identification and Equalisation in OFDM using Subspace-Based Methods", *Journal of Communications,* vol. 4, no. 7, pp. 472-480, 2009.

[70] B. Muquet, M. de Courville, and P. Duhamel: "Subspace-based Blind and Semi-Blind Channel Estimation for OFDM Systems", *IEEE Trans. Signal Processing*, vol. 50, no. 7, pp. 1699-1712, Jul. 2002.

[71] J. Heath, R. W. , and G. Giannakis: "Exploiting Input Cyclostationarity for Blind Channel Identification in OFDM Systems", *IEEE Trans. Signal Processing.*, vol. 47, no. 3, pp. 848-856, Mar. 1999.

[72] S. Zhou and G. Giannakis: "Finite-Alphabet based Channel Estimation for OFDM and related Multicarrier Systems", *IEEE Transactions on Communication*, vol. 49, no. 8, pp. 1402-1414, Aug. 2001.

[73] B. Kosko: *Neural Networks and Fuzzy Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1991.

[74] D. Nauck, F. Klawonn, and R. Kruse: *Foundations of Neuro-Fuzzy Systems*, Wiley, Chichester, U.K., 1997.

[75] K. Leiviskä: *Basics of Soft Computing Methods: Industrial Applications of Soft Computing - Paper, Mineral and Metal Processing Industries*, Physica-Verlag, Heidelberg, New York, 2001.

[76] L. Zadeh: "Fuzzy Logic and Softcomputing", in Proceedings of *IEEE International Workshop on Neuro Fuzzy Control*, Muroran, Japan 1993.

[77] L. Zadeh: "Fuzzy Sets", *Information and Control*, vol. 8, no.3, pp. 338-353, 1965.

[78] E. H. Mamdani and S. Assilian: *"An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller"*, *International Journal of Man-Machince Studies,*, vol. 7, pp. 1-13, 1975.

[79] T. Takagi and M. Sugeno: *"Fuzzy Identification of Systems and its Application to Modeling and Control"*, *IEEE Transaction of Systematics, Man and Cybernetics,* vol.SMC - 15, pp. 116-132, 1985.

[80] R. Fuller: *Neural Fuzzy Systems*, Abo Akademi University, Abo, 1995.

[81] J. Zhou, E.Juuso and K. Leiviskä: "Intelligent Methods in Peat Production", in Proceedings of 5^{th} *European Congress on Intelligent Techniques and Soft Computing (EUFIT'97)*, Verlag Mainz, Aachen, Germany, vol. 3, pp. 2103-2107, 1997.

[82] S. Kumar: *Neural Networks A Classroom Approach*, Tata McGraw Hill, 8^{th} Reprint, 2009.

[83] Laurene Fausett: *Fundamentals of Networks Architectures, Algorithms and Applications*, Pearson Education, 1^{st} ed., 1993.

[84] H. Jaeger: "A Tutorial on Training Recurrent Neural Networks, covering BPPT, RTRL, EKF and the "Echo State Network" Approach", *Fraunhofer Institute for Autonomous Intelligent Systems (AIS), German National Research Center for Information Technology*, Mar., 2005.

[85] S. L. Goh and D. P. Mandic: "A Complex-Valued RTRL Algorithm for Recurrent Neural Networks", *Neural Computation*, vol. 16, pp. 2699-2713, 2004.

[86] T. Kohonen, S. Kaski, K. Lagus, J. Salojarvi, J.Honkela, V. Paatero, and A. Saarela: "Self Organization of a Massive Document Collection", *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp. 574-585. May, 2000.

[87] E. Alhoneiemi, J. Hollmn, O. Simula, and J. Vesanto: "Process Monitoring and Modeling using the Self-Organizing Map", *Integrated Computer Aided Engineering*, vol. 6, no. 1, pp. 3-14, 1999.

[88] S. Kaski and K. Lagus: "Comparing Self-Organizing Maps", in Proceeding of *International Conference on Neural Networks*, pp. 809-814, 1997.

[89] H. U. Bauer and K. Pawelzik: "Quantifying the Neighborhood Preservation of Self-Organizing Feature Maps", *IEEE Transactions on Neural Networks*, vol. 3, no. 4, pp. 570-579, 1992.

[90] S. Kartalopoulos: "Understanding Neural Networks and Fuzzy Logic: Basic Concepts and Applications", *IEEE Press, NJ, USA*, 1996.

[91] H. Bolcskei and E. Zurich: "MIMO - OFDMA Wireless Systems: Basics, Perspectives, and Challenges", *IEEE Wireless Communications*, pp. 31-37, August, 2006.

[92] H. Sampath, S. Talwar, J. Tellado, V. Erceg and A. Paulraj: "A Fourth Generation MIMO - OFDMA Broadband Wireless System: Design, Performance and Field Trial Results", *IEEE Communications Magazine*, vol. 40, no. 9, pp.143-149, 2002.

[93] I. Koffman and V. Roman: "Broadband Wireless Access Solutions based on OFDM Access in IEEE 802.16 ", *IEEE Communication Magazine*, vol. 40, no. 4, pp. 96-103, Apr. 2002.

[94] R. Bercovich: "OFDM Enhances the 3G High-Speed Data Access", in Proceedings of *GSPx 2004 Conf.*, available online at http:// www.techonline.com/ pdf/ pavillions/ gspx/ 2004/ 1084.pdf, Santa Clara, CA, 2004.

[95] Y. Ogawa, K. Nishio, T. Nishimura and T. Ohgane: "A MIMO - OFDM System for High-Speed Transmission", in Proceedings of 58^{th} *IEEE Vehicular Technology Conference (VTC 2003)*, pp. 493-487, Orlando, Florida, USA, 2003

[96] M. Collados and A. Gorokhov: "Antenna Selection for MIMO - OFDMWLAN Systems", *International Journal of Wireless Information Networks*, vol. 12, no. 4, 2005.

[97] W. Bocquet, K. Hayashi and H. Sakai: "A Power Allocation Scheme for MIMO OFDM Systems", in Proceedins of 2^{nd} *International Symposium on Communications, Control and Signal Processing,* Marrakech, Morocco, 2006.

[98] S. L. Hijazi: "Multiuser Detection for Multicarrier Communication Systems", Disseration for the award of PhD degree submitted to *Department of Electrical and Computer Engineering, College of Engineering, Kansas State University, Manhattan, Kansas, USA,* 2006.

[99] H. Bolcskei: " Principles of MIMO-OFDM Wireless Systems", as a chapter in *CRC Handbook on Signal Processing for Mobile Communications, M. Ibnkahla, Ed., Ch. 12,* 2004.

[100] M. Collados and A. Gorokhov: " Antenna Selection for MIMO-OFDM WLAN Systems", *International Journal of Wireless Information Networks*, vol. 12, no. 4, Dec., 2005

[101] M. Ibnkahla, J. Yuan and R. Boutros: "Neural Networks for Transmission over Nonlinear Channels", chapter of a book titled *Signal processing for mobile communications handbook*, ed. M. Ibnkahla, CRC Press, pp. 552-579, USA, 2005.

[102] F. Cadini, E. Zio and N. Pedroni: " Validation of Infinite Impulse Response Multi layer Perceptron for Modelling Nuclear Dynamaics", *Hindawi Publishing Corporation Sceicne and Technology Installations*, vol. 2008.

[103] T. Koskela, M. Lehtokangas, J. Saarinen and K. Kashi: "Time Series Prediction with Mult Layer Perceptron, FIR and Elman Neural Networks", in Proceedings of *World Congress on Neural Networks,* pp 491-494, San Diago, CA, USA, 1996.

[104] A. Back, E. .A. Wan, S. Lawrence and A. C. Tsoi: " A Unifying view of Some Training Algorithms for Multi Layer Perceptrons with FIR Filter Synapses", *Neural Networks for Signal Processing 4*, IEEE Press, pp. 146-154, 1994.

[105] J. W. Tukey: "Discussion, Emphasizing the Connection Between Analysis of Variance and Spectrum Analysis", *Technometrics*, vol. 3, pp. 191-219, 1961.

[106] S. Haykins: *Adaptive Filter Theory*, Pearson Education, 4^{th} Ed., New Delhi, 2002.

[107] T. Kim and T. Adali: "Fully Complex Backpropagation for Constant Envelope Signal Processing", in Proceedings of *IEEE Signal Processing Society Workshop: Neural Networks for Signal Processing*, vol. 1, pp. 231-240, Piscataway, NJ, USA, Dec., 2000.

[108] K. Tokunaga, T. Furukawa and S. Yasui: "Modular Network SOM: Extension of SOM to the Realm of Function Space", in Proceedings of the *WSOM2003*, pp.173-178, Hibikino, Kitakyushu, Japan, 2003.

[109] G. J. Klir and B. Yuan: *Fuzzy Sets and Fuzzy Logic Theory and Applications*, Prenitce Hall of India, 1^{st} ed., New Delhi, 2002.

[110] T. J. Ross: *Fuzzy Logic with Engineering Applications*, Wiley India, 2^{nd} ed., New Delhi, 2008.

[111] R. A. Aliev, B. G. Guirimov, B. Fazlollahi and R. R. Aliev: "Evolutionary Algorithm-based Learning of Fuzzy Neural Networks. Part 2: Recurrent Fuzzy Neural Networks", Elsavier Journal of Fuzzy Sets and Systems, vol. 160, pp. 2553-2566, 2009.

[112] P. Stavroulakis: *Neuro-Fuzzy and Fuzzy-Neural Applications in Telecommunications*, Springer-Verlag, Berlin, 2004.

Publications out of the Thesis Work

- **Journals**

1. K. K. Sarma and A. Mitra: "Modeling MIMO Channels using a Class of Complex Recurrent Neural Network Architectures", *Elsevier AEU International Journal of Electronics and Communication* (In Press), doi:10.1016 / j.aeue.2011.08.008 (Online published on 21.9.2011).
2. K. K. Sarma and A. Mitra: "Fuzzy-based hybrid MIMO Channel Estimator with variable Membership and Inference Rule Set", to appear in *International Journal of Computer and Communications*, Dec., 2011 -Jan., 2012.
3. K. K. Sarma and A. Mitra: "Recurrent Fuzzy-Neural MIMO Channel Estimation", Communicated to *Elsevier AEU International Journal of Electronics and Communication.*
4. K. K. Sarma and A. Mitra: "Decision making in Fuzzy-Neural MIMO Channel Estimator," Communicated to *WASET International Journal of Information and Communication Engineering.*

- **Independent Book Chapter**

1. K. K. Sarma and A. Mitra: "Estimation of MIMO Wireless Channels using Artificial Neural Networks" as a chapter of a book titled *Cross-Disciplinary Applications of Artificial Intelligence and Pattern Recognition: Advancing Technologies*, IGI Global, 701 E. Chocolate Ave. Hershey, PA 17033, USA, 2011 (In Press).

- **Book Chapters Derived out of Conference Proceedings**

1. K. K. Sarma and A. Mitra: "A Class of Recurrent Neural Network (RNN) Architectures with SOM for Estimating MIMO Channels", as a chapter of a book titled *Advances in Computing and Communications in Computer and Information Science*, Springer-Verlag, Heidelberg, Germany, vol. 192, part 4, pp. 512-521, July, 2011.
2. K. K. Sarma and A. Mitra: "Estimation of Multipath Fading Channel of MIMO-OFDM System using ANN", as a chapter of a book titled *Advanced Computing Applications Databases and Networks* edited by Shahin Ara Begum and Prodipto Das, Narosa Publishing House, New Delhi, pp. 215-220, June, 2011.

- **Conference Proceedings**

1. K. K. Sarma and A. Mitra: "MIMO Channel Modeling: Suitability between Neuro-Fuzzy and Fuzzy-Neural Approaches," Communicated to *IEEE CISP-2012*, Don Bosco University, Guwahati, Assam.
2. K. K. Sarma and A. Mitra: "Membeship and Inference Rule Generation for Fuzzy-Neural MIMO Channel Estimator," to appear in Proceedings of *IEEE WICP-2011*, Mumbai, India.

3. K. K. Sarma and A. Mitra: "MIMO Channel Modeling with Cluster Configuration of Complex Time Delay Fully Recurrent Neural Network", in Proceedings of *IEEE International Conference on Recent Advances in Intelligent Computational Systems 2011*, pp. 1-4, Trivundrum, India, Sept., 2011.

4. K. K. Sarma and A. Mitra: "MIMO Channel Prediction: Neuro-Fuzzy or Fuzzy-Neural?", in Proceedings of *IETE Zonal Conference,* pp. 39-42, Guwahati, Assam, India, Aug., 2011.

5. K. K. Sarma and A. Mitra: "Estimation of MIMO Channels using Complex Time Delay Fully Recurrent Neural Network ," in Proceedings of 2^{nd} *IEEE National Conference of Emerging Trends and Applications in Computer Science (NCETACS - 2011), St. Anthony's College, Shillong,* pp. 6-11, Meghalaya, India, Mar., 2011.

6. K. K. Sarma and A. Mitra: "MIMO Channel Modeling using Temporal Artificial Neural Network (ANN) Architectures", in Proceedings of 1^{st} *ACM International Conference on Intelligent Interactive Systems and Multimedia (IITM),* pp. 37- 44, IIIT Allahabad, India, Dec., 2010.

7. K. K. Sarma and A. Mitra: "Rician Multipath Fading Channel Estimation using ANN of a MIMO-OFDM system", in Proceedings of *IEEE National Conference in Electrical, Power Engineering, Electronics and Computer Tech Meet-2010*, pp. 110-115, Pailan, West Bengal, India, Feb., 2010.

8. K. K. Sarma and A. Mitra: "ANN based Rayleigh Multipath fading channel estimation of a MIMO-OFDM system", in Proceedings of *IEEE* 1^{st} *Asian Himalayas International Conference on Internet AH-ICI 2009*, pp. 1-5, Kathmandu, Nepal, Nov., 2009.