ARTIFICIAL NEURAL NETWORK BASED SPARSE CHANNEL ESTIMATION FOR OFDM SYSTEMS

AbdurRehman Bin Tahir

2014.11.01.003

KADIR HAS UNIVERSITY

2017

ABDURREHMAN BIN TAHIR M.S. Thesis 2017

ARTIFICIAL NEURAL NETWORK BASED SPARSE CHANNEL ESTIMATION FOR OFDM SYSTEMS

ABDUR REHMAN BIN TAHIR

M.Sc., Electronics Engineering, Kadir Has University, 2017

Submitted to the Graduate School of Science and Engineering

In partial fulfillment of the requirements for the degree of

Masters of Science

in

Electronics Engineering

KADIR HAS UNIVERSITY

2017

KADIR HAS UNIVERSITY

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

ARTIFICIAL NEURAL NETWORK BASED SPARSE CHANNEL ESTIMATION FOR OFDM SYSTEMS

ABDUR REHMAN BIN TAHIR

APPROVED BY:

Asst. Prof. Dr. Habib Şenol (Kadir Has University) [illegible]
(Thesis Advisor)

Asst. Prof. Dr. Atilla Őzmen (Kadir Has University) [illegible]
(Thesis Co-Advisor)

Assoc. Prof. Dr. Serhat Erkűçűk (Kadir Has University) [illegible]

Prof. Dr. Hakan Ali Çırpan (Istanbul Technical University) [illegible]

APPROVAL DATE: 16/01/2017

ARTIFICIAL NEURAL NETWORK BASED SPARSE CHANNEL ESTIMATION FOR OFDM SYSTEMS

Abstract

In order to increase the communication quality in frequency selective fading channel environment, orthogonal frequency division multiplexing (OFDM) systems are used to combat inter-symbol-interference (ISI). In this thesis, a channel estimation scheme for the OFDM system in the presence of sparse multipath channel is studied. The channel estimation is done by using the artificial neural networks (ANNs) with Resilient Backpropagation training algorithm. This technique uses the learning capability of artificial neural networks. By means of this feature we show how to obtain a channel estimate and how it allows the proposed technique to be less computationally complex; as there is no need for any matrix inversions. This proposed method is compared with the Matching Pursuit (MP) algorithm that is well known estimation technique for sparse channels. The results show that the ANN based channel estimate is computationally simpler and a small number of pilots are required to get a better estimate of the channel especially in low SNR levels. With this setting, the proposed algorithm leads to a better system throughput.

Keywords – Orthogonal Frequency Division Multiplexing (OFDM), Sparse Channel Estimation, Matching Pursuit Algorithm, Artificial Neural Network (ANN).

OFDM SİSTEMLER İÇİN YAPAY SİNİR AĞI TABANLI SEYREK KANAL KESTİRİMİ

Özet

Frekans seçici sönümlemeli kanal ortamında, haberleşme kalitesini arttırmak için dik frekans bölmeli çoğullama (OFDM) sistemleri semboller arası girişimle baş edebilmek için kullanılmaktadır. Bu tezde, seyrek çok-yollu kanalın bulunması durumunda, OFDM sistemlerinde kanal kestirimi çalışılmıştır. Kanal kestirimi, Esnek Geri Yayılım eğitim algoritması kullanan yapay sinir ağları (YSA) ile gerçekleştirilmiştir. Bu teknik yapay sinir ağlarının öğrenme yetisini kullanmaktadır. Bu özellik sayesinde, kanal kestiriminin nasıl yapıldığı ve önerilen yöntemin herhangi bir matris tersine ihtiyaç duymadan daha az hesaplama karmaşıklığına nasıl sahip olabildiği gösterilmektedir. Önerilen bu yöntem, en uyguna yakın Eşleştirme Arama (MP) algoritması ile karşılaştırılmıştır. Sonuçlar, özellikle düşük SNR seviyelerinde daha iyi kanal kestirimi elde edebilmek için, YSA tabanlı kanal kestiriminin hesaplama kolaylığı sağladığını ve daha az sayıda pilot veriye ihtiyaç duyulduğunu göstermiştir. Böylece, önerilen yöntemin daha iyi bir sistem çıkışına olanak sağladığı gösterilmiştir.

Anahtar Kelimeler – Dik Frekans Bölmeli Çoğullama (OFDM), Seyrek Kanal Kestirimi, Eşleştirme Arama Algoritması, Yapay Sinir Ağları.

Acknowledgements

In the Name of Allah, the Most Beneficent, the Most Merciful. All the praises and thanks be to Allah, the Lord of the *'Alamin.* To Allah everything belongs and to Him it shall return.

First, I would like to convey my earnest gratefulness to my advisor Dr. Habib Şenol and co-advisor Dr. Atilla Őzmen from Kadir Has University for their unremitting care during my M.S. study and research, for their endurance, inspiration, passion, and vast knowledge. They enlightened me through every step of the way and I could not have thought of having better advisors and mentors for my study. I am justly indebted to you sirs for all the help.

Further, I express my gratitude to my instructors for conveying the knowledge and insight throughout the Master's program that made it possible today to finish and publish the work. Also, special thanks to my friends A. Ahad, A. Saad and M. Sohaib for their moral backing and for standing beside me through thick and thins of this long journey.

A final shout out to my brother and sisters, who associatively establish one of the strongest pillars on which this assembly stands today without a worry. This accomplishment would not have been possible without them. Thank you!

To my parents...

Table of Contents

List of Figures

List of Tables

1. Introduction

In a general setup of wireless communication systems, a signal is transmitted which passes through a wireless medium and is received at the receiver. The wireless medium – generally called wireless channel – is modeled as a pseudo-differential operator. In data modulation sense, the basic communication systems modulate the data onto a single carrier frequency due to which the available bandwidth is completely occupied by the transmitted symbol. However, in modern communication systems, the available spectrum is divided into equal sub channels. This is the base of orthogonal frequency division multiplexing (OFDM), in which the sub channels are mutually orthogonal that helps in mitigating the inter-symbol interference (ISI) and hence, providing large data rates and radio channel impairments. The attraction of OFDM is due to the fact that it handles multipath effect at the receiver which causes ISI and frequency selective fading. In this communication setup, the wireless channel estimation refers to carefully calculating the operator and equalization refers to commuting the transmitted signal. In this thesis, we study the problem of channel estimation for OFDM in time-invariant sparse mobile communication channels. We use already established mathematical models to describe the signal and wireless communication channel. Our study largely focuses on the channel estimation part of the system; to devise an architecture that uses a more efficient and computationally les complex algorithm i.e. the artificial neural networks (ANNs) as channel estimators. Making use of the learning power of ANNs, a technique has been developed for channel estimation and the performance of this technique is studied. The performance – in terms of efficiency/correctness and computational complexity – of this suggested architecture is compared with the existing sub-optimal Matching Pursuit (MP) algorithm for the channel estimation under sparse setting.

Furthermore, not much work has been previously done on the implementation of the ANN as a channel estimator when the channel's impulse response in the communication system is sparse. Another purpose that this study serves is the insight it provides to the channel estimation algorithms from the computational perspective. The Computational complexity theory is a separate field of study itself that is partly based on the theoretical computer science and mathematics; hence, studies have been done on the complexities of ANNs and MP algorithms. Making use of the proven theoretical models, we compare the time-complexities of both algorithms. Using the results from intense simulations in *Matlab®* environment, we show that given the right architectural parameters, ANN is superior than MP algorithm not just in terms of symbol error rate (SER) performance (information throughput) but is thought to be better in computational performance.

1.1. Literature Review

The wireless channels that occur just because of the multipath propagation of the transmitted signal and are static within one OFDM symbol are modeled with a convolutional operator i.e. the pseudo-differential operator reduces to a Fourier multiplier e.g. finite impulse response (FIR) filters. Such operators are perfectly diagonal in frequency domain and are known as frequency selective channels. Channel estimation of OFDM systems [1] in frequency selective channels with the help of pilot symbols are quite common [2] and is used in many applications. [3] discusses another approach of channel estimation by banded approximation of the channel matrix for OFDM systems. [4] and [5] discusses some optimal techniques for channel estimation and equalization and efficiency of the estimation algorithms. Some communication problems for OFDM contain channels with large delay spread and a smaller non-zero support [6]. These channels are known as sparse channels and are faced in a number of practical applications like high definition television (HDTV) where there are few echoes but the channel response spans hundreds of data symbols [7]. Some of the delay profiles like underwater acoustics or hilly terrain (HT) delay profile in broad-band wireless communication systems, comprise of a sparsely distributed multipath [8]. Different channel estimation techniques have been in use for such systems and proven to have different performance under different settings; the properties of the Least Square's estimators are investigates in [9]. An assemblage of sparse channel estimation algorithms has lately appeared and is known as compressed sensing [10]. Compressive sampling MP (CoSaMP) is used as sparse channel estimator in [11] based on the CoSaMP algorithm studied in [12]. The estimation of the channel taps one by one is done by using Matching Pursuit (MP) algorithm in [6] that is based on the work originally brought in light in [13].

However, all the mentioned multipath sparse channel estimation algorithms either suffer from an error floor or have high computational overhead. Artificial neural networks are computationally cheap and efficient algorithms that have caught much attention these days for channel estimation problems. In [14], authors have discussed an artificial neural network (ANN) based channel estimator that estimates the multipath channel in the frequency domain. The study uses multilayer perceptron NN (MLP-NN) with Levenberg-Marquardt as the learning algorithm.[15] evaluates the performance of pilot based OFDM system for multipath channel by using Generalized regression neural networks (GRNN) and adaptive network based fuzzy interference systems (ANFIS). A number of other studies use the same basic OFDM models and estimate the multipath channel either in frequency or time domain using ANN; with a slight change in the ANN architecture. [16] proposes a channel estimation using ANN for the LTE uplink system. Received pilot symbols are used in this study to first train the network and then estimate the whole channel. [17] – [19] also discuss the use of ANN as an estimator for OFDM systems under different assumptions. [20] – [21] are some further studies that make use of the recurrent neural networks (RNN) as channel estimator for different scenarios like multiple input multiple output OFDM (MIMO-OFDM) systems, STBC systems, OFDM interleave division multiple access (OFDM-IDMA) etc. A recent study [22] involves ANN based sparse channel estimation for MIMO-OFDM systems.

Computational complexity of the estimation algorithm is one of the major factors that define the quality of the estimator. Complexity theory being a separate field of study, much effort has been made on the complexity analysis and comparison of diverse algorithms. A comparative complexity study on the several invariants of MP algorithm is done in [27] and [28]. The complexity of ANN, however, is still an open research question and a lot of effort has been made towards the satisfactory solution to this problem. The author in [29] argues that for neural networks, measuring the computing performance entails to new gears from information theory and computational complexity. The complexity of learning with regards to the Multi-Layer Perceptron ANN (MLP-ANN) is comprehensively discussed in [30] in which author ponders upon the expected computational complexity of the learning problem.

1.2. Research Methodology

The motivation behind this work was driven by the fact that with the advent of technology, the need to find better and easier ways to go from point A to point B has increased. This shows researchers a way of combining the classical algorithms with the advanced ones to find better and faster solutions. Under the umbrella of Digital Signal Processing in modern day's wireless communications with high demand of bandwidth efficient algorithms, the channel estimation with better accuracy and efficiency becomes hard and computationally complex. This increase in complexity makes the machines' ability to learn, an important factor that cannot be neglected. Various machine learning algorithms namely Artificial Neural Networks (ANNs), Support Vector Machines (SVMs) etc. have been previously used in some earlier studies for the purposes of channel estimation in different wireless communication settings. With this motivation, we formulate the research question of our thesis that which machine learning architectures can be used for channel estimation under sparse setting, and how? Further, to devise an architecture to find the algorithm that is more efficient and less computationally complex than the current sub-optimal algorithms.

This work answers the above questions and formulates a comprehensive solution to the given sparse problem. The author aims to extend this work to the case of Time-Varying Sparse Channels in which we observe a movement between the transmitter and receiver; which in result introduces a phenomenon known as the Doppler's effect. The channel estimation problem in this case becomes difficult, hence, more complex mathematical models and powerful channel estimation schemes are required in order to solve the said problem. Furthermore, the author believes that this work paves the way for the future work that can produce a better performing communication systems with lesser complexity. The use of ANNs impose some questions as the ANN optimization is still an open problem, so, with the advancements in the field of ANNs an optimized solution to such problems with lesser computational complexity and better communication system throughput can be obtained.

Firstly, this research aims to develop the system model in chapter 2, that includes the OFDM and channel properties and model derivations given in the successive sections. Thereafter, a matrix form of the channel model is also obtained towards the end of the chapter. Secondly, the fundamentals of channel estimation are discussed in chapter 3 that covers the classical estimation algorithms; the sub-optimal Matching Pursuit (MP) algorithm based sparse channel estimator that forms the benchmark for our study is discussed in the final subsections of the chapter. It continues into the chapter 4 with the introduction, working and architectural models for Artificial Neural Networks; where the ANN model for the proposed estimator's architecture is discussed towards the end of the chapter. Finally, the results of this research obtained from simulations are presented and compared in Chapter 5 and conclusions are made in the final chapter 6.

2. System Model

2.1. Orthogonal Frequency Division Multiplexing (OFDM)

Before the advent of OFDM, FDM was used in order to transmit more than one signal through the telephone lines. As the name suggests, FDM divides the whole channel bandwidth into smaller sub-channels and multiple low rate signals are transmitted over the different sub-channels having different frequencies, fig. 1(a). However, this method is inefficient due to the fact that the sub-channels interfere with one another and have inadequacies even after being separated by a guard interval.

The urge to solve the bandwidth efficiency problem, gave birth to OFDM in which multiple, orthogonal, narrow-band sub-channels are transmitted in parallel hence, being more bandwidth efficient as pictured in fig. 1(b). Due to the demands of high data rates in wireless multimedia applications, OFDM has become more popular in recent times and becomes the modem of choice in modern wireless communication systems [31].

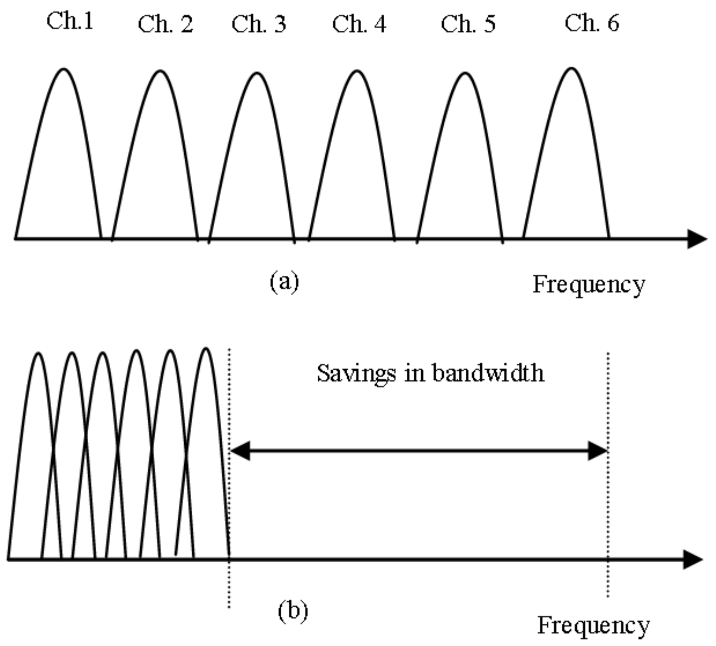


Figure 1: (a) Multiple Carriers in FDM, (b) Multiple Orthogonal Carriers in OFDM

Other than combatting with the multi-path fading and Inter-Symbol Interference (ISI), OFDM is also robust in high speed wireless communications. It is considered to be a remarkable method that is capable of decreasing the frequency-selective fading into flat fading by dividing the available bands into several subcarriers [32]. For this reason, OFDM symbol is quite longer than any other single-carrier system symbol, that makes it more robust against the delay dispersion of the channel and the fading caused by it [33].

2.1.1. OFDM Basics

An OFDM signal is generated on the transmitter side digitally because of the difficulty in designing the signal and receiver in the analog domain [34]. Fig.2 shows a typical OFDM system model in an AWGN channel [35].

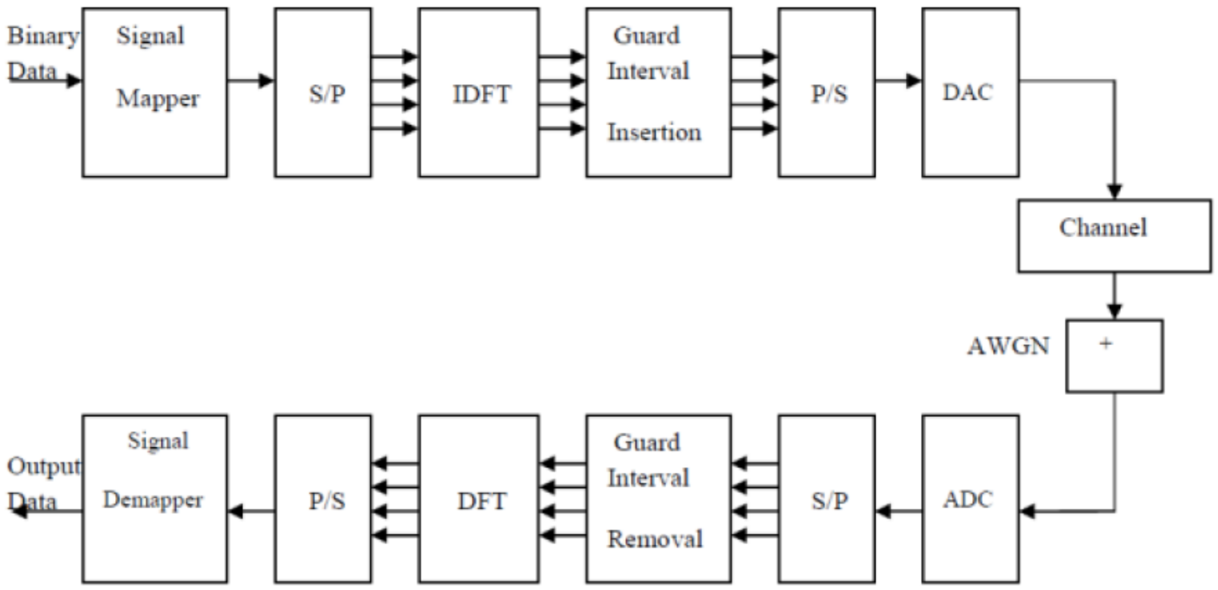


Figure 2: The basic block diagram of an OFDM system in AWGN channel

The incoming data stream is first modulated using any specific modulating schemes i.e. QPSK, QAM etc. and the symbol from these schemes are then serial to parallel converted. This parallel data is then fed to an N-point Inverse Discrete Fourier Transform (IDFT) and converted to the time domain sequence. IFFT is more cost efficient than IDFT, hence, is widely used as well.

The efficiency of OFDM lies in the fact that all the sub-carriers are closely spaced that is allowed because there exists orthogonality between them. This orthogonality can be checked by multiplying and then integrating any two subcarriers. Following section explain the signal model which hold the orthogonality principle for any given subcarriers. The carriers are linearly dependent if the carrier spacing is $1/T_{sym}$, where T_{sym} is the OFDM symbol duration; this can be held when the OFDM signal is defined by the FFT procedures. The OFDM symbol duration is defined as $T_{sym} = N \times T_s$, where N is the total number of subcarriers and T_s is the sampling duration.

After the IFFT block, a cyclic prefix (CP) is added to the signal that helps to mitigate the ISI. CP is the copy of a fraction, typically 25%, of the last part of the individual OFDM symbol that is added to the start in order to allow the receiver to capture the starting point of the symbol with the probability of the length of CP [32].

2.1.2. Signal Model

We reflect an OFDM system with N total subcarriers, the transmitted OFDM signal in discrete-time is then given as:

$$s[n] = \frac{1}{N} \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} d[k] \, exp(j2\pi k \, \frac{n}{N}) \tag{1}$$

Where, *N* is the total number of sub-carriers, $d[k]$ is the frequency domain data symbol transmitted at discrete time n and subcarrier k. By the Central Limit Theorem, $s[n]$ can be modeled as zero-mean complex Gaussian sequence, given that N is sufficiently large [36]. A set of known data symbols with known locations – branded as pilot symbols – are also imbedded in the transmitted signal. These uniformly-spaced pilot symbols are used on the receiver side in channel estimation step and, hence, estimation is called Pilot-Aided channel estimation. Throughout this study, the symbol Υ will be used to represent the pilot spacing, hence, greater the value of Υ, smaller will be the total number of pilot symbols. The total number of pilot symbols are denoted by N_p.

After the CP insertion of length L_c, the multiple OFDM symbols are now ready to be transmitted over consequent sub-carriers. These are then parallel to serial converted and transmitted over the Time-Invariant sparse multipath channel.

2.2. Wireless Channel

Ideally, the wireless channel should leave the transmitted signal unchanged so that the received signal equals the transmitted signal, i.e.$y(t) = s(t')$, where $t' = t - \tau_0$ defines the time shift equivalent to the time τ_0 it takes for the signal to reach the receiver from the transmitter. The ideal channel never happens in the practice obviously, hence, this subsection gives a brief review of some important properties of the wireless channel and its effect on the signal once it propagates through the channel. It also discusses the mathematical channel model used for this study.

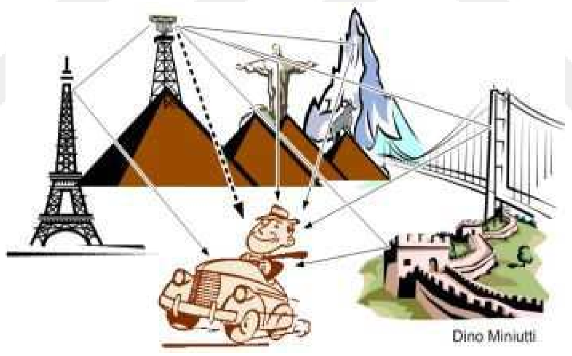


Figure 3: Wireless Multipath Channel

2.2.1. Multipath Fading

The transmitted signal is affected by a number of factors while it propagates through the wireless medium. Firstly, electromagnetic waves deteriorate as it passes through the radio channel and the power of the received signal is decreased with the increased distance between the transmitter and receiver; known as path loss. Additionally, as the signals' direct line of sight (LOS) path might be blocked by some physical objects, it might take other paths to the receiver; fig. 3 [37]. This multipath propagation occurs where the single transmitted signal arrives at the receiver through multiple different paths with different delays and attenuation factors because of the reflection, diffraction, scattering etc. instigated by the physical objects (collectively assumed as Scatterers). This makes the received signal's power fluctuate over time/frequency and is called fading. This multipath fading comes under the umbrella of small-scale fading and can be defined by the Rayleigh statistical model.

The communication problems concerning the estimation and equalization of the communication channels, that have a larger delay spread and small non-zero support, are being studied. These channels are known to have sparse channel impulse response. Sparse multipath channels (SMPC) can be seen in many real-world applications and the immobile channel impulse response in continuous time is written as:

$$c(t) = \sum_{l=0}^{L-1} \alpha_l \, \delta(t - \tau_l) \tag{2}$$

for $l = 0, \dots, L-1$. where,

L is the total number of multi-paths

α_l is the attenuation factor for each path

τ_l is the delay for the path l

The α_l are a subset of complex natural distribution with zero-mean and variance of σ_l^2 i.e. $\sim \mathcal{CN}(\mathbf{0}, \sigma_l^2)$ with normalized unit power.

As a sparse problem, it is not possible to directly show the discrete-time equivalent of (2), hence a discrete-time sparse representation is developed in the next section.

2.2.2. Sparsity in Wireless Channels

Given the fact that the transmitter, receiver and all the scatterers are static, the system can be modeled as linear time-invariant (LTI) system. Considering this, some communication environments involve channels with large delay spread and a small non-zero support; such problems occur quite often in practical implementations. The channel response of such channels spans many hundreds of data symbols in HDTV where there are a very few numbers of echoes due to multipath. The theory of LTI-SMPC suggests that $L << N_{CP}$, where N_{CP} is the length of the cyclic prefix (CP). An illustration of sparse multipath channel i.e. $c[n] \neq 0$ for really limited values of n, is given in Fig. 4. The channel length in the figure is 40, however, the non-zero or dominant taps amount to 15.

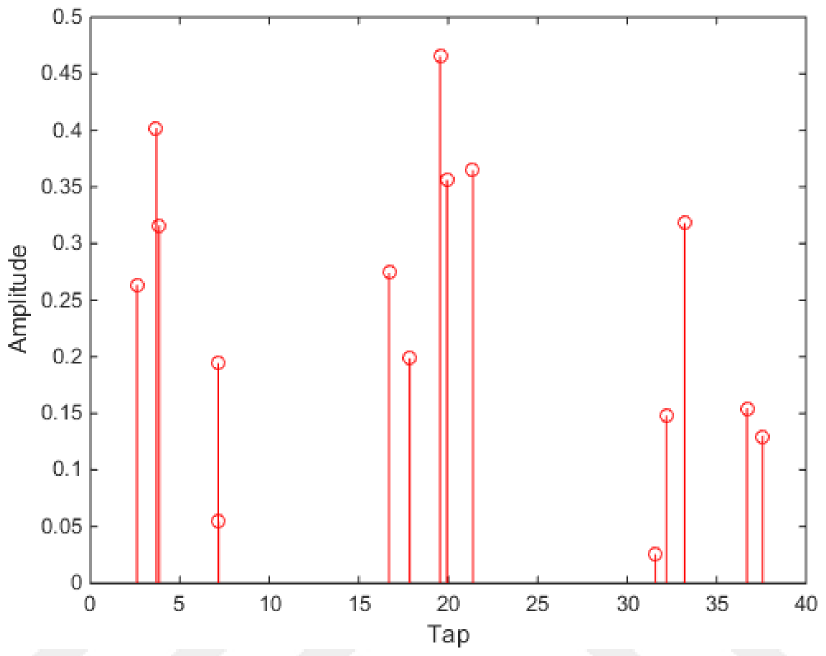


Figure 4: Sparse Multipath Channel

Recently, Compressive Sensing (CS) technique has gained popularity because of its efficiency in signal acquisition frameworks where the signal considered as sparse or compressible in frequency or time-domain; meaning that for continuous signals, the information rate is much smaller than as depicted by the signal's bandwidth. Also, a discrete-time signal relies on a certain degree of freedom that is fairly smaller than the signal's finite span [38]. As a result, length of the training sequence can be shortened as compared to linear estimation techniques. CS explores the sparsity of the signals and it has shown to be possible because an exact depiction of such signals is possible in terms of a suitable basis.

Working on the discrete-time sparse signals is common as it is simpler than its continuous-time counterpart and also that it is more developed. Taking the Fourier transform (FFT) of CIR given in (2), we get:

$$H(f) = \int_0^{T_{sym}} c(t) e^{-j2\pi f t}.dt \tag{3}$$

Replacing (2) in (3) and simplifying the equation gives:

$$H(f) = \sum_{l=0}^{L-1} \alpha_l \, e^{-j2\pi f \tau_l} \tag{4}$$

where, $f = (-\frac{N}{2} + k)\Delta f$ for $k = 0,1, \ldots, (N-1)$ and $\Delta f = \frac{1}{T_{sym}}$; Δf being the carrier spacing and T_{sym} is the symbol duration. Also, the continuous-time path delays τ_l can be represented in discrete-time, with a resolution factor ρ, as $\tau_l = \eta_l'(\rho T_s{}')$; here, $\eta_l' \times \rho = \eta_l$. Similarly, $T_{sym} = N(\rho T_s{}')$ and $T_s = \rho T_s{}'$. Then (4) can be written in simplified form as:

$$H(k) = \sum_{l=0}^{L-1} \alpha_l \, e^{-j2\pi(-\frac{N}{2}+k)(\frac{\eta_l}{\rho N})} \tag{5}$$

Eq. (5) represents the channel frequency response for the sparse channel impulse response given in (2). Now, in order to get the equivalent representation of discrete channel impulse response $h_{eq}[n]$, we take inverse Fourier Transform (IFFT) of (5) as:

$$h_{eq}[n] = \sum_{k=-\frac{k}{2}}^{\frac{K}{2}-1} H[k] \, e^{\frac{j2\pi nk}{\rho N}} \tag{6}$$

After replacement and rearrangement of (6), we get the discrete time equivalent channel impulse response representation of the multipath channel. We can write it in matrix form as $\boldsymbol{h_{eq}} = \boldsymbol{F_\eta^\dagger} \, \boldsymbol{H}$, where, $\boldsymbol{H}$ is given in (5) and $\boldsymbol{F_\eta^\dagger} \in \mathbb{C}^{\rho N_{CP} \times N}$ is inverse of $F_\eta = \frac{1}{\rho N} e^{-\frac{j2\pi}{\rho N} k(n-\eta_l)}$ and N_{CP} is the length of CP. The equivalent representation of channel's frequency response is then given as: $\boldsymbol{H}_{eq} = \boldsymbol{F_\eta} \, \boldsymbol{h_{eq}} \in \mathbb{C}^{N\times 1}$. $\boldsymbol{h_{eq}}$ will be used during the training part of neural networks as the target set. It is explained in detail in Chapter 4.

The multipath sparse channel with 15 multipath was shown in fig. 4; its equivalent discrete time-domain representation $\boldsymbol{h}_{eq}$ with resolution $\rho = 16$ is presented in fig. 5.

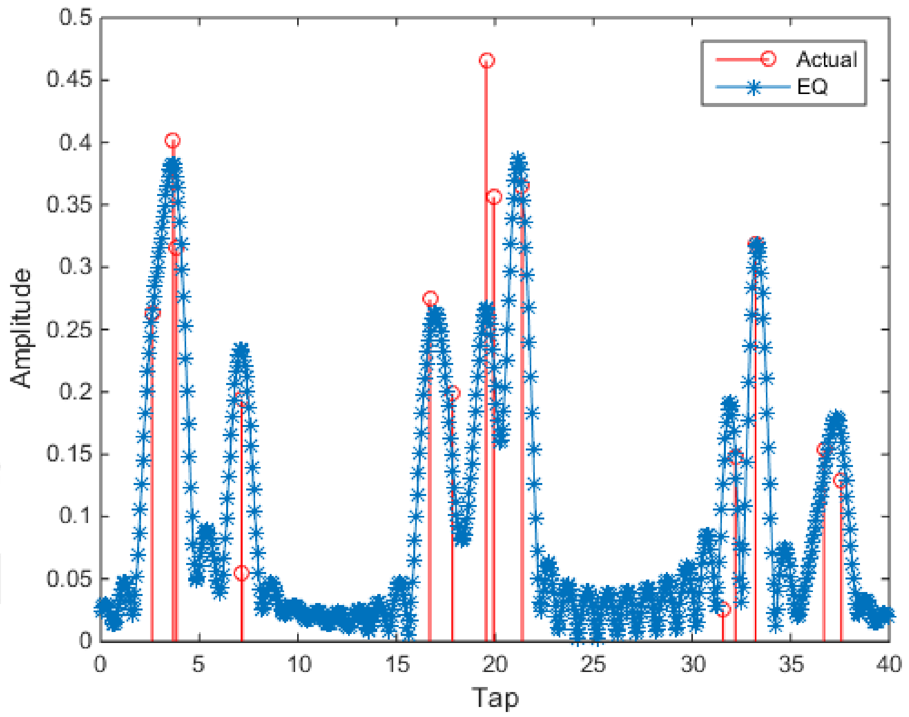


Figure 5: Time-Domain representation of the Sparse-multipath channel and its equivalent channel ($\rho = 16$)

The total number of multipath used for this study is 3 and fig. 6 presents the sparse multipath channel and its equivalent discrete time representation. The ripples that can be observed in the plots are due to the $sinc$ functions that arise because of the band-limited property of the channel. Also, as we can see that most of the values are either zero or near zero, such points are not required to be estimated and are mostly ignored at the channel estimation step by using a threshold factor. This is the reason why we require lesser pilots symbols for estimation and in return the throughput is increased.

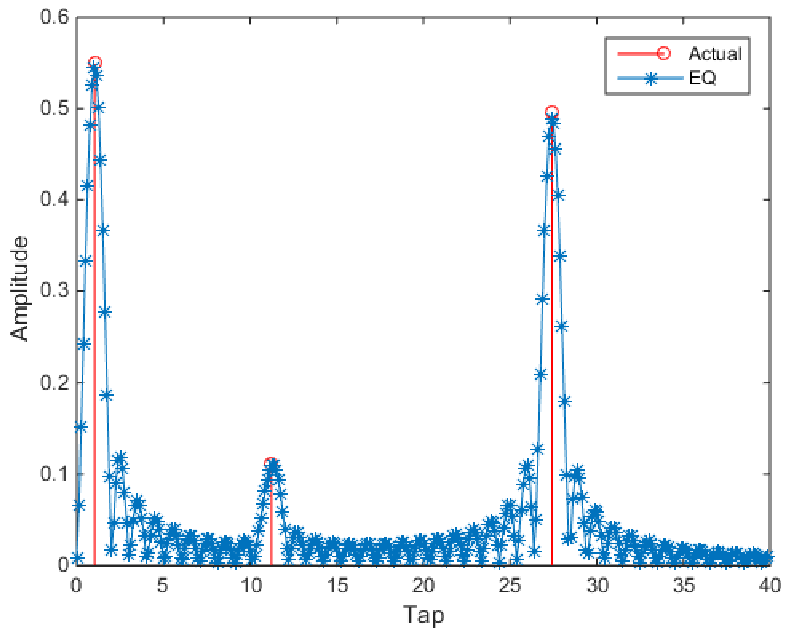


Figure 6: Time-Domain representation of the Sparse-multipath channel and its DT equivalent channel (ρ=8)

Thus, the frequency domain representation $\boldsymbol{H}$ and $\boldsymbol{H}_{eq}$ of $\boldsymbol{h}$ and $\boldsymbol{h}_{eq}$, respectively, is presented in the following Fig. 7.

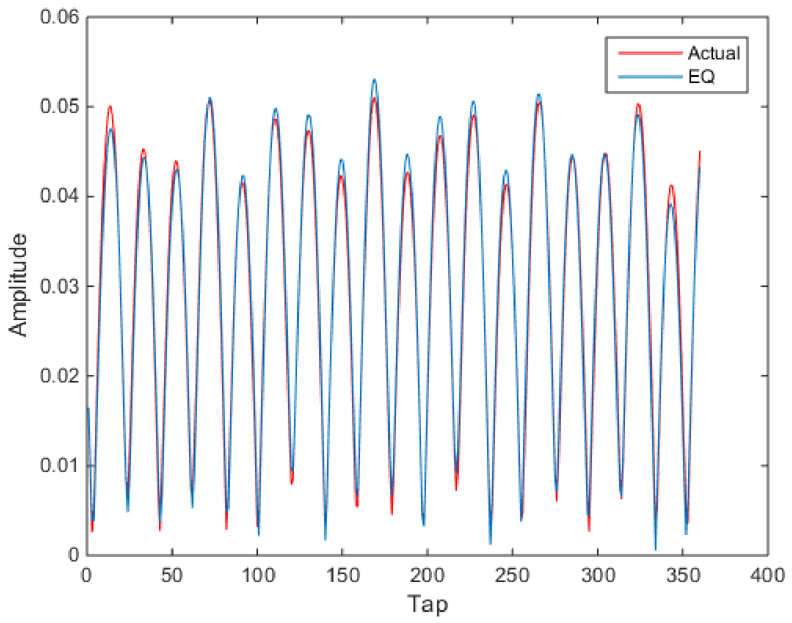


Figure 7: Frequency-domain representation of sparse channel and its equivalent channel

We'll see in the channel estimation sections that the resolution plays an important role in the efficiency of the estimation algorithm. Algorithms like MP that exploit the sparsity of the signal show difference in the performance on the basis of different values of ρ. The effect of ρ can be observed by comparing figures 5 and 6.

2.3. Unaccountable Additive Noise

Other than the fading effects on the propagating signals, there are some other minor electromagnetic effects that are unaccountable for in the channel. The combination of these effects are catered for by adding a noise into the signal when it passes through the channel. The noise process is usually additive and is categorized by its intensity and distribution. The noise process is measured in Signal to Noise Ratio (SNR) which is a ratio, hence, have no units and is represented in term of decibels dB. In the frequency modulation techniques, SNR is usually measured as the ratio of the Energy per Bit to the Noise Spectral Density *(E_b/N_0).* Being a process of noise, additive noise has zero as its first moment and the color of the noise is shown by its second moment. From the several statistical models, commonly used noise process is the Additive White Gaussian Noise (AWGN); with Gaussian being its distribution and White means that the white light is also a noise process, hence, affects the signal. In this thesis, we will use AWGN as the noise process with zero-mean and variance of N_0 and will be represented in the following sections with *w[-]* in time domain and with *W[-]* in frequency domain.

2.4. Observation Equation

At the receiver, after removing the CP i.e. discarding the samples falling in the CP and symbol rate sampling, the received signal at the input of the Fast Fourier Transform (FFT) can be expressed as:

$$y[n] = \sum_{l=0}^{L-1} s[n-l]c[l] + w[n] \qquad (5)$$

Where, n=0, 1…, N-1. $w[-]$ is the zero mean complex additive white Gaussian noise with variance N_0.

The above input-output relationship is further evaluated and converted to a matrix form in order to apply a suitable channel estimation algorithm.

The input-output model of the signal can now be represented in a matrix form. The matrix form of the model makes it easier to represent the input-output relationship and use it for the further evaluation of the relationship in the channel estimation steps. As our problem belongs to the wider sparse representation problem, hence, according to the channel representation discussed in section 2.2, we can represent our observation model given in (5). Eq. (5) can be written in matrix form as:

$$\begin{bmatrix} y(0) \\ \vdots \\ y(N-1) \end{bmatrix} = \begin{bmatrix} s(0) & \cdots & s(-M+1) \\ \vdots & \ddots & \vdots \\ s(N-1) & \cdots & s(N-M) \end{bmatrix} \begin{bmatrix} c(0) \\ \vdots \\ c(M-1) \end{bmatrix} + \begin{bmatrix} w(0) \\ \vdots \\ w(N-1) \end{bmatrix} \qquad (6)$$

Which can be condensed to:

$$\boldsymbol{y} = \boldsymbol{Ac} + \boldsymbol{w} \qquad (7)$$

From eq. (7), it is known that the channel is sparse and the problem is to approximately calculate the received vector $\boldsymbol{y}$ in terms of a linear combination of a small number of columns from the matrix $\boldsymbol{A}$. In other words, we must find $\boldsymbol{c}$ in (6) such that $\boldsymbol{y} \approx \boldsymbol{Ac}$ [6]. The matrix $\boldsymbol{A}$ is called the dictionary matrix or in CS theory as the sensing matrix that satisfies some characteristic conditions.

3. Channel Estimation

3.1. Introduction

The problem of obtaining the transmitted symbols from the received demodulated symbols is known as equalization as explained in the previous sections. One of the widely used approach is to estimate the effect of the channel and revert them. This poses a challenging sub-problem of the channel estimation. The purpose of the channel estimation is to approximately compute the effect of the channel on the transmitted signal, other than the unaccountable environmental noise. The estimation of this effect on the transmitted signal is calculated in terms of the coefficients of the channel matrix as explained in the above sections or in terms of the related system functions. An accurate channel estimation is required at the receiver in order to properly equalize the received signal so that the transmitted data can be extracted.

The associations between these system functions that defines a sparse multipath channel [39] suggests that the estimation of any of these parameters are enough for finding the transmitted data at the receiver. The entries of a channel matrix don't say much about the identity of the channel due to the limited bandwidth [40], hence, a model is defined that outlines the wireless channel and the estimation of model's parameters is carried out at the receiver to equalize the channels effect. The use of pilot symbols – as discussed in the previous sections – for channel estimation purposes is known to be the pilot-aided channel estimation. Many classical algorithms are used under the above mentioned settings.

3.2. Classical Algorithms

There are a number of channel estimation techniques used in wireless communication systems. Most of these techniques make use of the pilot symbols that are transmitted with the transmitted data. With these pilot symbols, some of the channel coefficients are calculated and then the rest are estimated using classical estimation algorithms like LS or MMSE. A number of different pilot arrangements are used that have shown to have different efficiencies for different systems.

3.2.1. Linear Minimum Mean Square Error Estimator (LMMSE)

LMMSE is a special case of the Bayesian estimator MMSE that works on the principle of minimizing the Mean Square Error (MSE) of the estimated values of the dependent variable. MSE is usually the measurement of an estimator's quality in most of the estimators that operate by minimizing the MSE. MMSE is the type of estimator that uses the quadratic cost function, therefore, the posterior mean of the estimated parameter is required by MMSE that is burdensome to calculate. This makes LMMSE a better choice as they are very flexible and are the core of many popular estimators like Kalman Filters. [41]

Furthermore, channel estimation can be done in frequency domain or in time domain. At the receiver, after matched filtering and removing CP [36], either the channel estimation can be performed on that time-domain signal or the signal can be passed through the FFT block and then the estimation is performed on the frequency domain signal at the output of the FFT block. But, the use of LMMSE for the bigger sample space gets prohibited and hence cannot be used as a channel estimator. Yet, it can be used as a secondary algorithm that assists the major, less complex channel estimator e.g. to estimate the values of the received pilot symbols. The use of LMMSE is the same as mentioned and is explained in the subsequent sections.

3.3. Matching Pursuit (MP) Algorithm:

As the fact has been established in the previous sections that the problem under discussion can be viewed as the sparse representation problem so the MP algorithm has proven to be a suboptimal solution to this problem [6].

3.3.1. MP as Channel Estimator:

MP is a greedy algorithm that works by selecting the waveform from dictionary matrix $\boldsymbol{A} = [a_1, a_2, \dots, a_M]$, at each iteration, that best matches the approximate part $b_0 = b$ of the signal and is denoted as a_{l_1}. The residual b_1 is obtained by negating the projection of b_0 in this direction from b_0. Again, the best aligned column a_{l_2} of $\boldsymbol{A}$ with b_1 is found and so is its residual. The algorithm continues sequentially in the same manner until a stopping criteria is met. The stopping criteria used for this study is the when the residual gets really small i.e. $\|b_p\| < \epsilon$ for some constant ϵ and iteration p. MP algorithm in its most elementary arrangement [42] is applied to our problem of sparse channel estimation using the observation model given in (7).

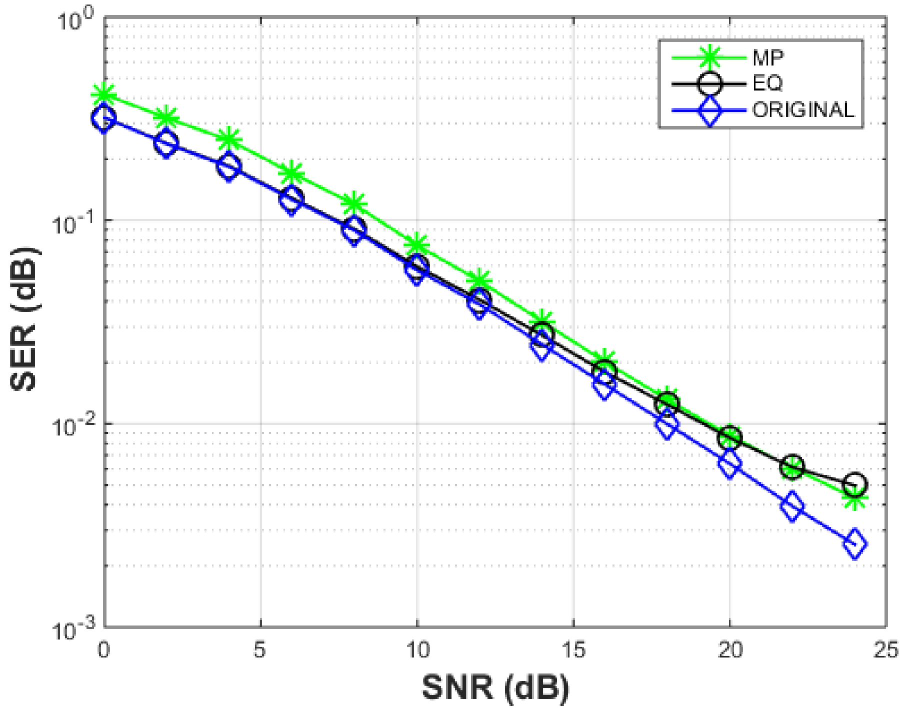


Figure 8: Performance comparison for MP estimate, original and equalized channel ($\rho = \Upsilon = 8$)

Figure 8 gives a comparison between the SER performance against multiple SNR values for MP estimate, original channel and its equivalent representation. It can be seen that MP performs quite well as a sparse channel estimator.

Effect of ρ

Keeping the same pilot spacing Υ and changing the resolution ρ, we observe that the signal is expanded in the time-domain and the number of columns of matrix $\boldsymbol{A}$ are increased, for MP algorithm. This increase in the number of atoms, increases the complexity of the algorithm. Figure 9 shows the symbol error rate (SER) comparison for the different values of SNR with $\rho = 20$ and $\Upsilon = 8$.

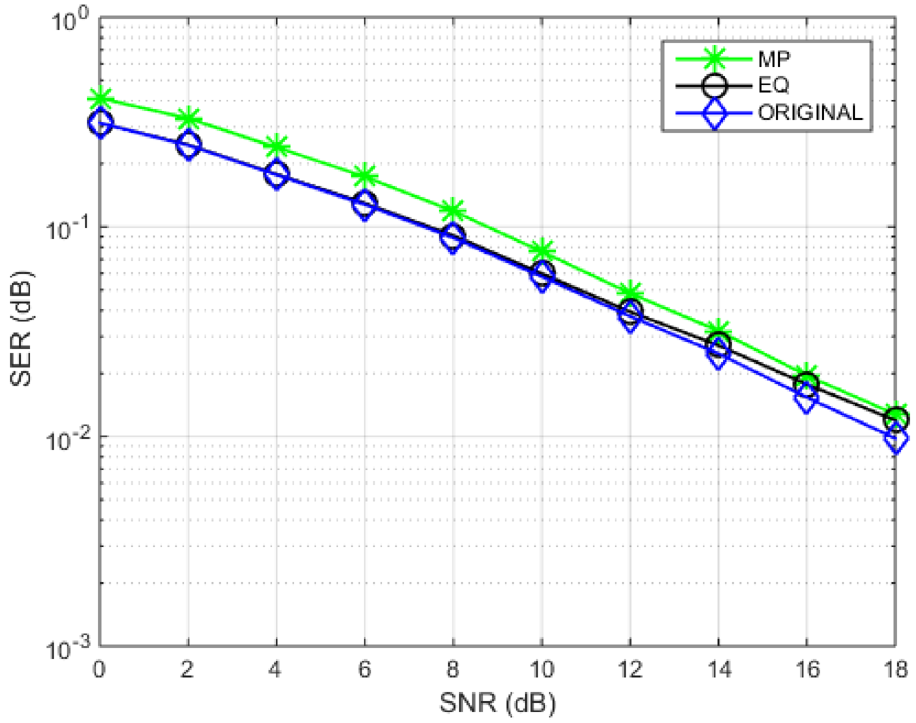


Figure 9: Performance comparison for MP estimate, original and equalized channel ($\rho = 20, \Upsilon = 8$)

Effect of Υ

Now if we keep the resolution constant and compare the efficiency of MP for different values of pilot spacing Υ, we will observe that with the increase in the spacing between pilots i.e. less number of pilots being used, the efficiency of MP is decreased. Efficiency increases with the increased number of pilots in the system and that is quite obvious for MP being used as an estimation algorithm. If the algorithm is already performing sub-optimally, then increase in the pilot symbols will not make much difference. Fig. 10 confirms it when compared with fig. 9.

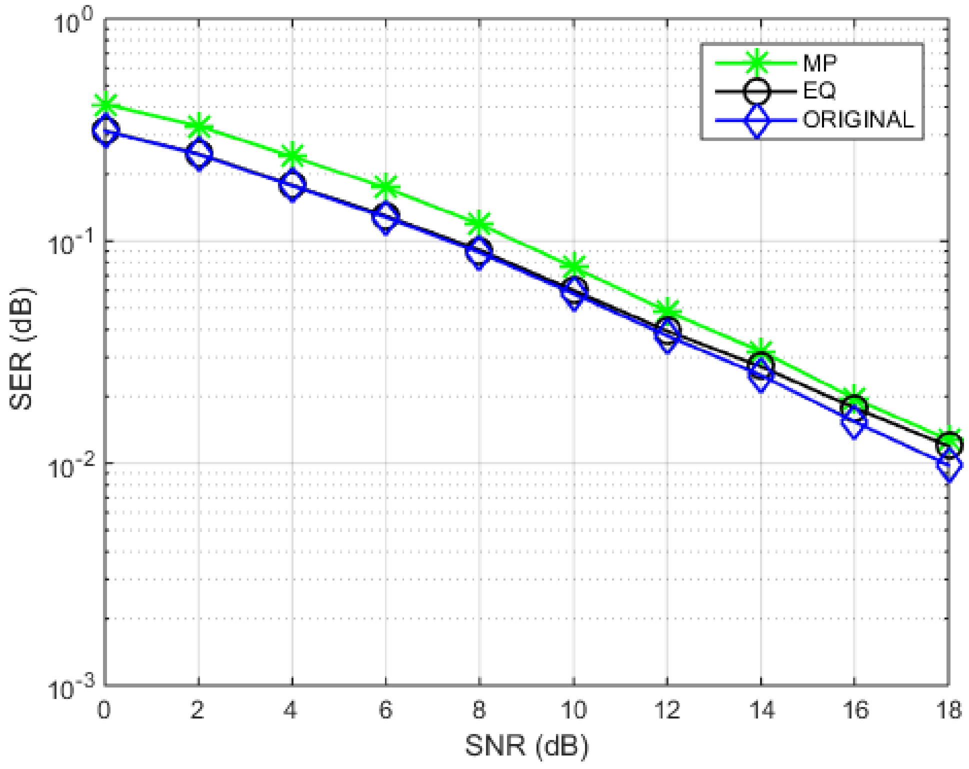


Figure 10: Performance comparison for MP estimate, original and equalized channel ($\rho = 20, \Upsilon = 4$)

4. Artificial Neural Network (ANN) Based Channel Estimation

4.1. ANN Basics

According to the basic definition, ANNs are comprised of a number of extremely linked, adaptive and simple groups of elements that are capable of exceptionally complex and parallel computations for data processing and artificial intelligence (AI) [43]. ANNs are inspired from the actual structures of the biological neurons and their functionality and construction can be seen in the modern computing like AI. The structure of biological neuron is quite simple, having a cell body containing a nucleus that acts as the command center, axons that connects the body part to the synapses and dendrites that act as the transmitters for the neuron. Human brain consists of a large number of such neurons building complexly interconnected nodes. An input is taken by these nodes from the other nodes or the external environment that is then independently processed in the same node causing an activation to produce output that triggers a response to the next layer of nodes or to an external output [44]. Fig. 11 shows the basic structure of the biological neuron [45].

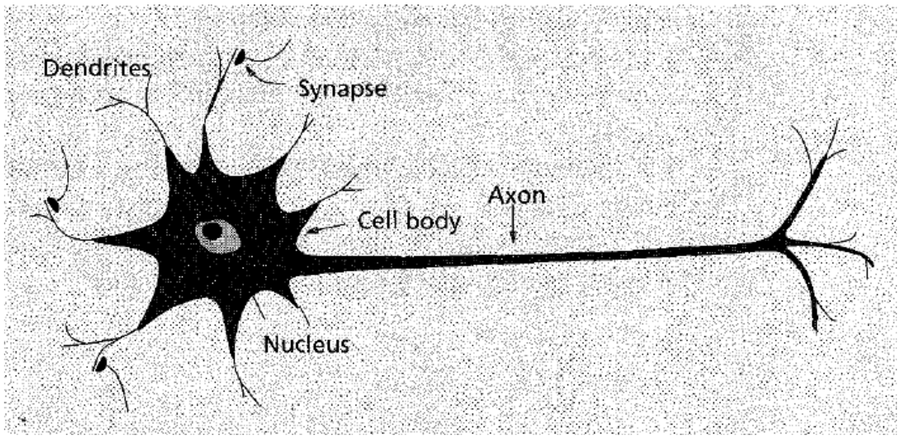


Figure 11: Basic Structure of Biological Neuron

ANN structure includes three types of layer levels. First is the input layer through which the ANN takes the input, next is the hidden layer where the inputs are actually processed and ANN gives the output through the next in line output layer. The input layer isn't considered to be a part of the layer structure because it doesn't perform any computations and hence, have no neurons. Hidden layer level may or may not have multiple number of layers and each hidden layer with multiple neurons. The outputs are specified automatically according to the problem so, the only layer and its number of neurons that are required to be specified is the hidden layer. Fig. 12 specifies the structure of an ANN [46].

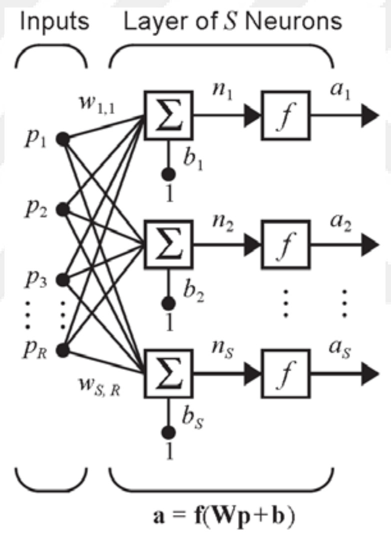


Figure 12: Structure of a simple Feed Forward ANN with R inputs, and S number of neurons in hidden layer

The number of layers and neurons in those layers specify the network topology and are important as the classification of ANNs are done on this basis which in turn defines the usage of the network. Due to the complex connections between the nodes and layers, ANN can learn and adapt to a set of sample data and can generalize a vast types of problems. The training of a network and its usage is a relatively easier task, however, selection of a network topology and its parameters is bigger chunk of the work.

4.1.1. Classification of ANNs

ANNs can be classified on the basis of two factors, its usage and the architecture as discussed in [43]. The network architecture consists of the input layer, hidden layer including its neurons and output layer together with their transfer or activation functions. A bias is sometimes also added to the total weighed sum of the network layer.

- *Architecture*

Classification of a network on the basis of its layers' structure, number of neurons, addition of bias and the connections between the layers comes under the umbrella of its architecture. For simple problems, a simple network with one layer might be enough, but as the problem gets complex, a relatively complex network might be required. A network comprising of more than one layer is known as a multi-layer network and is shown in fig. 13 [46].

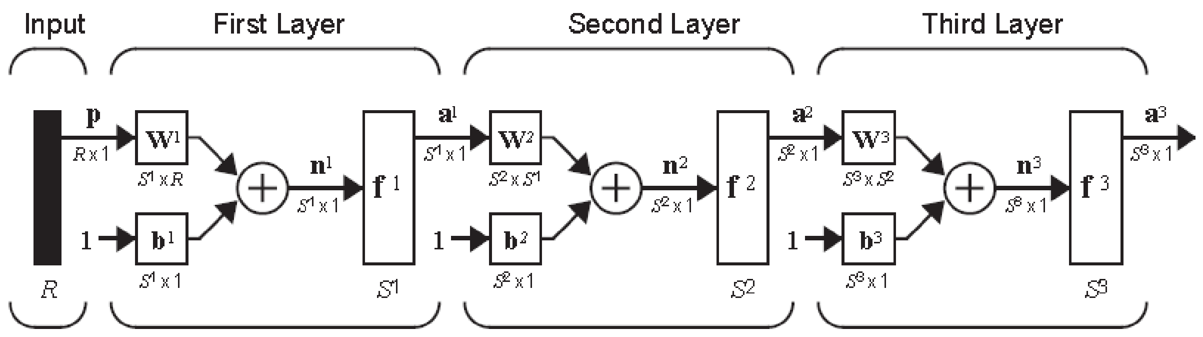


Figure 13: A Multi-Layer Feed Forward Network with 3 hidden layers

Further difference in the network can be caused by the backward connections from one layer to the previous ones. A network is called recurrent network if it has a feedback connection from its output to the input of the previous layer and are potentially more powerful than the FFNN [46].

Every layer is connected to the other layer via a weight value and each layers has its own bias value as well. These weights and bias value are updated according to the learning rule that is used to train the network. The functions that define the relationship between the input and outputs of a layer are called transfer functions and any transfer functions from many given functions can be used according to the problem's requirement. Most commonly used transfer functions are Linear, Hyperbolic Tangent Sigmoid and Log Sigmoid functions.

- *Usability*

The other classification factor is the usage of the network. ANNs are used for a vast types of problems including classification, pattern recognition, function fitting, time-series analysis, data reduction, prediction and control etc. Several types of ANN architectures have been proposed in order to solve these problems. Some of them are SOMs, Kohonen network, Hopfield network, back propagation multi-layer perceptron [47].

4.1.2. Network Learning

There are two types of learning, namely Supervised and Unsupervised learning. In unsupervised learning, the network is not presented by a training sequence, instead it is given a dataset and the network clusters the data into different classes. The method in which a learning algorithm is used to adjust network parameters according to the given training set, is called supervised learning. Supervised learning problems are further classified into two categories, regression and classification problems. In classification, the network is used to map the input vector to one of the discrete output values. As the name suggests, it learns to classify that which specific output class the input vector belongs to. However, in regression, the network maps the input vector to a continuous output, meaning it maps the input variables to some continuous function.

Before the network can be used to solve any given problems, it has first to be trained by giving a sample data on which the network adjusts its weights and biases. The training is done by giving the network a training set of the data and a target set. Network learns on the base of the training set and adjusts its parameters so that it can map the training sequence to the given target sequence. Target sequence is the required output we expect from our network after it is given a new set of data to generalize. During the training part, network learning is carried out i.e. network adjusts its parameters according to specific set of rule. These set of rules are defined by different specified Training Algorithms. Different training algorithms are used for different set of problems and for different network architectures. The problem of channel estimation comes under the umbrella of regression, and Resilient Backpropagation (Rprop) algorithm is used as the training function.

4.2. Multilayer Perceptron (MLP)

In supervised learning standard, network is provided by a required output *d* for each input pattern. During the learning process, the output *y* generated by the network may not be equal to the required output *d.* The difference between these network output and the desired output gives the error i.e. $e = d - y$. The concept of perceptron is to use this error to readjust the layer weights and biases so that this error can be minimized [45]. A network with this error correction rule and having more than one layer, is known as a Multilayer Perceptron (MLP). An MLP is a FFNN with having multiple layers and an error back-propagation rule. The network learning is performed only when the network makes an error [45]. Due to its simplicity and better error correction performance, MLP combines with Rprop algorithm, is our first choice to use as the network architecture for the channel estimation.

4.2.1. MLP Training with Resilient Backpropagation (Rprop)

The resilient backpropagation learning algorithm is a gradient-based batch update setup that works on the basis of Manhattan Update rule. The need of Rprop arises from the fact that the ANNs use sigmoid functions in the layers that work on the basis of slope approaching zero when input gets large. This causes problem when we try to train the ANNs with Steepest Descent algorithm. Rprop solves these issues with the magnitude of the direction and, hence, only takes in account the sign of the derivative.

After the cascade-correlation algorithm and Levenberg-Marquardt (LM) algorithm, Rprop is the fastest weights update algorithm. LM being fast takes more memory and this prohibits to use LM in our case of large network. As we'll see in the next chapters that the size of the network is directly proportional to the resolution ρ and the number of subcarriers N used for our communication system model because of sparsity, so the use of Rprop is the best choice as the training algorithm for our network.

4.3. Proposed ANN for Channel Estimation

As discussed in the previous section that the type of network used in this study is the multilayer perceptron with Rprop algorithm as the training function. Further details about the complete ANN models will be discussed in the subsequent sections. The simulations, data generation and all other functions being performed are done in *Matlab®* environment. In this chapter, the name of those variables will be defined and used for notational simplicity.

4.3.1. ANN Model

After the decision about the network type and the training function have been made, the other parameters of the network architecture are to be found and the complete model is then devised using those parameters. The most important steps making use of afore-mentioned parameters are given as follows:

- Generation of sample space
    - Data preparation
        - Training set
        - Target set
        - Ratio of training set to be used as train, validation and test subsets
        - Normalization of data vectors with zero mean and unit variance
- Layers transfer functions
- Find optimal initial values for
    - Weights' matrix
    - Number of hidden layers
        - Number of neurons in each hidden layer
- Training and optimization of the network
- Use of the optimized network

After a hit and trial method and training a number of NNs with different initial conditions, several different number of neurons and hidden layers, we select the NN with the best performance. This is further discussed in the following subsection.

Please note that there is no proved way of finding an optimized NN for a specific problem. However, by optimization here we mean to select the best NN on the base of its performance, among several trials with different initial conditions.

4.3.2. Generation of Sample Space and Preparation of Inputs

Before creating the required neural network, it is important that we have the sample data in the form as required by the input of the network. The input to the network will be a collection of pilot symbols of the received OFDM signals generated by the models as discussed in the previous chapters. The target vectors will be a collection of the known equivalent channel vectors c specific to the received pilots. Each received signal in the training set have a corresponding target set of the channel coefficients specific to its received signal.

For the generation of the training and target dataset, the observation model given in (7) is used. The training set used in our simulations is made up of multiple parts, each part contains multiple samples (i.e. 200) of the received pilot symbols collected for a specific SNR level. Similarly, the next parts of the training dataset are produced for other SNR values and all the parts are collected in one matrix. The SNR values used for the production of training and target datasets are from $SNR = 11$ to $SNR = 30$ with the step of size 3. This completes our training dataset. The target dataset is the collection of the equivalent discrete-time channel vector $\boldsymbol{h}[\boldsymbol{n}]$ specific to the received pilot symbols in the training set. The total number of samples in the sample space of training and target sets are: $\{\# \, of \; samples \; for \; each \; SNR \; value \; (500) \times \# \, of \; SNR \; values \; (7) = 3500\}$.

Training method plays a vital role in NN performance and the data used for the training impacts the training significantly. This gives basis of the fact that we trained our NN with signals for different SNR levels so that it can learn the impact of SNR on the system; and can use it to effectively predict the channels with differing SNR levels.

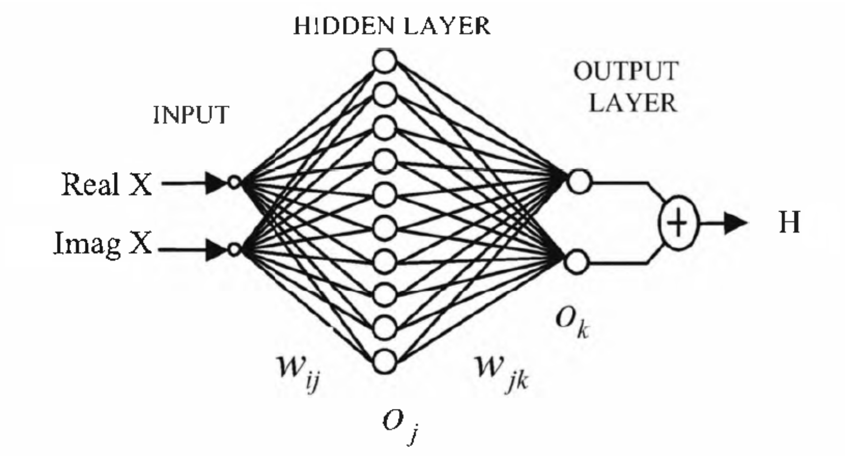


Figure 14: MLP-ANN model used with separate Real and Imaginary Inputs

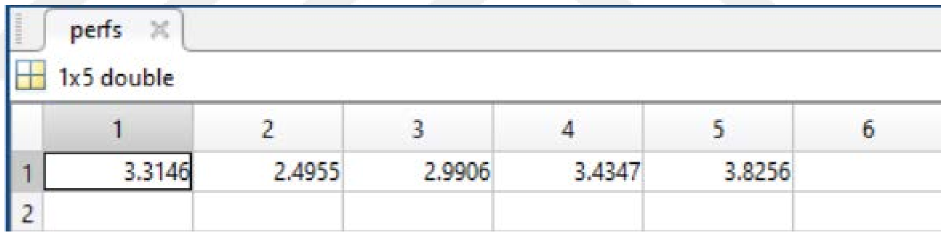
A fact to keep in mind at this point is that the network does not take arbitrary values as inputs and only takes real valued sample space. Therefore, we separate the samples of sample space into real and imaginary parts. Separating the real and imaginary values of input and target vector and vertically concatenating them into a training and target sequences solves the above hurdle and network now have two inputs with the vector from real values on top and the one from imaginary on the bottom. Let us denote the matrix consisting of the samples space for training and target set as $\boldsymbol{X}$ and $\boldsymbol{H}$ respectively.

The output of the network will also be the same and the real and imaginary parts are then joined at the output by doing the reverse process that was done at the input. Therefore, the model of our network looks like as shown in fig. 14.

The training and target sets are then divided into three subsets with train subset to train the network, validation subset to check the how well network performs on generalization and a test set to test the performance of the network on unseen data. The actual training and target sets are divided, however, the indices of the sample space are made fixed for subsequent subsets. The indices are get by using the dividerand() function of Matlab® by giving it the ratios of the subsets. The indices gotten from this function are then used to update the network parameter related to the index values of the subsets, which will be discussed in the following sections.

4.3.3. Network Creation and Training

Next step is to create a network with the above given functions and an optimal set of parameter such as initial values of weight and bias matrices, and hidden layers' matrix $\boldsymbol{O}$. $\boldsymbol{O}$ contains a combination of different number of hidden layers with different number of neurons for each layer. The optimal parameters are obtained by training the network in a nested loop with the outer loop running for each combination of hidden layers $\boldsymbol{O}$ and the number of neurons while the inner loop runs a number of times e.g. 5 for different random initial values, and the performances for each of these combinations stored in a vector. At the end of this nested loop, the performances are compared and the network with the best performance is used in estimation process. The performances of these network architecture are calculated using the *Matlab®'s perf()* function that calculates MSE for a set of input variable specified to a network. The network with the lowest value is the network with optimal parameters under the current settings.


perfs

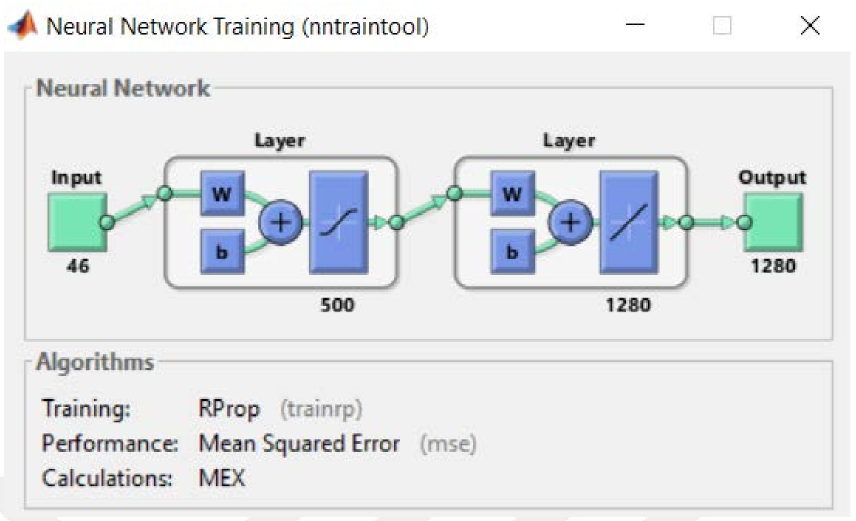
1x5 double

	1	2	3	4	5	6
1	3.3146	2.4955	2.9906	3.4347	3.8256	
2						

Figure 15: Performance vector for multiple trainings

Fig. 15 gives an insight to the performance vector when a network is trained five times. It can be seen that each run of the training produced different performance. The network with performance in index 2 was used. As it can be seen in Fig. 16, that the default transfer functions for each layers are used. The transfer function for hidden layers is *tansig*.

The final network model for training is shown in the following figure:



Training: RProp (trainrp)
Performance: Mean Squared Error (mse)
Calculations: MEX

Figure 16: Neural Network model after training (nntraintool)

The neural network used for the channel estimation uses one hidden layer with 500 neurons and one output layer. The size of the output layer of the neural network is directly related to the resolution factor ρ while the number of input connections are dependent upon the pilot spacing Υ. Table 1 shows the final neural network parameters and functions used in training in reference to the model showed in fig. 16.

Table 1: ANN Parameters and Functions

Parameter	Value	Parameter	Value
# of outputs	1	**# of inputs**	1
# of hidden layers	1	**# of hidden neurons**	500
Input size	$N_p \times 2$	**# of samples**	4000
# of layers	1	**Bias connect**	$[1;\ 1]$
Input connect	$[1;\ 0]$	**Layer connect**	$[0\ \ 0;\ 1\ \ 0]$
Output connect	$[0\ \ 1]$	**Adapt function**	*'defaultderiv'*
Divide function	*'dividerand'*	**Divide Parameters**	$[0.5\ \ 0.2\ \ 0.3]$
Performance Func.	*'MSE'*	**Train function**	*'trainrp'*

4.4. Estimated Channel:

After the network has been trained with the training sequence, it is now ready to be used as a channel estimator. The estimated channel is then equalized and the symbol error rate (SER) is calculated for several values of SNR. Fig. 17 shows the SER-SNR plot for the system using the NN based estimator. The estimator has performed well for the overall system performance and SER is not far from the equivalent channel vector in performance.

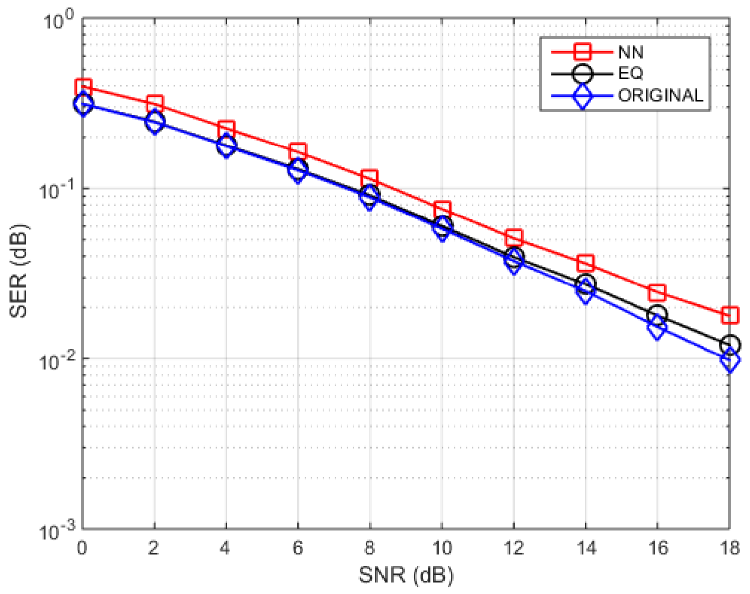


Figure 17: Performance comparison for ANN estimate, original and equalized channel (ρ=8, ϒ=8)

5. Simulations Results

Simulations for a number of different OFDM parameters, MP algorithm and with different network topologies and architectures were carried out using *Matlab®'s* simulation environment. A computer with the following specification was used for running the simulations (table 2).

Table 2: PC Resources and Capabilities

Item	Value
System Type	X64-based PC
Processor	Intel core i7-4500U CPU @ 1.81 GHz
Total Physical Memory	16 GB
Total Virtual Memory	25GB
Parallel Computing Capability	Yes (NVidia GPU)

Following table shows the parameters used for the communication system under consideration in this study – unless specified otherwise.

Table 3: Communication System's Parameters

Parameter	Value
No. of sub-bands (N)	512
No. of used sub-bands (K)	180 * N/256;
Subcarrier spacing (Δf) – LTE	15*KHz
Bandwidth	10*MHz * N/1024
Carrier frequency (f_c)	2.5*GHz
Constellation type	QPSK
No. of multipath	3

5.1. Performance Analysis of ANN and MP Algorithms

The performance based comparisons are done in this chapter for both channel estimators discussed in the preceding sections. Monte-Carlo simulations were done on both methods and the comparative figures are drawn from such simulations. We compare the efficiencies in terms of SER and SNR values and the simulation results show the gain and loss in terms of SER. The channel estimators are also compared in terms of the factors that have been discussed in the previous sections that affect the efficiency of the estimators.

Fig. 18 is SER-SNR comparison of original & equivalent channel representations, MP and NN algorithms. It can be seen here that the SER performance of the system using NN estimator is equal to the MP estimator at the SNR of 12dB. The NN performs better at the lower values of SNR than the MP algorithm; where the noise factor in the system is quite large. However, MP algorithm has better performance for the higher values of SNR.

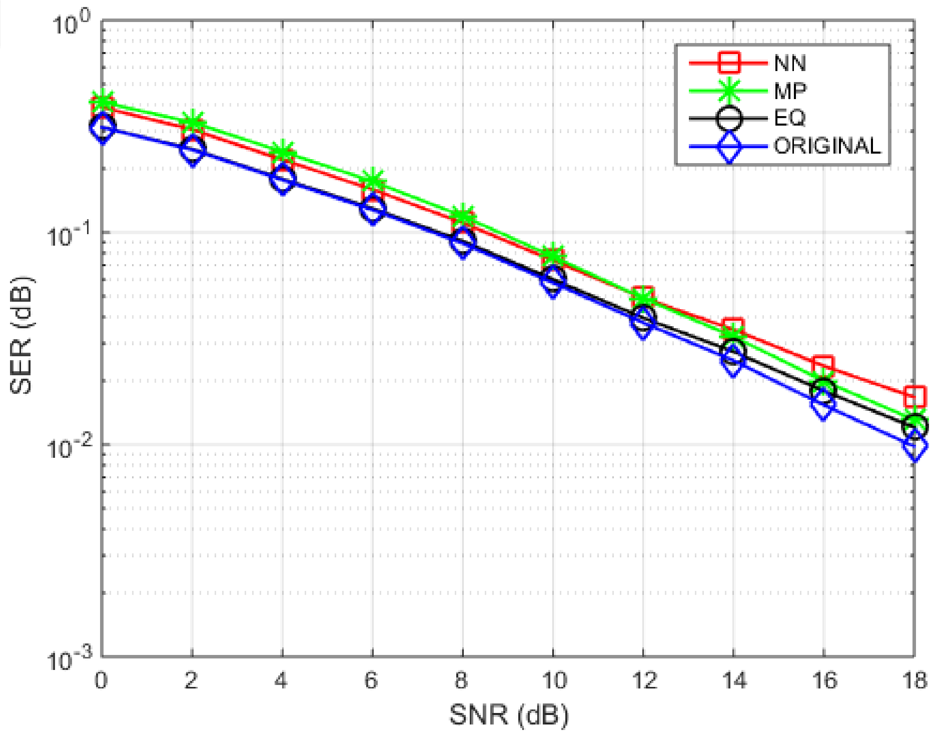


Figure 18: Performance comparison for ANN & MP estimate, original and equalized channel (ρ=8, ϒ=8)

These results can further be improved by applying linear MMSE for estimating the pilot symbols; for the received signals that are used for the training of the ANN algorithm. Until this point, the NN was trained with the received pilot symbols as the training set and the actual equivalent channel vector as the target sequence. However, now we show that the LMMSE channel estimator can be used to estimate the channel for the received pilot symbols which in turn become the input of the NN for the training period. This will train the NN in a way that the inputs and the outputs will be closer to each other and will in turn enhance the estimation capability of the NN. The target sequence for ANN training will be kept same as before.

The required information for the application of LMMSE are the Noise power N_0 and the correlation matrix $\boldsymbol{R}$ of the channel. With these requirements discussed in above sections, LMMSE can now be applied to the pilot symbols. The input matrix for the input of neural network $\boldsymbol{X}$ is now modified and more refined values are provided to the NN. The improvement in the performance of NN algorithm is considerable and is shown in fig. 19. With the SER improvement, the SNR cutoff point for ANN and MP has also moved from $SNR = 12$ to $SNR = 14$.

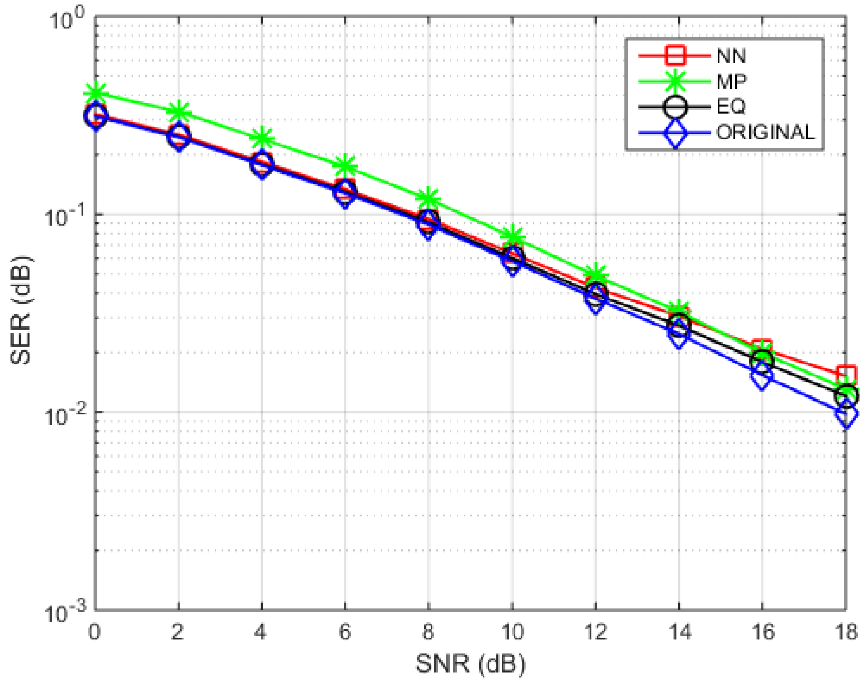


Figure 19: Performance comparison for LMMSE-ANN & MP estimate, original and equalized channel ($\rho = \Upsilon = 8$)

The above figure forms the major contribution of this work. It can be seen that by using LMMSE, the ANN based channel estimation produce better SER performance of the system.

5.1.1. Effect of ρ and Υ values

Keeping the updates in the settings mentioned above, the size of pilot spacing Υ and ρ will also affect the performance of both algorithms. Increasing ρ to 20 and keeping $\Upsilon = 8$; we show in fig. 20 that – though the difference is small – the performance for ANN has dropped. NN becomes sensitive to the lower SNR values and now equals to the performance of MP at SNR=10.

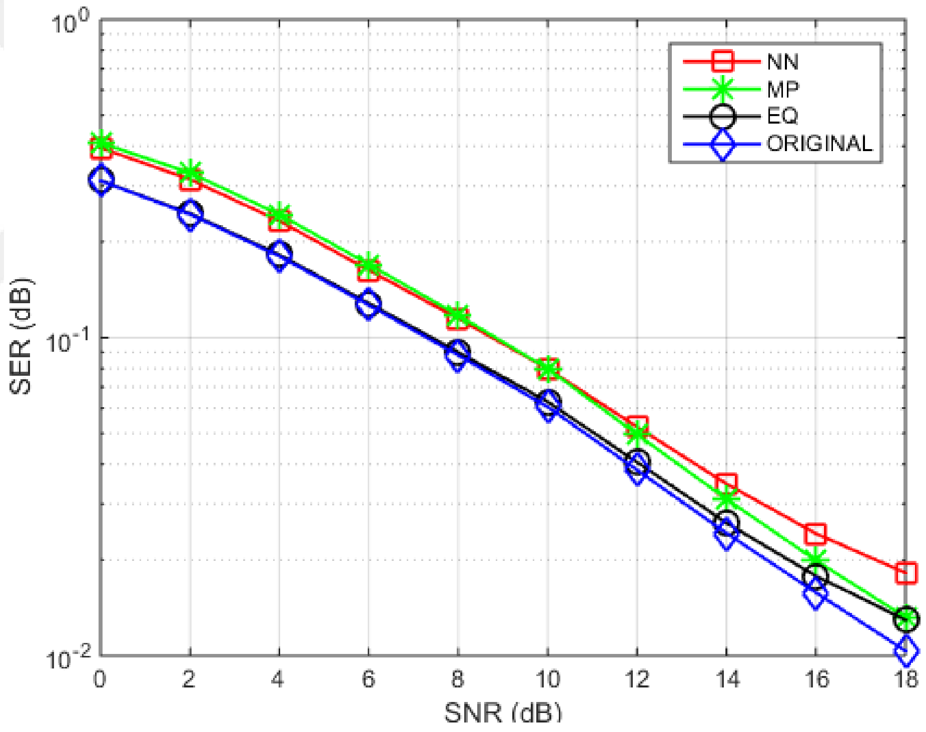


Figure 20: Performance comparison for ANN & MP estimate, original and equalized channel (ρ=20, ϒ=8)

Using the same parameters as in the fig. 20 but with LMMSE, the comparison is shown in fig. 21. If we relate fig. 19 with fig. 21, we see that even with LMMSE, increasing the resolution decreases ANN's performance slightly. One interesting fact that needs a mention here is that performance of ANNs is quite sensitive to the change in its input in terms of system performance. The reason being, this network architecture was carefully selected after intense simulations and is specific to the problem size and type. However, the complexity of the system is not much affected by these changes. On the other hand, the overall performance in terms of SER is improved for all the systems as shown in the following figure.

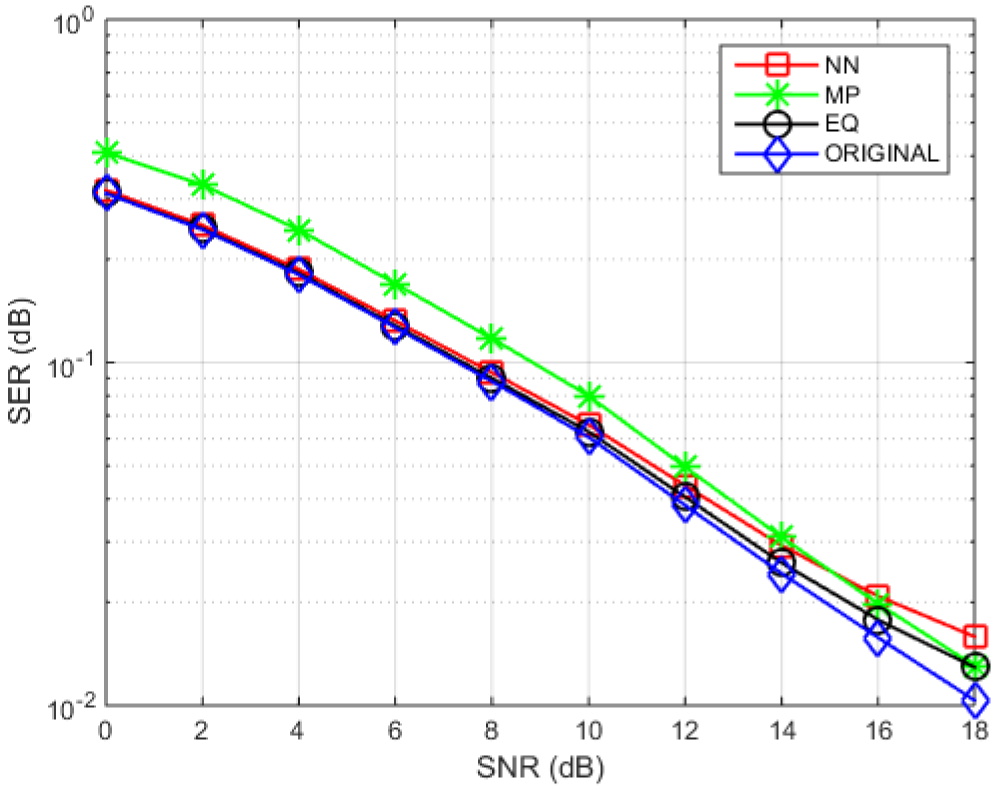


Figure 21: Performance comparison for LMMSE-ANN & MP estimate, original and equalized channel (ρ=40, ϒ=8)

To further the discussion, lets now change ρ to 16 and Υ to 16; the effect on the performance of both algorithms can be observed in Fig. 22. From fig. 22 it is clear that the pilot spacing cannot be increased from a certain point. This decreases the total number of pilot symbols and the overall efficiency of the system becomes unacceptable. Most acceptable pilot spacing is $\Upsilon = 8$, after that the system performs poorly regardless the algorithm used for estimation. However, we see that the proposed NN algorithm performs a little better under tougher conditions.

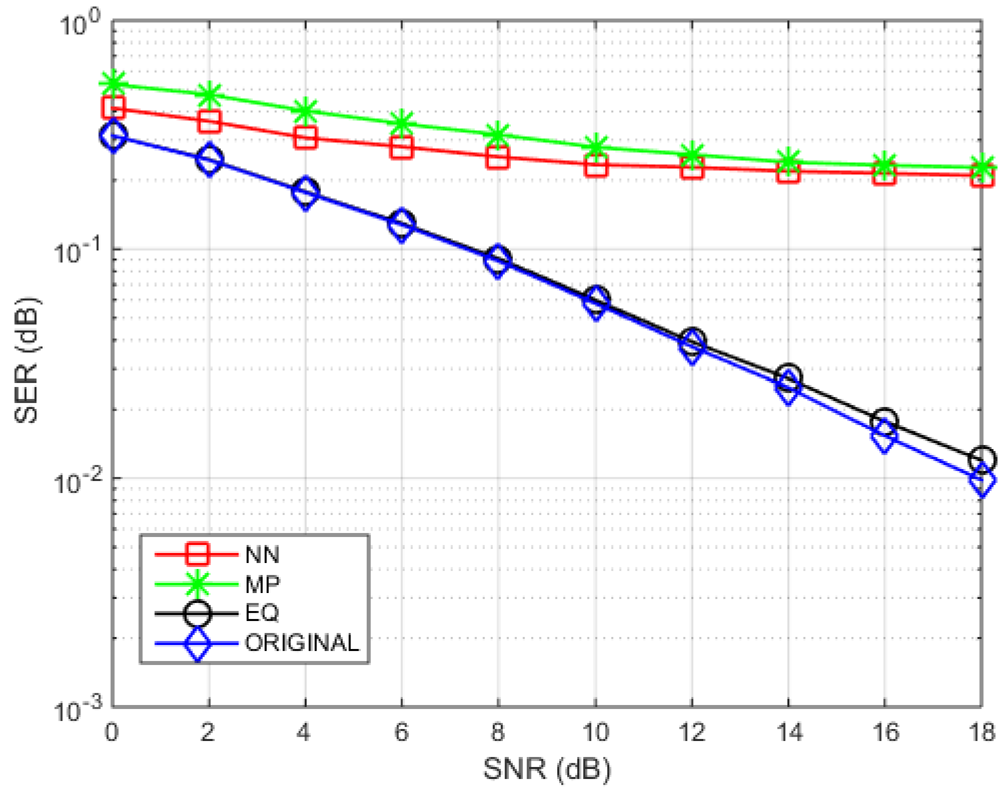


Figure 22: Performance comparison for ANN & MP estimate, original and equalized channel (ρ=16, Υ=16)

After the SER comparisons of the systems being discussed have been debated above, following figures will compare the mean-squared error (MSE) of these systems. The following fig. 23 is the MSE comparison between the MP and ANN based systems for $\rho = 40$ and $\Upsilon = 8$. Here we can see that the ANN based systems perform a lot better even in terms of MSE as compared to the MP based systems. ANN based system performs a lot better than MP based system in the presence of smaller SNR values up to the point where $SNR = 16$. The system parameters for this MSE comparison are same as system parameters used in fig. 21. Fig. 23 gives the MSE comparison between MP and ANN based systems for $\rho = 40$ and $\Upsilon = 8$. It can be observed in the following figures that the cutoff point between the performances of ANN and MP based algorithms is at $SNR = 14$ and this MSE value is better for greater value of ρ – even though the difference is small.

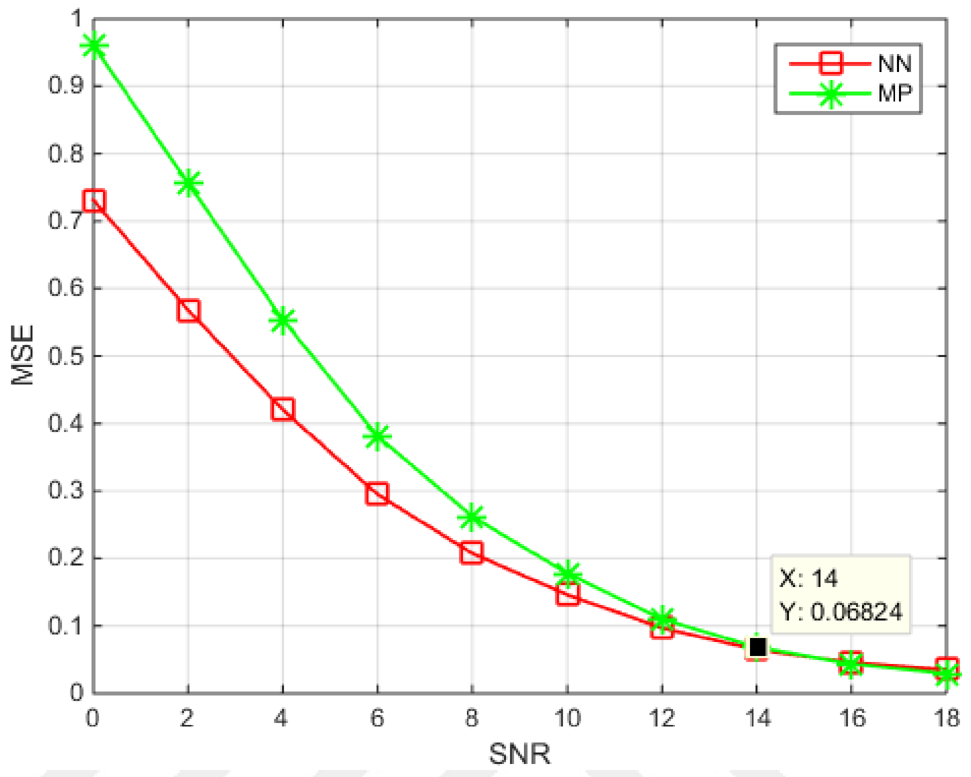


Figure 23: MSE vs SNR comparison of MP and ANN based systems ($\rho = 40, \Upsilon = 8$)

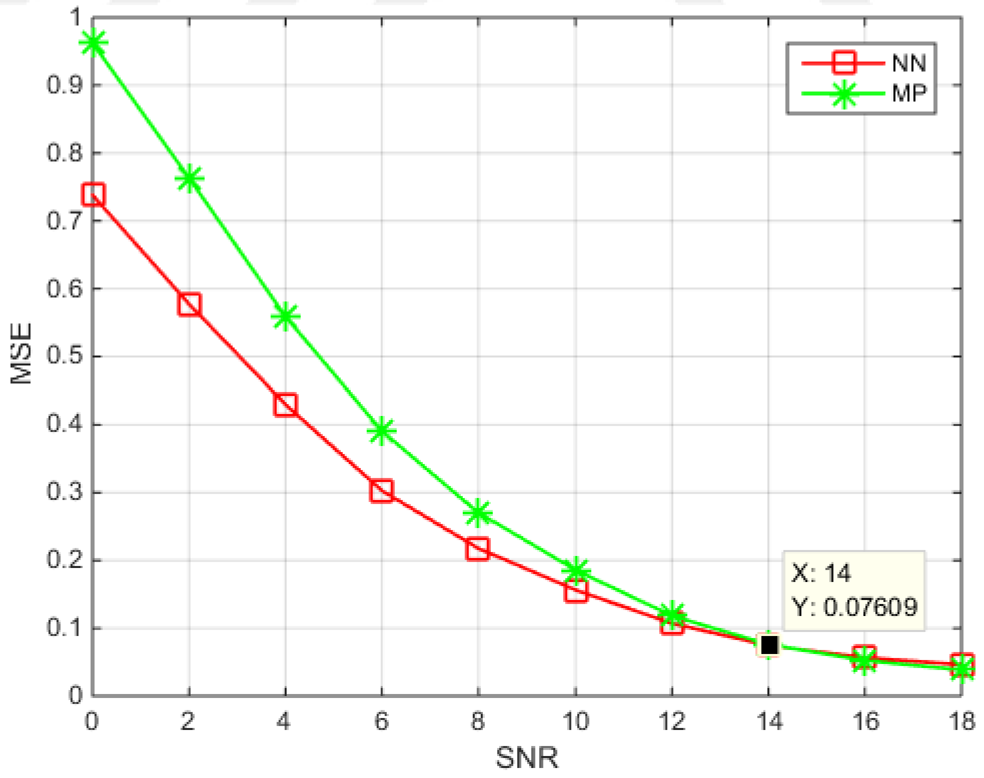


Figure 24: MSE vs SNR comparison of MP and ANN based systems ($\rho = 2, \Upsilon = 8$)

5.2. Complexity Analysis

The channel estimation algorithms can become computationally inefficient as a matrix inversion is required in most of the classical algorithms. The size of this matrix increases with the increased number of inputs and so does the overhead due to the inversion. ANNs, however, do not require any matrix inversion, therefore, are proven to be less complex. Computational complexity is a separate field of study altogether that deals with the classification of computational problems according to their difficulty and relate these complexity classes with each other. In other words, computational complexity theory helps in determining the practical limits on what computers are capable of and what they're not.

A problem is supposed as difficult if it requires significant resources to be solved no matter what algorithm is used. The theory discusses different problems on the basis of their difficulty by giving models of computations, and measures the resources required to solve these problems – like storage and time. Several classes of complexity are used to classify problems on the basis of above facts and are supposed to have different resource requirements. Non-deterministic Polynomial (NP) is one of the classes of computational complexity with variates as NP-hard and NP-complete. A sparse approximation of a signal is an important issue in many fields today and many algorithms have been proposed for a good sparse approximation in polynomial time like MP. But to design an algorithm that gives a good approximation performance and is flexible in complexity for the large signal dimensions. Also, the universal problem of finding the best m-term estimate is non-deterministic polynomial (NP-Complete) [27].

The computational complexity of algorithm is defined as the number of steps/calculations it has to make in order to solve a problem given an amount of space and time. The space complexity is ignored when comparing the computational complexity of algorithms and time complexity is used. The complexity of an algorithm is often expressed using big O notation.

5.2.1. Big-Oh Notation

Big O notation, also called Landau's symbol, is a representation used in complexity theory, computer science, and mathematics to label the asymptotic conduct of functions. Fundamentally, it expresses how fast a function grows or declines. The letter O is used because the rate of growth of a function is also called its order. The big O notation is used to classify different algorithms by how the algorithms react to the change in their input size. For example, how the running time of an algorithm changes with the increase in the problem size. There is no mechanical way of finding the Big O of an algorithm, however, there are some agreed upon methods that define the required resources for most of the calculations performed in the algorithms.

The Big O notation is useful in the analysis of the programs' efficiency e.g. the amount of time or number of steps it takes for an algorithm to solve a problem of size n. If the time it takes is said to be for example $T(n) = 4n^2 - 2n + 2$, then the lower order terms are neglected as the n^2 term dominates when the problem size is increased sufficiently. The big O notation of this program is then said to be $T(n) \in O(n^2)$.

- *MP*

MP algorithm has shown to have lower computational complexity than its other variants like OMP and GP but it also suffers from a higher error floor [27]. The computational bottleneck for MP is faced at the maximization step where a matrix multiplication is required [28]. The computational complexity of MP algorithm in its most basic form is compared with its other variant in the following table [27].

Table 4: Complexity order of greedy (MP) algorithms for a given iteration

	Step	**MP**	**OMP**	**GP**
Selection	Correlations	$\rho * N^2$	$\rho * N^2$	$\rho * N^2$
	Maximum	$\rho * N$	$\rho * N$	$\rho * N$
Update	Gram Matrix	0	$i * N$	0
	Coefficients	0	i^2	$i * N$
	Residual	N	$i * N$	N

From the above table it can be seen that the computational complexity of MP algorithm is:

$$T(n) = \rho \times N^2 + \rho \times N + N \tag{8}$$

for each iteration and N denotes the total number of columns in the dictionary matrix $\boldsymbol{A}$.

Consequently,

$$T(n) \in O(N^2) \tag{9}$$

Hence, it is identified that the computational time of MP algorithm increases exponentially with the increase in the input size.

- *ANN*

ANNs on the other hand are relatively hot topic for researchers as they pose some challenges when it comes to their optimization or the most optimized algorithm for a given problem. This topic has long been debated that which computational complexity class they belong to and what class of problems can be solved using ANNs. Just like any other algorithm, the computational complexity of a NN with predefined set of weights, initial condition, number of layers and number of neurons in those layers, is dependent on the size of the network input. When we talk about the input size of a NN in terms of computational complexity, it is dependent on a number of factors, mentioned bellow:

- Total number of layers $\boldsymbol{O}$
- Total number of neurons in each layer $\boldsymbol{o}$
- Size of the input
- Number of epochs
- Size of the sample space i.e. train/target pairs

The important parameters when comparing different networks are the size, depth and weight of the network. They are defined as the number of neurons, distance from an input neuron to the output neuron and the sum of all the weight values in the network. A theorem mentioned in [48] suggests that the polynomial size threshold networks with size s can be represented in $O(s\ log\ s)$. Similarly, the author of [49] argues that the NN is powerful enough to solve the class of problems known as NP-hard. This is the base of the fact that the sparse channel estimation problem can be effectively solved by NNs and with lower complexity than the MP algorithm.

The total number of connections – including weights and biases for all nodes – in a NN architecture can be found by:

$$\# \, connections = (o_1 \times o_2 \times \dots) \times 2 + o_1 + o_2 + \cdots \tag{10}$$

Where o_i denotes the number of neurons in $i-th$ layer and the addition of layers' nodes in the second part of (10) is due to the bias values for each node. Total number of weights for our architecture is calculated as:

$$\# \, connections = (500 \times 1280) \times 2 + 500 + 1280$$

$$= 1{,}281{,}780$$

The change in the output layer neurons is dependent on ρ and hence, change in resolution makes NN more complex and most probably to make them less efficient – as discussed in section 5.1.1. The size of NN input during training and testing phases, is dependent on pilot spacing but change in the input layer connections doesn't change the total number of neurons in the architecture; so the effect on complexity is not much but it can still degrade its predicting power.

Keeping the fact in mind that the training of the ANN is done offline, makes it easier to calculate the computational complexity of the ANN for the testing phase. In other words, the complexity of the ANN for the testing phase is just a bunch of multiplications and additions i.e. its linear to the size of the input. That been established, the complexity is then the total number of connections within the NN architecture as calculated above using (10).

Big O of a NN can be found and as mostly the operations for each iteration/epoch are counted for finding the polynomial of input space, we can collect all the operations to see how the NN reacts to the increasing input. Intensive simulation can be run to show that the time taken by NN to solve the discussed problem is dependent on total number of neurons and size of the sample space that actually define the input of a NN in terms of complexity. NN performs in liner time with respect to the input size for our problem. Though the optimization of NN is impossible until today, but, it doesn't have a computational bottleneck as opposed to MP algorithm as discussed above.

6. Conclusion

Two different algorithms have been used for the sparse channel estimation and compared in terms of the system's SER against several different SNR values. The simulation results have shown that the proposed ANN based estimation performs better in terms of SER for the smaller SNR values when it is trained with the LMMSE estimated data. MP algorithm have shown to have a little better performance for the larger values of SNR; as it would be expected of any estimation algorithm because the ratio of the noise is much smaller than the signal. However, for the smaller SNR values the noise factor is strong and the proposed algorithm has shown to be more affective. These claims are proved in Fig. 22 where it can also be observed that the NN estimation is now less sensitive to the SNR value of the system. If an optimized network can be found and trained for channel estimation, considering NNs' usefulness and power, it can be said that it will not give far better performance than MP in terms of BER but might as well be computationally a lot cheaper than the MP algorithm. Author aims to extend this idea of ANN based channel estimation to linear time varying Sparse channels as well. This study poses some further work as well; the optimization of NN for better communication system throughput and the way to compute the complexity of the optimized NN will help in designing faster and less complex systems.

7. References

[1] J.-J. Van de Beek, O. Edfors, M. Sandell, S. K. Wilson, and P. O. Borjesson, “On channel estimation in OFDM systems,” in *Vehicular Technology Conference, 1995 IEEE 45th*, 1995, vol. 2, pp. 815–819.

[2] S. Colieri, M. Ergen, A. Puri, and B. A, “A study of channel estimation in OFDM systems,” in *Vehicular Technology Conference, 2002. Proceedings. VTC 2002-Fall. 2002 IEEE 56th*, 2002, vol. 2, pp. 894–898 vol.2.

[3] T.-L. Liu, W.-H. Chung, S.-Y. Yuan, and S.-Y. Kuo, “On the Banded Approximation of the Channel Matrix for Mobile OFDM Systems,” *IEEE Trans. Veh. Technol.*, vol. 64, no. 8, pp. 3526–3535, Aug. 2015.

[4] S. Colieri, M. Ergen, A. Puri, and A. Bahai, “A study of channel estimation in OFDM systems,” in *Vehicular Technology Conference, 2002. Proceedings. VTC 2002-Fall. 2002 IEEE 56th*, 2002, vol. 2, pp. 894–898.

[5] M. Faran and P. Mor, “Comparison of Different Channel Estimation Techniques in OFDM Systems,” *Int. J. Innov. Technol. Explor. Eng. ISSN*, pp. 2278–3075.

[6] S. F. Cotter and B. D. Rao, “Sparse channel estimation via matching pursuit with application to equalization,” *IEEE Trans. Commun.*, vol. 50, no. 3, pp. 374–377, 2002.

[7] W. F. Schreiber, “Advanced television systems for terrestrial broadcasting: Some problems and some proposed solutions,” *Proc. IEEE*, vol. 83, no. 6, pp. 958–981, Jun. 1995.

[8] S. Ariyavisitakul, N. R. Sollenberger, and L. J. Greenstein, “Tap-selectable decision feedback equalization,” in *1997 IEEE International Conference on Communications, 1997. ICC ’97 Montreal, Towards the Knowledge Millennium*, 1997, vol. 3, pp. 1521–1526 vol.3.

[9] J. Homer, I. Mareels, R. R. Bitmead, B. Wahlberg, and A. Gustafsson, “LMS estimation via structural detection,” *IEEE Trans. Signal Process.*, vol. 46, no. 10, pp. 2651–2663, Oct. 1998.

[10] D. L. Donoho, “Compressed Sensing,” *IEEE Trans Inf Theor*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[11] G. Gui, Q. Wan, W. Peng, and F. Adachi, “Sparse Multipath Channel Estimation Using Compressive Sampling Matching Pursuit Algorithm,” *ArXiv10052270 Cs Math*, May 2010.

[12] D. Needell and J. A. Tropp, “CoSaMP: Iterative signal recovery from incomplete and inaccurate samples,” *Appl. Comput. Harmon. Anal.*, vol. 26, no. 3, pp. 301–321, May 2009.

[13] S. F. Cotter and B. D. Rao, “Matching pursuit based decision-feedback equalizers,” in *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100)*, 2000, vol. 5, pp. 2713–2716 vol.5.

[14] C. Çiflikli, A. T. Özşahin, and A. Ç. Yapici, “Artificial Neural Network Channel Estimation Based on Levenberg-Marquardt for OFDM Systems,” *Wirel. Pers. Commun.*, vol. 51, no. 2, pp. 221–229, Nov. 2008.

[15] B. K. Engiz, Ç. Kurnaz, and G. Kayhan, “Performance evaluation of ANN based channel interpolation for OFDM system,” in *Innovations in Intelligent Systems and Applications (INISTA), 2012 International Symposium on*, 2012, pp. 1–5.

[16] A. Omri, R. Bouallegue, R. Hamila, and M. Hasna, “Channel estimation for LTE Uplink system by Perceptron neural network,” *Int. J. Wirel. Mob. Netw.*, vol. 2, no. 3, pp. 155–165, Aug. 2010.

[17] K. Sharma and S. Varshney, “Artificial Neural Network Channel Estimation for OFDM System,” *Int. J. Electron. Comput. Sci. Eng. IJECSE ISSN*, pp. 2277–1956, 2012.

[18] N. Taşpinar and M. N. Seyman, “Back propagation neural network approach for channel estimation in OFDM system,” in *Wireless Communications, Networking and Information Security (WCNIS), 2010 IEEE International Conference on*, 2010, pp. 265–268.

[19] Ş. Şimşir and N. Taşpınar, “Channel estimation using neural network in Orthogonal Frequency Division Multiplexing-Interleave Division Multiple Access (OFDM-IDMA) system,” in *Telecommunications Symposium (ITS), 2014 International*, 2014, pp. 1–5.

[20] A. Engelhart, W. G. Teich, J. Lindner, G. Jeney, S. Imre, and L. Pap, “A survey of multiuser/multisubchannel detection schemes based on recurrent neural networks,” *Wirel. Commun. Mob. Comput.*, vol. 2, no. 3, pp. 269–284, 2002.

[21] L. Zhang and X. Zhang, “MIMO channel estimation and equalization using three-layer neural networks with feedback,” *Tsinghua Sci. Technol.*, vol. 12, no. 6, pp. 658–662, 2007.

[22] K. Hiray and K. V. Babu, “A neural network based channel estimation scheme for OFDM system,” in *2016 International Conference on Communication and Signal Processing (ICCSP)*, 2016, pp. 0438–0441.

[23] K. K. Sarma and A. Mitra, “MIMO channel modeling with cluster configuration of Complex Time Delay Fully Recurrent Neural Network,” in *Recent Advances in Intelligent Computational Systems (RAICS), 2011 IEEE*, 2011, pp. 348–351.

[24] P. Gogoi and K. K. Sarma, “Recurrent neural network based channel estimation technique for STBC coded MIMO system over Rayleigh fading channel,” in *Proceedings of the CUBE International Information Technology Conference*, 2012, pp. 294–298.

[25] S. J. Nawaz, S. Mohsin, and A. A. Ikaram, “Neural Network Based MIMO-OFDM Channel Equalizer Using Comb-Type Pilot Arrangement,” 2009, pp. 36–41.

[26] W. Liu, L.-L. Yang, and L. Hanzo, “Recurrent neural network based narrowband channel prediction,” in *2006 IEEE 63rd Vehicular Technology Conference*, 2006, vol. 5, pp. 2173–2177.

[27] B. Mailhé, R. Gribonval, F. Bimbot, and P. Vandergheynst, “A low complexity orthogonal matching pursuit for sparse signal approximation with shift-invariant dictionaries,” in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2009, pp. 3445–3448.

[28] T. Blumensath and M. E. Davies, “In greedy pursuit of new directions:(nearly) orthogonal matching pursuit by directional optimisation,” in *Signal Processing Conference, 2007 15th European*, 2007, pp. 340–344.

[29] Y. S. Abu-Mostafa, “Information theory, complexity and neural networks,” *IEEE Commun. Mag.*, vol. 27, no. 11, pp. 25–28, 1989.

[30] R. Rojas, “The Complexity of Learning,” in *Learning and Generalisation - With Applications to Neural Networks*, Springer, 1996, pp. 263–285.

[31] H. Liu and G. Li, *OFDM-Based Broadband Wireless Networks: Design and Optimization*. Wiley-Interscience, 2005.

[32] B.-H. Chiang, D.-B. Lin, and J.-L. Yu, “OFDM in multipath mobile fading channel,” in *Proc. of the 8th Mobile Computing Workshop*, 2002, pp. 196–201.

[33] A. F. Molisch, M. . Toeltsch, and S. . Vermani, “Iterative Methods for Cancellation of Intercarrier Interference in OFDM Systems,” *IEEE Trans. Veh. Technol.*, vol. 56, no. 4, pp. 2158–2167, Jul. 2007.

[34] E. P. L. Be, “Adaptive Techniques for Multiuser OFDM,” *Degree Dr. Philos. James Cook Univ.*, 2001.

[35] S. Jain, “Time-Frequency Training OFDM using Matlab for high speed environments,” *Int. J. Sci. Res. Publ.*, vol. 3, 2013.
[36] H. Senol, E. Panayirci, and H. V. Poor, “Nondata-Aided Joint Channel Estimation and Equalization for OFDM Systems in Very Rapidly Varying Mobile Channels,” *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4236–4253, Aug. 2012.
[37] “SatNav Model Page.” [Online]. Available: http://www.kn-s.dlr.de/satnav/LandMobile.html. [Accessed: 13-Jan-2017].
[38] E. J. Candes and M. B. Wakin, “An Introduction To Compressive Sampling,” *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2008.
[39] P. Bello, “Characterization of Randomly Time-Variant Linear Channels,” *IEEE Trans. Commun. Syst.*, vol. 11, no. 4, pp. 360–393, Dec. 1963.
[40] S. Das, “Mathematical methods for wireless channel estimation and equalization,” phd, uniwien, wien, 2009.
[41] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1993.
[42] S. G. Mallat and Z. Zhang, “Matching pursuits with time-frequency dictionaries,” *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.
[43] I. A. Basheer and M. Hajmeer, “Artificial neural networks: fundamentals, computing, design, and application,” *J. Microbiol. Methods*, vol. 43, no. 1, pp. 3–31, Dec. 2000.
[44] L. I. Zhang, H. W. Tao, C. E. Holt, W. A. Harris, and M. Poo, “A critical window for cooperation and competition among developing retinotectal synapses,” *Nature*, vol. 395, no. 6697, pp. 37–44, Sep. 1998.
[45] A. K. Jain, J. Mao, and K. M. Mohiuddin, “Artificial neural networks: a tutorial,” *Computer*, vol. 29, no. 3, pp. 31–44, Mar. 1996.
[46] M. T. Hagan, H. B. Demuth, and M. Beale, *Neural Network Design*. Boston, MA, USA: PWS Publishing Co., 1996.
[47] S. Lek and J. F. Guégan, “Artificial neural networks as a tool in ecological modelling, an introduction,” *Ecol. Model.*, vol. 120, no. 2–3, pp. 65–73, Aug. 1999.
[48] P. Orponen, “Computational Complexity of Neural Networks: A Survey,” *Nord. J Comput.*, vol. 1, no. 1, pp. 94–110, Mar. 1994.
[49] J. Bruck and J. W. Goodman, “On the power of neural networks for solving hard problems,” *J. Complex.*, vol. 6, no. 2, pp. 129–135, Jun. 1990.