Channel Estimation for OFDM using Deep Learning Methods

Atharv Ramesh Nair
EE20BTECH11006

Nithish S
EE20BTECH11044

***Abstract*—In this project, we investigate various channel estimation methods in Orthogonal Frequency Division Multiplexing (OFDM) systems. OFDM is a widely used data transmission technique that splits a frequency band into several closely spaced narrow bands, enabling parallel data transmission. Our study begins with a comprehensive analysis of classical channel estimation approaches, including Least Squares (LS) and Minimum Mean Square Error (MMSE) methods. After simulating and verifying these methods, we explore deep learning-based approaches for channel estimation.Deep learning has significantly advanced in recent years, and we propose a Convolutional Neural Network (CNN) model with skip connections that estimates complex channel coefficients from received signal samples. Our model demonstrates improved performance compared to the LS estimator, particularly at low SNR values, although it does not outperform the MMSE estimator. We highlight the potential for further improvement through the use of advanced deep learning techniques, such as bi-directional Long Short-Term Memory (LSTM) networks.**

I. Introduction

In this report we will be explaining various methods used for channel estimation in OFDM systems. Orthogonal Frequency Division Multiplexing(OFDM), is method of data transmission in which frequency band is split into several closely spaced narrow bands, so that data can be transmitted parallely. The following block diagram explains the various steps involved in OFDM system,

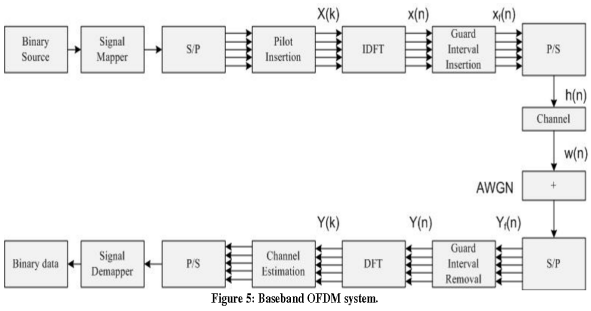

Fig. 1: OFDM Block Diagram

II. Classical Methods

Classical channel estimation algorithms can be classified into three types: Training based channel estimation, Blind channel estimation, Semiblind channel estimation.

Training based channel estimation involves sending pilot symbols(symbols known to the receiver), to estimate the channel state. The blind channel estimation involves evaluating statistical information of the channel such as fading distribution, channel gain, spatial correlation along with some properties of the transmitted signal. Semiblind channel estimation is a hybrid combination of Training based estimation and Blind channel estimation. In fast fading channels, Blind Channel estimation is the preferred method since the channel characteristics vary faster than the symbol duration and in slow fading channels Training based estimation is the preferred method.

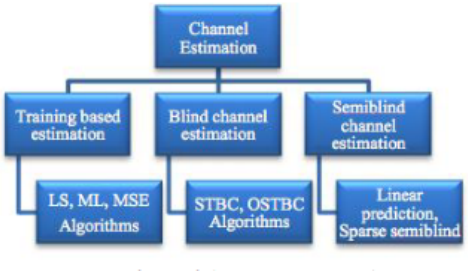

Fig. 2: Classical Channel Estimation Methods

A. Type of Pilots

There are three types of pilots, block type, comb type and lattice type pilots. In block type pilots, the pilot symbols are sent in all frequency sub carriers and periodically in time. We estimate the channel state at the instants when pilots are sent and assume that it remains constant till the next instant when the pilots are sent. In comb type pilots, the pilot symbols are sent only in certain frequency subcarriers but in all time instants. We estimate the channel response in frequency carriers where pilot symbols are sent, and use interpolation methods to estimate the channel response in other frequency sub carriers. For slow fading channels, it is better to use block type pilots and for fast fading channels it is better to use comb type pilots.

B. Least Squares Estimate

In Least Squares Estimate we try to find that value of channel response that minimizes the squared error loss.

$$H_{LS} = \underset{\hat{H}}{\operatorname{argmin}}\{(Y - HX)^H (Y - HX)\} \tag{1}$$

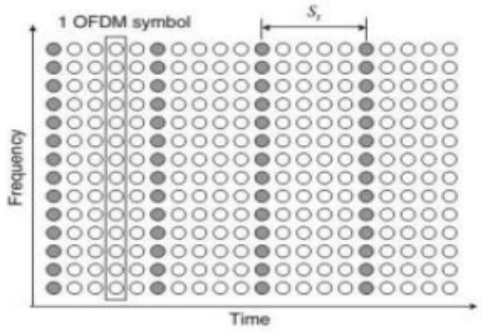

Fig.3(a)- Block Type Pilot

Fig. 3: Block Type Pilots

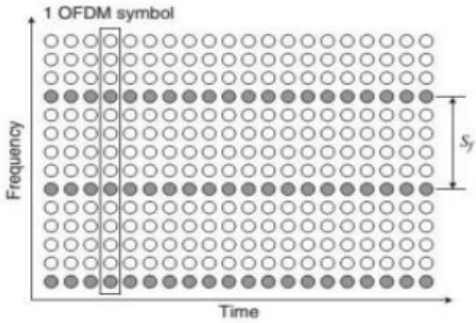

Fig. 3(b)- Comb Type Pilot

Fig. 4: Comb Type Pilots

The solution to the above minimization is given by,

$$H_{LS} = X^{-1}Y \tag{2}$$

The above solution can be obtained by setting the gradient of the objective function wrt H to zero. As you can see LS estimate doesn't use the underlying distribution of the channel coefficients to estimate the channel response.

C. Minimum Mean Squared Error(MMSE) Estimate

The mean squared error is defined by,

$$MSE = \mathbb{E}\{(H - \hat{H})^H (H - \hat{H})\} \tag{3}$$

Then the MMSE estimate of H is defined as that $\hat{H}$ that minimizes MSE,

$$H_{MMSE} = \underset{H}{\operatorname{argmin}}(\mathbb{E}\{(H - \hat{H})^H (H - \hat{H})\} \tag{4}$$

The following second order statistics will be used in MMSE estimation,

$$R_{HH} = \mathbb{E}[HH^H] = \mathbb{E}[(Fh)(Fh)^H] = FR_{hh}F^H \tag{5}$$
$$R_{hY} = \mathbb{E}[hY^H] = \mathbb{E}[h(XFh + N)^H] = R_{hh}F^H X^H \tag{6}$$
$$R_{YY} = \mathbb{E}[YY^H] = XFR_{hh}F^H X^H + \sigma^2 I_N \tag{7}$$

where F is the Fourier transform matrix, X is diagonal matrix containing pilot symbols, Y is vector of received symbols, σ denotes the noise variance and h denotes the channel response. Then the MMSE estimate of H that is given by,

$$H_{MMSE} = Fh_{MMSE} \tag{8}$$
$$= F[(F^H X^H)^{-1} R_{hh}^{-1}\sigma^2 + XF]^{-1}Y \tag{9}$$
$$= FR_{hh}[(F^H X^H XF)^{-1}\sigma^2 + R_{hh}]F^{-1}H_{LS} \tag{10}$$
$$= R_{HH}[R_{HH} + \sigma_N^2 (XX^H)^{-1}]^{-1}H_{LS} \tag{11}$$

To find the MMSE estimate we need to know the underlying distribution of the channel coefficients.

The following is the derivation of the experession for finding MMSE estimate,

Let,

$$\hat{H}_{MMSE} = MY \tag{12}$$

where M is a linear estimator. If M is MMSE estimator then it satisfies the orthogonality principle,

$$\mathbb{E}[(H - \hat{H}_{MMSE})Y^H] = 0 \tag{13}$$
$$\implies \mathbb{E}[HY^H] = \mathbb{E}[\hat{H}_{MMSE}Y^H] \tag{14}$$
$$\implies F\mathbb{E}[hY^H] = M\mathbb{E}[YY^H] \tag{15}$$
$$FR_{hY} = MR_{YY} \tag{16}$$
$$M = FR_{hY}R_{YY}^{-1} \tag{17}$$

Substituting the expressions for R_{hY} and R_{YY} in the above equations give,

$$M = FR_{hh}F^H X^H (XFR_{hh}F^H X^H + \sigma^2 I_N)^{-1} \tag{18}$$
$$M = R_{HH}X^H (XR_{HH}X^H + \sigma^2 I_N)^{-1} \tag{19}$$
$$M = R_{HH}X^H X^{-1}(R_{HH} + \sigma^2 (XX^H)^{-1})^{-1}(X^H)^{-1} \tag{20}$$

Then $\hat{H}_{MMSE}$ is given as,

$$\hat{H}_{MMSE} = R_{HH}(R_{HH} + \sigma^2 (XX^H)^{-1})^{-1}(XX^H)^{-1}X^H Y \tag{21}$$
$$\hat{H}_{MMSE} = R_{HH}(R_{HH} + \sigma^2 (XX^H)^{-1})^{-1}H_{LS} \tag{22}$$

III. Newer Methods for Channel Estimation

A. Deep Learning Based

In [1] the authors suggest a fully connected neural network to perform end to end signal detection using the received signal as input.

*1) **Model Architecture**:* The exact model architecture that we use consists of five layers. The number of neurons in the input layer is 256, in the first layer is 500, in the second layer is 250, in the third layer is 120, and in the final layer is 16. We used Relu activation function for the intermediate layers and sigmoid activation function in the final layer.

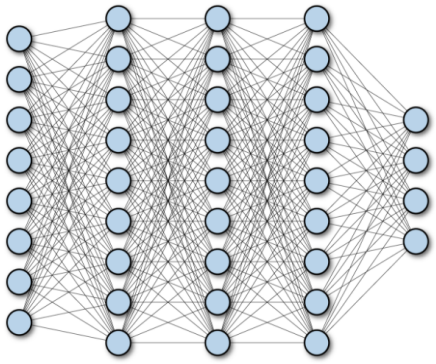

Fig. 5: Model Architecture

*2) **Training**:* Deep learning models are trained assuming that the entire OFDM block and the wireless channels to be a block box, and the Deep learning model is trained to model that function. The authors suggest to use the mean square error between transmitted bits and output of the network as the loss function to minimize.

We assume each OFDM block to have 64 carriers. The input layer has 256 neurons, because we pass as input two OFDM blocks and the real and imaginary part of the OFDM symbols ar e concatenated to form a input vector of length 256.

We predict the transmitted bits in groups of 16, and concatenate them to form the output vector.

Another popular deep learning based method is the ChannelNet model mentioned in the [4]. The authors of this paper pose the channel estimation problem as an image super resolution problem, and they use two popular CNNs called DnCNN and SRCNN. DnCNN is originaly used for the task of image denoising and SRCNN is used for the task of image super resolution.

B. Significant Taps Detection for Sparse Channel Estimation in OFDM Systems

In [2] the authors suggest an algorithm to effectively find the sparse channel estimate by finding the most significant taps.

We first get a Least Squares estimate of the channel impulse response. We then try to get an estimate of the noise standard deviation using this LS estimate. Let $\hat{h_{LS}}$ be the LS estimate of the channel impulse response, then we get an initial estimate of the noise standard deviation by,

$$\hat{\sigma_n} = \sqrt{2}\frac{\text{median}(|\hat{h}_{LS}|)}{\sqrt{\ln 4}}\sqrt{2}\frac{\text{median}(|\hat{h}_{LS}|)}{\sqrt{\ln 4}} \tag{23}$$

Then the initial estimate of threshold is defined as,

$$T = \sqrt{2\ln L_{CP}}\hat{\sigma_n} \tag{24}$$

We define the vector of noise coefficients $\mathbf{c}$ to be the coefficients $\hat{h}_{LS}[j]$ with amplitude smaller than or equal to T.

$$\mathbf{c} = [\mathbf{c}[0], \mathbf{c}[1], \cdot, \mathbf{c}[L_{CP}^{'} - 1]], \text{ where } L_{CP}^{'} \leq L_{CP} \tag{25}$$

The final estimate of noise standard deviation is obtained by,

$$\hat{\sigma}_n^{'} = \sqrt{2}\frac{\text{median}(|\hat{c}|)}{\sqrt{\ln 4}} \tag{26}$$

and an effective threshold is obtained by,

$$\eta = \sqrt{2\ln L_{CP}^{'}}\hat{\sigma_n^{'}} \tag{27}$$

We then soft threshold LS estimate of the channel impulse response $\hat{h}_{LS}$ with the η.

$$\hat{h}[n] = \hat{h}_{LS}[n]; \text{ if } |\hat{h}_{LS}[n]| > \eta \tag{28}$$

$$= 0; \text{ if } |\hat{h}_L S[n]| \leq \eta \tag{29}$$

$$0 \leq n \leq L_{CP} - 1 \tag{30}$$

The above soft thresholding operation makes the final estimate of channel impulse response sparse. .

IV. Proposed CNN Model Architecture with Skip Connections

The model presented in Section 3.1 is outdated and does not incorporate recent advancements in the field of deep learning. In this section, we introduce a proposed deep learning model for channel estimation, designed to learn complex channel coefficients from received signal samples. The model consists of multiple layers, including convolutional and fully connected layers, enhanced with skip connections to improve the learning process. The model accepts the received OFDM symbol (Y) as input and outputs the estimate of the channel frequency response (H). Given that we are dealing with complex-valued signals, the real and imaginary components are separated.

A. Model Structure

The proposed model architecture consists of the following layers:

1) **Input Layer**: The input layer takes a 64x2 matrix representing the real and imaginary components of the received signal samples.
2) **First Convolutional Block**: This block contains a 1D convolutional layer with 64 filters and a kernel size of 3, followed by batch normalization and a ReLU activation function. The output of this block retains the same spatial dimensions as the input.
3) **Second Convolutional Block with Skip Connection**: This block starts with a skip connection from the output of the first convolutional block. The main path of this block includes two 1D convolutional layers, each with 64 filters and a kernel size of 3, followed by batch normalization and ReLU activation. The output of the main path is combined with the skip connection, and the result is passed through another ReLU activation function.
4) **Fully Connected Layers**: The output of the second convolutional block is flattened and passed through two fully connected layers. The first layer has 128 neurons with ReLU activation, while the second layer has 128

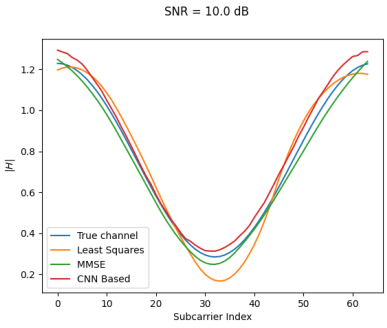

Fig. 6: Channel Estimation for SNR = 10dB

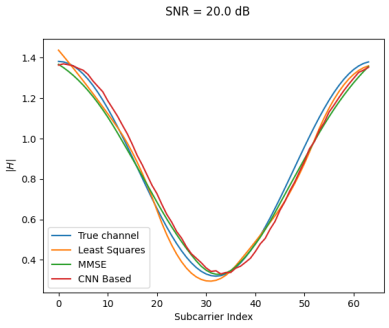

Fig. 7: Channel Estimation for SNR = 20dB

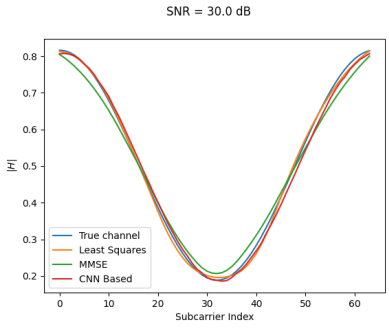

Fig. 8: Channel Estimation for SNR = 30dB

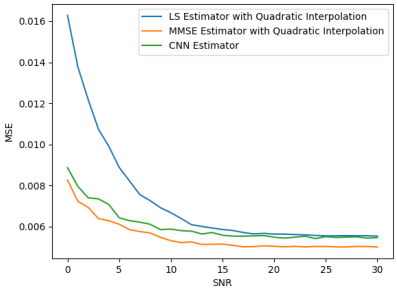

Fig. 9: MSE vs PSNR for different methods

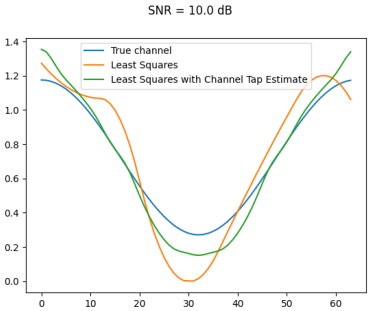

Fig. 10: Improvement of Least Square Estimate

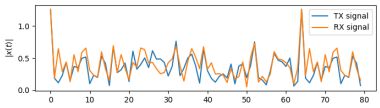

Fig. 11: TX and RX

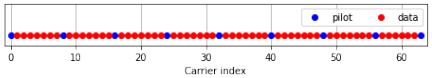

Fig. 12: Pilot Placement for Simulation

neurons without activation.

5) **Output Layer**: The output of the last fully connected layer is reshaped into a 64x2 matrix, representing the real and imaginary components of the estimated channel coefficients.

B. Data Generation and Training

To train our CNN model for channel estimation, we generated synthetic data using the basic OFDM block shown in Fig 1. The data generation and training process involves the following steps:

1) Generate random binary symbols: 220 random bits are generated and mapped to 16-QAM symbols to create 55 data symbols.
2) Generate pilot symbols: Certain indices in the OFDM block are set to a fixed value ($(3+3j)$ in our case) to act as pilot symbols. These symbols are known to the receiver and are used to estimate the channel response. To facilitate interpolation, pilots are added at the beginning and end of the block. Specifically, the 64-symbol OFDM data stream has pilot symbols at positions *0,8,16,24,32,40,48,56,63*. The remaining symbols (i.e., those that are not pilot symbols) are filled with the data symbols. Fig 12 shows the placement of pilots in the OFDM block.
3) Generate transmitted signal: The data symbols generated in the previous step are transformed into the time domain using a 64-point IFFT. The resulting signal is then appended with a cyclic prefix (CP) of length 16 samples to guard against inter-symbol interference.
4) Generate channel response: A complex-valued channel response h (in our case, a 3-tap response) is generated and weighted with specific values. The channel response is then transformed into the frequency domain using a 64-point FFT to obtain H_{FFT}, which is stored since it is used for training the model
5) Channel: The transmitted signal is convolved with the channel response h to obtain the received signal. Gaussian noise is added to the received signal to achieve a specific signal-to-noise ratio (SNR) specified in dB.
6) Remove Cyclic Prefix: At the receiver's end, the cyclic prefix is first removed to obtain a 64-symbol block denoted by Y.
7) The input to the model is Y, and the output to be generated is H. Both these values are fed to the model for training.

We generated a total of 900,000 samples for training and 100,000 samples for testing using the above method. Y and H values were transformed into 64x2 vectors by separating the real and imaginary components since the model is designed to work with real values. Training was performed for data with three sets of SNR values: 10dB, 20dB, and 30dB. Each model was trained for 50 epochs. Training took around 1 hour each. Since the SNR was fixed for one set of SNR values, each model is skewed for data having a particular SNR. Through testing, it was observed that the model trained for SNR = 30dB had better overall performance for a range of SNRs from [0, 30]. During prediction time, an appropriate model was chosen based on the SNR value.

V. Simulations

We simulated the OFDM block diagram shown in Fig 1. The specifics have been mentioned used in Section 4.b . The Channel Response(64 x1) estimation was done using three

methods - Least Squares, MMSE (Minimum Mean Square Estimation) and CNN based estimation. Once the channel vector was estimated, the following steps were performed.

- Equalisation: The recieved vector (Y) was divided by the channel estimate (H)
- Extraction of Data Symbols : Symbols present in pilot indices were removed and data symbols were extracted
- Demodulation: The recieved symbols were demodulated based on minimum distance rule using a demapping Look Up Table based on 16 QAM

VI. Results and Discussions

Figure 11 illustrates the input and output symbols of the channel, highlighting the distortion introduced by the channel. Figures 6, 7, and 8 display the magnitude of the channel estimate vector at various SNR values, with the x-axis representing all 64 sub-carriers. At lower SNRs, both the MMSE and CNN-based models outperform the LS-based model. However, as the SNR increases, the performance gap narrows, and all three models estimate the channel with minimal error.

Figure 9 depicts the Mean Square Error's (MSE) variation with SNR. The performance of our CNN-based model falls between that of the MMSE and LS models. Notably, at lower SNRs, it significantly surpasses the LS estimate in performance.

Figure 10 demonstrates the enhancement in LS estimation achieved through the time-domain thresholding method described in Section 3.B.

VII. Conclusions and Learnings

In this project, our initial focus was to gain a comprehensive understanding of classical methods for estimating the Channel Response (H), specifically the Least Squares (LS) and Minimum Mean Square Error (MMSE) estimation techniques. After simulating and verifying these methods, we shifted our attention to deep learning-based approaches for channel estimation. Our starting point was the work by [4]; however, we were unable to replicate their results. Consequently, we turned to [1], one of the pioneering papers in employing deep learning for channel estimation.

Although deep learning has significantly advanced since the publication of [1], we were inspired to develop our own deep learning model for channel estimation. The model described in Section 4.1 was our attempt at this. Despite limited training and only one hour of training time, the results were reasonably satisfactory. Our model outperformed the LS estimator at low Signal-to-Noise Ratio (SNR) values, but it did not compare favorably with the MMSE estimator.

Recent research by [6] appears promising, as they have utilized bi-directional Long Short-Term Memory (LSTM) networks, in addition to Convolutional Neural Networks (CNNs) and Fully Connected Neural Networks (FCNNs). They claim that their model's performance can surpass the MMSE estimator. As a future extension of our work, we plan to explore this approach further.

References

[1] H. Ye, G. Y. Li and B. -H. Juang, "Power of Deep Learning for Channel Estimation and Signal Detection in OFDM Systems," in IEEE Wireless Communications Letters, vol. 7, no. 1, pp. 114-117, Feb. 2018, doi: 10.1109/LWC.2017.2757490.

[2] H. Xie, G. Andrieux, Y. Wang, J. F. Diouris and S. Feng, "Threshold based most significant taps detection for sparse channel estimation in OFDM system," 2013 IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC 2013), KunMing, China, 2013, pp. 1-5, doi: 10.1109/ICSPCC.2013.6663907.

[3] S. Coleri, M. Ergen, A. Puri and A. Bahai, "Channel estimation techniques based on pilot arrangement in OFDM systems," in IEEE Transactions on Broadcasting, vol. 48, no. 3, pp. 223-229, Sept. 2002, doi: 10.1109/TBC.2002.804034.

[4] M. Soltani, V. Pourahmadi, A. Mirzaei and H. Sheikhzadeh, "Deep Learning-Based Channel Estimation," in IEEE Communications Letters, vol. 23, no. 4, pp. 652-655, April 2019, doi: 10.1109/LCOMM.2019.2898944.

[5] Channel Estimation in OFDM Systems by Yushi Shen and Ed Martinez https://www.nxp.com/docs/en/application-note/AN3059.pdf

[6] X. Ren, L. Chen and Z. Liu, "A Deep Neural Network with Residual Skip Connections for Channel Estimation," 2022 3rd International Conference on Electronics, Communications and Information Technology (CECIT), Sanya, China, 2022, pp. 298-303, doi: 10.1109/CECIT58139.2022.00059.