Mohammad Abu Shams 1200549, Yazeed Hamdan 1201133, Mahmoud Hamdan 1201134

Line Echo Cancellation

Abstract — this project focuses on improving voice quality in communication systems by addressing line echoes through the use of adaptive line echo cancellers. The project involves analyzing impulse responses and source signals, generating and evaluating echo signals. An adaptive line echo canceller is trained using the e-NLMS algorithm, and its performance is assessed. The project also includes a comparison of adaptive algorithms for line echo cancellation. The ultimate goal is to enhance communication quality by effectively canceling echoes and improving signal intelligibility.

I. INTRODUCTION

Line echo cancellation is crucial in communication systems to enhance voice quality by mitigating the interference caused by echoes. Adaptive line echo cancellers analyze the far-end signal and generate a replica of the echo, enabling the removal of the echo component from the received signal.

II. PROBLEM SPECIFICATION

This project addresses the issue of degraded voice quality in communication systems due to echoes. The objective is to develop an adaptive line echo canceller that can estimate and cancel the echo, improving the clarity of the received signal. Key challenges include analyzing the echo path's impulse response, examining the source signal, generating the echo signal, and evaluating the performance of the adaptive algorithm. By addressing these challenges, our aim is to enhance communication quality by effectively eliminating echoes and improving the intelligibility of transmitted signals.

III. DATA

In this project on line echo cancellation, we will be utilizing two main datasets to develop and evaluate our work. These datasets provide the necessary signals for simulating and analyzing the echo path and echo cancellation algorithms. The details of the datasets are as follows:

1. Echo Path Impulse Response:
    - Dataset Name: "path.mat"
    - Description: This dataset contains the impulse response sequence of a typical echo path. The impulse response represents the behavior of the echo path, including the reflections and delays introduced in the communication system over phone lines.
    - Usage: We will load this dataset to plot the impulse response and frequency response of the echo path. It serves as a reference for understanding the characteristics of the echo path and evaluating the performance of the echo cancellation algorithm.
2. Composite Source Signal (CSS):
    - Dataset Name: "css.mat"
    - Description: This dataset comprises 5600 samples of a synthetic signal that emulates the properties of speech. It consists of segments with pauses, periodic excitation, and white-noise properties, representing various speech patterns.
    - Usage: We will load this dataset to generate the far-end signal for the echo cancellation system. By concatenating multiple blocks of the CSS signal, we can simulate a continuous speech signal that travels through the echo path. The CSS signal will be used as the input to the adaptive line echo canceller and to generate the echo signal for training and evaluation purposes.

These datasets provide the necessary information to simulate the echo path and evaluate the performance of the echo cancellation algorithms. By using the impulse response of the echo path and the CSS signal as input, we can analyze the characteristics of the echo signal, estimate the input/output powers, train the adaptive filter, and evaluate the cancellation performance. Throughout the project, we will employ various signal processing techniques, algorithms, and analysis methods to achieve effective line echo cancellation. The chosen datasets play a crucial role in providing realistic and representative signals that allow us to develop and evaluate our work in a practical context.

The project involves loading and analyzing these signals to perform various tasks such as plotting their time-domain and frequency-domain representations, evaluating the echo-return-loss (ERL), training an adaptive line echo canceller, comparing different adaptive algorithms, and analyzing the estimated FIR channel's response.

IV. EVALUATION CRITERIA

Echo Cancellation Quality: This refers to how effectively the system can eliminate the echo. This can be measured by calculating the percentage of echo reduction or Echo Reduction Rate (ERR), where a higher percentage indicates a more effective cancellation. Audio Quality: The system should not distort the desired signal while removing the echo. This can be measured through the Perceptual Evaluation of Speech Quality (PESQ) score, which is an ITU standard for objective measurement for perceived speech quality. Real-Time Performance: An effective LEC should work in real-time and adapt quickly to changes in the echo path. This can be measured by tracking the time it takes for the system to adapt to a change (Adaptation Time). Double-Talk Performance: A good LEC should be able to handle scenarios where both the near-end and far-end speakers are talking simultaneously. This can be evaluated using Double-Talk Detection Rate (DTDR), where a higher rate indicates better performance. Residual Echo: This is the amount of echo left after the cancellation process. It can be measured in terms of Residual Echo Rate (RER). The lower the rate, the better the echo cancellation. System Stability: The echo cancellation system should maintain a consistent performance over time, without any significant fluctuations. Stability can be evaluated by tracking performance metrics over a certain period and ensuring they remain within acceptable ranges. Resource Efficiency: The echo canceller should be efficient in terms of computational resources it utilizes. This could be measured by evaluating the CPU usage, memory usage, and other system resources. These criteria provide a comprehensive way to evaluate the performance of your Line Echo Canceller, allowing you to identify areas that need improvement and measure the effectiveness of your solution.

V. APPROACH

Approach 1: Time-Domain Echo Cancellation in this method, we directly process the signals in the time domain using adaptive filtering techniques. Steps: Filter Initialization: We initialize an adaptive filter with arbitrary weights, typically zeros. Filter Adjustment: As the signal comes in, we calculate the error, which is the difference between the desired signal and the output of our adaptive filter. We then update our filter weights based on this error using an algorithm like Least Mean Squares (LMS) or Normalized Least Mean Squares (NLMS). Echo Removal: We use the adjusted filter to predict and subtract the echo from the incoming signal, resulting in an echo-free signal. The main advantage of time-domain methods is their simplicity and straightforward implementation. However, they can be computationally intensive for long echo paths.

Approach 2: Frequency-Domain Echo Cancellation This method involves transforming the signals into the frequency domain before processing, which can be more efficient for handling long echo paths. Steps: Transformation: We first convert the signals from the time domain to the frequency domain using the Fast Fourier Transform (FFT). Filter Initialization and Adjustment: Similar to the time-domain method, we initialize and adjust an adaptive filter. However, in this case, the filter operates in the frequency domain. Echo Removal: After applying the filter, we perform echo cancellation in the frequency domain. Then, we convert the signal back to the time domain using the Inverse Fast Fourier Transform (IFFT), resulting in an echo-free signal. This approach is generally more efficient in terms of computational complexity for handling long echo paths, but it introduces additional complexity due to the need for transforming signals between the time and frequency domains. By contrasting these two approaches, we can investigate the trade-offs between computational efficiency and implementation complexity. We aim to achieve the most effective echo cancellation by balancing these factors according to the specific requirements and constraints of the system.

VI. ALGORITHMS USED

The normalized least-mean-square (NLMS) algorithm is the most widely applied algorithm for adaptive filters such as communication, control, and acoustic processing. Unfortunately, it is very sensitive to impulsive measurement noise. Therefore, it suffers from performance degradation in the presence of impulsive measurement noise. To combat such impulsive measurement noise, many different algorithms have been proposed in the literature.

In many applications, however, not only is the system measurement corrupted by noise, which includes impulsive noise, but the input may also be corrupted by additive noise. Although algorithms which have robustness against impulsive noise work well for situations where there is impulsive noise in the system measurement, they yield biased parameter estimation for the noisy input. Therefore, the bias due to noisy input should be compensated by an unbiased estimation. However, conventional bias-compensation approaches cannot yield unbiased estimates because of the adverse effect of a large estimation error by impulsive noise.

In this Letter, we propose a bias-compensated, error-modified NLMS (BCE-NLMS) algorithm to reduce the negative impacts of a noise corrupted input signal and impulsive noise corrupted measurement. We use the saturation-type error nonlinearity. The nonlinearity was modelled as the modified Huber function. The error signal is passed through the modified Huber function and is then used in an error-modified algorithm. To compensate for the bias due to noisy input properly for the error-modified algorithm, we introduce an unbiasedness criterion. The BCE-NLMS dynamically selects a bias-compensation vector satisfying this criterion that provides unbiased estimates of the parameter against the noisy input and impulsive measurement noise. To use the proposed bias-compensation method, the input noise variance should be estimated. However, in general, input noise variance estimates also do not work well if there is impulsive noise. Therefore, we present a new estimation method. The estimation of input noise variance is updated or maintained by determining whether impulsive noise is occurring or not. In simulations, the BCE-NLMS algorithm had more robustness against both impulsive noises and noisy inputs than the existing algorithms.

Error-modified-NLMS algorithm

Consider desired signal $d_i = \boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{h} + v_i$ obtained from an unknown system, where $\boldsymbol{h} \in \mathscr{R}^{M\times 1}$ is the unknown system to be identified; $\boldsymbol{u}_i = [u_i, u_{i-1}, \ldots, u_{i-M+1}]^{\mathrm{T}}$ denotes the noise-free input vector; u_i is a white Gaussian sequence; v_i is the measurement noise, which is a contaminated-Gaussian impulsive noise defined as $v_i = gi + b_i w_i$, and g_i and w_i are each zero-mean white Gaussian sequences with variances σ_g^2 and σ_w^2; and b_i is a switch sequence of ones and zeros, which is modelled as a Bernoulli random process with occurrence probability $P_r(b_i = 1) = p_r$. The noisy input vector of the adaptive filter is given by

$$\bar{\boldsymbol{u}}_i = \boldsymbol{u}_i + \boldsymbol{n}_i \quad (1)$$

Where $\boldsymbol{n}_i = [n_i, n_{i-1}, n_{i-M+1}]^{\mathrm{T}}$ and n_i is a white Gaussian noise with zero-mean and variance σ_n^2. Moreover, n_i, g_i, w_i and the input vector $\boldsymbol{u}_i$ are assumed to be independent of each other. The filter output error is defined by $\bar{e}_i = d_i - \bar{\boldsymbol{u}}_i^{\mathrm{T}}\boldsymbol{w}_i$, where $\boldsymbol{w}_i \in \mathscr{R}^{M\times 1}$ is the weight of the adaptive filter. The filter output error for the noise-free input vector is defined by $e_i = d_i - \boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{w}_i$. When the impulsive noise occurs, the value of e_i fluctuates considerably. Therefore, the impulsive noise disturbs the precise estimation of $\boldsymbol{h}$. To obtain adequate estimates of $\boldsymbol{h}$ in the system that is experiencing impulsive noise, a modified Huber function $\psi(\cdot)$ is applied to the error signal of the NLMS

$$\boldsymbol{w}_{i+1} = \boldsymbol{w}_i + \mu\frac{\boldsymbol{u}_i\psi(e_i)}{\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{u}_i},$$
$$\psi(e) = \begin{cases} e & \text{for } -\xi \le e \le \xi \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Where ξ is a threshold parameter used to suppress the effect of the outliers.

Bias-compensated Error-modified-NLMS

When the input vector is noisy, only a noisy measurement $\bar{\boldsymbol{u}}_i$ and $\bar{e}_i$ are available instead of $\boldsymbol{u}_i$ and e_i. Therefore, (2) yields a biased estimate. Our main contribution is an unbiased robust NLMS algorithm that compensates for the bias in (2) due to the noisy input. To compensate for the bias, we introduce a bias-compensation vector $\boldsymbol{x}_i$ such as

$$\boldsymbol{w}_{i+1} = \boldsymbol{w}_i + \mu \frac{\bar{\boldsymbol{u}}_i \psi(\bar{e}_i)}{\bar{\boldsymbol{u}}_i^{\mathrm{T}} \bar{\boldsymbol{u}}_i} + \boldsymbol{x}_i \tag{3}$$

Then, the estimation error, $\widetilde{\boldsymbol{w}}_i \triangleq \boldsymbol{w}_i - \boldsymbol{h}$, follows the recursion

$$\widetilde{\boldsymbol{w}}_{i+1} = \widetilde{\boldsymbol{w}}_i + \mu \frac{\bar{\boldsymbol{u}}_i \psi(\bar{e}_i)}{\bar{\boldsymbol{u}}_i^{\mathrm{T}} \bar{\boldsymbol{u}}_i} + \boldsymbol{x}_i \tag{4}$$

The bias-compensation vector $\boldsymbol{x}_i$ will be designed to satisfy the following unbiasedness criterion:

$$E\left(\widetilde{\boldsymbol{w}}_{i+1} \mid \bar{\boldsymbol{u}}_i\right) = 0 \text{ whenever } E\left(\widetilde{\boldsymbol{w}}_i \mid \bar{\boldsymbol{u}}_i\right) = 0 \tag{5}$$

The proposed criterion provides unbiased estimates of the parameter under the noisy input circumstances. If $|\bar{e}_i| \le \xi \left(\psi(\bar{e}_i) = \bar{e}_i\right)$, after taking expectation on both sides of (4) for the given $\bar{\boldsymbol{u}}_i$ and assuming that $E\left(\widetilde{\boldsymbol{w}}_i \mid \bar{\boldsymbol{u}}_i\right) = 0$, the criterion holds that

$$E\left(\widetilde{w}_{i+1} \mid \bar{u}_i\right) = \mu E\left(\frac{\bar{u}_i \bar{e}_i}{\bar{u}_i^{\mathrm{T}} \bar{u}_i} \mid \bar{u}_i\right) + E\left(x_i \mid \bar{u}_i\right) = \mu E\left(\frac{\bar{u}_i e_i}{\bar{u}_i^{\mathrm{T}} \bar{u}_i} \mid \bar{u}_i\right)$$
$$-\mu E\left(\frac{n_i n_i^{\mathrm{T}} w_i}{\bar{u}_i^{\mathrm{T}} \bar{u}_i} \mid \bar{u}_i\right) + E\left(x_i \mid \bar{u}_i\right) = 0 \tag{6}$$

Where

$$E\left(\frac{u_i e_i}{\bar{u}_i^{\mathrm{T}} \bar{u}_i} \mid \bar{u}_i\right) = \frac{1}{\bar{u}_i^{\mathrm{T}} \bar{u}_i} E\left(u_i\left(-u_i^{\mathrm{T}} \widetilde{w}_i + v_i\right) \mid \bar{u}_i\right)$$
$$= \frac{1}{\bar{u}_i^{\mathrm{T}} \bar{u}_i} \left\{-E\left(u_i u_i^{\mathrm{T}} \mid \bar{u}_i\right) E\left(\widetilde{w}_i \mid \bar{u}_i\right) + E\left(u_i v_i \mid \bar{u}_i\right)\right\} =$$

$$E\left(\frac{n_i n_i^{\mathrm{T}} w_i}{\bar{u}_i^{\mathrm{T}} \bar{u}_i} \mid \bar{u}_i\right) = \frac{1}{\bar{u}_i^{\mathrm{T}} \bar{u}_i} E\left(n_i n_i^{\mathrm{T}}\right) E\left(w_i \mid \bar{u}_i\right)$$

Which leads to

$$E\left(\boldsymbol{x}_i \mid \bar{\boldsymbol{u}}_i\right) = \mu \frac{\sigma_n^2}{\bar{\boldsymbol{u}}_i^{\mathrm{T}} \bar{\boldsymbol{u}}_i} E\left(\boldsymbol{w}_i \mid \bar{\boldsymbol{u}}_i\right) \tag{7}$$

Using this condition and stochastic approximation, we have

$$\boldsymbol{x}_i = \mu \frac{\sigma_n^2 \boldsymbol{w}_i}{\bar{\boldsymbol{u}}_i^{\mathrm{T}} \bar{\boldsymbol{u}}_i} \tag{8}$$

However, there is no bias from input noise when $|\bar{e}_i| > \xi$ (i.e. $\psi(\bar{e}_i)$ is saturated by zero). Therefore, bias-compensation should only be executed for $|\bar{e}_i| \le \xi$. The proposed algorithm is obtained by modifying (3) as follows:

$$\boldsymbol{w}_{i+1} = \begin{cases} \boldsymbol{w}_i + \mu \dfrac{\bar{\boldsymbol{u}}_i \psi(\bar{e}_i)}{\bar{\boldsymbol{u}}_i^{\mathrm{T}} \bar{\boldsymbol{u}}_i} + \boldsymbol{x}_i & \text{if } |\bar{e}_i| \le \xi \\ \boldsymbol{w}_i & \text{otherwise} \end{cases} \tag{9}$$

To use the proposed algorithm (9), the input noise variance σ_n^2 should be estimated. If $\boldsymbol{h}$ is well-estimated by $\boldsymbol{w}$, then from the independent properties of n_i, g_i and w_i, it is easy to show that autocovariance $\bar{e}_i$ is given by $E\left[\bar{e}_i^2\right] = \sigma_g^2 + p_r \sigma_w^2 + \sigma_n^2 E \|\boldsymbol{w}_i\|^2$, where $\|\cdot\|$ denotes the Euclidean norm of a vector. We use $E\left[\bar{e}_i^2\right]$ to estimate the input noise variance σ_n^2. However, $\bar{e}_i$ cannot be used directly in the estimation of σ_n^2 because $\bar{e}_i$ fluctuates considerably when impulsive noise occurs. Therefore, σ_n^2 should be estimated only if $b_i = 0$ (i.e. there are no occurrences of impulsive noise in v_i). Because $E\left[\bar{e}_i^2 \mid b_i = 0\right] = \sigma_g^2 + \sigma_n^2 E \|\boldsymbol{w}_i\|^2$, it leads to

$$\sigma_n^2 = E\left[\bar{e}_i^2 \mid b_i = 0\right] / \left(E \|\boldsymbol{w}_i\|^2 + \beta\right) \tag{10}$$

Where $\beta = \sigma_g^2 / \sigma_n^2$, and it is assumed to be available. To decide whether impulsive noise has occurred or not, we investigate the conditions based on probability and assume the following conditions on the signal:

$$\sigma_n^2 \le E\left[\boldsymbol{u}_i^{\mathrm{T}} \boldsymbol{u}_i\right] = E \|\boldsymbol{u}_i\|^2, \quad \sigma_g^2 \le E\left[\left(\boldsymbol{u}_i^{\mathrm{T}} \boldsymbol{h}\right)^2\right] = \| \boldsymbol{h}$$

From the above assumptions, $E\left[\bar{e}_i^2 \mid b_i = 0\right]$ can be upper-bounded as follows:

$$E\left[\bar{e}_i^2 \mid b_i = 0\right] = \sigma_g^2 + \sigma_n^2 E \|w_i\|^2 \le \|\boldsymbol{h}\|^2 E \|\boldsymbol{u}_i\|^2 + E \|\boldsymbol{w}_i\|^2 E \|\boldsymbol{u}_i\|^2 \tag{11}$$

From (1), it holds that $E \|\bar{\boldsymbol{u}}_i\|^2 = E \|\boldsymbol{u}_i\|^2 + M\sigma_n^2$. Substituting this equation into the right side of (11), we obtain

$$E\left[\bar{e}_i^2 \mid b_i = 0\right] \le \|\boldsymbol{h}\|^2 \left(E \|\bar{\boldsymbol{u}}_i\|^2 - M\sigma_n^2\right) + E \|\boldsymbol{w}_i\|^2 \left(E \|\bar{\boldsymbol{u}}_i\|^2 - M\sigma_n^2\right)$$
$$\le \|\boldsymbol{h}\|^2 E \|\bar{\boldsymbol{u}}_i\|^2 + E \|\boldsymbol{w}_i\|^2 E \|\bar{\boldsymbol{u}}_i\|^2 \tag{12}$$

Under the condition that $\boldsymbol{h}$ is well-estimated by $\boldsymbol{w}$, $\|\boldsymbol{h}\|^2$ can be approximated as $\| \boldsymbol{h} \|^2 \simeq E \| \boldsymbol{w}_i \|^2$. Using this approximation in (12) yields

$$E\left[\overline{e}_i^2 \mid b_i = 0 \right] \leq 2E \| \boldsymbol{w}_i \|^2 E \| \overline{\boldsymbol{u}}_i \|^2 \qquad (13)$$

From (13), we can determine that there is no impulsive noise in the desired signal if $E\left[\overline{e}_i^2\right]$ satisfies $E\left[\overline{e}_i^2\right] \leq 2E \| \boldsymbol{w}_i \|^2 E \| \overline{\boldsymbol{u}}_i \|^2$. Replacing the expected values of (13) by their instantaneous values, we obtain the following condition:

$$\overline{e}_i^2 \leq 2 \| \boldsymbol{w}_i \|^2 \| \overline{\boldsymbol{u}}_i \|^2 \qquad (14)$$

From (10) and (14), it is inferred that the method based on selective estimation of the input noise variance is given by

$$\hat{\sigma}_n^2 = \begin{cases} e_i^2 / \left(\| \boldsymbol{w}_i \|^2 + \beta \right) & \text{if} \quad \overline{e}_i^2 \leq 2 \| \boldsymbol{w}_i \|^2 \| \overline{\boldsymbol{u}}_i \|^2 \\ \hat{\sigma}_n^2 & \text{otherwise} \end{cases}$$

(15)

Using (15), σ_n^2 is estimated precisely when the impulsive noise has not occurred [1].

Also in part F, We used Recursive least squares filter (RLS) Recursive least squares (RLS) is an adaptive filter algorithm that recursively finds the coefficients that minimize a weighted linear least squares cost function relating to the input signals. This approach is in contrast to other algorithms such as the least mean squares (LMS) that aim to reduce the mean square error. In the derivation of the RLS, the input signals are considered deterministic, while for the LMS and similar algorithms they are considered stochastic. Compared to most of its competitors, the RLS exhibits extremely fast convergence. However, this benefit comes at the cost of high computational complexity [2].

VIII. RESULT AND ANALYSIS

Part A:

By running the code of part a, we will obtain a plot of the impulse response and a magnitude response plot of the corresponding frequency response. These plots can provide insights into the characteristics of the system or filter represented by the impulse response, such as its impulse response shape, magnitude, and frequency response characteristics.

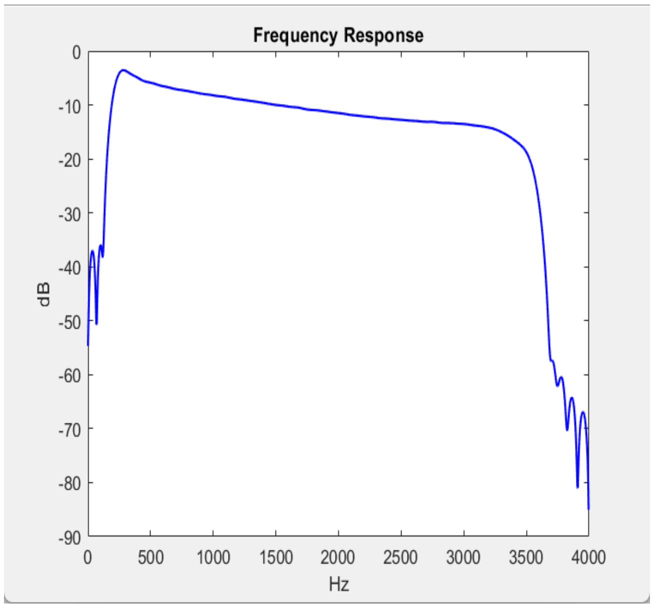


Figure 1

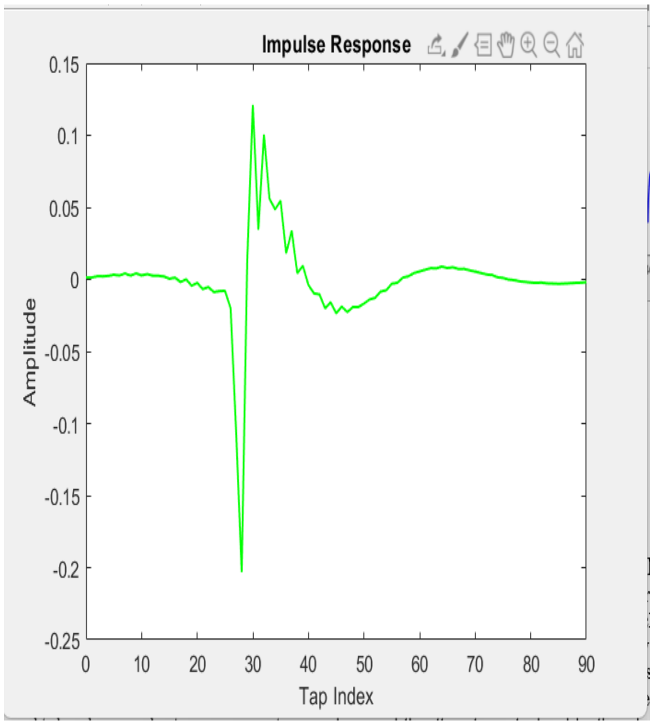


Figure 2

Part B:

For part b, we will obtain a plot of the data signal and a plot of the normalized power spectral density. These plots can provide insights into the frequency content and power distribution of the signal, allowing us to analyze its spectral characteristics and identify any prominent frequency components or noise present in the signal.

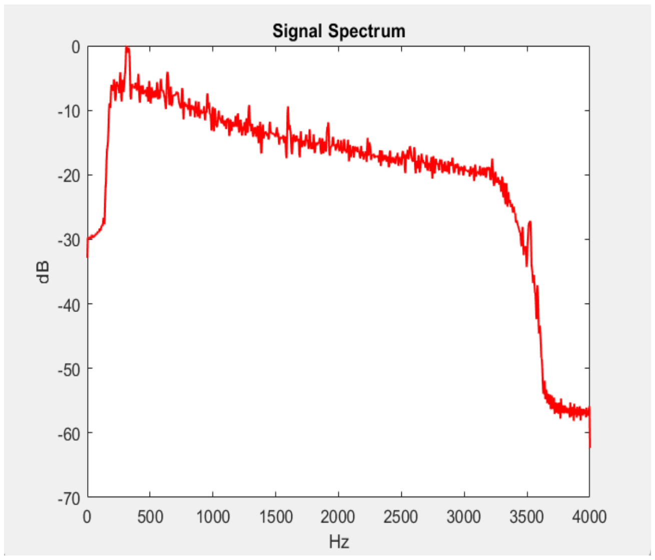


Figure 3

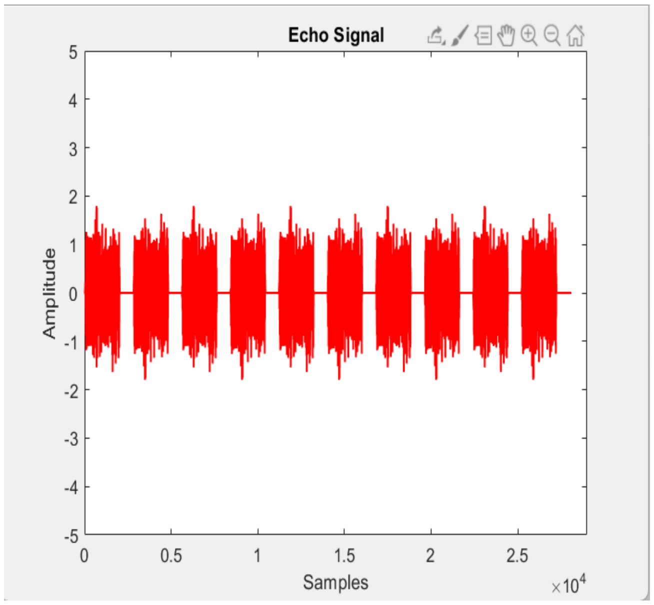


Figure 6

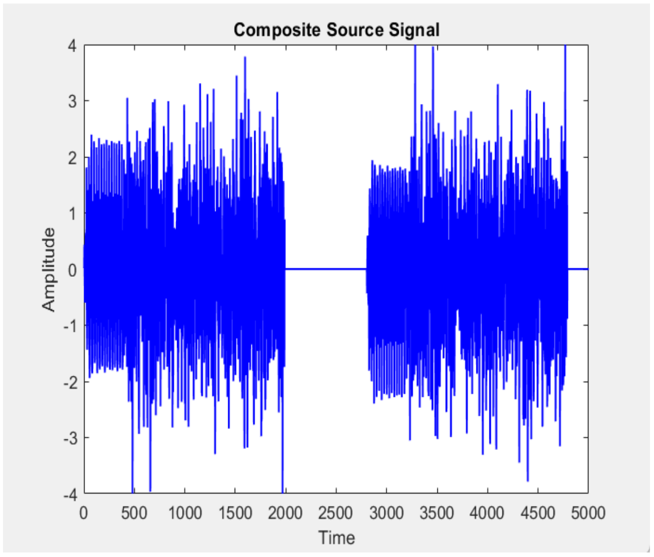


Figure 4

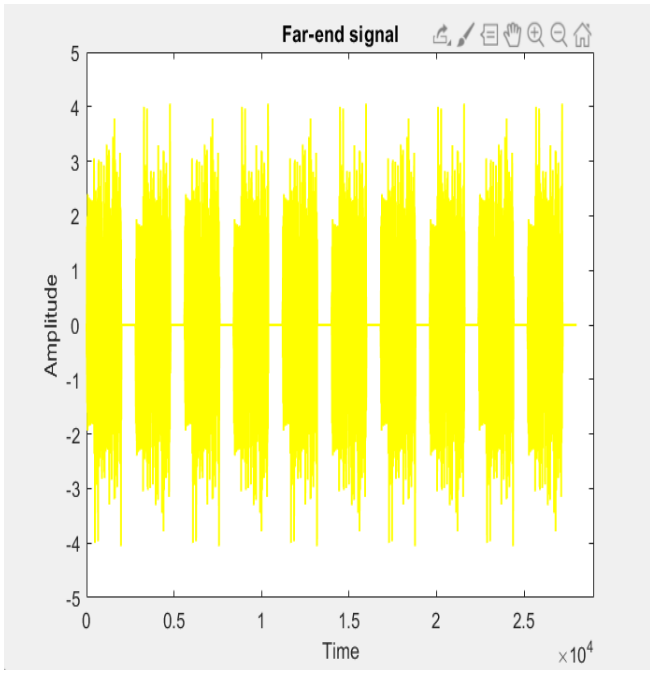
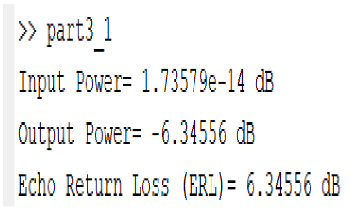


Figure 7

Part C:

Results for part c were observed as shown in the figures, and it has been noticed that two plots were obtained showing the far-end signal and the echo signal, respectively. Additionally, the input power, output power, and ERL (Echo Return Loss) values will be printed. These results provide insights into the level of echo attenuation achieved by the system, which is essential for assessing the quality of echo cancellation.


```
>> part3_1
Input Power= 1.73579e-14 dB
Output Power= -6.34556 dB
Echo Return Loss (ERL)= 6.34556 dB
```
Figure 5

Part D:

Results of part d were obtained as shown in the figures, the obtained figures shows the far-end signal, echo signal, error signal, true echo path, and estimated echo path. These figures allow you to see the performance of the adaptive line echo canceller algorithm in canceling the echo from the far-end signal. The comparison between the true and estimated echo paths provides insights into the convergence of the adaptive filter weights to the actual echo path.

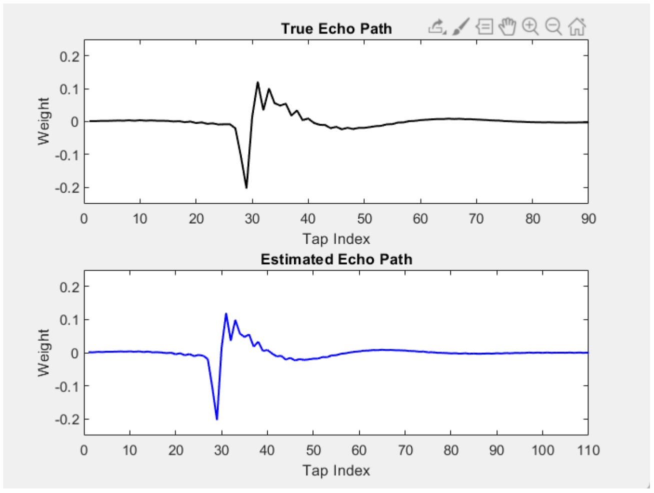


Figure 8

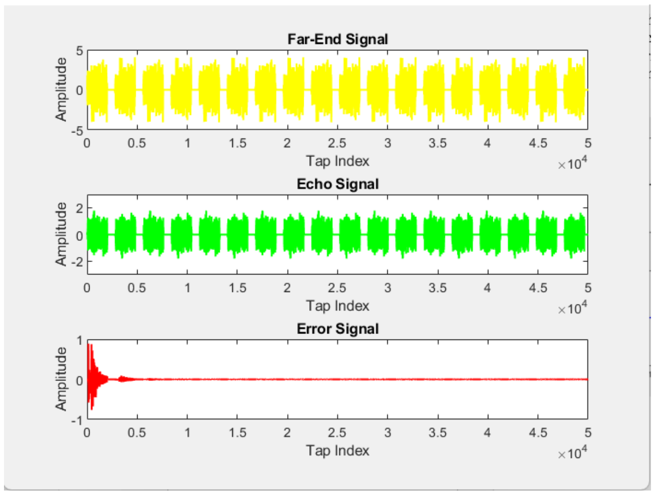


Figure 9

Part E:

By running part E code, we will obtain figures showing the estimated FIR channel response and the given FIR system response. These figures allow us to compare the frequency and phase characteristics of the estimated and given FIR responses. This comparison helps assess the effectiveness of the adaptive line echo canceller in approximating the given FIR system.

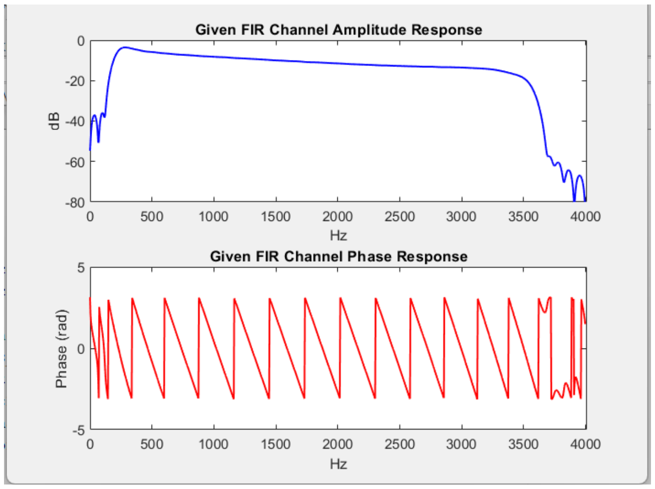


Figure 10

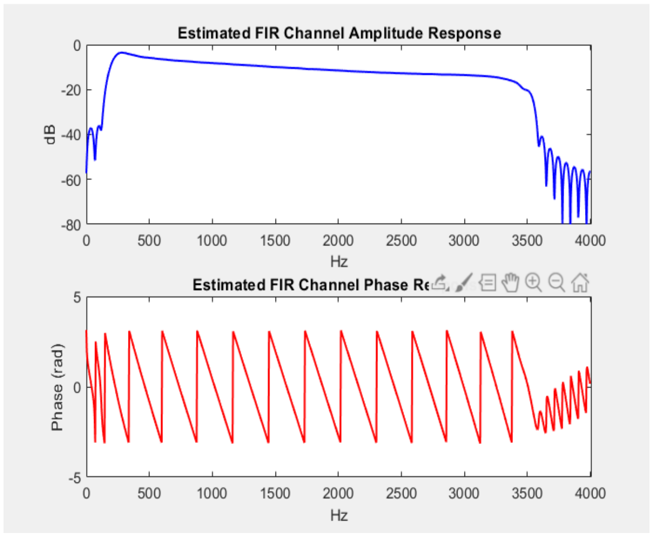


Figure 11

Part F:

For part F, we will obtain figures showing the estimated FIR channel response using RLS and the given FIR system response. These figures allow us to compare the frequency and phase characteristics of the estimated and given FIR responses, specifically using the RLS algorithm.

It has been noticed that the algorithm that has been used (RLS) gives the same results as the algorithm (eNLMS) in the plots of the given FIR channel Amplitude and phase responses, while a small difference is noticed between the plots of the (eNLMS) algorithm and the plots of the (RLS) algorithm for the estimated FIR channel Amplitude and phase responses.

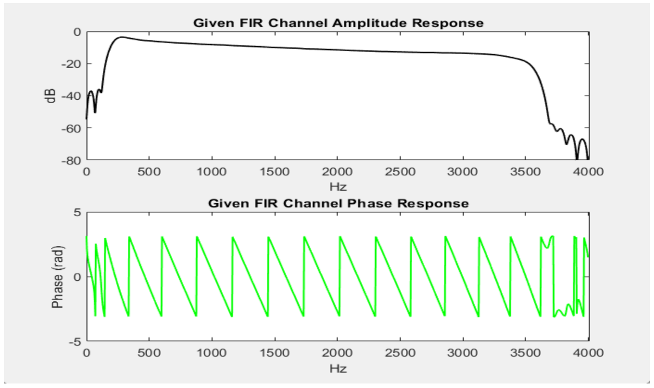


Figure 12

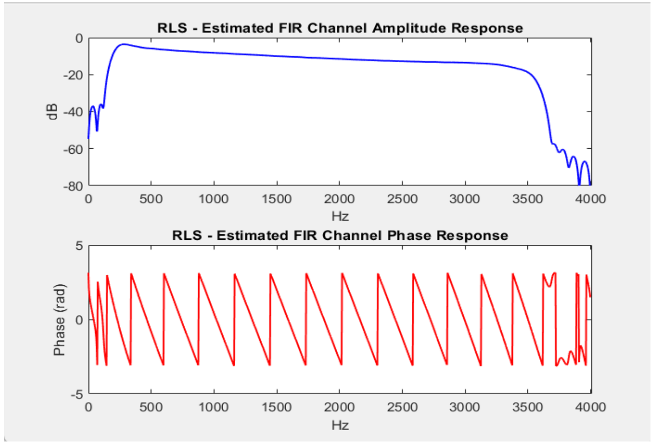


Figure 13

IX. CONCLUSION

In this project, we addressed the issue of echo interference in phone line communications by implementing an adaptive line echo canceller. Through the analysis of the echo path's impulse and frequency responses, we gained insights into its characteristics. By processing composite source signals and evaluating the resulting echo signal, we estimated the echo-return-loss (ERL) and assessed the attenuation introduced by the echo path. We trained an adaptive echo canceller using the e-NLMS algorithm, which effectively reduced the echo interference. The estimated FIR channel closely matched the original echo path, indicating the success of the cancellation process. Additionally, we proposed an alternative adaptive algorithm and compared its performance with e-NLMS, contributing to the exploration of different techniques for echo cancellation. Overall, this project demonstrates the effectiveness of adaptive line echo cancellation techniques in improving voice quality and mitigating echo interference in communication systems.

X. REFERENCES

[1] *Normalised least-mean-square algorithm for adaptive filtering of impulsive measurement noises and noisy inputs*. (2023, 7 2). Retrieved from IET: https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/el.2013.2482#ell2bf00784-bib-0001

[2] *Recursive least squares filter*. (2023, 7 2). Retrieved from WIKIPEDIA: https://en.wikipedia.org/wiki/Recursive_least_squares_filter#:~:text=Recursive%20least%20squares%20(RLS)%20is,reduce%20the%20mean%20square%20error.

XI. APPENDIX

This is code for part A:

```
%PART A.
%Mahmoud Hamdan 1201134.
%Yazzed Hamdan 1201133.
%Mohammad Abu Shams 1200549.
%SEC2.
%Load the impulse response.
load('path.mat');
impulse_response=path;
Fs=8000;%Sampling frequency=8Khz.

%Plot the impulse response.
figure;
plot(0:length(impulse_response)-
1,impulse_response,'g','linewidth',1.2);
xlabel('Tap Index');
ylabel('Amplitude');
title('Impulse Response');
xlim([0,90]);% X axsis from 0-90.
%Compute the frequency response.
[H,frequencies]=freqz(impulse_response,1,1024,Fs);
%Magnitude response.
magnitude_response=abs(H);

%Plot the magnitude response.
figure;
plot(frequencies,20*log10(magnitude_response),'b','linewidth',
1.3);
xlabel('Hz');
ylabel('dB');
title('Frequency Response');
```

This is the code for Part B:

```
%PART B.
%Mahmoud Hamdan 1201134.
%Yazzed Hamdan 1201133.
%Mohammad Abu Shams 1200549.
%SEC2.
%Load the data.
load('css.mat');
css_data=css;
N=length(css_data);
Fs=8000;%Sampling frequency=8Khz.

%Plot the CSS data.
figure;
plot(1:N,css_data,'b','linewidth',1.1);
xlabel('Time');
ylabel('Amplitude');
title('Composite Source Signal');
xlim([0,5000]);% X axsis from 0-5000.
ylim([-4,4]);% Y axsis from -4-4.

%Compute(PSD).
[psd,frequencies]=pwelch(css_data,[],[],[],Fs,'power');

%Normalize the PSD.
max_psd=max(10*log10(psd));
psd_normalized=10*log10(psd)-max_psd;

%Plot the normalized PSD.
figure;
plot(frequencies,psd_normalized,'r','linewidth',1.3);
xlabel('Hz');
ylabel('dB');
title('Signal Spectrum');
xlim([0,4000]);% X axsis from 0-4000.
```

Mohammad Abu Shams 1200549, Yazeed Hamdan 1201133, Mahmoud Hamdan 1201134

This is the code for part C:

```
%PART C.
%Mahmoud Hamdan 1201134.
%Yazzed Hamdan 1201133.
%Mohammad Abu Shams 1200549.
%SEC2.
%Load the data.
load('css.mat')
%Load the impulse response.
load('path.mat')

%Concatenate five blocks of css.
css_concat=repmat(css,1,5);

%Plot the far-end signal.
figure;
plot(css_concat,'y','linewidth',1.2);
title('Far-end signal');
xlabel('Time');
ylabel('Amplitude');
xlim([0,29000]);% X axsis from 0-29000.
ylim([-5,5]);% Y axsis from -5-5.

%Feed the concatenated signal into the echo path.
echo_signal=conv(css_concat,path);

%Plot echo signal.
figure;
plot(echo_signal,'r','linewidth',1.2);
title('Echo Signal');
xlabel('Samples');
ylabel('Amplitude');
xlim([0,29000]);% X axsis from 0-90.
ylim([-5,5]);% Y axsis from 0-90.


%Estimate the input and output powers in dB.
Ncss=length(css_concat);
Necho=length(echo_signal);
Pin=10*log10(sum(css_concat.^2)/Ncss);
Pout=10*log10(sum(echo_signal.^2)/Necho);

%Evaluate the echo return loss (ERL).
ERL=Pin-Pout;


fprintf('Input Power= %g dB\n',Pin);
fprintf('Output Power= %g dB\n',Pout);
fprintf('Echo Return Loss (ERL)= %g
dB\n',ERL);
```

This is the code for part D:

```
%PART D.
%Mahmoud Hamdan 1201134.
%Yazzed Hamdan 1201133.
%Mohammad Abu Shams 1200549.
%SEC2.

%Load the data.
load('css.mat')
%Load the impulse response.
load('path.mat')
%Concatenate Ten blocks of css.
css_concat=repmat(css,1,10);
%Concatenate Ten blocks of css.
far_end_signal=repmat(css,1,10);

concat_Css_10=[css_concat css_concat];
echo_signal=conv(concat_Css_10, path);

M=128;%Taps.
u=0.25;%Step Size.
epsilon=1e-6;
w=zeros(M,1);
y=zeros(size(concat_Css_10));%Echo.
e=zeros(size(concat_Css_10));%Error.

%adaptive line echo canceller.
for n=M:length(far_end_signal)
   x_n=far_end_signal(n:-1:n-M+1).';%tap inputs(column
vector).
   y(n)=w' * x_n;%echo replica.
   e(n)=echo_signal(n)-y(n);%error signal.
   w=w+(u/(epsilon+x_n'* x_n))* e(n)* x_n;%update weights.
end

figure;
subplot(3,1,1);
plot(far_end_signal,'y','linewidth',1.2);
xlabel('Tap Index');
ylabel('Amplitude');
title('Far-End Signal');
xlim([0,50000]);% X axsis from 0-50000.
ylim([-5,5]);% Y axsis from -5-5.

subplot(3,1,2);
plot(echo_signal,'g','linewidth',1.2);
xlabel('Tap Index');
ylabel('Amplitude');
title('Echo Signal');
```

```
xlim([0,50000]);% X axsis from 0-50000.
ylim([-3,3]);% Y axsis from -3-3.

subplot(3,1,3);
plot(e,'r','linewidth',1.2);
xlabel('Tap Index');
ylabel('Amplitude');
title('Error Signal');
xlim([0,50000]);% X axsis from 0-50000.
ylim([-1,1]);% Y axsis from -1-1.


figure;
subplot(2,1,1);
plot(path,'k','linewidth',1.2);
xlabel('Tap Index');
ylabel('Weight');
title('True Echo Path');
xlim([0,90]);% X axsis from 0-90.
ylim([-0.25,0.25]);% Y axsis from -0.25-0.25.

subplot(2,1,2);
plot(w,'b','linewidth',1.2);
xlabel('Tap Index');
ylabel('Weight');
title('Estimated Echo Path');
xlim([0,110]);% X axsis from 0-110.
ylim([-0.25,0.25]);% Y axsis from -0.25-0.25.
```

This is the code for part E:

```
%PART E.
%Mahmoud Hamdan 1201134.
%Yazzed Hamdan 1201133.
%Mohammad Abu Shams 1200549.
%SEC2.

%Load the data.
load('css.mat')
%Load the impulse response.
load('path.mat')

%Concatenate Ten blocks of CSS.
css_concat=repmat(css,1,10);

M=128;%Taps.
u=0.25;%Step Size.
epsilon = 1e-6;
w=zeros(M, 1);
y=zeros(size(css_concat));%Echo.
e=zeros(size(css_concat));%Error.

%Adaptive line echo canceller.
for n=M:length(css_concat)
   x_n=css_concat(n:-1:n-M+1).';%Tap inputs (column
vector).
   y(n)=w' *x_n;%Echo replica.
   e(n)=echo_signal(n)-y(n);%Error signal.
   w=w+(u/(epsilon + x_n' * x_n))*e(n)*x_n;%Update
weights.
end

%Plot the estimated FIR channel response.
[H_estimated,frequencies_estimated]=freqz(w,1,1024,Fs);
amplitude_response_estimated=abs(H_estimated);
phase_response_estimated=angle(H_estimated);

figure;
subplot(2,1,1);
plot(frequencies_estimated,
20*log10(amplitude_response_estimated), 'b', 'linewidth', 1.2);
xlabel('Hz');
ylabel('dB');
title('Estimated FIR Channel Amplitude Response');
xlim([0,4010]);% X axsis from 0-4010.
ylim([-80,0]);% Y axsis from -80-0.

subplot(2,1,2);
plot(frequencies_estimated, phase_response_estimated, 'r',
'linewidth', 1.2);
xlabel('Hz');
ylabel('Phase (rad)');
title('Estimated FIR Channel Phase Response');
xlim([0,4010]);% X axsis from 0-4010.
ylim([-5,5]);% Y axsis from -5-5.

% Plot the given FIR system (Path) response for comparison.
[H_given, frequencies_given]=freqz(path, 1, 1024, Fs);
amplitude_response_given=abs(H_given);
phase_response_given=angle(H_given);

figure;
subplot(2,1,1);
plot(frequencies_given, 20*log10(amplitude_response_given),
'b', 'linewidth', 1.1);
xlabel('Hz');
ylabel('dB');
title('Given FIR Channel Amplitude Response');
xlim([0,4010]);% X axsis from 0-4010.
ylim([-80,0]);% Y axsis from -80-0.

subplot(2,1,2);
```

```
plot(frequencies_given,
phase_response_given,'r','linewidth', 1.1);
xlabel('Hz');
ylabel('Phase (rad)');
title('Given FIR Channel Phase Response');
xlim([0,4010]);% X axsis from 0-4010.
ylim([-5,5]);% Y axsis from -5-5.
```

This is the code for part F:

```
% PART F.
% Mahmoud Hamdan 1201134.
% Yazzed Hamdan 1201133.
% Mohammad Abu Shams 1200549.
% SEC2.

% Load the data.
load('css.mat')
% Load the impulse response.
load('path.mat')

% Concatenate Ten blocks of CSS.
css_concat = repmat(css, 1, 10);

M=128; % Taps.
lambda=0.98;
delta=1;

w=zeros(M,1);
P=delta*eye(M);

y = zeros(size(css_concat)); % Echo.
e = zeros(size(css_concat)); % Error.

% Recursive Least Squares adaptive filter.
for n = M:length(css_concat)
   x_n = css_concat(n:-1:n-M+1).'; % Tap inputs
(column vector).
   y(n) = w' * x_n; % Echo replica.
   e(n) = echo_signal(n) - y(n); % Error signal.

   % Recursive Least Squares updates.
   k = ((lambda + x_n' * P * x_n)^-1) * P * x_n;
   alpha = e(n);
   w = w + alpha * k; % Weight update.
   P = (lambda^-1) * P - (lambda^-1) * k * x_n' *
P; % Inverse correlation matrix update.
end

% Plot the estimated FIR channel response.
[H_estimated, frequencies_estimated] = freqz(w, 1, 1024, Fs);
amplitude_response_estimated = abs(H_estimated);
phase_response_estimated = angle(H_estimated);

figure;
subplot(2,1,1);
plot(frequencies_estimated,
20*log10(amplitude_response_estimated), 'b', 'linewidth', 1.1);
xlabel('Hz');
ylabel('dB');
title('RLS - Estimated FIR Channel Amplitude Response');
xlim([0,4010]);% X axsis from 0-4010.
ylim([-80,0]);% Y axsis from -80-0.

subplot(2,1,2);
plot(frequencies_estimated, phase_response_estimated, 'r',
'linewidth', 1.1);
xlabel('Hz');
ylabel('Phase (rad)');
title('RLS - Estimated FIR Channel Phase Response');
xlim([0,4010]);% X axsis from 0-4010.
ylim([-5,5]);% Y axsis from -5-5.

% Plot the given FIR system (Path) response for comparison.
[H_given, frequencies_given] = freqz(path, 1, 1024, Fs);
amplitude_response_given = abs(H_given);
phase_response_given = angle(H_given);

figure;
subplot(2,1,1);
plot(frequencies_given, 20*log10(amplitude_response_given),
'k', 'linewidth', 1.1);
xlabel('Hz');
ylabel('dB');
title('Given FIR Channel Amplitude Response');
xlim([0,4010]);% X axsis from 0-4010.
ylim([-80,0]);% Y axsis from -80-0.

subplot(2,1,2);
plot(frequencies_given, phase_response_given, 'g', 'linewidth',
1.1);
xlabel('Hz');
ylabel('Phase (rad)');
title('Given FIR Channel Phase Response');
xlim([0,4010]);% X axsis from 0-4010.
ylim([-5,5]);% Y axsis from -5-5.
```