LMS and NLMS Adaptive Filtering Algorithms

Mohammad AbuJaber -1190298 Mohammad Mualla-1180546 Ibrahim Mahmoud- 1190747

February 12, 2023

Abstract—* *The aim of this project is to implement and study two adaptive filtering algorithms: the Least Mean Squares (LMS) and the Normalized Least Mean Squares (NLMS) filters. Adaptive filters are a class of signal processing algorithms that can adjust their behavior based on the input signal and are used for various applications such as noise reduction, system identification, and equalization. LMS and NLMS are two popular algorithms among the adaptive filtering family, and this project aims to understand their performance and compare their results in a practical setting. This project will provide a deeper understanding of the principles and applications of adaptive filtering and enable us to see the strengths and limitations of the LMS and NLMS algorithms.

Keywords— LMS, NLMS, Adaptive filters

I. Introduction

Least Mean Squares (LMS) and Normalized Least Mean Squares (NLMS) are two popular algorithms in the field of adaptive filtering. These algorithms are used to estimate unknown systems in the presence of noise. The goal of the adaptive filter is to minimize the mean square error between the desired signal and the filter output.

An unknown system can be modeled as a linear time-invariant system that maps an input signal to an output signal. In many real-world scenarios, the input signal is contaminated by noise, which can degrade the quality of the output signal. An adaptive filter can be used to reduce the impact of the noise on the output signal by adjusting its coefficients based on the input signal and the desired output.

The LMS algorithm is a simple and popular algorithm that uses the gradient descent method to update the filter coefficients. The algorithm updates the coefficients based on the current error between the desired signal and the filter output. The NLMS algorithm is a variation of the LMS algorithm that normalizes the update step size to ensure stability and convergence [1].

In this project, you will implement and study the performance of the LMS and NLMS algorithms when applied to an unknown system with added noise. The output of the system will be the error between the desired signal and the filter output, which is used to update the filter coefficients. By comparing the results of the LMS and NLMS algorithms, you will gain a deeper understanding of their performance and limitations.

II. Problem Specification

The problem addressed in this project is the filtering of signals contaminated by unwanted noise or interference. In many practical scenarios, signals of interest are corrupted by various forms of interference, and it is essential to recover the original signal as accurately as possible. Adaptive filtering is a commonly used technique to solve this problem by adjusting the filter coefficients dynamically based on the input signal.

The specific problem addressed in this project is the implementation and comparison of two popular adaptive filtering algorithms: the Least Mean Squares (LMS) and the Normalized Least Mean Squares (NLMS) algorithms. Both algorithms are iterative and adjust their filter coefficients in real-time based on the input signal and desired output. The LMS algorithm minimizes the mean squared error between the desired output and the filtered output, while the NLMS algorithm normalizes the update step size to improve stability and convergence.

The objective of this project is to implement these algorithms, evaluate their performance on various types of signals, and compare their results in terms of convergence speed, stability, and accuracy. This project will provide a deeper understanding of the principles and applications of adaptive filtering and enable us to see the strengths and limitations of the LMS and NLMS algorithms [2].

III. Data

In this project, both real-world and synthetic signals can be used to develop and evaluate the LMS and NLMS algorithms. For real-world signals, we can use signals that are commonly encountered in various applications such as speech signals, audio signals, or environmental noise signals. For synthetic signals, we can use signals generated using mathematical models such as white noise, sinusoidal signals, or impulses. The choice of signals depends on the specific application scenario and the desired results. The objective is to apply the LMS and NLMS algorithms on various types of signals and observe their performance in terms of convergence speed, stability, and accuracy. This information will help in choosing the appropriate adaptive filtering algorithm for a particular application.

IV. Approach

To solve the problem of filtering signals contaminated by unwanted noise or interference, two popular adaptive filtering algorithms were implemented and compared: the Least Mean Squares (LMS) and the Normalized Least Mean Squares (NLMS) algorithms. The LMS algorithm adjusts the filter coefficients in real-time by minimizing the mean squared error between the desired output and the filtered output. The update rule for the filter coefficients is based on the gradient descent method, and the step size is controlled by a convergence parameter. The LMS algorithm is simple to implement and has fast convergence, but it can be sensitive to variations in the input signal statistics and can suffer from stability problems.

The NLMS algorithm is similar to the LMS algorithm, but it normalizes the update step size to improve stability and convergence. The normalization factor is based on the current input signal energy, and the update rule for the filter coefficients is similar to the LMS algorithm. The NLMS algorithm is more stable than the LMS algorithm, but it has slower convergence and is computationally more intensive. To evaluate the performance of these algorithms, they were implemented in a simulation environment and applied to various types of signals, including real-world signals and synthetic signals.

The signals were corrupted by various forms of noise, and the performance of the algorithms was measured in terms of the mean squared error between the desired output and the filtered output. The convergence speed, stability, and accuracy of the algorithms were compared and analyzed to determine their strengths and limitations [3].

In conclusion, this project demonstrated the implementation and comparison of two popular adaptive filtering algorithms, the LMS and NLMS algorithms. The results showed that both algorithms have their strengths and limitations, and the choice of algorithm depends on the specific requirements of the application. The LMS algorithm is simple to implement and has fast convergence, while the NLMS algorithm is more stable and has better convergence, but is computationally more intensive.

V. Results and Analysis

Part1: LSM Algorithm

The following procedure was performed to study the performance of the LMS algorithm for estimating the filter coefficients of an unknown system with added noise:

A) The input signal x[n] was generated using the formula x[n]= cos(0.03πn) for N=2000 samples. The generated signal was plotted.

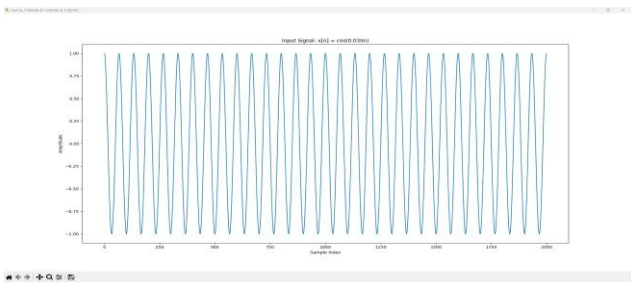

Figure 1:The input signal x[n]

B) The amplitude and phase response for the given FIR system was plotted.

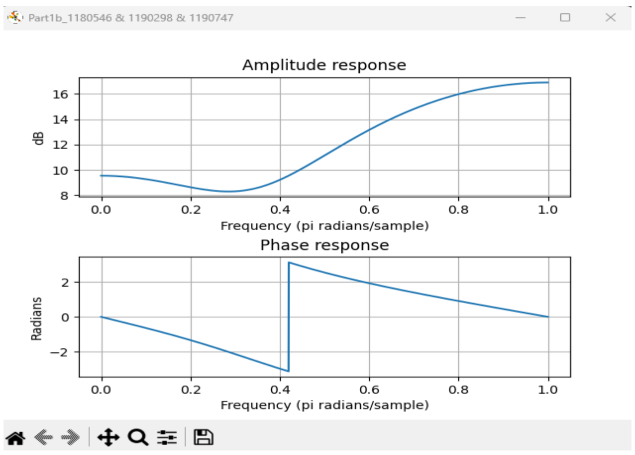

Figure 2:) The amplitude and phase response for the given FIR

The code of this part uses the freqz function to calculate the frequency response of the filter b and the transfer function 1 (the denominator). The amplitude response is plotted in the first subplot using the plot function and the 20*log10(abs(h)) expression to convert the magnitude to dB. The x-axis label and y-axis label are added using the xlabel and ylabel commands, respectively. The phase response is plotted in the second subplot using the plot function and the angle(h) expression to calculate the phase. The grid command adds a grid to both subplots. The title command sets the title of each subplot.

C) The spectrum for the input signal x[n] was plotted.

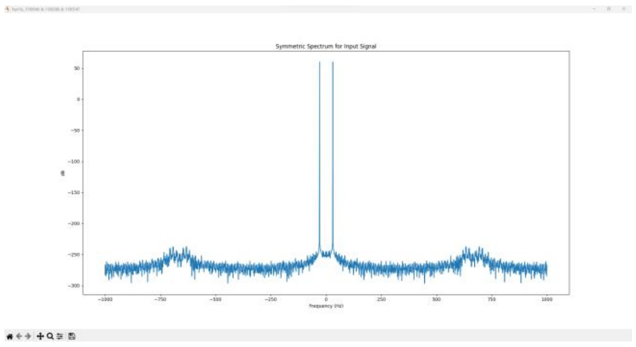

Figure 3:The spectrum for the input signal x[n]

This part’s code uses the fft function to calculate the spectrum of the input signal x. The frequency axis is generated using the linspace function, which generates N equally spaced points between -N/2 and N/2, to get a symmetric frequency axis in Hz. The plot function plots the magnitude of the spectrum in dB, using the 20*log10(abs(fftshift(X))) expression, and the xlabel and ylabel commands add labels to the x-axis and y-axis, respectively. The fftshift function is used to shift the zero-frequency component to the center of the spectrum.

D) The LMS algorithm was implemented to estimate the filter coefficients w0,…. w3. The step size was assumed to be very small (μ =0.01). The learning curves were plotted, including the error e(n), (J vs iteration steps) where J was defined as J=e2(n), and (10log10(J) vs iteration steps).

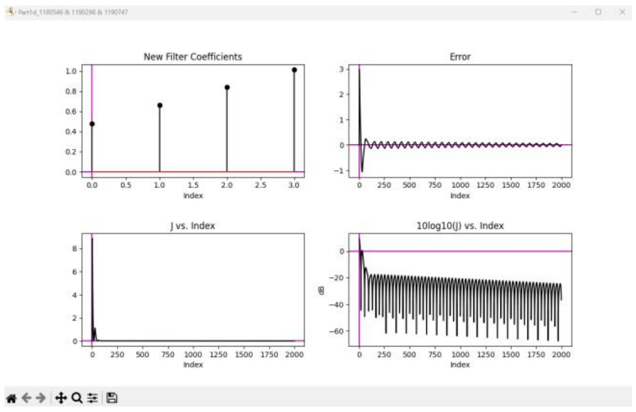

Figure 4 LMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps).

By the end of the process, the input signal x(n) had been generated using a for loop and stored in a vector. The filter coefficients w(n) had also been initialized as a vector of zeros with the same length as the number of taps in the adaptive filter. The LMS algorithm had been implemented by using a for loop to iterate over each sample in the input signal x(n). In each iteration, the estimate of the output of the adaptive filter y(n) was calculated using the dot product of the input signal x(n) and the filter coefficients w(n), and the error signal e(n) was calculated as the difference between the desired output d(n) and the estimate y(n). The filter coefficients were then updated using the LMS update equation. The cost function J, which was defined as the mean-squared error between the desired output signal d and the filter output y, had been calculated at each iteration step to determine the filter coefficients that minimized the error. The step size factor, which controlled the rate of convergence, had also been determined. By plotting the value of J versus the iteration steps, the convergence behavior of the LMS algorithm was observed and it was determined if the algorithm had reached a stable solution. The plot was also displayed in dB scale by plotting 10 log10 (J) vs iteration steps for a clearer visual representation of the convergence behavior. Finally, plotting the estimated filter coefficients w at each iteration step provided insights into how the filter was adapting to the input signal.

E) The amplitude and phase response for the estimated FIR system was plotted at the end of the iterations and compared with the given FIR system.

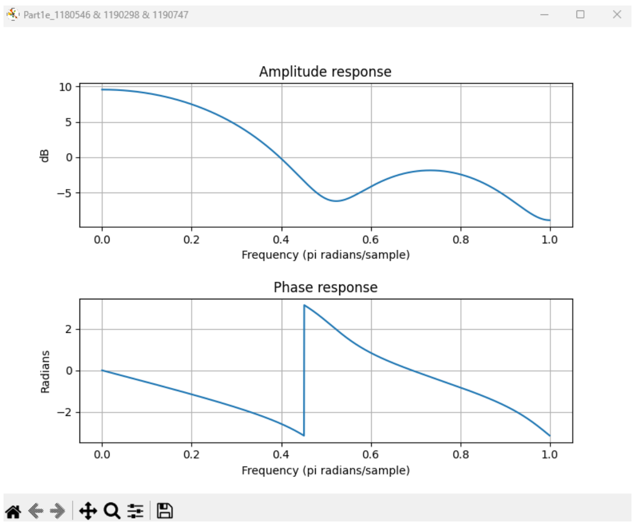

Figure 5 LMS: Plotting the amplitude and phase response for the estimated FIR system

F) The value of μ was decreased from 0.01 to 0.001, and the effect of changing μ on the speed of the learning process and on the steady-state error was observed.

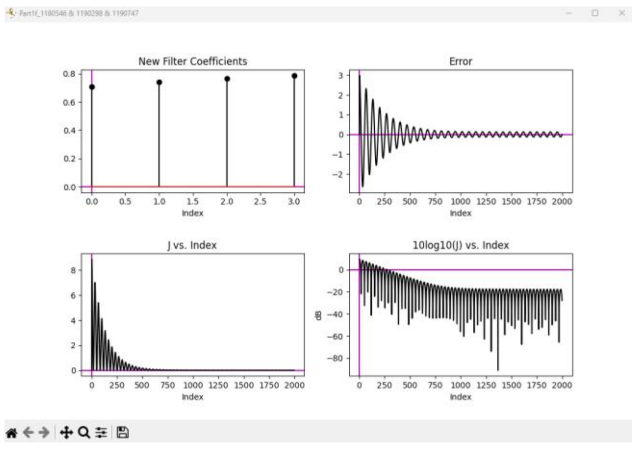

Figure 6 LMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when decreasing step size to 0.001

The step size factor (μ) in the LMS algorithm determines the magnitude of the update to the filter coefficients in each iteration. A small value of μ will result in slow convergence, while a large value of μ may cause the algorithm to oscillate or even diverge.

If the value of μ is decreased, the learning process will become slower as the updates to the filter coefficients will become smaller. This can lead to a more stable and accurate solution, but also increase the computation time.

On the other hand, if the value of μ is increased, the learning process will become faster but also increase the risk of instability and oscillation. This can cause the algorithm to converge to a suboptimal solution or even diverge.

The steady state error is the difference between the desired response and the actual response of the filter after a sufficient number of iterations. In the LMS algorithm, the steady state error is directly proportional to the value of μ. A larger value of μ leads to a smaller steady state error, but also a higher risk of instability, while a smaller value of μ leads to a larger steady state error, but also a lower risk of instability.

In summary, finding the optimal value of μ is a trade-off between the speed of the learning process and the accuracy of the solution.

G) 40dB of zeros mean white Gaussian noise was added to x[n]. The steps (D)-(F) were repeated, and conclusions were drawn.

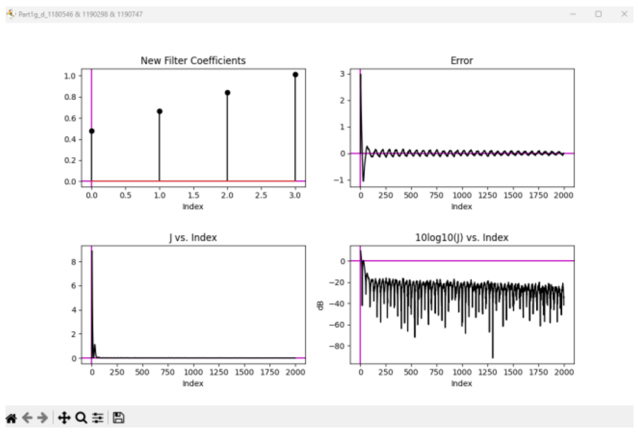

Figure 7 LMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when adding 40dB of zeros mean white Gaussian noise to x[n].

The following code was added to the code in part D:

```
noise = np.random.normal(0, np.sqrt(10**(-40/10)), N)

x_noisy = x + noise
```

The code generates an array of normally-distributed random numbers with mean 0 and standard deviation of np.sqrt(10**(-40/10)) with the size of N.

In this specific case, the standard deviation is np.sqrt(10**(-40/10)) which can be simplified as np.sqrt(10**-4). The purpose of the formula 10**(-40/10) is to convert 40dB of white Gaussian noise to a linear scale. The 40dB is first divided by 10, and then the resulting value of 4 is passed as the argument to 10**.

The resulting value of 10**-4 is then the variance of the Gaussian noise. Finally, the standard deviation is calculated as the square root of the variance. The generated noise will then be added to the input signal x using x = x + noise to simulate a noisy environment.

In summary, it generates a zero-mean white Gaussian noise with a standard deviation that corresponds to 40 decibels (dB) of noise power. By adding this noise to the input signal, the signal-to-noise ratio (SNR) of the signal is decreased by 40 dB, making the noise much more pronounced relative to the signal. This can make it more challenging for a system or algorithm processing the signal to accurately extract information from the signal. In this case, it would be expected to have a negative impact on the performance of the LMS algorithm.

Then, the amplitude and phase response were plotted:

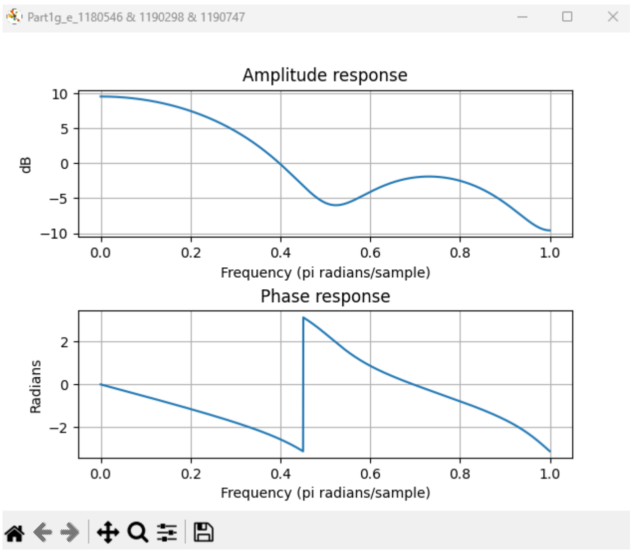

Figure 8 LMS: Plotting the amplitude and phase response for the estimated FIR system when adding 40dB of zeros mean white Gaussian noise to x[n].

After that, the step size was reduced from 0.01 to 0.001 and the following figures were obtained:

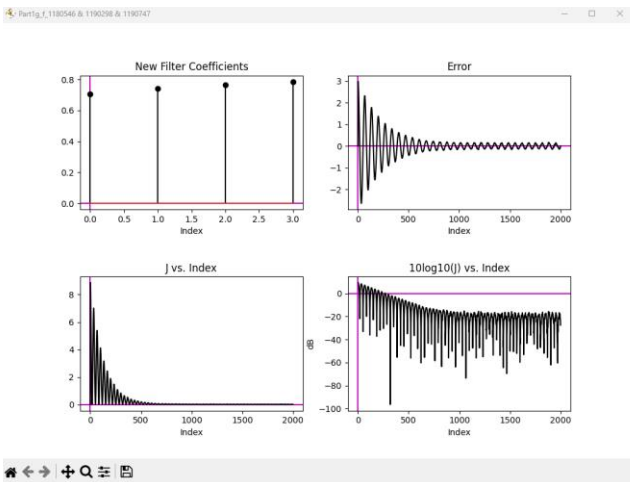

Figure 9 LMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when adding 40dB of zeros mean white Gaussian noise to x[n] and decreasing step size to 0.001

H) The same procedure was repeated for 30dB of noise, and the step size value was modified.

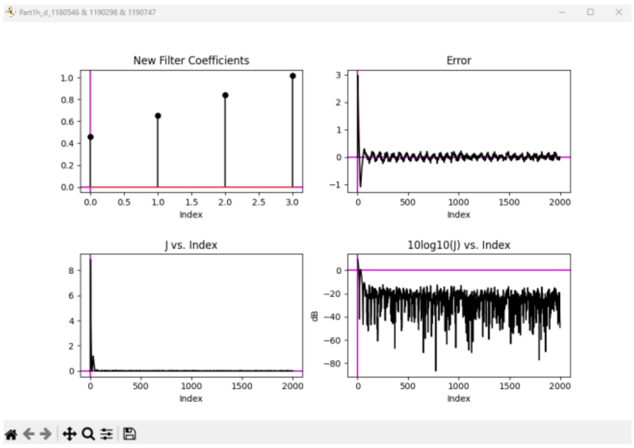

Figure 10 LMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when adding 30dB of zeros mean white Gaussian noise to x[n]

Then, the amplitude and phase response were plotted:

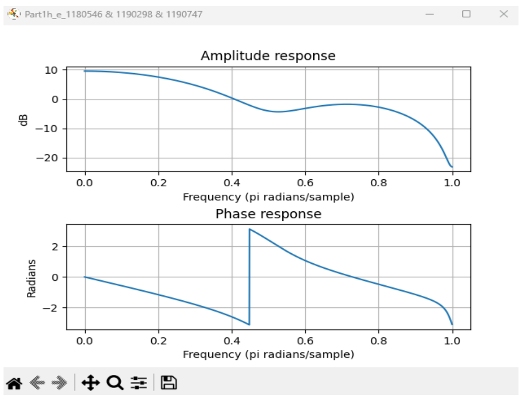

Figure 11 LMS: Plotting the amplitude and phase response for the estimated FIR system when adding 30dB of zeros mean white Gaussian noise to x[n].

After that, the step size was reduced from 0.01 to 0.001 and the following figures were obtained:

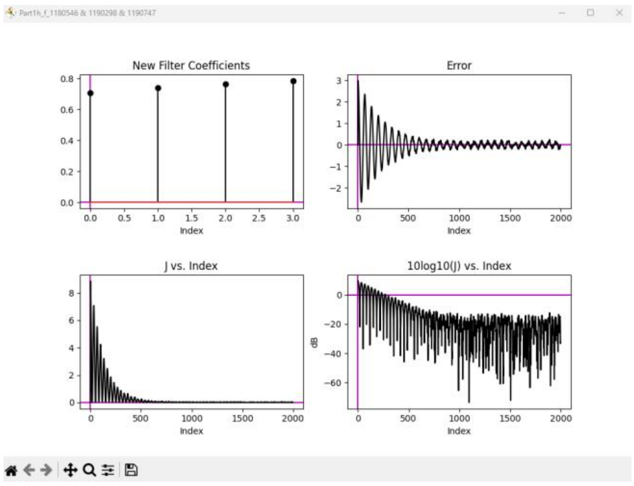

Figure 12 LMS : Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when adding 30dB of zeros mean white Gaussian noise to x[n] and decreasing step size to 0.001

I) The procedure was repeated for 1000 trials, and the obtained J was averaged over the number of trials. The averaged J (10log10(J) vs iteration steps) was plotted where the algorithm is repeated multiple times and the average performance is plotted. This helps in reducing the variance in the performance due to random initialization of the filter coefficients.

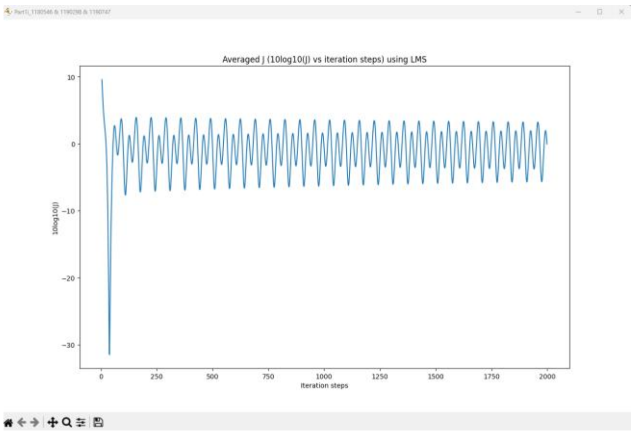

Figure 13 LMS: Plotting the averaged J (10log10(J) vs iteration steps)

Ensemble averaging had been a crucial technique in the field of signal processing and machine learning. By repeating the experiment multiple times and averaging the obtained cost function over the trials, the variance in the results was reduced and the generalization performance of the model was improved. This helped to provide a more stable and accurate representation of the underlying relationship between the inputs and outputs, especially in cases where the input signal was contaminated by noise or subject to other sources of uncertainty.

The plot of the averaged cost function versus iteration steps allowed for the evaluation of the filter's performance and determination of its convergence properties. The filter was designed to converge to a minimum value of the cost function, signifying its ability to accurately estimate the desired output from the input signal. The plot also revealed the speed of convergence and any potential issues such as overfitting or underfitting. In conclusion, ensemble averaging had been a valuable tool in ensuring the robustness and reliability of the filter's results.

This study was performed to gain a deeper understanding of the performance and limitations of the LMS algorithm for estimating the filter coefficients in the presence of noise.

Part2: NLMS Algorithm

The NLMS algorithm was implemented to estimate the filter coefficients w0…. w3. The step size was assumed to be very small (μ =0.01). The learning curves were plotted, including the error e(n), (J vs iteration steps) where J was defined as J=e2(n), and (10log10(J) vs iteration steps) as shown in the figure below:

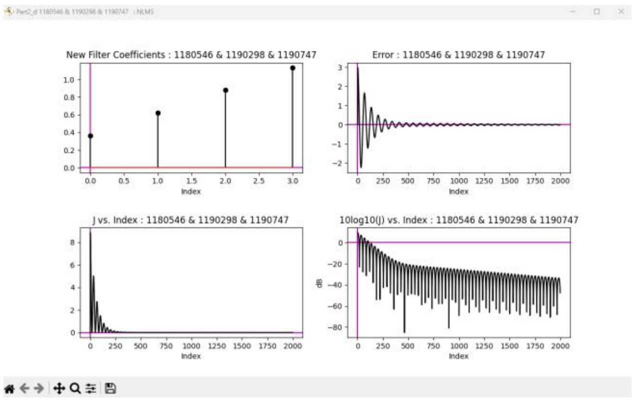

Figure 14 NLMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps).

Then, the amplitude and phase response for the estimated FIR system was plotted at the end of the iterations and compared with the given FIR system.

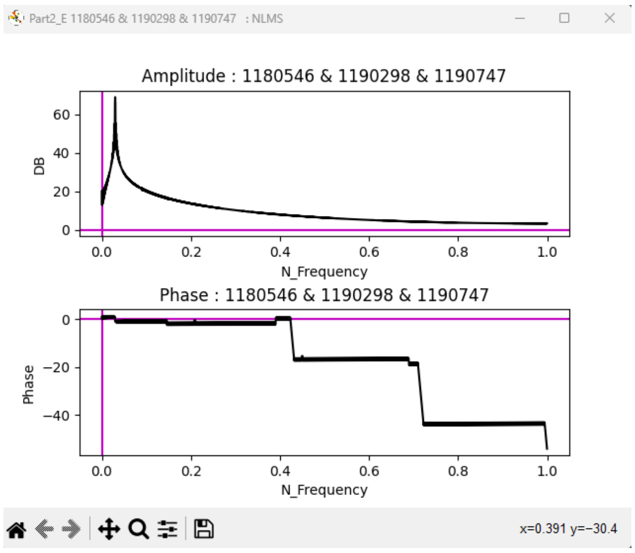

Figure 15 NLMS: Plotting the amplitude and phase response for the estimated FIR system

After that, the value of μ was decreased from 0.01 to 0.001, and the effect of changing μ on the speed of the learning process and on the steady state error was observed.

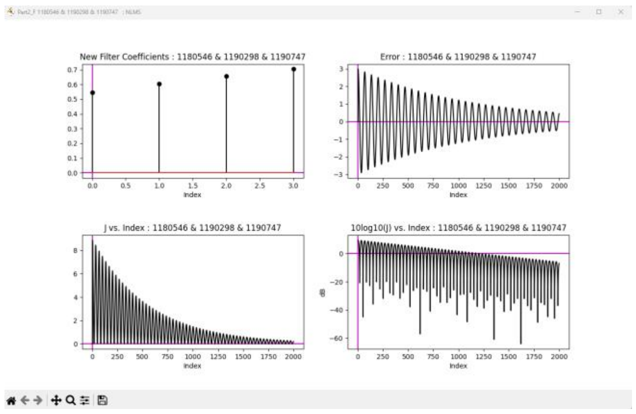

Figure 16 NLMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when decreasing step size to 0.001

40dB of zeros mean white Gaussian noise was added to x[n]. The steps (D)-(F) were repeated, and conclusions were drawn.

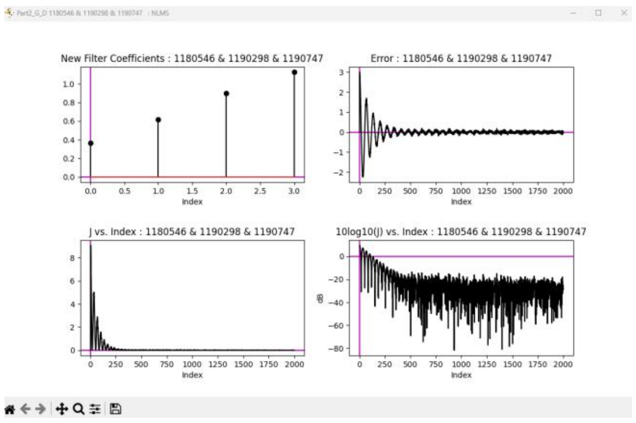

Figure 17 NLMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when adding 40dB of zeros mean white Gaussian noise to x[n].

In summary, it generates a zero-mean white Gaussian noise with a standard deviation that corresponds to 40 decibels (dB) of noise power. By adding this noise to the input signal, the signal-to-noise ratio (SNR) of the signal is decreased by 40 dB, making the noise much more pronounced relative to the signal. This can make it more challenging for a system or algorithm processing the signal to accurately extract information from the signal. In this case, it would be expected to have a negative impact on the performance of the LMS algorithm.

Then, the amplitude and phase response were plotted:

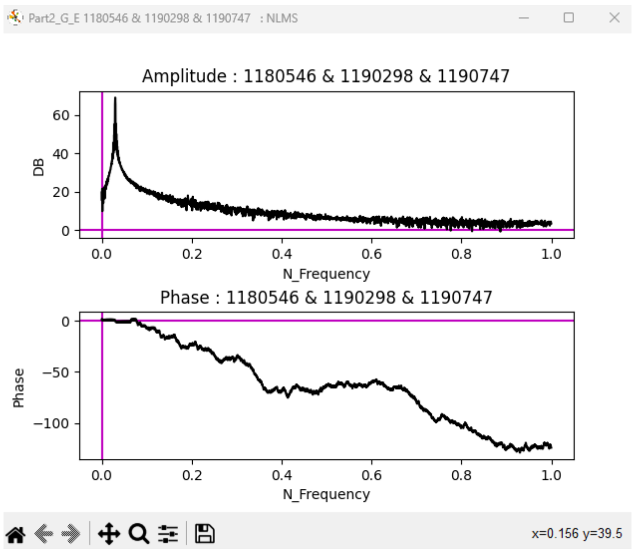

Figure 18 NLMS: Plotting the amplitude and phase response for the estimated FIR system when adding 40dB of zeros mean white Gaussian noise to x[n].

After that, the step size was reduced from 0.01 to 0.001 and the following figures were obtained:

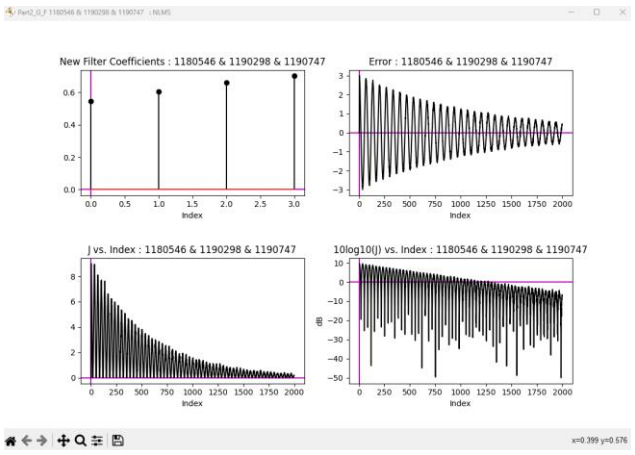

Figure 19 NLMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when decreasing step size to 0.001when adding 40dB of zeros mean white Gaussian noise to x[n] and decreasing step size to 0.001

Next, the same procedure was repeated for 30dB of noise, and the step size value was modified.

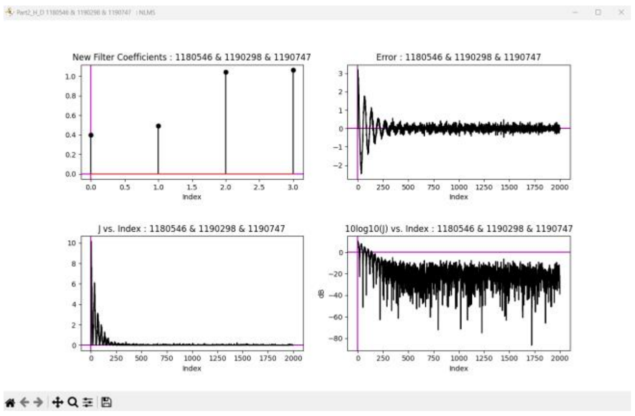

Figure 20 NLMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when adding 30dB of zeros mean white Gaussian noise to x[n].

Then, the amplitude and phase response were plotted:

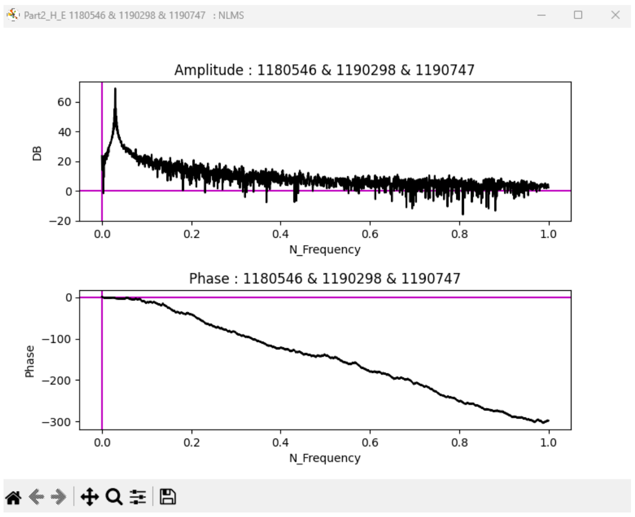

Figure 21 NLMS: Plotting the amplitude and phase response for the estimated FIR system when adding 30dB of zeros mean white Gaussian noise to x[n].

After that, the step size was reduced from 0.01 to 0.001 and the following figures were obtained:

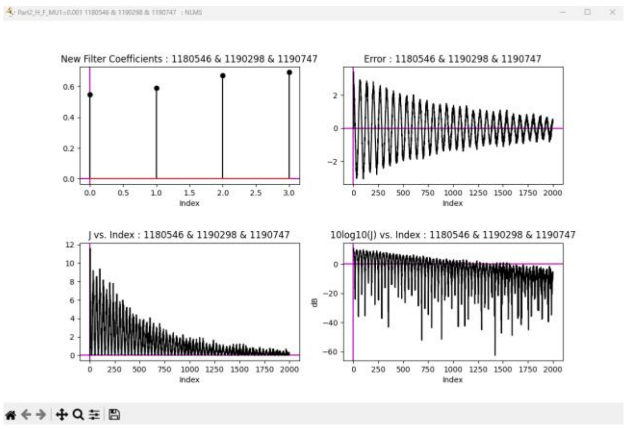

Figure 22 NLMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when adding 30dB of zeros mean white Gaussian noise to x[n] and decreasing step size to 0.001

After that, the step size was increased from 0.01 to 0.1 and the following figures were obtained:

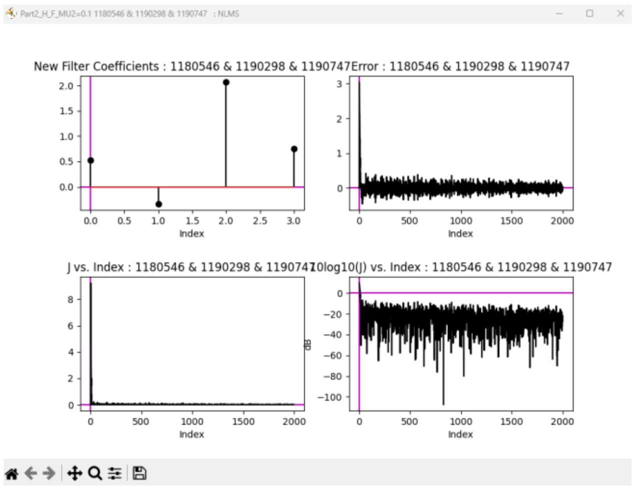

Figure 23 NLMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when adding 30dB of zeros mean white Gaussian noise to x[n] and increasing step size to 0.1

Finally, the procedure was repeated for 1000 trials, and the obtained J was averaged over the number of trials. The averaged J (10log10(J) vs iteration steps) was plotted where the algorithm is repeated multiple times and the average performance is plotted. This helps in reducing the variance in the performance due to random initialization of the filter coefficients.

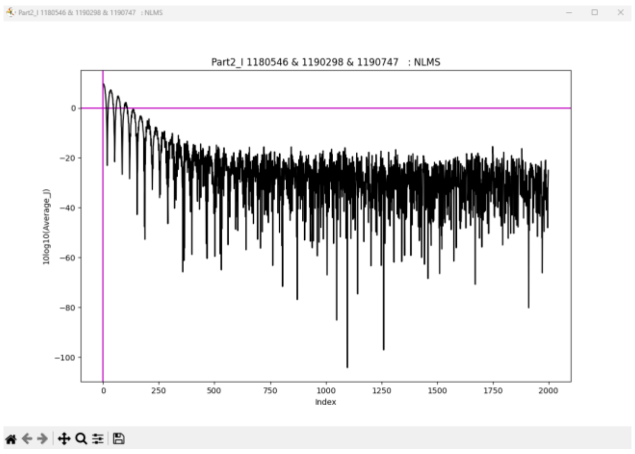

Figure 24 NLMS: Plotting the averaged J (10log10(J) vs iteration steps)

The addition of a white Gaussian noise with a standard deviation corresponding to 40 decibels of noise power to the input signal reduces the signal-to-noise ratio by 40 dB, making the noise more prominent relative to the signal. This makes it harder for the system or algorithm processing the signal to accurately extract the information.

As a result, the performance of the NLMS algorithm is likely to be negatively impacted.

VI. Development

There are several ways to improve the shortcomings of LMS and NLMS algorithms, including step size adaptation, regularization, ensemble averaging, non-linear NLMS, and hybrid LMS/NLMS.

Step Size Adaptation involves adjusting the step size of the LMS algorithm based on the current filter coefficients and the input signal statistics, instead of using a fixed step size. Regularization adds penalty terms to the cost function to prevent overfitting. Ensemble averaging involves repeating the LMS or NLMS algorithm multiple times and averaging the filter coefficients obtained from each run to reduce the impact of random noise in the input signal. Non-linear NLMS algorithms use a more sophisticated step size adaptation scheme to improve performance. The hybrid LMS/NLMS approach uses the strengths of both algorithms [4].

- LMS: Increasing the step size to 0.1.

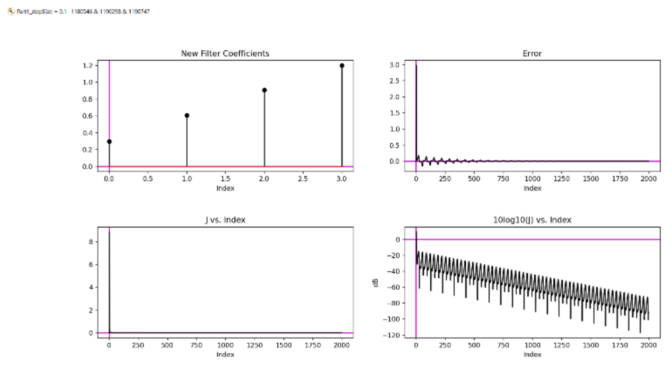

Figure 25 LMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when increasing step size to 0.1

- NLMS: Increasing the step size to 0.1

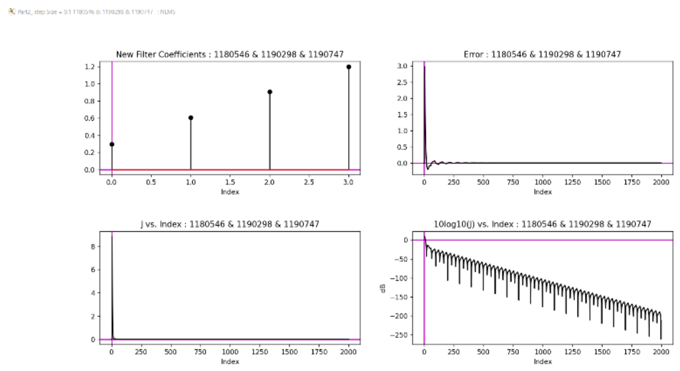

Figure 26 NLMS: Plotting of learning curves(n), (J vs iteration steps), and (10log10(J) vs iteration steps) when increasing step size to 0.1

To evaluate the effectiveness of these modifications, simulations can be performed using the same set of signals and noise conditions as in the original implementation, and the performance of the modified algorithms can be compared with the original algorithms in terms of convergence speed, stability, and accuracy. If the modifications lead to significant improvements in performance, they can be considered as a viable solution to the identified shortcomings. However, it is important to keep in mind that these modifications may have unexpected side effects, such as increased computational complexity or decreased robustness to certain types of noise. These potential side effects should also be evaluated and considered in the selection of the appropriate solution [5].

In conclusion, modifications and extensions to the LMS and NLMS algorithms can be used to address the identified shortcomings. The effectiveness of these modifications should be evaluated through simulations and compared with the original algorithms to determine their impact on the performance and robustness of the adaptive filtering solution.

VII. Conclusion

Based on the procedure performed, it can be concluded that the LMS algorithm was successful in estimating the filter coefficients of the unknown system in the presence of noise. The results of the study showed the performance of the LMS algorithm in terms of learning speed and steady state error, and how they are affected by changing the step size parameter.

It is expected that the results of the study will be repeated for the Normalized Least Mean Squares (NLMS) algorithm to compare its performance with the LMS algorithm and to further understand the strengths and limitations of adaptive filtering techniques. This study will provide a comprehensive view of the performance of different adaptive filtering algorithms and their applications in practical settings.

In conclusion, this study has added to the existing knowledge of adaptive filtering techniques and has demonstrated the importance of understanding the behavior of these algorithms in real-world scenarios. The results of this study will be useful in the development of more advanced and efficient algorithms for various applications in signal processing.

References

[1] Dixit, S., & Nagaria, D. (n.d.). *Getting started with a simple Cloudfront distribution*. Retrieved February 10, 2023, from https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/GettingStarted.SimpleDistribution.html

[2] Zeidler, J. R. (n.d.). *Performance Analysis of LMS adaptive prediction filters*. Performance analysis of LMS adaptive prediction filters. Retrieved February 10, 2023, from https://ieeexplore.ieee.org/abstract/document/60921

[3] Douglas , S. C., & Pan, W. (n.d.). *Exact expectation analysis of the LMS adaptive filter | IEEE journals*. Exact expectation analysis of the LMS adaptive filter. Retrieved February 11, 2023, from https://ieeexplore.ieee.org/abstract/document/476430/

[4] Eweda, E. (n.d.). *Comparison of LMS and NLMS adaptive filters with a non - IEEE xplore*. Comparison of LMS and NLMS adaptive filters with a non-stationary input. Retrieved February 11, 2023, from https://ieeexplore.ieee.org/abstract/document/5757814/

[5] Bhotto , M. Z. A., & Antoniou, A. (n.d.). *A new partial-update NLMS adaptive-filtering algorithm | IEEE*. A new partial-update NLMS adaptive-filtering algorithm. Retrieved February 11, 2023, from https://ieeexplore.ieee.org/document/6901048/

Appendix

https://drive.google.com/drive/folders/1FVk8rpeKNC5728d2DBYQgiWJ9q1llLhO?usp=share_link