System Identification

Mahmoud Qaisi
Birzeit University
Palestine
1190831@student.birzeit.edu

Ibrahim Nobani
Birzeit University
Palestine
1190278@student.birziet.edu

***Abstract*—In this paper the problem of system identification, a powerful method in signal processing called adaptive filtering makes it possible to identify unidentified systems in real time is tackled. It is investigate how two well-liked adaptive filtering algorithms—Recursive Least Squares (RLS) and Least Mean Squared (LMS)—can be used to identify systems. While the LMS algorithm is well known for its simplicity and ease of use, the RLS method is well known for its quick convergence and minimal steady-state error.**

The performance of these algorithms is compared by applying them to a system identification problem, in which the objective is to identify an unknown system from its input and output signals. First, a set of input signals were generated and applied to an unknown system, whose transfer function is unknown. Next, the RLS and LMS algorithms were used to estimate the transfer function of the unknown system.

***Index Terms*—Adaptive filter, least mean square, recursive least squares, Transfer function, System Identification.**

I. Introduction

Understanding and modeling the behavior of dynamic systems requires the use of system identification. It has numerous real-world uses. System identification, for example, can provide information about a system's parameters, which can later be utilized to develop control systems that are tailored for that system. This can increase the control system's performance and stability. Adaptive filters are a system identification approach. Based on observed input/output data, they are used to predict the parameters of a mathematical model. The least mean square (LMS) method has long been a workhorse of adaptive filtering, with numerous successful applications since its inception in 1959. The LMS converges to the Wiener-Hopf solution wop (i.e. the optimal solution) based on the reduction of an instantaneous approximation to the mean square error, and its 'instantaneous' character makes it an algorithm of choice for non-stationary signal processing. Another frequent method for obtaining the Wiener solution, wop, is to use the method of least squares, specifically the recursive least squares algorithm (RLS). Varying the step size in the LMS is the main focus to accelerate the convergence and reduce ready state misalignment. For RLS on the other side the main focus has been on effective implementation and numerical robustness.

II. Problem Specification

The aim is to identify the system that maps the given input and output signals, This is achieved using adaptive filter algorithms like Least Mean Squared (LMS) and Recursive Least Squares (RLS).

The technical issue that arises in a number of applications is the system identification, or determining the relationship between its input-output response. We set up the system configuration as Fig. 1 shows below, in order to locate the filter coefficients that represent the unknown system. Assuming that the unknown system is time invariant, which indicate that the coefficients of its impulse response are constants and finite such that the desired response is given by:

$$d[n] = \sum h[k]x[n-k] \tag{1}$$

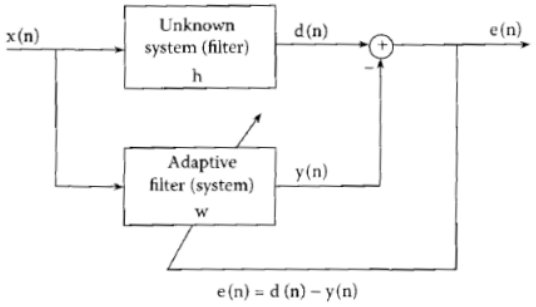

Fig. 1. System Identification Problem

III. Solution Approach

The approach to identifying a system where the input and the output data of the system are provided, is using, as mentioned previously, adaptive filters that dynamically adjusting the transfer function w until it can generate an output that matches the given output from the unknown system.

The output of an adaptive FIR filter with the same number of coefficients, M, is given by

$$y[n] = \sum w[k]x[n-k] \tag{2}$$

Two types of adaptive filters were used and the LMS and the RLS. Both were implemented using python.

- LMS algorithm:
- RLS algorithm:

IV. Evaluation Criteria

The evaluation criteria used for system identification using adaptive filters includes the accuracy and speed of convergence of the algorithm, also the robustness if their exits a disturbance

Inputs: M: filter length
μ: step−size factor
x(n): input data to the adaptive filter of length N (Vector)
w (0): initialization filter (vector) =zeros of length M
Outputs at each iteration (n) y(n)=wT(n)x(n)
e(n)=d(n)−y(n)
w(n+ 1) =w(n)+ 2μe(n)x(n), the updated filter coefficients

Fig. 2. LMS algorithm [2]

Inputs: M: filter length
ρ: step−size factor
x(n): input data to the adaptive filter of length N (Vector)
w (0): initialization filter (vector) =zeros of length M
Outputs at each iteration (n) y(n)=wT(n)x(n)
e(n)=d(n)−y(n)
P(n+ 1) = P(n) − (ρ ∗ P(n)x(n)x(n)T ∗ P(n)) / (1 + x(n)T ∗ P(n) ∗ x(n)), the updated autocorrelation matrix
w(n+ 1) =w(n) + P(n+ 1) ∗ x(n) ∗ e(n), the updated filter coefficients

Fig. 3. LRS algorithm [2]

such as noise. Another factor is the computational complexity of each algorithm.

Both LMS and RLS algorithms have their own advantages and disadvantages, The LMS is much simpler and can be easily applied while taking longer time to converge, The RLS on the other hand has increased complexity and computational cost with faster convergence. LMS has Larger steady state error that RLS with respect to the unknown system [1].

V. Results & Analysis

After both adaptive filters using Python the following set of results were refined and analyzed. The input for the system at hand is:

$$x[n] = cos(0.03n) for N = 2000 samples. \quad (3)$$

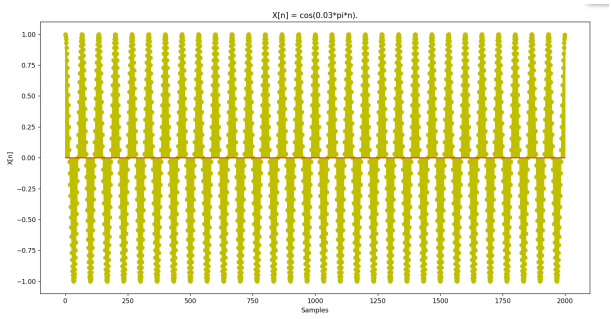

Fig. 4. X[n]

The amplitude and phase response was plotted for the given FIR system, which is the unknown system, these plots will be used later to compare with the output generated from the algorithms to test the validity of the system.

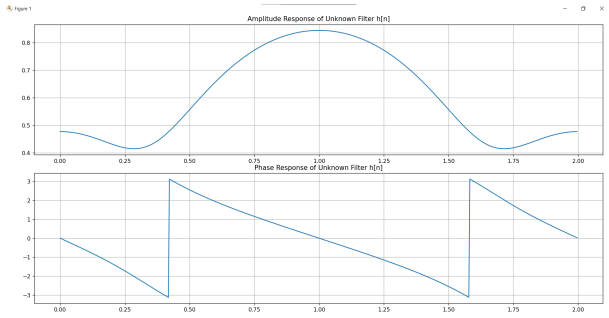

Fig. 5. Frequency Response of h[n]

After that the spectrum for X[n] was plotted.

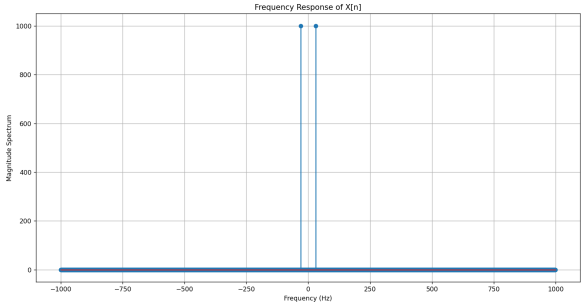

Fig. 6. X(f)

In order to estimate the coefficients w0,...,w3 the LMS algorithm was implemented, with the learning rate initialized to 0.01.

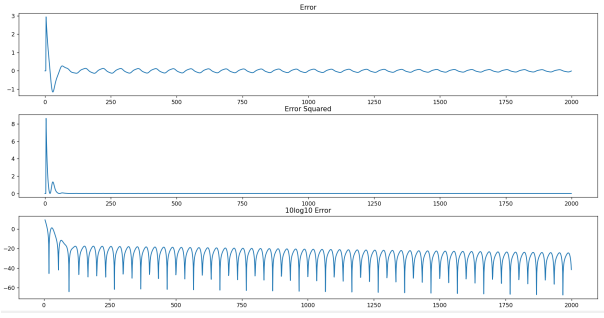

Fig. 7. LMS error rates when μ =0.01

The adaptive filter will eventually produce final values for the filter coefficients. The filter displayed the following behaviour in figure 8:

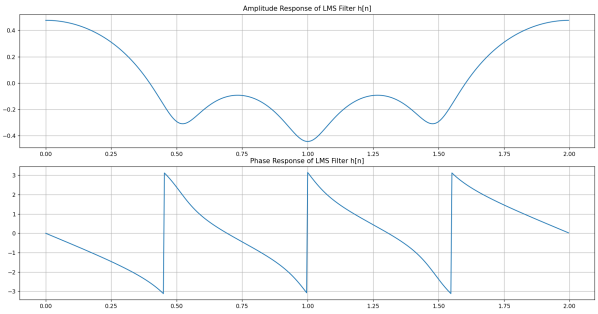

Fig. 8. LMS filter frequency response

The phase response of the LMS filter should be as close as possible to the unknown system phase response, in the figure 9 below that compares the two phase response, The similarity between the two is noticed.

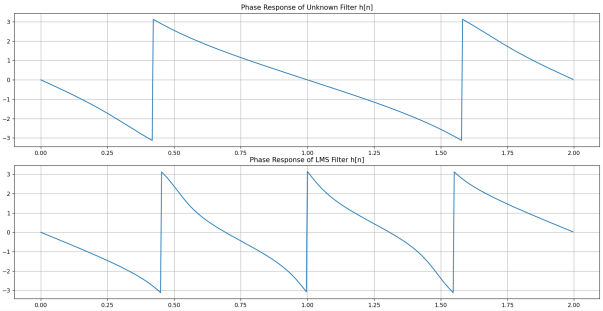

Fig. 9. LMS filter Phase & Unknown Filter phase

This plot in figure 10 shows the unknown filter plot and the adaptive filter plot using LMS, the adaptation process from the adaptive system to the unknown system can be noticed in the plot, as the two systems come together.

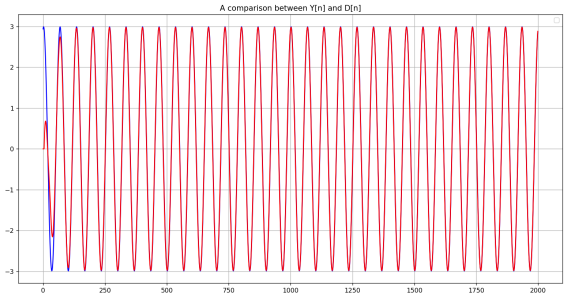

Fig. 10. Y[n] & D[n]

Upon decreasing the learning rate in the LMS the error rate also decreases, which results in the system being more stable and accurate of its parameters, but the adaptation process is much slower because of this decrease of step size, thus means more computation time. (If the learning rate is too small the system might not converge or converge slowly), here we decreased the learning rate from 0.01 to 0.005, we can see the error rate for this in the figure below, it can be noticed the decrease in this error thus increase on the system overall accuracy estimate.

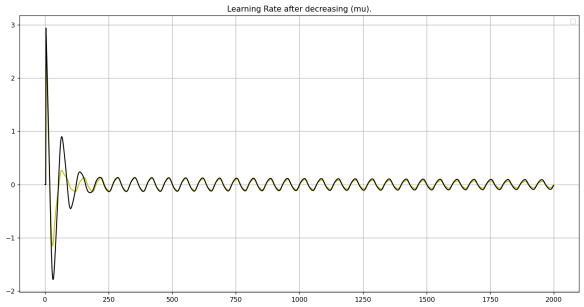

Fig. 11. Learning Rate with decreased mu

It can be seen how significantly can the noise affect the behaviour of the adaptive filter. Figure 12 shows the error resulting from using a noisy input and how it causes unpredictable behaviour for the filter.

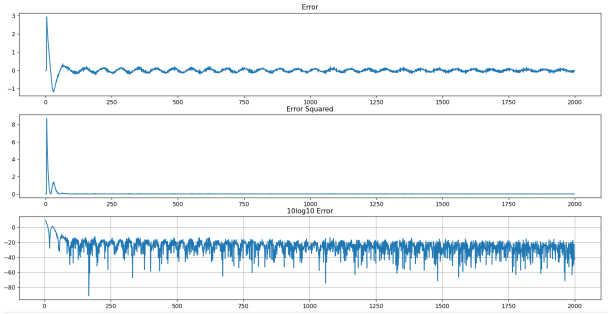

Fig. 12. Error with noisy input X[n]

Because 40dB is relatively high level of noise, it can cause the filter coefficients to converge to an incorrect solution, leading to residual noise or distortion in the filtered output. It can be seen in figure 13 how the red plot which represents the output of the filter does not exactly match the output of the noisy system. Even though the noise value is pretty high phase response was fairly similar and was not impacted a lot.

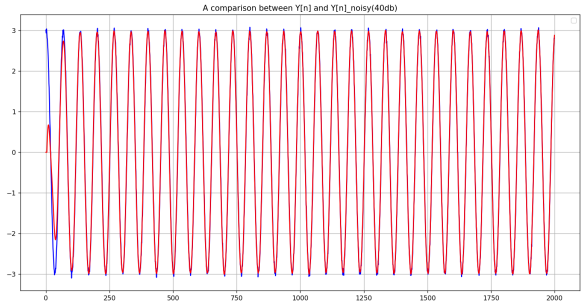

Fig. 13. noisy D[n] & Y[n]

If the amount of noise on the input data decreases to 30 dB, it will have a less significant impact on the behavior of

the adaptive filter compared to 40 dB of noise. A noise level of 30 dB is still relatively high, but the reduced noise level will allow the adaptive filter to converge to a more accurate solution, resulting in a filtered output with less residual noise or distortion.

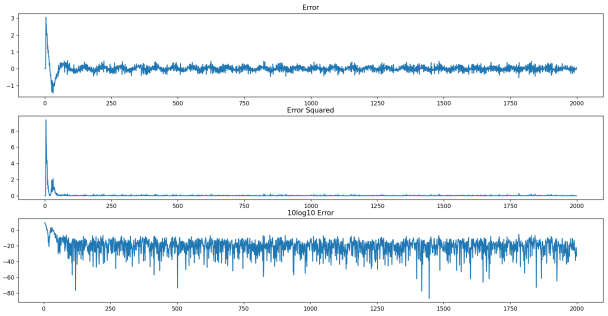

Fig. 14. Error with noisy X[n] (30dB)

It can be seen in figure 15 how the noise impacted the output data from the filter Y[n] which is unable to match D[n] within the given sample window.

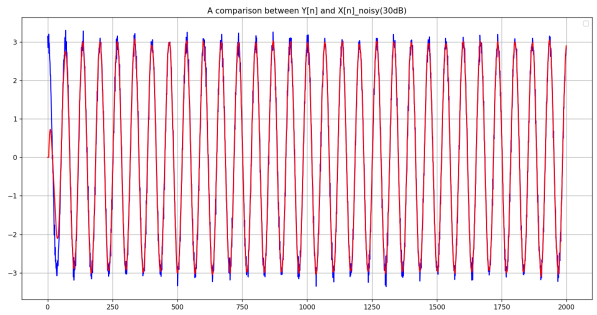

Fig. 15. noisy D[n] (30dB) & Y[n]

Figure 16 shows the different learning rates with the given noise values. The learning rate for the data with 30dB noise is the black plot and converges faster.

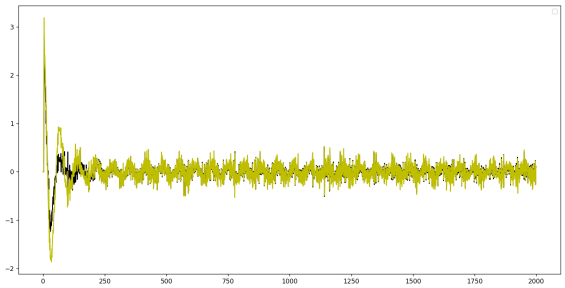

Fig. 16. Learning Rate after decreasing the noise value from 40dB to 30dB

Systems in nature are far from perfect. Every system naturally has noise in it. In order to decrease the effect it has on the adaptive filter s, the ensemble method takes the input data a given number of times and use the average of these iteration to produce a more acceptable output.

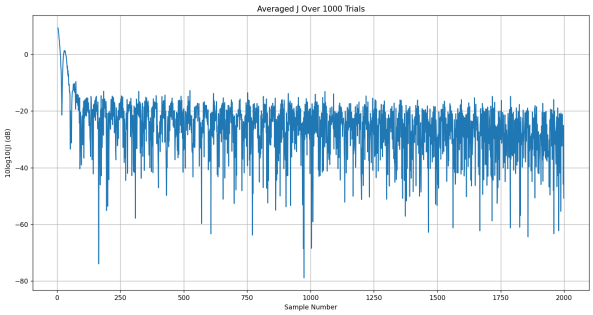

Fig. 17. averaged J

The second adaptive filter algorithm we chose is the RLS, it was implemented to estimate the filter coefficients, the RLS has a forgetting factor (Initially set to 0.9), the following error learning curves were obtained:

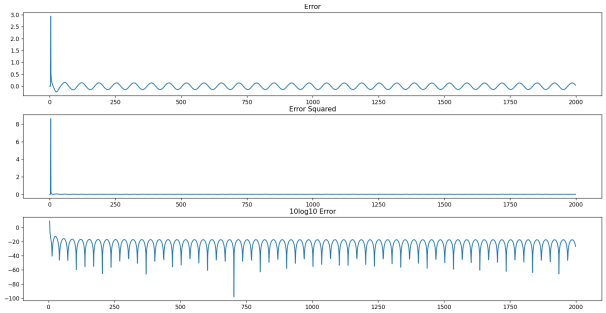

Fig. 18. RLS Error rates

Then the amplitude and phase response of the RLS estimated FIR system were plotted after the end of iterations.

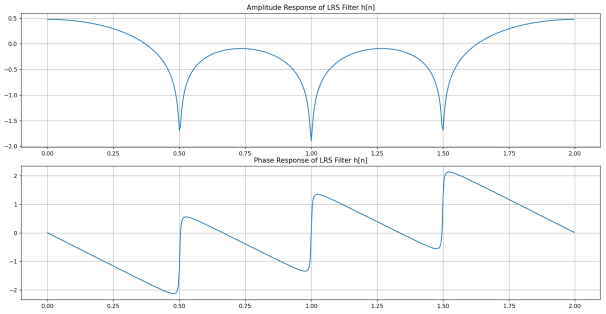

Fig. 19. RLS Frequency Responcse fr = 0.9

The phase response of the RLS filter should be as close as possible to the unknown system phase response, in the figure 21 below that compares the two phase response, The similarity between the two is there, but not that relatively high similarity, a change in the forgetting factor could help.

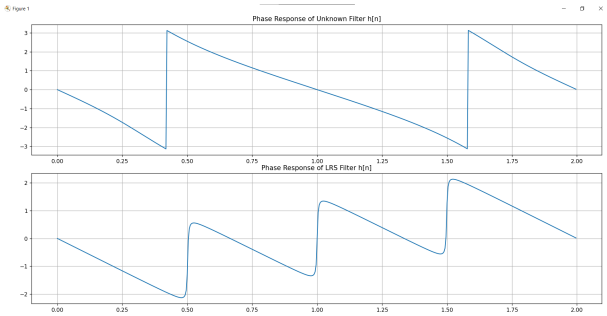

Fig. 20. RLS Phase fr = 0.9 & Unknown Filter Phase

This plot in figure 21 shows the unknown filter plot and the adaptive filter plot using RLS, the adaptation process from the adaptive system to the unknown system can be noticed in the plot, as the two systems come together.

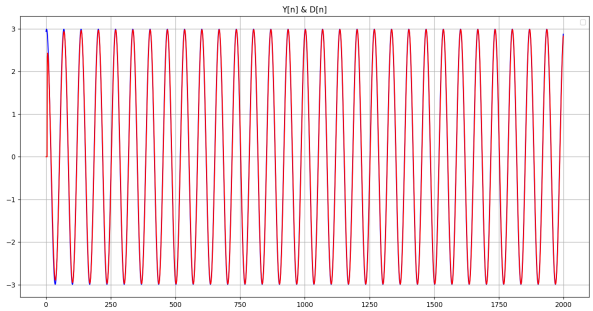

Fig. 21. Y[n] & D[n]

Upon decreasing the forgetting factor, more weight is given to the older data meaning the filter is less responsive to changes in the input, thus reducing the impact of noise, same as the LMS it means slower convergence rate with more stability, so the phase response may become closer to that of the unknown system, the error rate did not differ much from when the forgetting factor was 0.9, you can notice these changes in the figures below:

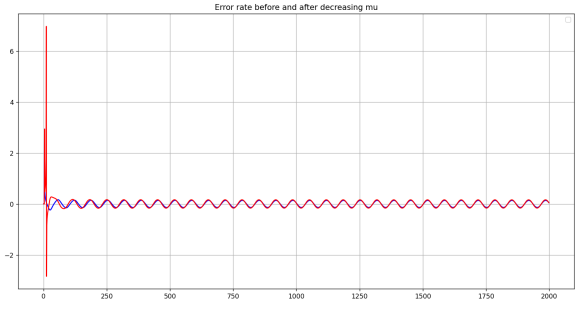

Fig. 22. Error rate before and after decreasing fr

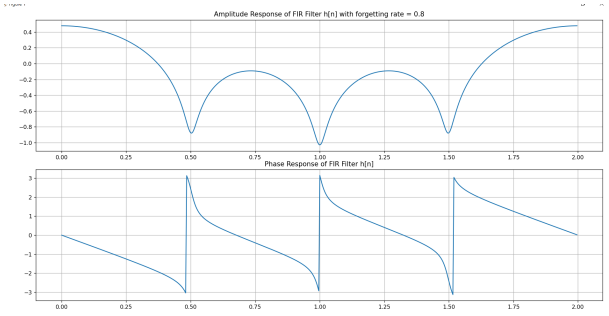

Fig. 23. RLS frequency response fr = 0.8

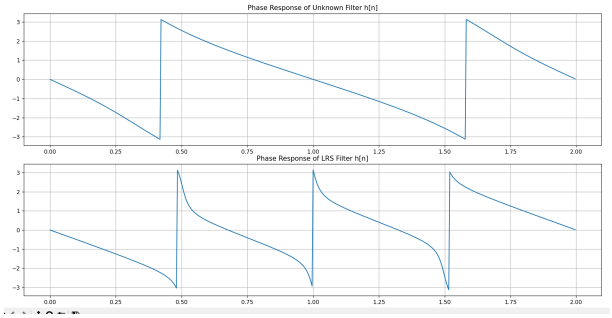

Fig. 24. RLS Phase fr = 0.8 & Unknown Filter Phase

Due to the different approaches taken in each algorithms, RLS and LMS respond to noise in data differently. RLS algorithms use a recursive least squares optimization technique, which can handle Gaussian noise more effectively than LMS algorithms. However, RLS algorithms are more sensitive to impulsive noise, and can become unstable if the noise spikes are too large. 40dB is large value for noise and it can be seen how it affects Y[n] in figure 26 where the spikes are too high them regulated later.

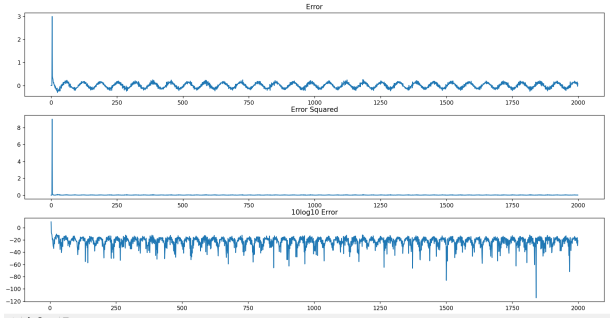

Fig. 25. RLS Error rates with noisy data (40dB)

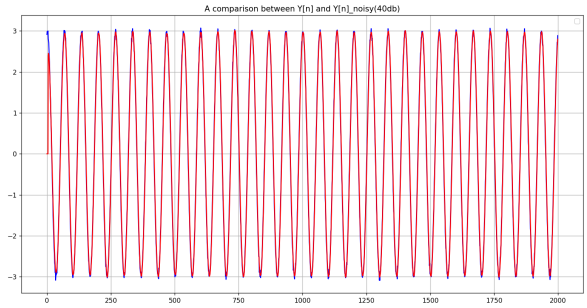

Fig. 26. Y[n] & noisy D[n] (40dB)

Decreasing the noise value to 30dB will give the RLS algorithm a better chance to filter it out since it can handle high spikes of noise very well.

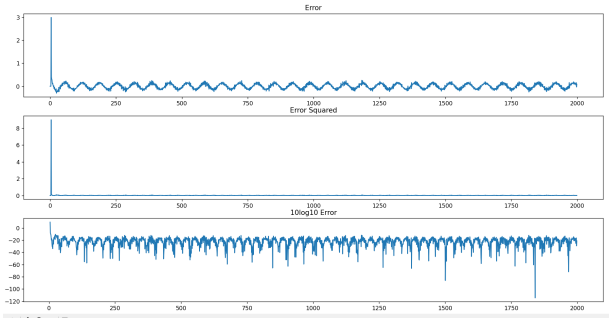

Fig. 27. RLS error rates (30dB)

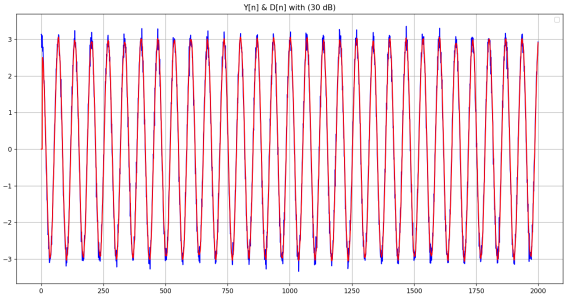

Fig. 28. Y[n] vs noisy D[n] (30dB)

As mentioned previously, ensemble methods will be use to average out the noisy values to smoothen the curves.

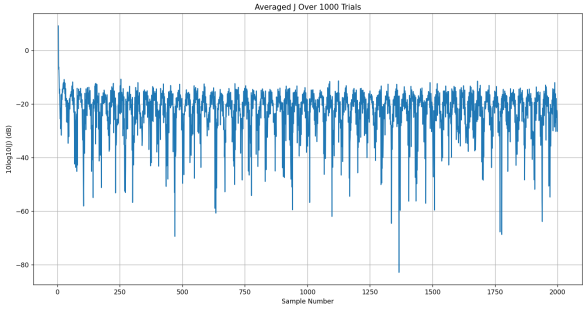

Fig. 29. averaged J

There are multiple differences noticed between LMS and RLS from the results before. RLS algorithms converge faster than LMS algorithms, and their convergence rate is independent of the learning rate. LMS algorithm is less sensitive to instability than RLS algorithm, as the step-size parameter in LMS algorithm can be adjusted to control the convergence rate and stability. LMS algorithm is relatively slow compared to RLS algorithm, as LMS algorithm requires multiple operations for each iteration. Both LMS and RLS algorithms produce an estimate of the unknown system parameters based on the input-output data. The output from an LMS algorithm is the filtered signal, while the output from an RLS algorithm is the estimated system parameters. Finally, changing the learning rate in LMS algorithm may not necessarily change the phase response. However changing the forgetting factor in RLS algorithm can affect the phase response, because the forgetting factor controls the weight given to past observations, and determines how quickly the algorithm forgets the past observations.

VI. Improvements

A naive development approach I tried while implementing the RLS, as the error was sometimes low (close to zero), while the adaptive phase response was close to the unknown phase response (Also the opposite of this happened), so I tried to bring them closer making the error close to zero and make both the phase responses more similar, the approach was to add another parameter to the RLS other than the forgetting factor, which is μ (gain or step size), this was added in order to balance the convergence speed and stability, with smaller values giving slower convergence but greater stability.

In the code below you can see this slight difference, an identity matrix is created from the order of the system 'M' then divided by the step size 'mu' to scale it by this factor, this controlling the rate of convergence of the algorithm.

```python
def rls2(x, d, M, mu=0.1, lam=0.98):
    N = len(x)
    w = np.zeros(M)
    P = np.eye(M) / mu
    y = np.zeros(N)
    e = np.zeros(N)
    for n in range(M, N):
        x1 = x[n - M + 1:n + 1][::-1]
        y[n] = np.dot(w, x1)
        e[n] = d[n] - y[n]
        k = np.dot(P, x1) / (lam + np.dot(np.dot(x1, P), x1))
        w = w + k * e[n]
        P = (P - np.dot(k[:, np.newaxis], x1[np.newaxis, :]) * lam) / lam
    J = e ** 2

    Jnew = 10 * np.log10(J)
    return w, y, e, J, Jnew
```

Fig. 30. RLS code with a step size

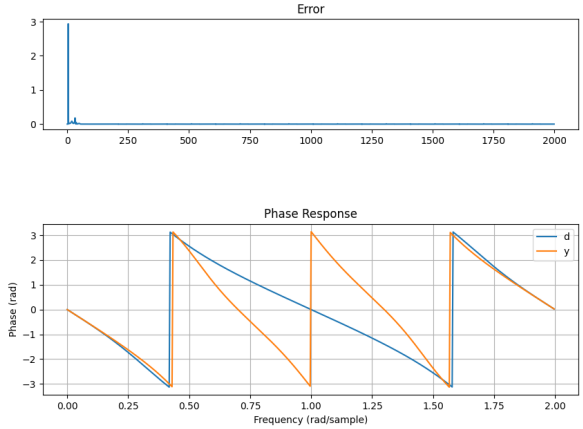

Fig. 31. Error and Adaptive filter phase response & Unknown filter phase response

VII. Conclusion

In this project the system identification was tackled using adaptive filters, using two adaptive filter algorithms, LMS and RLS, first we generated the input X[n] and X[f], then the amplitude and phase response for the given FIR system to compare with the one generated from the algorithms, For LMS both phase responses were relatively close even after decreasing the learning rate from 0.01 to 0.005, while the RLS the phase response got more precise when the forgetting factor was decreased from 0.9 to 0.8, as for the error, it got lower after decreasing the LMS learning rate, on the other hand the RLS didn't change much because the forgetting rate was not decreased significantly.

Also in this project it was observed how the noise affects the signal identifying process and how measures were taken to decrease it's effect on the desired outcome since it can't be completely eliminated. The ensemble method was displayed and how it works in improving the quality of the output when there is noise.

Note: The Codes are in a separate file.

References

[1] A Compare RLS and LMS Adaptive Filter Algorithms - MATLAB Simulink. (n.d.). https://www.mathworks.com/help/dsp/ug/compare-rls-and-lms-adaptive-filter-algorithms.html

[2] Digital Signal Processing: Principles and Applications (Illustrated). Cambridge University Press. Meyer, R., Gerstacker, W. H., Schober, R., Huber, J. B. (2006). A single antenna interference cancellation algorithm for increased gsm capacity. IEEE Transactions on Wireless Communications, 5(7), 1616–1621. https://doi.org/10.1109/twc.2006.1673070

[3] Ramirez, P.S. (2002). The Least-Mean-Square (LMS) Algorithm. In: Adaptive Filtering. The Kluwer International Series in Engineering and Computer Science, vol 694. Springer, Boston, MA. https://doi.org/10.1007/978-1-4757-3637-3$_3 Finite - window RLS algorithms. (2022, September). Scince Direct. Retrieved February 12, 2