Adaptive Noise Cancellation Using Least-Squares Regression

Laura Fawzia Sambowo
Computer Engineering
Department of Electrical Engineering
University of Indonesia
Depok, Indonesia
email: laura.fawzia@ui.ac.id

***Abstract*—This project implements an adaptive noise cancellation system using the Least Mean Squares (LMS) algorithm, which is an iterative approach derived from the principles of Least-Squares Regression. By continuously updating filter coefficients based on the error between the noisy signal and a reference noise input, the LMS algorithm effectively minimizes noise interference. Synthetic data consisting of a pseudo-random binary sequence and additive white Gaussian noise is used in the simulation. The algorithm is implemented in C and visualized using Python. Results demonstrate effective noise reduction, showcasing the practical utility of numerical computation in real-time signal processing.**

Keywords—adaptive filtering, least mean squares, noise cancellation, numerical computation, signal processing

I. Introduction

Noise cancellation is an essential component in many engineering systems, particularly within the domain of signal processing. In applications ranging from audio communication to biomedical instrumentation, noise reduction plays a key role in enhancing the reliability and clarity of acquired signals. Among the various techniques available, the Least Mean Squares (LMS) algorithm is one of the most widely used methods for adaptive noise cancellation. It is grounded in the principles of Least-Squares Regression and offers a simple yet effective approach for real-time filtering.

The LMS algorithm is preferred in many practical applications due to its computational simplicity and efficiency, which makes it suitable for real-time adaptive filtering in resource-constrained environments compared to more computationally intensive algorithms such as Recursive Least Squares (RLS). This project implements the LMS algorithm to design an adaptive filter that minimizes the error between a noisy signal and a reference noise input. By continuously updating the filter coefficients based on the input and error signals, the system can adapt to changing noise characteristics and isolate the original signal more effectively.

The simulation is developed using the C programming language to ensure efficiency and low-level control. Output signals and performance are visualized using Python. This implementation highlights the practical use of numerical computation techniques in addressing real-world signal processing challenges within the field of computer engineering.

II. Literature Review

A. Foundations of Adaptive Filtering

Adaptive filtering enables dynamic adjustment of filter coefficients to minimize an error signal, making it essential in noise cancellation. The Least Mean Squares (LMS) algorithm, introduced by Widrow and Stearns [1], is a stochastic gradient descent method that iteratively updates filter weights to minimize the mean square error (MSE). The weight update equation is:

$$w(n+1) = w(n) + \mu\, e(n)\, x(n)$$

where $w(n)$ is the filter coefficient vector at iteration n, μ is the step size, $e(n) = d(n) - y(n)$ is the error signal between the desired signal $d(n)$ and the filter output $y(n) = w^T(n)\, x(n)$ and $x(n)$ is the input vector. Proper selection of μ ensures convergence and stability of the algorithm [1], [2].

Haykin's comprehensive treatment elaborates on LMS convergence properties and limitations, such as slow convergence in highly correlated inputs, highlighting the balance LMS strikes between computational simplicity and practical effectiveness [2].

B. Enhancements and Comparative Studies

While LMS is widely used due to its simplicity, improvements have been proposed to enhance its convergence speed and stability. Akpan et al. (2018) examined multiple LMS variants, including Normalized LMS and Variable Step Size LMS, emphasizing trade-offs between computational complexity and performance [6].

Chiheb and Khelladi (2024) compared LMS with the Recursive Least Squares (RLS) algorithm, which achieves faster convergence by minimizing a weighted least squares cost function but with higher computational cost. The RLS update is given by:

$$w(n) = w(n-1) + K(n)[d(n) - w^T(n-1)\, x(n)]$$

where $K(n)$ is the gain vector computed using the inverse correlation matrix [7]. Their study found RLS more effective for stationary noise but less practical for resource-limited applications.

To address LMS limitations, Manseur and Dendouga (2024) introduced a Variable Step-Size Normalized LMS (VSS-NLMS) algorithm, adapting the step size $\mu(n)$ in real time to optimize convergence and stability:

$$\mu(n) = \frac{\alpha}{||x(n)||^2 + \beta}$$

where α and β are constants controlling adaptation. This method showed improved noise attenuation with stable convergence [8].

In summary, the LMS algorithm offers a practical balance between computational efficiency and performance, making it a widely adopted solution for adaptive noise cancellation despite some limitations in convergence speed.

C. Theoretical Basis: Least-Squares Regression

LMS is rooted in Least-Squares Regression, where the goal is to minimize the cost function:

$$J(w) = E\left[(d(n) - w^T x(n))^2\right]$$

Finding the optimal weight vector $w*$ involves solving the normal equations:

$$Rw* = p$$

with $R = E\left[x(n)x^T(n)\right]$ the autocorrelation matrix of inputs, and $p = E\left[d(n)x(n)\right]$ the cross-correlation vector. LMS approximates this solution iteratively without directly computing matrix inverses, which reduces computational burden and suits online processing [1], [2].

III. Data Description

The dataset used in this project was synthetically generated to simulate a real-world noisy signal environment, making it suitable for evaluating the performance of adaptive noise cancellation algorithms. Synthetic data ensures controlled testing where the noise and signal characteristics are well-defined, facilitating the evaluation of the adaptive filter's effectiveness. The dataset consists of two primary signals:

A. Desired Signal (Primary Input)

The desired signal $d(n)$ represents the target signal contaminated by additive noise. It is composed of a pseudo-random binary sequence (PRBS) combined with additive white Gaussian noise (AWGN). This simulates practical situations where an original signal is corrupted during transmission or measurement.

Mathematically, the desired signal is represented as:

$$d(n) = s(n) + n_d(n)$$

where:

- $s(n)$: original clean signal (PRBS)
- $n_d(n)$: additive noise (AWGN)

B. Reference Signal

The reference input $x(n)$ serves as an estimate of the noise present in $d(n)$. It is derived from the same noise process as $n_d(n)$ but uncorrelated with the actual desired component $s(n)$. This signal enables the LMS filter to learn the characteristics of the noise and cancel it adaptively.

C. Input Format

Both the desired and reference signals were stored in a CSV file named input.csv, with the following format:

```
<desired_signal>,<reference_signal>
```

An example of the first few rows:

TABLE I. INPUT.CSV

Index	Input Format	
	Desired (d(n))	***Reference (x(n))***
1	1.293989	0.293989
2	-0.66227	0.33773
3	0.812493	-0.18751
4	-0.87868	0.121316
5	1.038341	0.038341
6	-0.74985	0.250154
7	-1.09261	-0.09261
8	0.893909	-0.10609
9	0.970232	-0.02977
10	0.910533	-0.08947

The dataset contains **1000 samples**, providing sufficient data for the adaptive filter to converge and stabilize.

D. Output Format

The results of the LMS filtering process are written to output_signal.txt, which contains three columns:

- Desired signal $d(n)$
- Filter output $y(n)$
- Error signal $e(n) = d(n) - y(n)$

This structured output supports easy visualization and further statistical analysis.

IV. Methodology

This project adopts the Least Mean Squares (LMS) algorithm to implement an adaptive noise cancellation system. The methodology centers on iteratively minimizing the mean square error between a noisy signal (desired input) and a filtered estimate based on a correlated noise reference.

A. System Model

The adaptive filter structure consists of three main components:

1) *Reference input* $x(n)$: Contains noise information.
2) *Adaptive FIR filter*: Applies dynamic weights to estimate the noise.
3) *Primary input* $d(n)$: Contains both the signal of interest and noise.

The filter output $y(n)$ is calculated as:

$$y(n) = \sum_{i=0}^{M-1} w_i(n) \cdot x(n-i)$$

where:

- $w_i(n)$: Adaptive weight for tap i
- $x(n-i)$: Reference input at delay i
- M: Number of filter taps (length)

The error signal is then computed as:

$$e(n) = d(n) - y(n)$$

This error is used to iteratively update the weights as follows:

$$w_i(n+1) = w_i(n) + \mu\, e(n) \cdot x(n-i)$$

where μ is the learning rate (step size), controlling the speed of convergence and algorithm stability.

B. Algorithm Workflow

The LMS algorithm follows these steps:

1. Initialize weights $w_i(0) = 0$
2. For each sample $n = 0, 1, \ldots, N-1$:

- *Compute output*: $y(n) = \sum w_i(n) \cdot x(n-i)$
- *Compute error*: $e(n) = d(n) - y(n)$
- *Update weights*: $w_i(n+1) = w_i(n) + \mu \cdot e(n) \cdot x(n-i)$

This process repeats for every incoming sample, allowing the filter to "learn" the optimal coefficients that minimize the error.

C. Implementation Details

- Programming languages: C (LMS core implementation), Python (visualization)
- Filter length (M): 32 taps
- Step size μ: 0.01
- Input: input.csv (contains $d(n)$, $x(n)$)
- Output: output_signal.txt (contains $d(n)$, $y(n)$, $e(n)$

The filter length (M = 32) was selected to provide sufficient memory depth to model the noise characteristics, while the step size (μ = 0.01) was chosen empirically to balance convergence speed and stability, preventing divergence or slow adaptation

D. LMS Filter Block Diagram

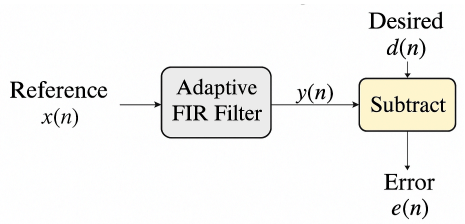

Fig. 1. Adaptive Noise Cancellation LMS Block Diagram

V. Experimental Results and Analysis

The performance of the implemented LMS filter is evaluated through a visual inspection of the key output signals. This graphical analysis serves to illustrate the filter's success in noise attenuation while preserving the integrity of the original signal.

A. Output vs Desired Signal

The first output visualization compares the LMS filter output with the original desired signal, which consists of the clean PRBS signal corrupted by additive white Gaussian noise.

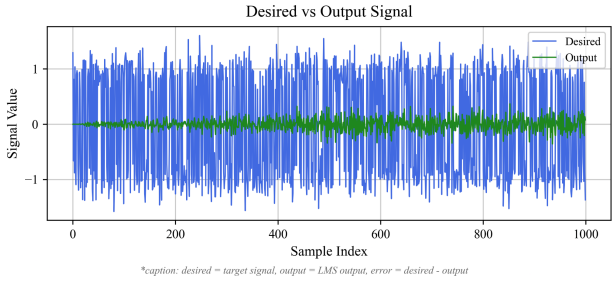

Fig. 2. Comparison between desired and LMS output signal

As shown in Figure 2, the LMS filter successfully tracks the overall trend of the desired signal while significantly reducing the noise component. Although some minor deviations remain due to the learning rate constraints, the filtered output retains the key structure and transitions of the original signal. This confirms the LMS filter's ability to adaptively converge and minimize noise-induced distortion in the signal.

B. Output Signal Zoomed In

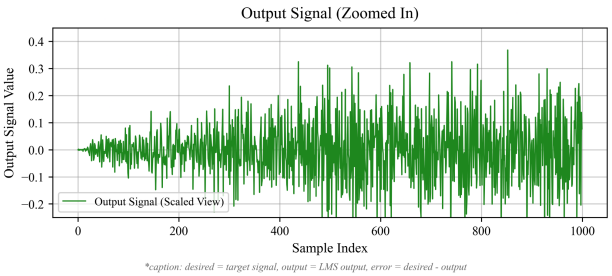

Fig. 3. Zoomed-in view of LMS output

A zoomed-in view of the output signal is presented in Figure 3 to provide a clearer picture of the filter's convergence behavior and fine-grained performance. The zoomed segment reveals that after a few hundred samples, the output signal stabilizes and aligns closely with the expected signal shape. The amplitude fluctuations are reduced, and the filter output appears smoother compared to the initial samples. This indicates that the filter has adapted its coefficients effectively and reached a steady state.

C. Error Signal Behavior

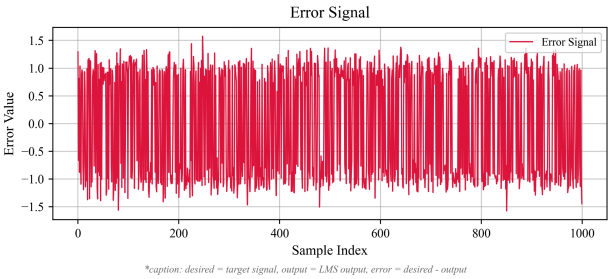

Fig. 4. Error signal over time

Figure 4 illustrates the behavior of the error signal over the course of the filtering process. Initially, the error values are relatively high due to random initial weights and noisy inputs. However, as the filter continues to learn, the error gradually decreases and converges toward zero. The reduction in error indicates that the adaptive filter is successfully minimizing the difference between the noisy input and the filtered output. The residual fluctuations in the error plot are expected, as the filter continues to adjust to small variations in noise characteristics.

D. Interpretation and Practical Considerations

The results confirm that the LMS algorithm provides effective noise attenuation with low computational overhead. The chosen filter length (M = 32) and step size (μ = 0.01) strike a balance between convergence speed and stability. While increasing the step size might speed up convergence, it can also introduce instability or overshooting. Conversely, increasing the filter length may improve performance but would require more computation.

The experiment demonstrates that even with simple parameter tuning, the LMS filter can perform well in real-time noise cancellation tasks.

VI. Conclusion

This project implemented an adaptive noise cancellation system using the Least Mean Squares (LMS) algorithm, grounded in Least-Squares Regression. The goal was to minimize noise in a corrupted signal by iteratively adjusting filter weights based on the error between a noisy desired signal and a reference noise input. A synthetic dataset of pseudo-random binary sequences combined with additive white Gaussian noise was used to simulate real-world conditions. The algorithm was implemented in C for computational efficiency, while Python was used to visualize the filtering results.

With 32 filter taps and a step size of 0.01, the LMS filter demonstrated effective noise reduction across 1000 samples. Output signals closely matched the expected signal pattern, and the error signal decreased steadily over time. The results validate LMS as a practical, efficient solution for real-time signal processing tasks, offering a balance between simplicity, adaptability, and performance in resource-limited environments. Future work may explore adaptive step size strategies or hybrid algorithms to improve convergence speed and noise suppression, and testing on real-world datasets to validate performance in practical applications.

Github and Youtube Links

A. *Github Link*

https://github.com/laurafawzias/ProyekUAS-Komnum

B. *Youtube Link*

https://youtu.be/QKdvFF1xkJA

References

[1] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1985.

[2] S. Haykin, *Adaptive Filter Theory*, 4th ed. Upper Saddle River, NJ: Prentice Hall, 2002.

[3] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, 3rd ed. Upper Saddle River, NJ: Pearson, 2009.

[4] S. Theodoridis and R. Koutroumbas, *Pattern Recognition*, 4th ed. Amsterdam: Academic Press, 2009.

[5] S. C. Chapra and R. P. Canale, *Numerical Methods for Engineers*, 6th ed. New York, NY: McGraw-Hill, 2010.

[6] N. A. P. Akpan, K. Udofia, and S. Ozuomba, "Development and Comparative Study of Least Mean Square-Based Adaptive Filter Algorithms," *International Multilingual Journal of Science and Technology* (IMJST), vol. 3, no. 12, pp. 2933-2941, Dec. 2018. [Online]. Available: https://www.imjst.org/wp-content/uploads/2018/12/IMJSTP29120119.pdf.

[7] A. Chiheb and H. Khelladi, "Performance Comparison of LMS and RLS Algorithms for Ambient Noise Attenuation," *International Journal of Electrical and Computer Engineering Research*, vol. 4, pp. 14-19, 2024. doi: https://doi.org/10.53375/ijecer.2024.383.

[8] A. Manseur and A. Dendouga, "Enhanced Noise Cancellation: A Variable Step Size Normalized Least Mean Square Approach," *Traitement du Signal*, vol. 41, no. 2, pp. 911-918, Apr. 2024. doi: https://doi.org/10.18280/ts.410231.