External Project Report on Signals and Systems (EET2051)

Submitted by

P.GOPINATH REDDY 2341018134

INSTITUTE OF TECHNICAL EDUCATION AND RESEARCH
(FACULTY OF ENGINEERING)
SIKSHA 'O' ANUSANDHAN (DEEMED TO BE UNIVERSITY), BHUBANESWAR, ODISHA

Declaration

We,the undersigned students, hereby declare that the projecttitled " Simulation and Analysis of Adaptive Noise Cancellation using MATLAB" has been undertaken and completed by us as a part of our academic

This project represents our sincere and collaborative effort, and we affirm that it is the result of our own work, carried out with dedication and responsibility. We also confirm that this work has not been submitted, either in part or in full, to any other university, college, or institution for the award of any degree, diploma, certificate, or other academic recognition. Throughout the course of this project, we have ensured that all sources of information, data, or references taken from existing literature, online platforms, or other external sources have been appropriately acknowledged and cited. Every effort has been made to maintain the authenticity and originality of our work, and we have not knowingly included any content that is plagiarized or misrepresented.

This project was completed under the valuable guidance and supervision of our project supervisor, whose support and feedback were instrumental in shaping the outcome of this work. We are proud to present this report as a reflection of our learning, research, and teamwork.

Contents

Abstract

This project implements an Adaptive Noise Cancellation (ANC) system in MATLAB to address noise interference in signals. Traditional filters struggle with time-varying noise; therefore, an adaptive algorithm is used. The system dynamically adjusts parameters to track noise characteristics. Performance is evaluated via SNR improvement and MSE reduction. Results demonstrate the ANC system's effectiveness in enhancing signal quality.

. This work highlights the practical application of adaptive filtering for noise cancellation.The LMS algorithm enables the filter to learn the noise behaviour and gradually minimise it from the corrupted signal. The simulation results clearly show that the adaptive filter significantly improves the signal quality by reducing the noise effectively. This study demonstrates the importance of adaptive filtering in signal processing and highlights MATLAB as a powerful tool for analysing and implementing such systems.

1. Introduction

In our daily surroundings, most useful signals get mixed with unwanted noise. This problem is common in audio recordings, communication systems, sensors, and many electronic devices. When the noise level is high, it becomes difficult to clearly hear or analyse the actual signal. Normal fixed filters do not always work well because noise keeps changing with time. To handle such situations, we use Adaptive Noise Cancellation (ANC), where the filter adjusts itself automatically according to the incoming noise.

ANC uses an adaptive filter that continuously updates its weights to reduce the noise and recover the clean signal. In this project, we use the Least Mean Squares (LMS) algorithm, which is one of the simplest and widely used adaptive algorithms because of its easy implementation and good performance

In our work, we generate a synthetic temple bell sound as the clean signal and then add noise to it through an unknown system to make it more realistic. Using MATLAB, we apply the LMS adaptive filter to remove the noise and try to restore the original bell sound. By comparing the clean, noisy, and filtered signals, we can clearly observe how effectively the ANC system works.

2. Problem Statemen

1. Real-world signals often get mixed with unwanted noise, which reduces their clarity and makes them difficult to analyse or process.
2. Traditional fixed filters cannot handle changing noise conditions, especially when the noise characteristics vary over time.
3. There is a need for a system that can automatically adjust itself to remove noise without affecting the useful part of the signal.
4. Recovering the original signal from a noisy version is challenging, especially when the noise passes through an unknown environment or channel.
5. An efficient adaptive filtering technique must be implemented and tested to observe how well it can reduce noise in practical situations.
6. MATLAB-based simulation is required to demonstrate the working of an Adaptive Noise Cancellation system using the LMS algorithm.

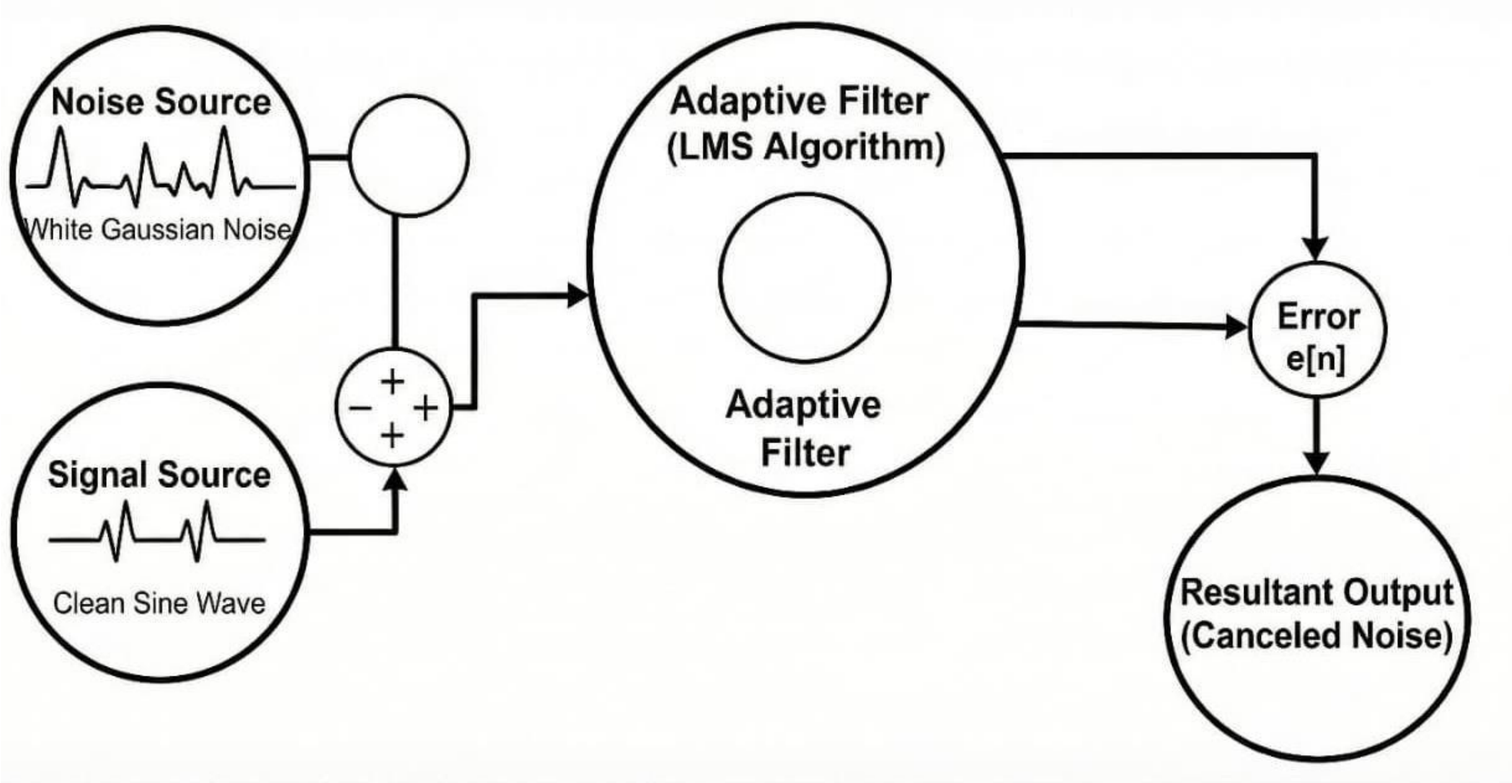
3. Methodology

The following steps were followed to carry out the Adaptive Noise Cancellation (ANC) project:

1. Generate Clean Signal:
A synthetic temple bell sound was created in MATLAB to serve as the original clean signal.
2. Add Noise to the Signal:
White Gaussian noise was passed through an unknown system (filter) to make the noise realistic, and this noise was added to the clean signal
3. Prepare Reference Noise Input:
The unfiltered noise signal was used as the reference input for the adaptive LMS filter.
4. Implement LMS Adaptive Filter:
An LMS filter of fixed length and step-size was used to adaptively estimate the noise present in the corrupted signal
5. Perform Noise Cancellation:
The adaptive filter continuously adjusted its weights to predict the noise, which was then subtracted from the noisy input to produce the restored signal.
6. Visual Analysis:
Time-domain plots of the clean signal, noisy signal, and filtered output were generated to compare the performance

4. Implementation

Blockdaigram:



MATLABCODE:

```
clear; clc; close all;
Fs_sig = 44100;
t = 0:1/Fs_sig:4;
% -------- Temple Bell Signal --------- %
s = (exp(-1.8*t).*sin(2*pi*440*t) + 0.6*exp(-1.6*t).*sin(2*pi*880*t))';
s = s / max(abs(s));
N = length(s);
t = (0:N-1)/Fs_sig;
% -------- Noise & Corrupted Signal--------- %
noise_power = 0.05;
n1 = sqrt(noise_power) * randn(N, 1);
Hz_coeffs = [0.05 0.1 0.2 0.4 0.2 0.1 0.05];
no = filter(Hz_coeffs, 1, n1);
d = s + no;
% -------- LMS Adaptive Filter ---------%
L = 32;
mu = 0.008;
lms_filter = dsp.LMSFilter('Length', L, 'StepSize', mu, 'Method', 'LMS');
fprintf('Processing audio stream ..\n');
[y_est, e] = step(lms_filter, n1, d);
segment_idx = 1:min(N, 5000);

% -------- Figure 1: Original--------- %
figure('Name', 'Original Temple Bell', 'Color', 'w');
plot(t(segment_idx), s(segment_idx), 'b');
```

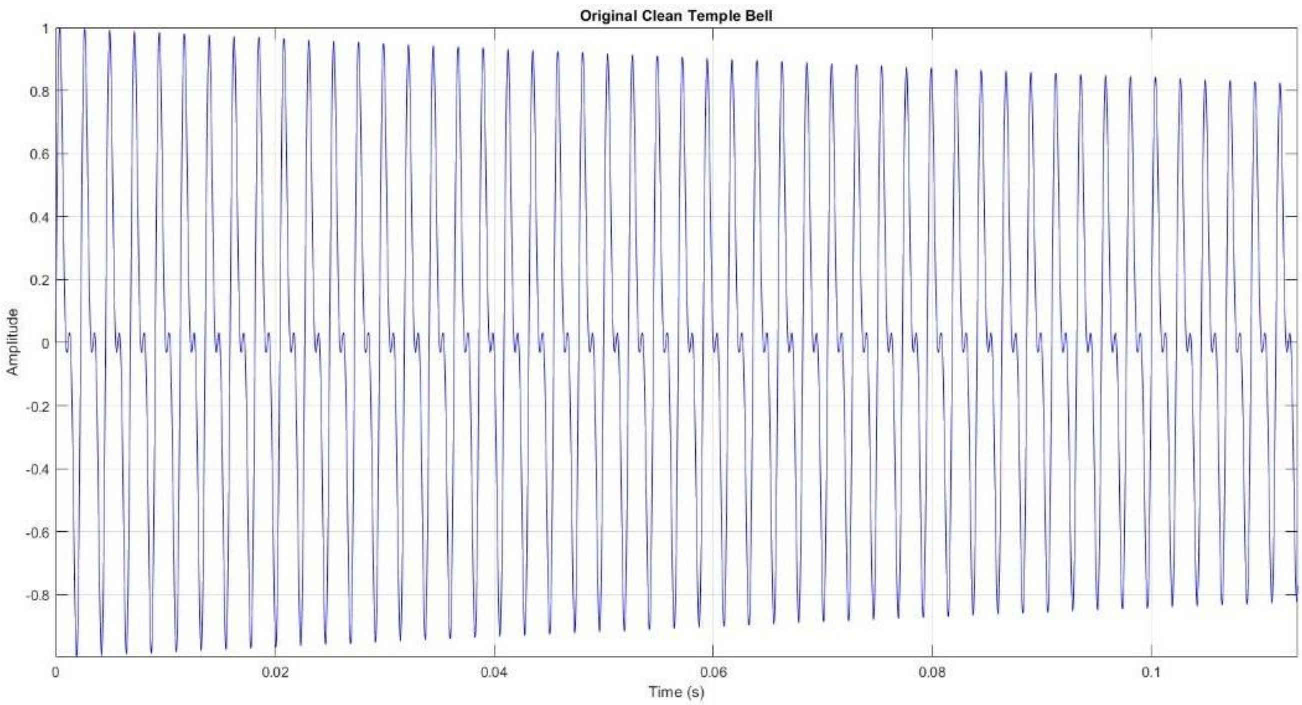
```
title('Original Clean Temple Bell');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;
axis tight;
fprintf('Playing Original Bell...\n');
soundsc(s, Fs_sig);
pause(length(s)/Fs_sig + 1);

% -------- Figure 2: Corrupted --------- %
figure('Name', 'Corrupted Temple Bell', 'Color', 'w');
plot(t(segment_idx), d(segment_idx), 'b');
title('Corrupted Input (Bell + Noise)');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;
axis tight;
fprintf('Playing Corrupted Bell .. \n');
soundsc(d, Fs_sig);
pause(length(d)/Fs_sig + 1);

% -------- Figure 3: Filtered--------- %
figure('Name', 'Filtered Temple Bell', 'Color', 'w');
plot(t(segment_idx), e(segment_idx), 'b');
title('Filtered Output (LMS Restored)');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;
axis tight;
fprintf('Playing Restored Bell .. \n');
 soundsc(e, Fs_sig);
```

5. Results & Interpretation

1. **Clean Signal:**
   i. **The original temple bell signal was smooth and clearly visible in the plot.**
   ii. **This served as the reference to compare how much noise was removed.**
2. **Noisy Signal:**
   i. **After adding filtered noise, the waveform became distorted, and the bell sound**
   ii. **was harder to recognize. This shows that the noise strongly affected the useful signal.**
3. **Filtered Output (LMS Restored Signal):**
   i. **The LMS adaptive filter gradually reduced the noise as it updated its weights.**
   ii. **The final output signal looked much closer to the original clean signal**
4. **Performance Observation:**
   i. **The LMS filter successfully learned the noise pattern.**
   ii. **Most of the random noise components were suppressed.**
   iii. **The restored signal retained the important features of the bell sound**
5. **Audio Quality Check:**
   i. **On listening to the results:**
   ii. **The noisy signal sounded rough and unclear.**
   iii. **The filtered output sounded noticeably cleaner and more similar to the original bell tone**
7. **Overall Interpretation:**
   i. **The results clearly show that the Adaptive Noise Cancellation system using the LMS algorithm is effective in reducing noise from audio signals.**
   ii. **The method works well even when the noise passes through an unknown system**

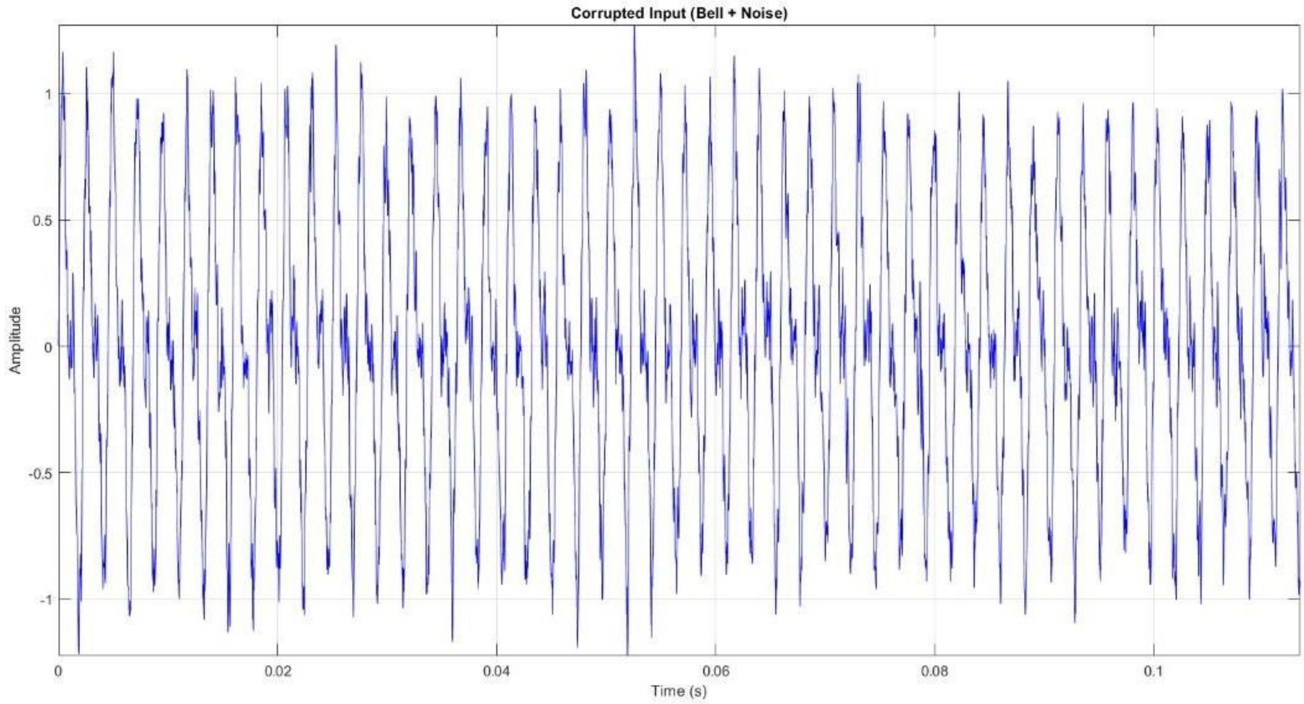
plot

Clean Signal:



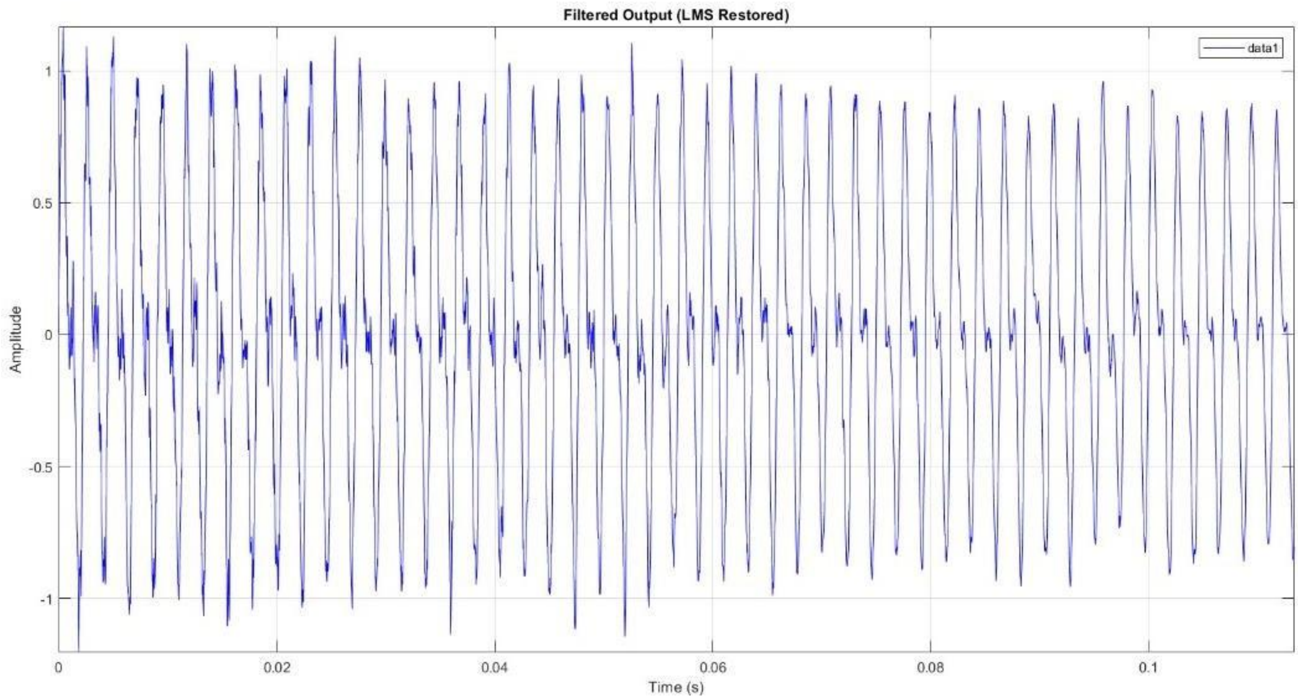
(Figure :1)

Noisey signal:



(figure:2)

Filtered output:



(figure :3)

6. Conclusion

i. In this project, an Adaptive Noise Cancellation system was successfully implemented using the LMS algorithm in MATLAB. The experiment showed that the adaptive filter can efficiently learn the noise pattern and reduce it from the corrupted signal. The restored output closely matched the original temple bell sound, proving that the LMS-based ANC method is effective for practical noise reduction.

ii. The results—both visual and audio—clearly demonstrate that adaptive filtering performs better than traditional fixed filters in environments where noise changes over time. This project helped us understand how ANC works, how the LMS algorithm updates its weights, and how adaptive systems can improve signal clarity in real-world applications

References

(As per the IEEE recommendations)

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

[2] S. Haykin, Adaptive Filter Theory, 5th ed. Upper Saddle River, NJ, USA: Pearson, 2014.

[3] A. V. Oppenheim and A. S. Willsky, Signals and Systems, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 1997.

[4] S. Haykin and B. Widrow, “Adaptive noise cancelling: Principles and applications,” Proc. IEEE, vol. 63, no. 12, pp. 1692–1716, Dec. 1975.

[5] MathWorks, “Adaptive Filters,” MATLAB Documentation, The MathWorks Inc., 2024. [Online]. Available: https://www.mathworks.com

[6] P. S. R. Diniz, Adaptive Filtering: Algorithms and Practical Implementation, 4th ed. New York, NY, USA: Springer, 2013

Appendices

i. Appendix A: MATLAB Code Used in the Project:

This appendix contains the complete MATLAB script used for generating the clean temple bell signal, adding noise, implementing the LMS adaptive filter, and plotting the results. The code can be referred to for understanding the step-by-step implementation and for reproducing the experiment.

ii. Additional Plots and Observations:

Extra waveform plots, if any, are included here to support the analysis. These plots help show how the signal changes during different stages such as noise addition, adaptation, and final noise cancellation.

iii. Audio Output Samples

The project used audio playback to compare the noisy and restored signals. This appendix documents the details of the audio files and the listening observations made during the experiment.