**EE 4745 Project 2**

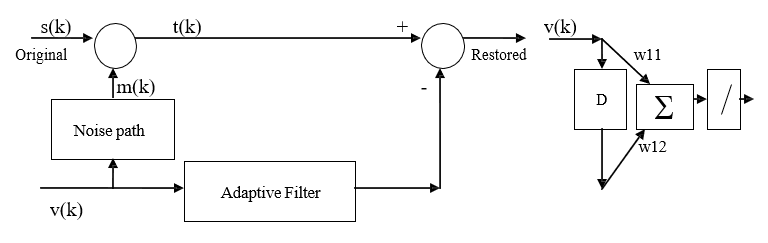
Bret Chalpin, Devon McDonald, Nurudeen Dauda, Bradley Gonzales

I. Abstract

The main objective of this project is to take what we have learned in the lectures and apply it to practical neural network problems using Matlab and Python. We are using equations to obtain minimum points, contour plots, and others signals from a noise cancellation system. Once obtained, implementing the Least Mean Square (LMS) algorithm will show us how the system is working. Our calculations as well as the programs we write demonstrate our understanding of the material and help us prepare for real-life applications.

II. Introduction

In this project, the adaptive filter is used for noise cancellation as shown in Fig. 1 below. It shows an original signal source s(k), uniformly distributed between the values -2 and +2, that is corrupted by the presence of noise m(k) derived from the path of a noise source v(k) of frequency 60Hz. The noise source v(k) is fed as the input to the adaptive filter to produce an output a(k) that is a close estimate of the noise m(k). This noise estimate a(k) is then subtracted from the corrupted signal t(k) to produce a close or restored version of the original uncorrupted signal s(k).



*Fig.1 Noise Cancellation System.*

The input signals used in this project are given as shown below:

1. s(k) = U(2, 2), (ii) v(k) 1.2 sin 2k / 3, and (iii) m(k) 0.12 sin 2k / 3 / 2.

**From the figure, the following can be deduced:**

t(k) = s(k) + m (k) (1)

Error e(k) = t(k) – a(k) (2)

E(k) = s(k) + m(k) – a(k) (3)

**From the general equation:**

F(x) **=** *E[t*2] – 2xTE[tz] + xTE[zz]*T*x; where c ***=*** E[t2], h = E[tz], and R = E[zz]*T* (4)

x\*=R–1 h (5)

R = h =

III. Method

Using MATLAB, the signals were applied as shown in the noise cancellation system and the LMS algorithm was implemented.

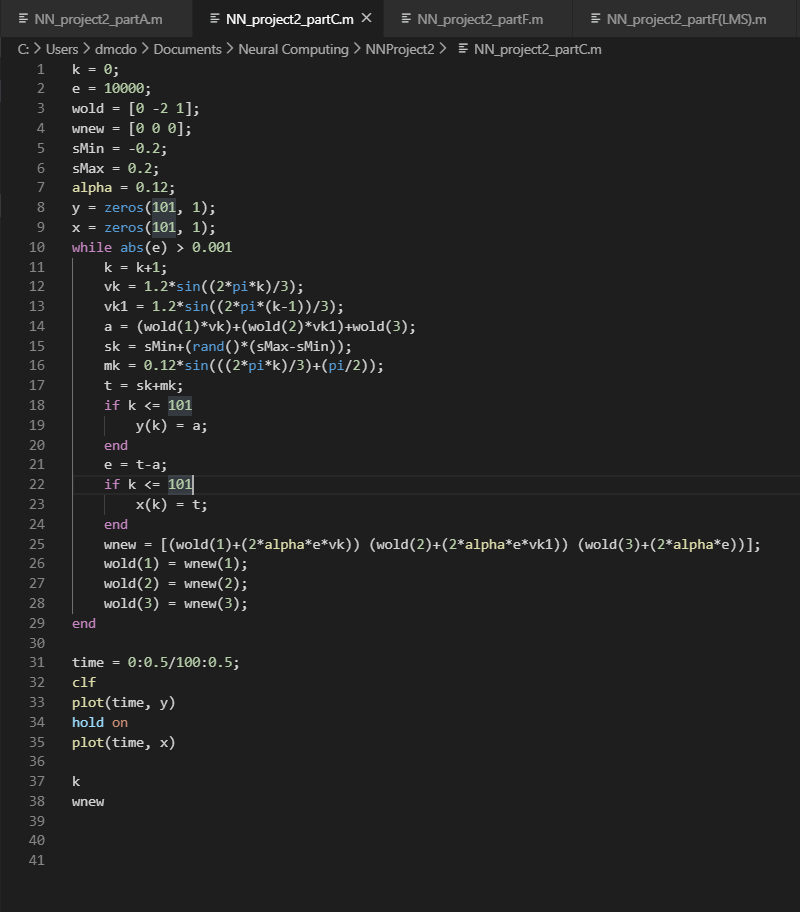
1. For part (a), we used Matlab to compute matrix “R”, which is comprised of and . We also used Matlab to calculate “h”. Since s(k) and v(k) are independent and zero mean, “h” is comprised of just

The eigenvalues and eigenvectors of the Hessian matrix for the mean-square error were calculated and the minimum point was located using x\* = R–1h. The contour plot was also drawn in Matlab using the eigenvalues and eigenvectors of R.

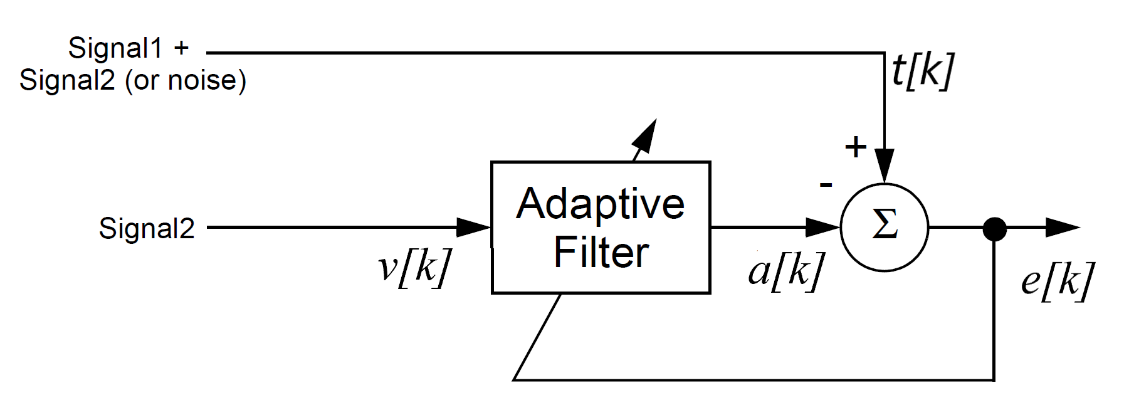
1. For part (b), we know the maximum stable learning rate for the LMS algorithm is:

**2 / max(eigenvalues)

1. For part (c), we coded the LMS algorithm in **<Python/Matlab>.** Here is a photo of the code:



1. For part (d), we plotted the original and restored signals in **<Python/Matlab>.** One graph shows both signals as time passes while the other graph shows the difference between the two signals.
2. For part (e), in the results section, we explained why the restored signal is not exactly the same as the original.
3. We repeated parts (a) through (d) for *m*(*k* ) 1.2 sin 2*k* / 3 3**/ 2.
4. For part (g), two recorded signals were used. The desired signal s(k) and the noise signal v(k). Both sounds were read using MATLAB and the system was set up as shown below in figure 2.



*Fig 2. Noise Cancellation System II*

IV. Results  
*Original m(k).*

1. For part (a), we used Matlab to compute matrix R. The Hessian matrix we calculated was 2R = . From here, we were able to find the two eigenvalues (**2.16** and **0.72**) and the two eigenvectors ()**.** After using Matlab to find h (which equals , we located the minimum point (x\* = R–1h)and got x\* **= .**

Here is a photo of the contour plot sketched in Matlab with **0.12.

A screenshot of a cell phone

Description automatically generated

1. For part (b), to find the maximum stable learning rate for LMS algorithm, we did **2 / max(eigenvalues), so ** 2 / 2.16 = **0.926.**
2. For part (c), we implemented the LMS algorithm and our weights and bias were

A screenshot of a cell phone

Description automatically generated

1. The resulting figures illustrate how the filter adapts to cancel the noise. The top graph shows the restored and original EEG signals. It takes about \_\_\_\_ seconds (with ** = 0.12 ) for the filter to adjust to give a reasonable restored signal. The difference between the original and restored signal is shown in the lower graph.

**A screenshot of a social media post

Description automatically generated**

1. **Comment on why restored isn’t same as original.**

*New m(k)*

1. For part (a), since v(k) didn’t change, the eigenvalues and eigenvectors are the same. Since m(k) changed, “h” will have to be recalculated. The new “h” is **.** Then, we located the minimum point (x\* = R–1h)and got x\* **= .**

A screenshot of a cell phone

Description automatically generated

1. For part (b), since the eigenvalues didn’t change, the maximum stable learning rate for LMS algorithm will stay the same at **0.926.**
2. For part (c), we implemented the LMS algorithm and our weights and bias were

A screenshot of a cell phone

Description automatically generated

1. The resulting figures illustrate how the filter adapts to cancel the noise. The top graph shows the restored and original EEG signals. It takes about \_\_\_\_ seconds (with ** = 0.12 ) for the filter to adjust to give a reasonable restored signal. The difference between the original and restored signal is shown in the lower graph.

A screenshot of a social media post

Description automatically generated

**For Part (g)….**

V. Conclusion

In conclusion, this project was very helpful in further understanding and applying the material. The LMS algorithm did a great job of filtering out the noise and restoring the signal almost completely back to its original form. The methods and results described above provide a snapshot of a few of the applications for the LMS algorithm. Overall, this project was a great overview of how to apply these concepts in the real world.