EE 4745 – Neural Computing

Project 2: Noise Cancellation using Adaptive Filter

Andres Ponce

Javier Solorzano

Donovan Groschang

Electrical Engineering Department

Louisiana State University

15 November 2019

Content

[Abstract 2](#_Toc24573150)

[Introduction 2](#_Toc24573151)

[I. Mean square error 2](#_Toc24573152)

[1) Eigenvalues and eigenvectors of the Hessian 2](#_Toc24573153)

[2) Minimum point 2](#_Toc24573154)

[3) Contour plot 3](#_Toc24573155)

[II. LMS algorithm 3](#_Toc24573156)

[1) Maximum stable learning rate 3](#_Toc24573157)

[2) Algorithm implementation 3](#_Toc24573158)

[3) Results 3](#_Toc24573159)

[III. Noise change 4](#_Toc24573160)

[IV. Implementation on an audio signal 6](#_Toc24573161)

[References 6](#_Toc24573162)

# Abstract

The objective of this project is to apply neural networks to a noise cancellation system with an adaptive filter. The team was tasked to implement the LMS algorithm for the system shown below and be able to successfully remove the signal noise from the original s(k) signal.

# Introduction

Noise removal is essential for many signals processing operations and using neural networks has shown to be an efficient method to remove unwanted noise from a signal. As shown above, the original signal will use an adaptive filter, whose inputs are manipulated to minimize the error, or noise, of the system. The adaptive filter, in short, will be attempting to copy the original signal using only the original noise source, so it can only reproduce the part of the signal that is linearly correlated with the original noise v(K). The team will be implementing the LMS algorithm to the adaptive filter to restore the signal. The procedure is shown below.

# Mean square error

## Eigenvalues and eigenvectors of the Hessian

Following equation 10.12 [1], the mean square error can be written as follow: and as a quadratic function: where A is the Hessian of F. This results in and .

R is defined as: with . Thus .

Thus, and therefore .

Now that with have the Hessian of the mean square error, we can determine its eigenvalues u and eigenvectors l using Matlab:

and therefore and .

## Minimum point

The minimum point is defined so . We have .

Therefore or .

h is defined as: where t is the target. In our case our target is the signal m. Therefore:

We can compute the inverse of R using Matlab and end up with: .

## Contour plot

The contour plot seems coherent with the eigenvalues, eigenvectors and minimal point previously computed (see Fig. 1).

# LMS algorithm

## Maximum stable learning rate

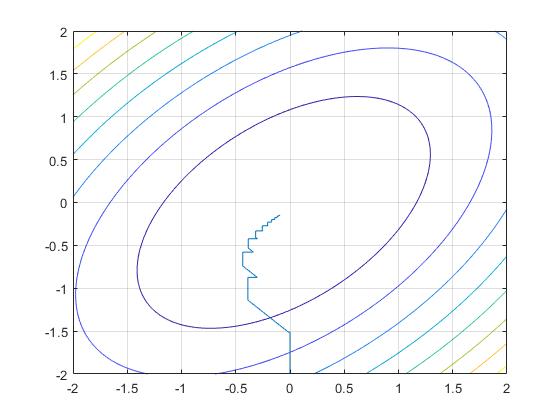
Figure : Contour plot of the mean square error

The learning rate should satisfy the following condition: where is the greatest eigenvalue of the Hessian. In our case therefore the learning rate should be so

## Algorithm implementation

To apply the LMS algorithm to this problem, we’ll consider a bias b = 0 and start with a initial point (or initial weight): . Each following weight will be determined as follow:

Where:

* e(k) is the difference between the noise added to the signal (actual target) and the output of the adaptive filter:
* is the learning rate; we’ll take as suggested but any value respecting the maximum learning rate condition should allow the algorithm to converge
* is the input vector:

The stopping condition of the algorithm will be: .

The code is attached in Annex 1.

## Results

First, plotting the weight onto the previous contour plot, we can see that the algorithm converges toward a solution which seems close to our minimal point (see Fig. 2). Then plotting the error, we can see that it is minimized accordingly (see Fig. 3). Finally, plotting the original signal and the recovered signal, we can see that those two become the same as the error is minimized (see Fig. 4).

Figure : LMS Algorithm onto the contour plot of the mean square error

The number of iterations needed for the algorithm to converge depends on the learning rate and the error threshold wanted. Here with our parameters ( and ) the program converges in 34 iterations.

However, we can note that the original and recovered signal will never be truly the same as the error can be minimized but can never reach a true 0. If we let run the algorithm for a greater number of iterations (for example 70) we can see that the result is satisfying as the error decrease and the original and recover signals merge (see Fig. 5 and Fig. 6).

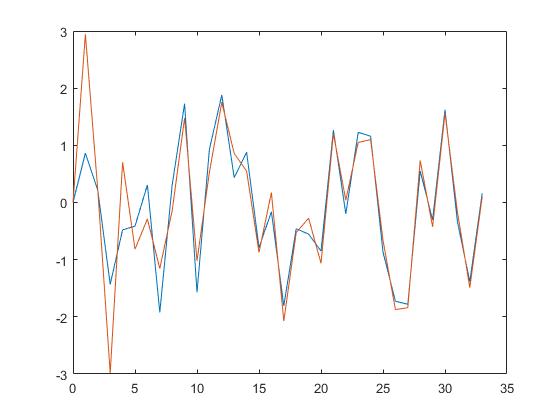
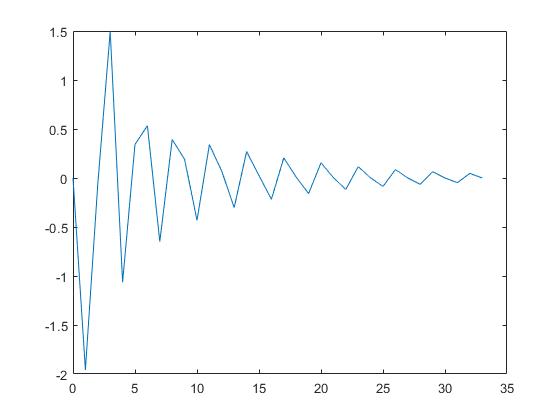
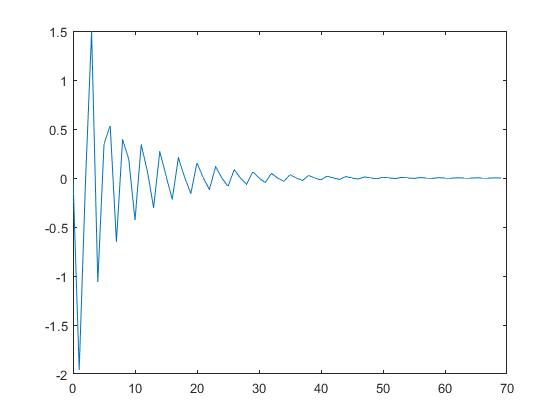
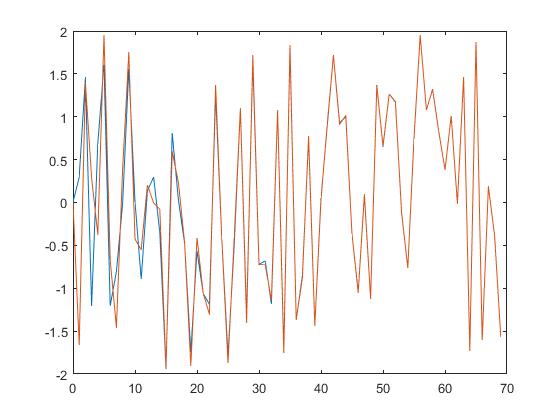


Figure 4: Original signal (blue) and recovered signal (orange)

Figure 3: Mean square error

Figure : Original signal (blue) and recovered signal (orange) extended to 70 iterations

Figure 5: Mean square error extended to 70 iterations

# Noise change

The noise path we used distorted the original noise v(k) on both phase and amplitude (which was reduced by a magnitude). Let’s now test our filter with a different noise path having an impact only on the phase. In other words, the noise contaminating the signal is now 10 times stronger. To modelize this, we’ll now take: . (Note: The phase distortion can seem different, but the sinus function is -periodic, which means than ).

If we look closely, the Hessian of the mean square error doesn’t depend of the noise m(k). Therefore, the eigenvalues and eigenvectors of the Hessian are the same as previously and so is R.

The minimum point however is going to be affected as and .

Thus,

And our minimal point is now:

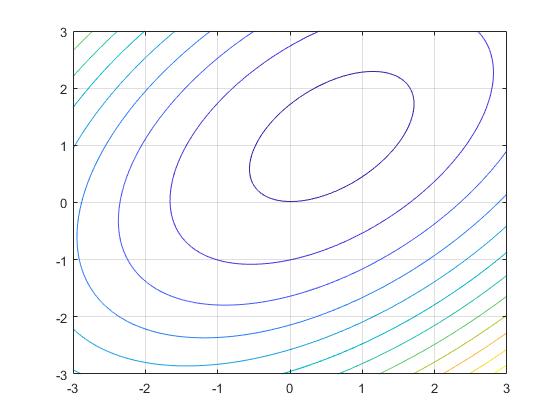
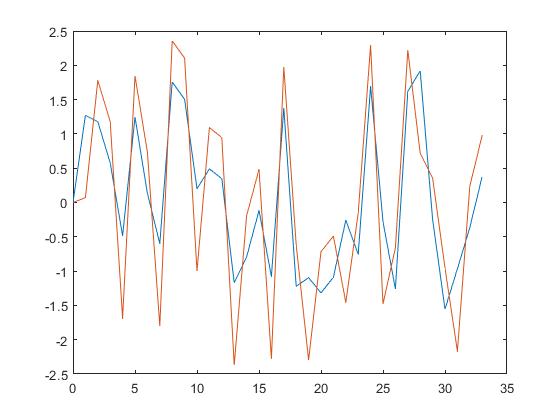
If we take a look at the contour plot, we can see that it is similar to the first one (as the eigenvectors and eigenvalues are the same) except for the minimum point (center of the ellipse) whose coordinates seem match those which just calculated for (see Fig. 7).

Figure 7: Contour plot of the mean square error

Again, as the eigenvalues of the Hessian of the mean square error are the same, our maximum learning rate will be the same as well:

When simulating the circuit with those new inputs, we can at first see how the noise is stronger than in the previous case when plotting the original signal and the contaminated signal (see Fig. 8). However, let’s see if the algorithm still manages to recover the original signal. Looking at the contour plot of the algorithm on the contour plot, we can see that it converges toward the minimum point previously calculated (see Fig. 9).

Looking at the error, we can see that it is still minimized and reach the error threshold in 34 iteration as well (see Fig. 10).

Finally, looking at the original and output signal, we can clearly see that those two merges as in the previous case (see Fig. 11). Therefore, even with a noise 10 times stronger, the LMS algorithm is still able to recover the original signal and do it as quickly.

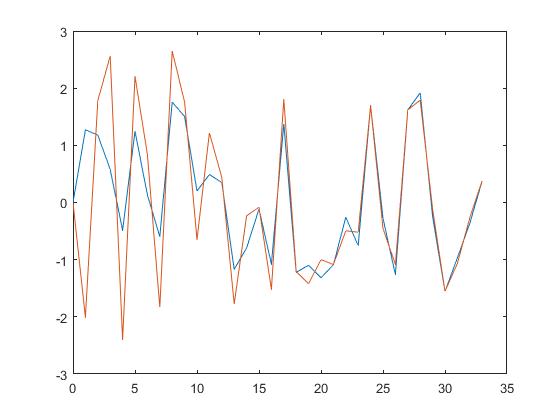
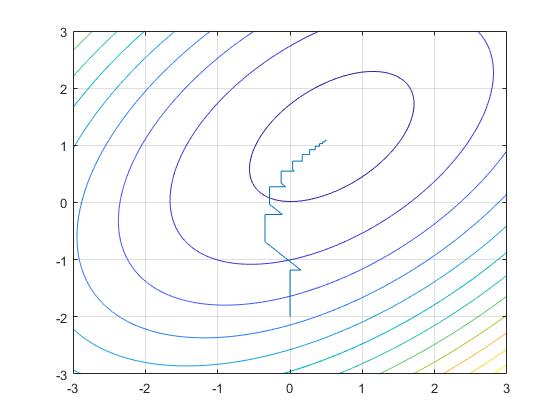
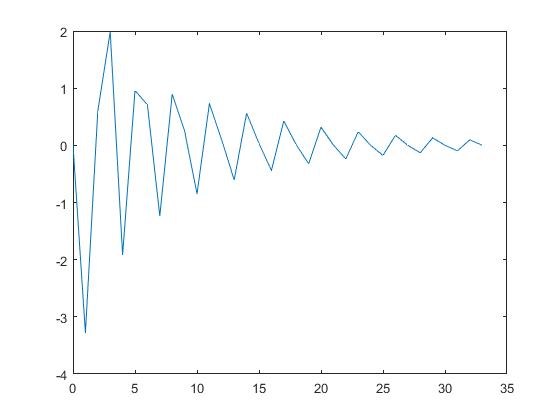


Figure 8: Original signal (blue) and contaminated signal (orange)

Figure 10: Mean square error

Figure 9: LMS algorithm onto the plot of the mean square error

Figure 11: Original signal (blue) and recovered signal (orange)

# Implementation on an audio signal

# References