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

[Introduction 2](#_Toc24479072)

[I. Mean square error 2](#_Toc24479073)

[1) Eigenvalues and eigenvectors of the Hessian 2](#_Toc24479074)

[2) Minimum point 2](#_Toc24479075)

[3) Contour plot 2](#_Toc24479076)

[II. LMS algorithm 2](#_Toc24479077)

[4) Maximum stable learning rate 2](#_Toc24479078)

[5) Algorithm implementation 2](#_Toc24479079)

[6) Results 2](#_Toc24479080)

[III. Noise change 3](#_Toc24479081)

[IV. References 4](#_Toc24479082)

# Introduction

# 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.

# LMS algorithm

## Maximum stable learning rate

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: .

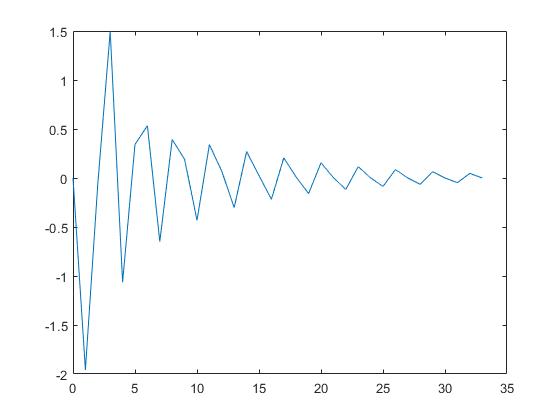
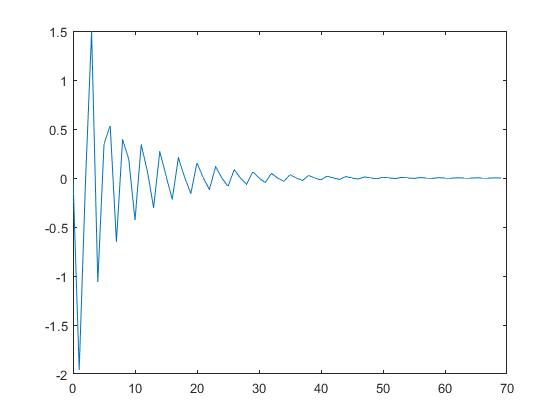
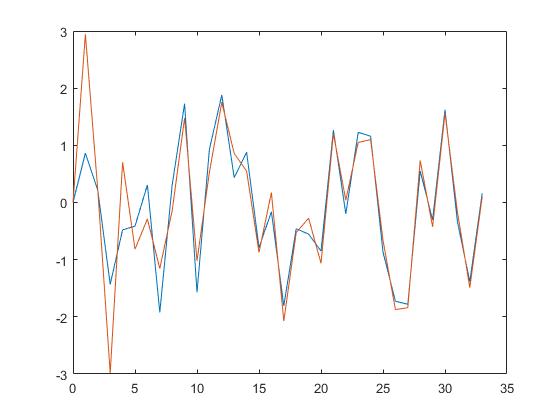
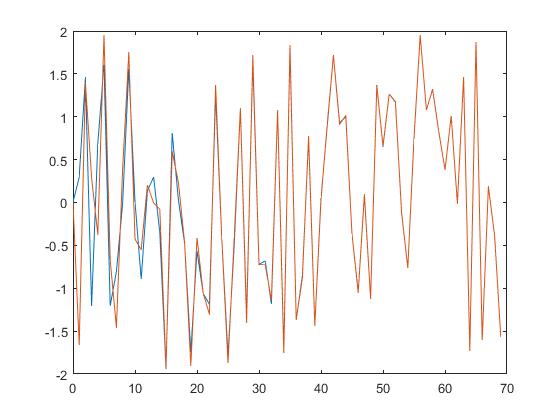
The code is attached in Annex 1.

## 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. XXXXXXXXX). Then plotting the error, we can see that it is minimized accordingly (see Fig. XXXXXXXXX). 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. XXXXXXXXX).

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 pretty satisfying ass the error decrease and the original and recover signals merge (see Fig. XXXXX and XXXXXX).



# Noise change

# References