Interfering Tone Suppression (June 2020)

Joel Delgado Garrido and Esther Martin Cuartero, EETAC

Abstract- This document serves as our answer to the SESSION 4 → Interfering Tone Suppression, for any further clarification on the subject debated in this document please send an email to esther.martin.cuartero@estudiantat.upc.edu joel.delgado@estudiantat.upc.edu

The main objective of this lab session is to design a digital system capable of improving the quality of a signal contaminated by a disturbing beep. In particular, the music is played by a record player. In order to 'clean' (remove the interference) the analogue signal, such signal has to be first digitized and then digitally processed.

I. INTRODUCTION

The signal (a little song) we will use, has an interference beep that has to be deleted. In order to achieve it, we have to sample the signal using MATLAB for detecting and gathering all the available information of the tone that is contaminating the original song.

To be succeed, we are going to follow two steps:

1. Analyse the spectral of the digital contaminating interference.
2. Filtering the algorithm to remove the unwanted beep.

II. PROCEDURE FOR PAPER SUBMISSION

Along this paper you will find the code for the required function.

First of all, we need to load on of the ***.wav*** files that we could find in Atenea (e.g. choose ***Ring(Beep).wav***). MatLab will automatically give us the sampling frequency of this signal, in this case fs=44100. This file contains two signals: the main signal is a song and the other one is a beep that disturb the main signal of the file. Our target is to delete this beep and in order to be able to do that we need to program a function that allow us to do it.

This function needs to have as inputs: the signal (x), the sampling frequency (fs) and a value r that we will use to create the filter (h(n)). In order to obtain our signal without the beep we need to convolve the signal x[n] with a filter h[n]:

y[n]=x[n]*h[n]

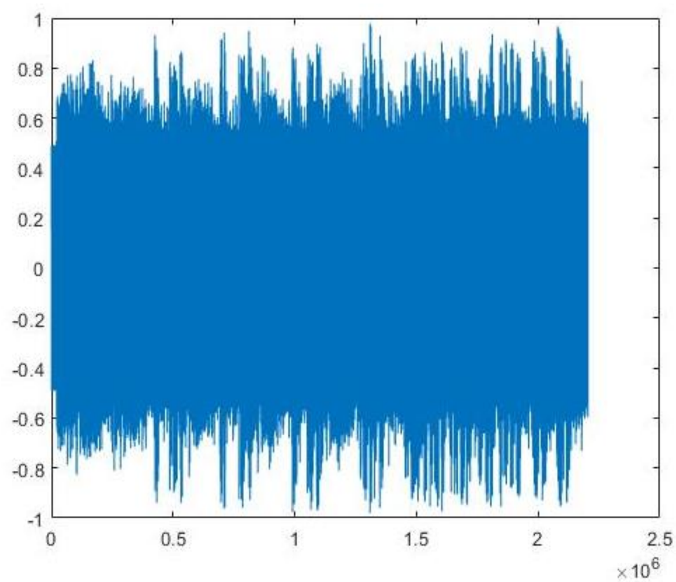

Figure1. Plot of the signal x[n].

In Figure1 we aren't able to differentiate the signal of the song and the signal of the beep. So we will do the Fast Fourier Transformation (FFT) of the signal x[n], plot the function X[k] and watch its spectrum.

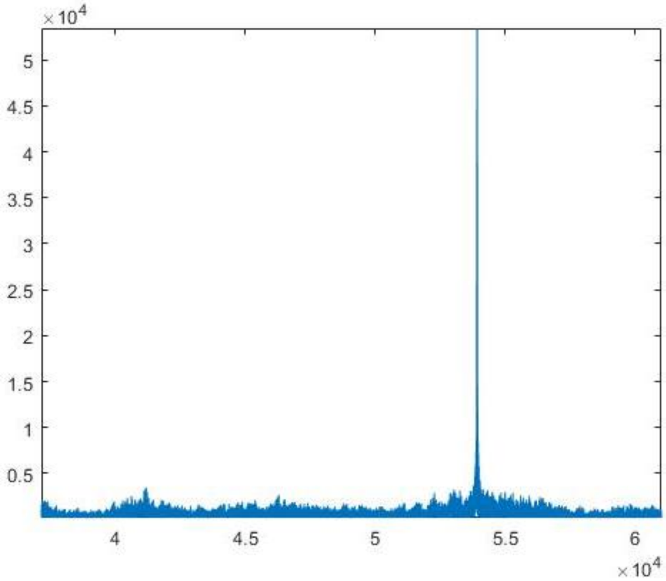

Figure2. Plot of the signal X[k].

Now we need to find and appropriate filter to wipe out the beep. As we can see in the picture, the beep has only one frequency so the filter we must use in this situation will be a band removed filter (BRF). To implement this BRF, the zeros of the filter must be placed in the same frequency of the beep interference. The filter could be FIR or IIR, the only difference will be the value of "r", if we want to implement a FIR filter, r needs to be 0 and if we want to implement a IIR filter, r needs to be higher than 0 and lower than 1, otherwise the filter will be unstable and wont work. Also, we prove that if we use a r=0.9 we will obtain a better exit signal audio quality.

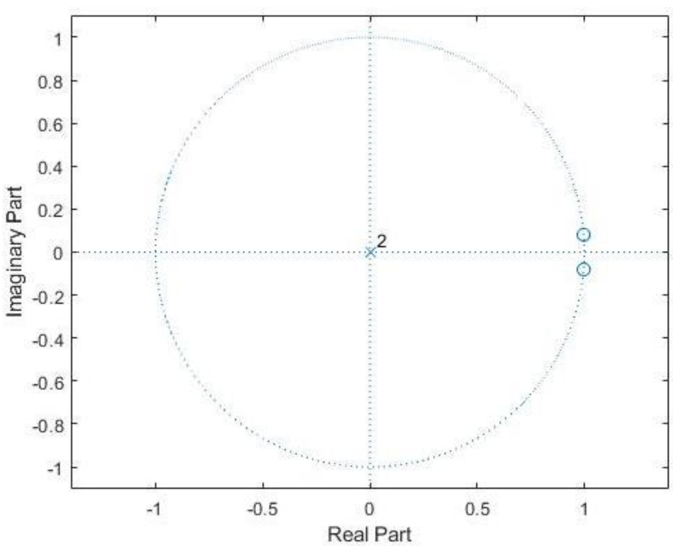

Figure3. Zero-Pole Diagram of the FIR filter.

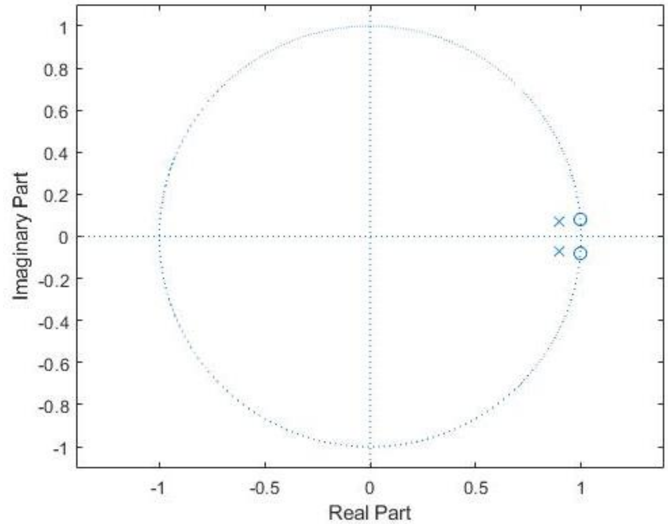

Figure4. Zero-Pole Diagram of the IIR filter.

The IIR filter follows the equation:

$$H(Z) = \frac{(Z - e^{j\Omega})(Z - e^{-j\Omega})}{(Z - re^{j\Omega})(Z - re^{-j\Omega})}$$

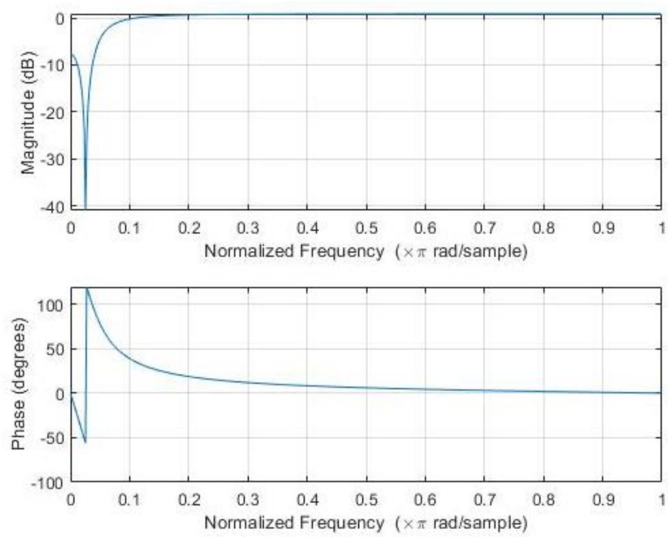

Figure5. IIR module response.

We will use the IIR filter because we think that using it we will obtain a better audio quality of y[n]. In this point the funtion that we need to program will find the maximum value of the X[k] spectrum, which will be the interference tone. Next, it will calculate the Ω of this interference in order to obtain the filter h[n].

Finaly we are going to convolve the signal x[n] with the filter h[n]. There are two ways to do this.

1. Convolve directly with the comand:

```
y = conv(x,h);
```

2. Using the Inverse Fourier Fast Transformation (IFFT) of the Y[k], that we obtain for doing the product of the spectrum of x[n] and the spectrum of h[n]:

```
Y=X.*fft(h,N);
y=ifft(Y,N);
```

After doing that we will obtain a signal y[n] that wont have the beep interference.

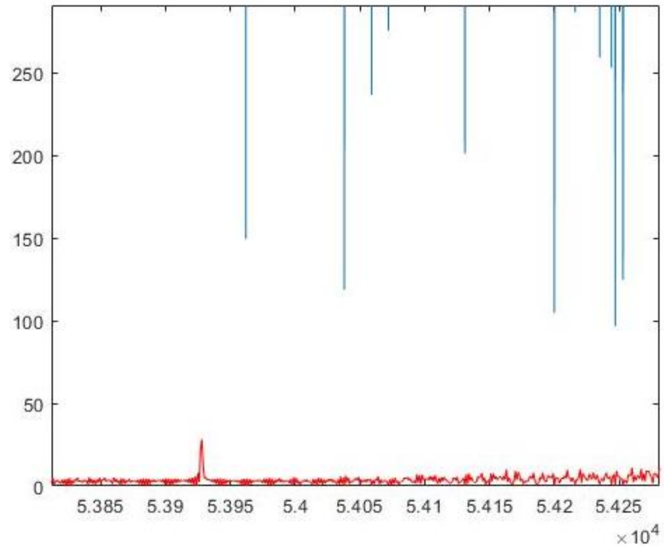

Figure6. Plot of spectrum signals.

In the picture below, there are two types of spectrums: in blue there is the spectrum of the signal x[n] and in red there is the spectrum of the signal y[n]. As we can see, the filter reduced the signal amplitude and erase the beep interference. Now, we can listen the song clearly.

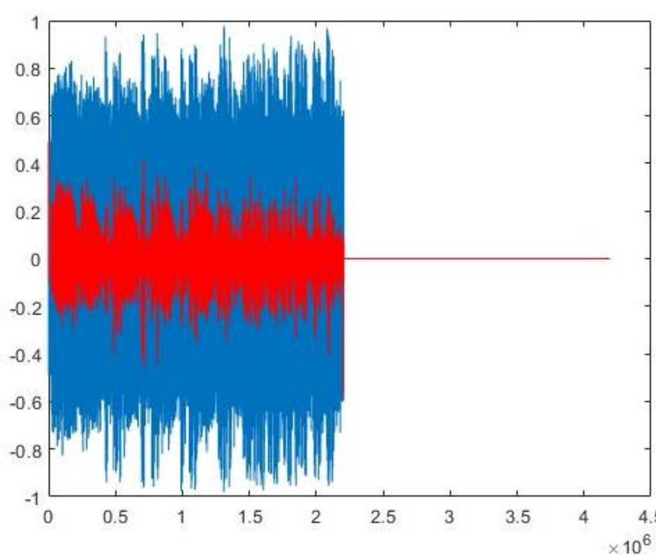

Figure7. Plot of both singals: x[n](blue) and y[y](red)

UNITS

These are the units we use during the entire lab session:

- "fs" in Hz.

III. SOME COMMON MISTAKES

Here we have some examples of mistakes we had during the lab session:

- Saving the functions in a folder where the program is not capable to find them.
- Not to put "N" as a power of two.
- Not to put "N" bigger than "L".

IV. CONCLUSION

Due to this practice about Interfering Tone Suppression, we have learned how we can suppress and interfering a tone on a using new functions and commands on MATLAB. It was important to choose the correct filter and know exactly what frequency we had to delete; this was possible thanks to the zero-pole diagram because we located the zeros.

Also, before doing the lab session, we had to do a background study for a better understanding of the concepts.

In our opinion, it was interesting to carry out all the knowledge we learned doing the background study and all the concepts of the first, second and third session of MATLAB. Finally, in the development of this practice, we enjoyed to play with the signal (the song) so, it was a practical mode to learn about the subject.

V. REFERENCES

[1] Introduction to MATLAB: *https://atenea.upc.edu/pluginfile.php/3123187/mod_resource/content/3/Background_Study_Introduction_to_MATLAB.pdf*

[2] Interfering Tone Suppression: https://atenea.upc.edu/pluginfile.php/3123197/mod_resource/content/3/PDS%20%20Interfering_Tone_Suppression.pdf

[3] Official web page of MATLAB: *https://es.mathworks.com/help/matlab/getting-*

started-with-matlab.html
[4] “Señales y sistemas de tiempo discreto”, Eduard Bertran, Ed. UPC: *https://bibliotecnica.upc.edu/edupc/locate4.asp?codi=TL030XXX*