# DIGITAL SIGNAL PROCESSING - LAB

Interfering Tone Suppression

## LABORATORY SESSION AND BACKGROUND STUDY

## INTRODUCTION

The objective of this laboratory 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.

To design the 'cleaning' algorithm (based on digital filtering) we have available a sample of the whole song (of about 30 seconds of duration) digitized in WAV format (16 bits, stereo, 44100 Hz).

Steps for the design of the 'cleaning' system

1.- Spectral analysis of the digitized signal containing the disturbance.
2.- Design of the necessary filtering algorithm to remove the unwanted tone or beep.

## BACKGROUND STUDY

To improve your knowledge of digital second order filters, please read page 303, section: "*7.7 Filtros de segundo orden*" of the book "*Señales y sistemas de tiempo discreto*", Eduard Bertran, Ed. UPC. You have access to the book online (you must be connected with an IP from UPC) at the following URL: http://bibliotecnica.upc.es/edupc/locate4.asp?codi=TL030XXX

Background study question 1: Which is the type of filter necessary to reject only one specific frequency? Plot the zero-pole diagram of the filter. Consider both FIR and IIR implementations.

Background study question 2: Write the discrete (or pulse) transfer function ($H(z)$) for both second order filters (FIR and IIR) described in the previous question.

Background study question 3: Plot the magnitude ($|H(e^{j\Omega})|$) of the frequency response of the filter.

## LABORATORY WORK

### a) Spectral analysis of the digitized signal containing the disturbance.

In a first approach we would like to detect and gather all the available information of the tone contaminating the original song. This can be done using the the Matlab instruction '*fft*'. A fast Fourier transform (FFT) is a computationally efficient algorithm to compute the discrete Fourier transform (DFT). The FFT algorithm has to be applied to sequences with a length equal to a power of 2 ($2^N$), which always can be easily accomplished by simply applying zero padding.

1. Copy the audio file (*jaguai.wav*) in your work path.
2. Using your headphones, listen to the *jaguai.wav* file. You will notice that the song has been contaminated by a disturbing beep. The song has been sampled at a *fs* of 44100 Hz and using 16 bits per sample.
3. Open Matlab and load the contents of the file *jaguai.wav* in a variable named *xstereo*. To load the contents of a wav file, search the Matlab help for the instruction '*audioread*'.
4. Split the stereo signal into two mono signals (one for the right channel and one for the left one):

```
>> xR=xstereo(:,1);
>> xL=xstereo(:,2);
```

5. Plot the waveform signal of one of the mono channels, for example, the right one (*xR*):

```
>> plot(xR)
```

6. Read the Matlab help for the following instructions: *max*, *abs*, *fft*. Plot the absolute value of the FFT of the right channel of the signal (*xR*). Identify the interference tone and answer the following questions:

Lab. question 1: Which is the digital frequency (Ω) of the interference tone?

Lab. question 2: Which is the analog frequency [Hz] of the interference tone?

## b) Design and implementation of the filtering algorithm

Lab. question 3: According with the background study questions, which kind of digital filter is able to suppress the interference tone?

Lab. question 4: Plot the zero-pole diagram of the *H(z)* of the filter and the magnitude ($|H(e^{j\Omega})|$) of the frequency response of the filter specifying the most significant values. Consider both FIR and IIR implementations.

7. Read the Matlab help for the following instructions: *roots*, *poly* and *tf*. Define the transfer function H(z) of the digital filter defined previously. Consider both FIR and IIR implementations of the designed filter.
8. Read the Matlab help for the following instructions: *pzmap, zgrid, zplane, freqz.* Plot the zero-pole diagram and the frequency response of the filter H(z). Check if the filter shows a frequency response able to filter the undesired tone.
9. Read the Matlab help for the following instruction: *dimpulse.* Find the impulse response (h[n]) of the filter. Note that for an IIR filter the impulse response is infinite. However, unless we specify a particular length for the impulse response of an IIR filter, the Matlab instruction *dimpulse* already truncates it.
10. Using the *fft* algorithm, perform the convolution of the contaminated signal *xR* and the impulse response of the filter *h[n]*. To do that, you need to remember how to calculate the linear convolution using the DFT, that is, how to proceed so that the linear convolution and the circular convolution are equal.

**REMEMBER:** $y_R[n] = IDFT_N\left\{DFT_N\left\{x_R[n]\right\}\cdot DFT_N\left\{h[n]\right\}\right\} = IDFT_N\left\{X_R[k]\cdot H[k]\right\}$

with $N=L_x+L_h-1$, and where $L_x$ and $L_h$ are the lengths of the input sequence $x_R[n]$ and the filter impulse response $h[n]$, respectively.

11. Alternatively, you can use the Matlab functions: *conv*, filter or *fftfilt*.

**REMEMBER:** $y_R[n] = x_R[n] * h[n] = \sum_{k=-\infty}^{\infty} x_R[k]h[n-k] = \sum_{k=-\infty}^{\infty} x_R[n-k]h[k]$

12. Plot the spectrum of the resulting filtered signal $y_R[n]$ and check if the interfering tone has been suppressed.
13. Finally, to compare results, play both the filtered signal in the right channel and the contaminated signal in the left channel using the '*soundsc*' function:

```
>> soundsc([yR,xL],fs)
```