**University of Nottingham**

**Faculty of Engineering**

**Department of Electrical and Electronic Engineering**

**H64DSP – Digital Signal Processing for Telecommunications, Multimedia and Instrumentation**

**Coursework- Filtering of a Sound Signal**

**1. Introduction**

The coursework component for H64DSP is worth 30% of the module assessment. The coursework makes use of MATLAB to process sound signals which have been corrupted by noise and interference.

There are Workshops that take place between 10-11 on Friday mornings in Room 402 Tower, where you have access to computers running MATLAB but you can of course work on this project outside this slot.

The main educational aim of this project is to reinforce the theoretical material introduced in the lectures. You will have the opportunity to apply a wide variety of filters and other signal processing techniques and you should, by the end of the project, have a better understanding of these techniques.

**2. Background to the Project**

**2.1 Data Files**

You are provided with MATLAB files containing a piece of music:

*clean.wav -* Clean signal

*corrupted.wav -* 10 different versions of the clean signal corrupted with noise and other artefacts.

Information about reading and writing sound waves in MATLAB can be found at:

<http://uk.mathworks.com/help/matlab/import_export/read-and-get-information-about-audio-files.html> (last accessed 29/9/15).

For example, if you want to read an audio file, then you use the command **audioread**:

e.g. [y, fs] = audioread(‘corrupted.wav’)

The array *y* will contain the data values (e.g. for *corrupted.*wav :10 columns and 50000 rows with each column corresponding to one of the ten corrupted signals). fs will give you the sampling frequency (Hz).

If you want to hear the signal, then you should use the **sound** command:

sound(y,fs)

Hearing the signal after processing may give you an indication of how close to the clean signal is the processed signal. However, if loudspeakers are not available, comparing the plots of the variations with time of the clean and processed signals is just as good a procedure for comparison.

After having processed the data, if you want to write it to a file with \*wav format then you should use the **audiowrite** command:

e.g. audiowrite(‘processed.wav’,y,fs)

**2.2 Aim of the Project**

The aim of the project is to process the 10 sets of data in *corrupted.wav* in order to recover as much as possible the original signal contained in *clean.wav*. **Please note:** *clean.wav* is provided for your reference – information from this file must not be used to process the signal in *noise.wav*.

You can use a variety of methods to process the signal. Digital filters are introduced in Lectures 5 and 6 but you are welcome to read ahead and use some of these filters earlier if you want to.

**3. Deliverables**

You should produce and hand in and submit to the ESLC the following:

(1) A report, approximately 15-20 pages in length, which should be word processed. See below for more details of the format of the report.

(2) A CD accompanying the report ,with your name and student id on the front, with the data files that you have produced after processing the signal. Your report should cross-reference to these files explaining clearly how you have processed the data to produce the filtered output. It should be possible for me to reproduce the data that you have produced.

**4. Report**

**4.1 Deadline**

You will need to hand in the report +C D by

**Monday, December 7th before 3 pm** .

**4.2 Structure of the Report**

The report is to be structured as follows:

*(1) Introduction*

What are the aims and objectives of the project?

*(2) Methods and Results*

This is the main section of the report and should describe the following:

* The signal processing methods you have used and how you have implemented them using MATLAB.
* The reason why you have used that particular method. Justification for using specific choice of input parameters.
* Results shown in graphical form
* Give filenames for processed data that are produced – these files should also be on the CD.

*(3) Discussion of Results*

* If a particular combination of signal processing methods worked well, then why is that the case?
* If a particular combination of signal processing methods worked badly, then why is that the case?

*(4) Conclusions*

Recommendation for the best combination of single processing methods to be used and the reasons you are recommending this combination.

**5. Assessment**

The assessment of the report will be as follows:

(1) Introduction – 10%

(2) Methods and Results – 40%

(3) Discussion of Results – 40%

(4) Conclusions – 10%

*Note on assessment:*

You will not be penalised for trying out a method that does not work – this happens all the time with research! You will be given credit for explaining why a method does not work (or works well). You will not receive much credit if you present a lot of results with no explanation.

**6. Connection with the Lectures**

Many of the signal processing methods introduced in the lectures will be relevant to the coursework. This project should help you in the understanding of these methods. Digital filter methods are introduced later in the module, but here are some activities you can do before then:

(1) Plot the corrupted and clean signals as a function of time. What features do you note in the noisy signal that are not present in the clean signal? How could you get rid of them?

(2) Use the FFT command to look at the amplitude spectrum of the corrupted signal. Are there any features there that can be attributed to noise and interference?

(3) **medfilt1**  - Could this be useful? Read up more on median filtering at:

<http://www.nptel.ac.in/courses/117104069/chapter_8/8_16.html>

During the course of the lectures, various MATLAB commands will be noted in the lectures – try reading up on these in the Mathworks web site to see if they could be useful:

**Spectral Analysis**

Demonstrations: **fftdemo**

• DFT: **fft ifft**

• Windows: **bartlett triang hanning hamming blackman chebwin kaiser**

• JTFA: **spectrogram**

• PSD (Welch) : **pwelch** (used to be called **psd**)

**Digital Filtering**

• **fir1** truncation +weighting for common filters (LP etc)

• **fir2** truncation + weighting for a sampled magnitude response

• **firls** least square design

• **firpm** chebyshev design

• Filter properties **grpdelay freqz impz zplane**

• **conv filter fftfilt** (to implement filtering)

• **SPTOOL** (versatile tool for design, filtering, analysis)

*Malcolm Woolfson, October 4th 2015*