ELEC6021 Research Methods coursework: Signal processing in Matlab

These exercises form your assessment for the "Introduction to Matlab" part of the Research Methods course, and contribute 25% of your mark for that course. The work handed in must be entirely your own. Hand in your results to ECS reception by 4pm on Thursday 13/11/2008. Your submission should consist of the *fully commented* listings of the Matlab programs that you are to write, the graphs that you are to produce and the answers to any specific questions asked. Ensure that your graphs (particularly the axes) are properly labelled and that they clearly show what they are intended to. A report is not required. Avoid using `for` or `while` loops in your code where you can. You may not use any of the signal processing toolbox functions for this coursework.

Exercise 1

Before we can perform signal processing on a signal, we typically have to *sample* it first. Sampling picks out the value of the signal at regular time intervals.

- In the lab session on 15/10/2008, we looked at how to write Matlab functions. Your first task in this coursework is to write a function that outputs a vector of samples that are taken from a sinusoidal signal $A\sin(2\pi ft+\theta)$, where A is the sinusoid's amplitude, f is its frequency and θ is the phase. The parameters of your function should be A, f, θ, f_s, t_{start} and t_{stop}, where the sampling is performed with a frequency f_s, over the time interval $[t_{\text{start}}, t_{\text{stop}}]$. Pick some values for these parameters (using $f_s = 10f$ for the moment) so that the output represents a few cycles of the sinusoid. Plot this output.

- If a sinusoid having a frequency f is sampled at a frequency f_s that is less than $2f$, the result becomes distorted due to aliasing. Repeat the function call from before but this time, use a sampling frequency of $f_s = 1.5f$ and plot the resultant under-sampled output in the same figure as your over-sampled one.

- Write some code to clip any values in your over-sampled output that exceed $\pm A/2$, so that they take on a value of $\pm A/2$. Plot the result in the same figure as before.
- Include a listing of your function and the code you used to clip its output. Also, provide the figure containing your three plots in your write-up. Explain clearly what aliasing is and how you have demonstrated it.

Exercise 2

Once a signal has been sampled (like in Exercise 1), we need to digitise it before we can do any digital signal processing. But, typical signals are analogue, meaning that the samples can have any value. On the other hand, *digital* samples can only have values taken from a discrete set. To elaborate, we could use n number of bits to represent each sample. Here, there are 2^n possible combinations of n number of 0s and 1s. We can use each combination to represent a different digital sample value. For example, if we use $n = 3$ bits per sample, there are $2^3 = 8$ combinations {`000`, `001`, `010`, `011`, `100`, `101`, `110`, `111`}. We can use each of these combinations to represent a different one of the digital sample values in {-0.875, -0.625, -0.375, -0.125, 0.125, 0.375, 0.625, 0.875}, for example. We can digitise (or *quantise*) the analogue samples by rounding them to whichever of the digital samples values is nearest and, hence, provides the best approximation.

- Write a Matlab function that quantises some samples using an n-bit representation. Assume that you'll be dealing with samples having values that are uniformly distributed between -1 and $+1$. However, make sure that your function can deal with samples that break from the uniform distribution and have values that exceed ± 1. Your function should take a value for n and vector of samples as its input. The output should be a vector of values taken from your discrete set. Do not use `uencode` or any other built-in or toolbox function in this exercise. Note that it is possible, though tricky, to write this function without using any loops.
- Include a fully commented listing of your function in your write-up and explain its features.

Exercise 3

In this exercise, we'll do both sampling and quantisation.

- Use the functions that you wrote in Exercises 1 and 2 for this exercise. Plot a single cycle of a sinusoid that has an amplitude of $A = 1$. In the same figure, plot a version that is quantised using a low number of bits.

- Include the figure in your write-up, together with a brief description and explanation of what it shows.

- Come up with a way for quantifying the distortion introduced by quantisation and explain this in your write-up. Quantify the distortion shown in your figure and include this in your write-up.

Exercise 4

A set of N points $\{(x_i, y_i)\}_{i=1}^{N}$ in a scatter plot can be approximated by a straight line $y = mx + c$ by solving a pair of simultaneous equations:

$$\sum_{i=1}^{N} x_i y_i - m \sum_{i=1}^{N} x_i^2 - c \sum_{i=1}^{N} x_i = 0, \tag{1}$$

$$\sum_{i=1}^{N} y_i - m \sum_{i=1}^{N} x_i - cN = 0. \tag{2}$$

- Write a Matlab function that outputs m and c when provided with a vector of x coordinates and a vector of y coordinates.

- Make up a set of points and test your function. Put a scatter plot of the points and its straight line fit in a single figure.

- Include a listing of your function and your figure in your write-up. Provide a brief description of how this might be useful in experimental work.

- Come up with a way for quantifying the "goodness of fit" that the line has to the data and explain this in your write-up. Include a value that quantifies the "goodness of fit" shown in your figure.

Exercise 5

A filter can be described by a difference equation, such as:

$$y_i = 0.3x_i + 0.6x_{i-1} + 0.3x_{i-2} - 0.9y_{i-2}, \qquad i \in [1 \ldots N]. \tag{3}$$

- Write a Matlab function to output the vector $\mathbf{y} = \{y_i\}_{i=1}^N$ that results when a vector $\mathbf{x} = \{x_i\}_{i=1}^N$ is input into the above equation. Assume that any values that you dont know, for example x_{-1}, are zero. I think you'll need a loop in this function, so feel free to indulge this time!
- Create an $\mathbf{x}$ vector comprising $N = 128$ elements, the first of which is 1 and the rest zeros. Plot the output vector $\mathbf{y}$ that results from inputting your $\mathbf{x}$ vector into your difference equation function.
- In your write-up, include a listing of your function, your plot of the $\mathbf{y}$ vector and a description of the behaviour shown in the plot.

Jeff Reeve 15/10/2005
James Wilkinson 15/10/2007
Rob Maunder 06/10/2008