Worksheet 2: Quantisation

Note: The quantiz function requires the Signal Processing and Communications Toolboxes in MATLAB to be installed. This should be setup for you in the online MATLAB version.

Exercise 1 (Quantisation Example) This exercise covers generating a sinusoidal voltage signal and quantising it in MATLAB.

a. Generate and plot 100 samples of a sine wave whose frequency is (1/50)=0.02 times the sample rate using the following code:

```
t = 1:1:100;  % define time samples
y=7.5*(sin(2*pi*0.02*t)+1);  % define sine wave with range 0-15
plot(t,y,'b-');  %  plot signal with a blue solid line
```

b. What is the DC level of the signal y? Why is this important?

c. Next we will create the reproducer values based on the example in the notes slide 12, We have to add two extra values at the start and end, for the overload region:

```
rval = 0:1:15;  % create reproducer values
repV = [rval(1) , rval, rval(16)]; % repeat first & last values for overload
```

d. We now need to create the decision boundaries – following the colon example given in the notes slide 12, create a vector *decB* that contains all 17 decision boundaries from -0.5V to 15.5V, in steps of 1V.

e. Now we can quantise the signal using the following code:

```
[I,R] = quantiz(y,decB,repV);
```

f. We can now plot our signal samples and data together using the following code:

```
plot(t,y,'b-',t,R,'r*');
```

g. Note in your lab book your observations about this plot. Try subtracting 1 from the vector *I* created at step e. What are these values?

Exercise 2 (Example for STM32 Board) In Exercise 1, the reproducer values were the same as the reproducer indexes. In this exercise changes the reproducer values to be in the range 0.0-3.3V, which matches the range of the analogue-to-digital converter available on the STM32 Nucleo board.

a. Repeat the steps above to design a 4-bit quantiser, but this time the minimum decision boundary value is 0V and the maximum 3.3V. **Hint -** The minimum reproducer value now will be 3.3V/32 = 0.103125 and the step between adjacent reproducer values will be 3.3V/16 = 0.20625.

b. Now create a sine wave similar to Exercise 1a, but this time with a minimum value of 0V and a maximum value of 3.3V. What DC level is needed this time?

Exercise 3 (Sampling of Audio Signals) In this exercise we will modify our MATLAB code to sample audio signals with different numbers of bits per audio sample.

a. Download the following 8 KHz sample rate WAV sound file:
http://www.signalogic.com/melp/EngSamples/Orig/male.wav
This is a high quality 16 bit recording that you can try to sample with lower bit rates to see the effect of quantising it. If using the online MATLAB version, click on the "Home" tab at the top left to see an "Upload" option for this file.

b. The code steps listed at the end of this sheet should enable you to load the file, quantise it at 8 bit resolution and then create a second WAV file to download and listen to on your computer, e.g. using Windows Media Player.

c. Try different bit rates in the range from 2 bits to 12 bits. Can you hear the "quantisation noise" present in the signal? What causes this noise effect? How many bits of resolution are needed to get a good quality recording?

Optional Exercise 4 (Sine Waves and Square Waves)

a. Using MATLAB's sin and square functions, create waveforms with amplitudes in the range -1 to +1 and then sample them following the steps you used in Exercise 3 and the Appendix. You will need to create longer time vector than Exercise 1a, e.g. make t count from 1 to 16000 for 2 seconds of samples at 8 kHz. How do these waveforms sound? Try changing the frequency of the signal to see how this changes the sound.

Appendix: MATLAB Code for Exercise 3b

We can design the quantiser for this question using the MATLAB commands as follows:

```
[Data,Fs] = audioread('male.wav');
Bits = 8; % Number of Bits in Quantiser
Levels = 2^Bits; % Number of Reproducer Values
Delta = 2/Levels; % Spacing between Reproducer Values
DecBnd = -1:Delta:1; % These are decision boundaries between quantizer intervals
C = (-1+(Delta/2)): Delta:1-(Delta/2); % These are the reproducer values
RepVal = [ C(1), C, C(Levels) ]; % Two extra values needed to cover the overload region
[I1,Q1] = quantiz(Data,DecBnd,RepVal); % Quantize the data – Q1 contains the samples
audiowrite('ENG_M_USQ.wav',Q1,Fs); % Save the results to a file to play it
```