Matlab/Octave Assignment 2
Signal Processing with FIR Filters

In this assignment will design FIR filters to process audio signals. This assignment requires answering questions and writing Matlab/Octave scripts and functions. You will summary of the results of this worlk in a "report". If you work in Matlab the report should be generated directly, from a Matlab script file using Matlab's `publish` function. Learning to use `publish` is part of the project. For those of you familiar with MS Word it is probably set to publish to a MS Word document. This will allow you to edit this document and then print a final version to **a single PDF file** for final submission and grading. Alternatively, if you use Octave, the report should be generated directly from Jupyter Notebook as a PDF file.

Note: To complete this assignment you need the *signal processing toolbox* which is by default available in Matlab or equivalently the *signal* package in Octave. Type `ver` to see the toolboxes (or packages) available. Octave loaded packages have a * in the listing, if they are listed but do not have * just load them, *e.g.*, type `pkg load signal`)

1 FIR Filters (50 points)

Consider two FIR filters with impulse responses $h_1[n]$ and $h_2[n]$ given below

```
h1 = {0.0030, 0.0050, 0.0067, 0.0000,-0.0252,-0.0721,-0.1306,-0.1801, 0.7979,
     -0.1801,-0.1306,-0.0721,-0.0252, 0.0000, 0.0067, 0.0050, 0.0030}.
h2 = {0.0030,-0.0050, 0.0067, 0.0000,-0.0252, 0.0721,-0.1306, 0.1801, 0.7979,
      0.1801,-0.1306, 0.0721,-0.0252, 0.0000, 0.0067,-0.0050, 0.0030},
```

(a) For each of the above filters
   1) Plot the impulse response as a "stem" plot (using `stem`). Lablel your plot clearly.
   2) Plot the frequency response (using `freqz`). Lablel your plot clearly.
   3) From the frequency response determine
      i. The type of filter, *e.g.*, low-pass, high-pass, band-pass, etc.
      ii. Whether of not the filter has linear phase. Jusfity your answer.

   Summarize your results tabular form.

Filter	Filter Type	Linear Phase
`h1`		
`h2`		
`h`		

(b) Connect the two filters h_1, h_2 in cascade (*e.g.*, in series) to form a new equivalent filter h
   1) From the impulse response of h_1 and h_2, calculate the impulse response of the resulting combined filter (using `conv`) and plot the resulting impulse response of the cascade (using `stem`).
   2) Find the frequency response of the cascaded filter system using the **impulse response** $h[n]$ of the combined system found in part (1). Then plot it (using `freqz`).
   3) Determine if the resulting filter has linear phase or not. Explain why.
   4) Now instead of using convolution to combine the filters, use an equivalent frequency domain approach to find the **frequency response** of the resulting cascaded filte. That is, find $H(e^{j\hat{\omega}})$ directly from $H_1(e^{j\hat{\omega}})$ and $H_2(e^{j\hat{\omega}})$, working entirely in frequency domain (use the convolution-multiplication property,

$$h_1[n] * h_2[n] \xleftrightarrow{DTFT} H_1(e^{j\hat{\omega}})H_2(e^{j\hat{\omega}})$$

   5) Compare the two frequency responses, in part (2) and part (4) by plotting them in the same figure with different line types and color. Are they the same ? Explain and justify what you observe.

2 FIR Filter Design (50 points)

Here you use the function `fir1` to design FIR filters. The first step is to find out what this function does and how it is used. For that type `doc fir1` at Matlab's prompt, to bring up the documentation and examples (alternatively use `help fir1`).

(a) **Generate test signals.** To test your filter you will use four three pure sinsusoidal signals signals of duration 10 sec. and frequencies $f_1 = 100$, $f_2 = 5000$ and $f_3 = 1000$ Hz sampled at $f_s = 20$ kHz and the combination of the three. These sampled signals will be called $x_1[n], x_2[n], x_3[n]$ and $x[n]$, respectively, and can be easily generated in Matlab/Octave using the code below.

```
% Sample Matlab/Octave code to generate the test signals
% Signal parameters
    dur = 10;        % duration [sec]
    fs = 20000;      % sampling frequency [Hz]
    t = 0:1/fs:dur;  % sampled time axis: n T_s (or n/f_s)
% generate sampled signal
    x1 = sin(2*pi*100*t);
    x2 = sin(2*pi*5000*t);
    x3 = sin(2*pi*1000*t);
     x = x1 + x2 + x3;
```

(b) **Plot Signals Spectrum.** We will soon study how the FFT computes numerical values of (samples of) the spectrum of sampled signals. Here you will do that using Matlab/Octave.
First, define a ***frequency vector***, `f`, with equidistant frequencies in the main frequency band of your filters, that is $\left[-\frac{f_s}{2}, \frac{f_s}{2}\right]$ Hz, using `f=linspace(-fs/2,fs/2,N)` (type `help linspace` for more details), where `N` is the number of points used in the FFT calculations (here you will use `N=2048`). Then plot, in the same figure (using `subplot`) the (approximate) ***magnitude spectrum*** of the test signals $s_1(t), s_2(t), s_3(t)$ and $s(t)$ as a function of the independen variable f in Hz (you will need 4 subplots). The correct spectral lines are obtained by **dividing the FFT coefficients by N**, the number of points used to compute the FFT. Type `doc fft` for details about this function.
Note: for the spectrum plot to be properly "centered" you need to use `fftshift`, find out what it does typing `help fftshift` or look at the online documentation at https://www.mathworks.com/help/matlab/ref/fftshift.html).
Discuss the main features of the spectrum found using the FFT and contrast these feature with what you expect from an exact analysis of the spectrum.

(c) **Low-Pass FIR Filter.** Using the Matlab function `fir1` design a **lowpass FIR** filter of order 40 and cutoff frequency $f_c = 500$ Hz. Note that you will need to convert the continuous-time frequency $f_c = 500$ Hz to its corresponding digital frequency, $\widehat{f}_c = f_c T_s = f_c / f_s$ since you need to use this frequency in `fir1`.

1) Find the frequency response of the filter and plot it along the spectrum of the signal calculated in part (a). Use `plotyy` to generate this plot. For example, if `Hb` is the frequency response of the filter, `Xf` the spectrum of the input signal and `f` the frequency grid used to compute `Hb` and `Xf`, the desired plot can be created with the following command:

```
plotyy(f,abs(Hb),f,abs(Xf));
```

The function `plotyy` *is useful to plot, in the same figure, two functions of the same variable but vastly different dependent variables,* i.e.*, vertical scales (type* `help plotyy` *for more details.)*

2) Use the function `filter` to process **the signal $s[n]$** with the filter you designed. Calculate and plot the magnitude spectrum of the output signal. Explain the result.
3) Use the function `sound`, to play in your computer's speakers the signals $s[n]$ (input to your filter), $s_1[n]$ (a reference signal) and the signal at the output of your filter. Describe briefly what you hear, and explain the result based on your analysis of the filter above.

(d) **High-Pass FIR Filter.** Use `fir1` to design a **highpass FIR** filter of order 40 and cutoff frequency $f_c = 3$ kHz. Repeat parts (1)-(3), using **the signal $s_3[n]$** as input signal.

(e) **Band-Pass FIR Filter.** Using `fir1` design a **bandpass FIR** filter of order 80 with a pass band (the range of frequencies that are not filtered out) between 300 Hz and 3 kHz. Repeat parts (1)-(3), using **the signal $s[n]$** as input signal.

Optional: You can listen to the signals before and after filtering to assess the result of the filter. If your working setup does not allow you to play the signals through the speakers, or if you cannot hear them well enough, save them to a `wav` *file so that you can play them elsewere.*