Use Python and Matlab for all questions below (unless some derivation is required which you can do better by hand). If using Python, use Jupyter notebook to format properly, comment when necessary, and include all required and necessary outputs (Jupyter allows LATEX embedding which would be great if you could use it). Convert the notebook to pdf and upload to blackboard against the assignment. If using MATLAB, use MATLAB publish to convert to pdf. Solution documents should be well commented.

Q1. Consider the sawtooth function

$$f(x) = \begin{matrix} x & 0 < x < 1 \\ 0 & 1 < x < 2 \end{matrix}$$

(a) Plot the sawtooth function for x between 0 and 2 like I plotted the square function in class.
(b) Approximate the sawtooth function with Fourier series. Plot the approximations for k=1,3,5,10,25,100

You may need the following integrals:

$$\int_0^1 x\sin(\pi kx)dx = \frac{(-1)^{k+1}}{k\pi}$$
$$\int_0^1 x\cos(\pi kx)dx = \frac{(-1)^k - 1}{k^2\pi^2}$$

Q2. Load the file dataF.mat in your workspace. This file has variables named s and sr which you will need to load. The data has been sampled at $f_s = 2000Hz$.

(a) Create an appropriate time vector (t) for s based on the information above. Plot s versus that time vector. You will see that this is a sinusoidal variation with 5 complete cycles. From this plot, calculate the frequency of this sinusoid. You can do this by determining the time T taken for 1 complete cycle of the sinusoid. The frequency is given by 1/T.
(b) The plot in (a) shows the functional variation of s with time variable 't'. So we have

$$s = f(t)$$

The function $f(t)$ can be approximated by the complex discrete Fourier series (as covered in class):

$$f(t_j) = \sum_{k=-N}^{N} C_k e^{i2\pi f_k t_j}$$

where $f_k = f_s k/n$, n is the total number of data-points in s, and the summation is carried over all the time data-points t_j. The complex coefficients C_k are given by

$$C_k = \frac{1}{n}\sum_{t_j} s(t_j)e^{-i2\pi f_k t_j}$$

Calculate $f_0, f_1, f_2, f_3, f_4, f_5, f_6$. These are the first 7 frequencies (in HZ) which will be present in the Fourier transform of s.
(c) Calculate the corresponding $C_0, C_1, C_2, C_3, C_4, C_5, C_6$ coefficients from the sum equation given in part b. To calculate these, note the following:
(a) The imaginary number i is given in Matlab with the symbol 1i and in python with the symbol 1j

(b) To calculate some C_k, first calculate f_k, then create a vector corresponding to the exponential term. Elementwise multiply the exponential vector with s and then sum all the elements (you can use sum function in matlab or numpy.sum in python).

(d) Calculate the corresponding power spectrum (energy) $P_0, P_1, P_2, P_3, P_4, P_5, P_6$ where $P_k = |C_k|^2/n$.

(e) Out of the frequencies $f_0, f_1, f_2, f_3, f_4, f_5, f_6$, which frequency has the maximum energy? The answer to this is also the frequency of the sinusoid signal in s and should match your answer in part a.

(f) Use the fft command to compute the Fourier coefficients now. While calculating these coefficients, MATLAB/Python does not multiply them with $1/n$ term as shown in part b. So multiply all of them with $1/n$. Now verify that the first 7 coefficients that you get from the fft command (after multiplication with $1/n$) are equal or close to the values $C_0, C_1, C_2, C_3, C_4, C_5, C_6$ you calculated in part c.

(g) Plot the absolute values of the Fourier coefficients (as determined from fft command) against an appropriate frequency vector. The plot should show the peak at the right value of the dominant frequency in f.

Q3. The data in sr has been sampled at 2000Hz. I have created the sr by mixing 4 different sines with random noise.

(a) Create an appropriate time vector for sr based on the information above. Plot sr versus that time vector.

(b) Plot the absolute values of the Fourier coefficients (as determined from fft command) against an appropriate frequency vector. The plot should show 4 distinct peaks.

(c) From the locations of these peaks, Determine the frequency (in HZ) of the sines mixed in the signal.

Q4. Read the file wilhelm.wav in python/matlab. This is an audio file for the famous Wilhelm scream:
https://en.wikipedia.org/wiki/Wilhelm_scream

which has been used in numerous Hollywood movies. In this question you will figure out the frequency content of this audio file.

To read the file In Python you can use

```
from scipy.io import wavfile
samplerate, data = wavfile.read('wilhelm.wav')
```

In MATLAB you can use the audioread function:

https://www.mathworks.com/help/matlab/ref/audioread.html

When you read the file as a matrix (stored in the variable *data* in the python example), the matrix will have two columns, one corresponding to each ear (left and right). For the purpose of this question it is sufficient to work with any one of the two columns. Let’s store one of the columns (any of your choosing) in the vector s. In python you can do it with s=data[:,0] and in matlab you can do this with s=data[:,1]. The audio file has a sampling frequency (also called sampling rate) of 44100 Hz.

(a) Plot s over the appropriate time vector

(b) Plot the power spectral density of s over the appropriate frequency vector. You can use any function you like to do this calculation. Identify the prominent frequencies in the Wilhem scream from this plot.