Practical 3: Hand in your solution to MyUni before Fri 23rd March at 1pm.

You should read the practical, and prepare before the actual session.

1. Analyse a set of signals using the Fourier transform.

   - Download the sound (FLAC) and image files (PBM) from http://www.maths.adelaide.edu.au/matthew.roughan/notes/AMP1/11other.html Have a look/listen to them.
   - Visualise the frequencies of the sinusoids present in the 1D sound files. A simple example is provided below, using MATLAB commands. If you don't understand a particular command, you can use MATLAB's `help` to find out more.

     ```
     [y,Fs] = audioread('cos_110.flac'); % y=signal, Fs=sampling rate

     figure(1)
     plot( y(1:10000) ); % plot the start of the sound

     z = fft( y );       % Fourier transform of signal (complex)
     z2 = abs( z.^2 );   % squared magnitude of the Fourier transform

     figure(2)
     semilogy( z2 );    % plot log of the squared magnitude
     ```

     The example shows how to analyse the simple cos wave. Now use this on the guitar pluck, and replicate the figure shown in your notes.

     Note that the figure is symmetric, so you only need to plot half of `z2`, or you can use the `fftshift` function to put the 0 frequency component in the middle.
   - Perform the same analysis on the other 1D sound signals to replicate the figure in your notes for the "guitar pluck" sounds.
   - Perform a similar analysis on the image file. Note that you will need commands like

     ```
     filename = 'pegasus.pbm';
     A = imread(filename); % reads the image from a file
     A2 = double(A);       % convert the image into an array of doubles

     figure(1)
     image(A2, 'CDataMapping', 'scaled');
     axis image
     colormap(gray)       % gray-scale image

     B = fft2(A2);         % two-d Fourier transform
     C = log(abs(B.^2));
     figure(2)
     image(C, 'CDataMapping', 'scaled');
     axis image
     colormap(gray)
     ```

   - Download a simple, repetitive image from the Internet, and perform the same analysis.

2. Generate a 2D pattern by combining a small set of sinusoids together into an image. The easiest way to start is to generate a set of x and y coordinates as follows:

```
M = 10;
x = (1:M)/M;
[X, Y] = meshgrid(x,x);
```

This is just an example – have a look at the resulting X and Y. You will want a larger value of M (the grid size) in your work.

Now you can create code in the form below to create and plot a 2D sinusoid.

```
f_0 = 3;
A = sin(2*pi*f_0*X);
figure(11)
image(A, 'CDataMapping', 'scaled');
colormap(gray)
```

Play around with different frequencies and directions, and then combine several such into an interesting pattern.

Put plots from Part 1 (the Fourier transform of the guitar, and your image and its Fourier transform) and Part 2 (your pattern) into an Overleaf document, output the PDF, and hand in via MyUni by the due date.

(Optional extension exercise) Create a 1D Fourier transform, k

and invert it using `ifft` to obtain

- a single sin wave;
- a pair of cosines added together; and
- a square wave.

Plot your results.

Hints and tips:

- The image we read in was a PBM file. It has a grey-scale image, with 1 bit per pixel (either the pixel is black or white). When you read in a more typical colour image (*e.g.,* a JPG), the data will have three numbers per pixel. Typically these give you the RGB (red-green-blue) values, though there are other colour representations. This will mean that `A = imread(filename)` results in a $n \times m \times 3$ array of numbers (3 numbers for each pixel). To get an approximate *intensity* grey-scale image from these, we can just take the mean of the three values, *i.e.,*

  ```
  Adash = mean( A, 3 )
  ```

  The second argument (the 3) says to take the mean over the third-dimension of the array. The output, `Adash`, will be an $n \times m$ array.

  If you want to do something more clever have a look at https://stackoverflow.com/questions/687261/converting-rgb-to-grayscale-intensity or https://stackoverflow.com/questions/17615963/standard-rgb-to-grayscale-conversion.

- When you plot a FFT, often the most interesting values are towards the left. You can use MATLAB's interactive zooming to explore this, or use `xlim` commands in the command window.

- The guitar signal is stereo, so when you load it, you will get a $n \times 2$ array, with the left and right channels. Either use one half, or plot both.