Image processing and data extraction

In this project, we will investigate the brain activity from a model with a specific gene removed. As all the data needed for analysis are from video files, we will first extract relevant data from the video, and then we will investigate the characteristics of brain activity. This project forms a part of my PhD thesis, in which the complete content can be found in the University of Auckland Doctoral Thesis repository.

For my PhD, I video recorded brain activity from a model where a specific gene important for brain cells communicating with each other was removed. The purpose of this scientific research was to gain insight into the role of the gene in brain functions. In part 1 of this project, we will extract relevant data from a video for data analysis.

We all know "a picture is worth a thousand words". In a scientific context, a picture is worth a thousand numbers. This is because all digital images are a matrix (a grid of numbers arranged in rows and columns), where each number represents how bright or dark a particular area of the image is. This number is known as a pixel. In a video file, a series of images (frames) is shown quickly to give the appearance of motion. This means the size of the data we are working with can be represented as:

$$size\ of\ data = number\ of\ rows * number\ of\ columns * number\ of\ frames$$

Combining that, it is clear how much data we have from a video and how important we extract the data that is relevant to our question.

All processing and analyses were done on MATLAB. Unlike Python, there is no need to import libraries because the core Python libraries such as *numpy*, *pandas* and *matplotlib* are built into the MATLAB environment. However, since we do need specialised functions, we need to install the *Image Processing and Computer Vision* toolbox.

Image processing and data extraction using MATLAB:

We will begin by importing the experiment information, which was kept as an Excel file into MATLAB. This provides information about whether an experiment has video files for analysis. We will also set parameters relevant to the video file for later.

[1] *Note: MATLAB uses % to comment*

```
>> % Load experimental information
>> T_raw = readtable(['Experiment/Experiment_Information.xlsx']);
>> T_raw
```

```
T_raw =

  29×14 table
```

ID	Genotype	Inclusion	file_1	file_2	file_3	file_4	file_5	file_6	file_7	file_8
{'AC01'}	{'KO'}	0	0	0	NaN	NaN	NaN	NaN	NaN	NaN
{'AC06'}	{'WT'}	0	0	0	NaN	NaN	NaN	NaN	NaN	NaN
{'AC12'}	{'WT'}	1	1	1	0	0	0	1	NaN	NaN
{'AC15'}	{'KO'}	1	1	1	0	0	NaN	NaN	NaN	NaN
{'AC17'}	{'WT'}	1	1	0	1	0	1	NaN	NaN	NaN
{'AC22'}	{'WT'}	1	0	0	0	1	1	0	0	1
{'AC23'}	{'WT'}	1	1	1	1	1	NaN	NaN	NaN	NaN
{'AC26'}	{'KO'}	0	0	0	0	NaN	NaN	NaN	NaN	NaN
{'AC29'}	{'KO'}	1	1	1	1	NaN	NaN	NaN	NaN	NaN
:	:	:	:	:	:	:	:	:	:	:
{'AC48'}	{'WT'}	1	1	0	NaN	NaN	NaN	NaN	NaN	NaN
{'AC49'}	{'WT'}	1	1	1	1	0	NaN	NaN	NaN	NaN
{'AC53'}	{'KO'}	1	1	1	1	NaN	NaN	NaN	NaN	NaN
{'AC57'}	{'WT'}	1	0	1	0	NaN	NaN	NaN	NaN	NaN
{'AC58'}	{'WT'}	1	1	1	1	1	NaN	NaN	NaN	NaN
{'AC59'}	{'KO'}	1	1	1	1	1	0	NaN	NaN	NaN
{'AC61'}	{'KO'}	1	1	1	0	0	1	1	1	1
{'AC62'}	{'KO'}	1	0	1	0	0	1	NaN	NaN	NaN
{'AC63'}	{'WT'}	1	0	0	1	1	NaN	NaN	NaN	NaN

[2]

```
>> % Remove all the experiments with no data from the list
>> No_data_row = T_raw.Inclusion == 0;
>> T2 = T_raw;
>> T2(No_data_row,:) = [];
>> T2
```

T2 =

24×14 table

ID	Genotype	Inclusion	file_1	file_2	file_3	file_4	file_5	file_6	file_7	file_8
{'AC12'}	{'WT'}	1	1	1	0	0	0	1	NaN	NaN
{'AC15'}	{'KO'}	1	1	1	0	0	NaN	NaN	NaN	NaN
{'AC17'}	{'WT'}	1	1	0	1	0	1	NaN	NaN	NaN
{'AC22'}	{'WT'}	1	0	0	0	1	1	0	0	1
{'AC23'}	{'WT'}	1	1	1	1	1	NaN	NaN	NaN	NaN
{'AC29'}	{'KO'}	1	1	1	1	NaN	NaN	NaN	NaN	NaN
{'AC30'}	{'WT'}	1	1	0	1	0	0	1	0	0
{'AC31'}	{'KO'}	1	1	0	0	0	0	0	0	0
:	:	:	:	:	:	:	:	:	:	:
{'AC49'}	{'WT'}	1	1	1	1	0	NaN	NaN	NaN	NaN
{'AC53'}	{'KO'}	1	1	1	1	NaN	NaN	NaN	NaN	NaN
{'AC57'}	{'WT'}	1	0	1	0	NaN	NaN	NaN	NaN	NaN
{'AC58'}	{'WT'}	1	1	1	1	1	NaN	NaN	NaN	NaN
{'AC59'}	{'KO'}	1	1	1	1	1	0	NaN	NaN	NaN
{'AC61'}	{'KO'}	1	1	1	0	0	1	1	1	1
{'AC62'}	{'KO'}	1	0	1	0	0	1	NaN	NaN	NaN
{'AC63'}	{'WT'}	1	0	0	1	1	NaN	NaN	NaN	NaN

Display all 24 rows.

[3]

```
>> % Set the frame rate to be the same as the camera
>> fps = 5;
>> scale = [(1/fps):(1/fps):8*60];
>>
>> % Make duplicates of the experimental information to hold extracted data later
>> T_frequency = T2;
>> T_frequency(:,'Inclusion')=[];
>> T_amplitude = T_frequency;
>> T_duration = T_frequency;
>>
>> % See how many video files are in each experiment
>> varnames = T2.Properties.VariableNames;
>> varnames(:,[1:2])= [];
```

Since we will be using the same functions for each of the experiments[1], we will put all the functions in a *for* loop. In Python, some loops should be written as a function to improve efficiency and readability. In MATLAB, however, *for* loops are the easier approach. I will break the loop into sections and explain each process.

[1] In a scientific study, we repeat experiments for the reason of 1. To reduce variability and improve accuracy and 2. Not all experiments will result in usable data.

The first loop is to go through all experiments that have video data. The second loop is to go through all video files in each experiment. These two loops ensure we are going through every video file from all experiments. When we have the list of all experiments and video files to go through, we will load each video file from one experiment into MATLAB.

[4]

```
>> % First loop: To go through all experiments that had video files
>> f1 = figure(1)
>> for ii = 1:size(T2,1)
     % List all video files from each experiment data
     ID = char(T2{ii,'ID'})
     files = dir(['Experiments/',ID,'/*.tif']);

     % Second loop: to work with the individual video file
     for iFiles = 1:size(files)
          filename = files(iFiles).name

          % Load video file into MATLAB
          I = tiffreadVolume(['Experiment/',ID,'/',filename]);
```

In the video, not everything in the picture is relevant to answering our questions in our project. Therefore, we must take out all the irrelevant objects and space in the picture so that only the relevant objects are left. To do this, we isolate the relevant object/space by "masking" (as in hiding) the irrelevant object/space (Figure 1).

[5]

```
          % Extracting relevant data: we are interested in a particular area
          % of the brain, so we will isolate the region of interest

          % To isolate the region of interest, we need to see the image
          figure(1)
          clf
          imshow(stack(:,:,1), [], 'InitialMagnification', 'fit');

          % Locate the region on the image to isolate and create a mask
          roi = drawellipse('Center',[128,128], 'SemiAxes', [60, 40]);

          roimask = createMask(roi);
          maskroi = imdilate(roimask,strel('disk',3));

          % Isolate the same object across all the frames in the video file
          ROIstack = NaN(size(stack));
          for ss = 1:size(stack,3)
              frame = stack(:,:,ss);
              im = ROIstack(:,:,ss);
              im(maskroi==1) = frame(maskroi);
              ROIstack(:,:,ss) = im;
          end
```

It is important to note that once video recording is loaded into MATLAB, we are no longer working with images (something we can see) and instead we are working with a matrix (a table of numbers arranged in rows and columns).

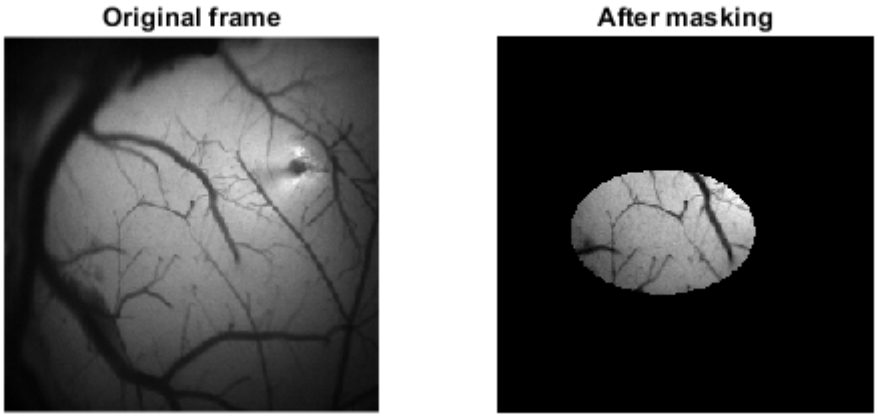


Figure 1: Before and after isolating the relevant area in the frame

The relevant information is the brain activity over time coming from the region we isolate. It is important to remember that the region we isolated is comprised of many pixels (numbers). Since all these pixels represent a signal from the same object in the image, plotting the data from each pixel would be messy, redundant and importantly, cannot be interpreted. Therefore, we average all the pixels in the object so that we can get the overall value of the signal coming from that object.

[6]

```
% Average all the pixels from the same object to get an overall value.
% This value represents the brain activity from the object and will
% change over time.

 trace = [];
 for tt = 1:size(ROIstack,3)
     trace(tt,1) = mean(ROIstack(:,:,tt),'all','omitnan');
 end
```

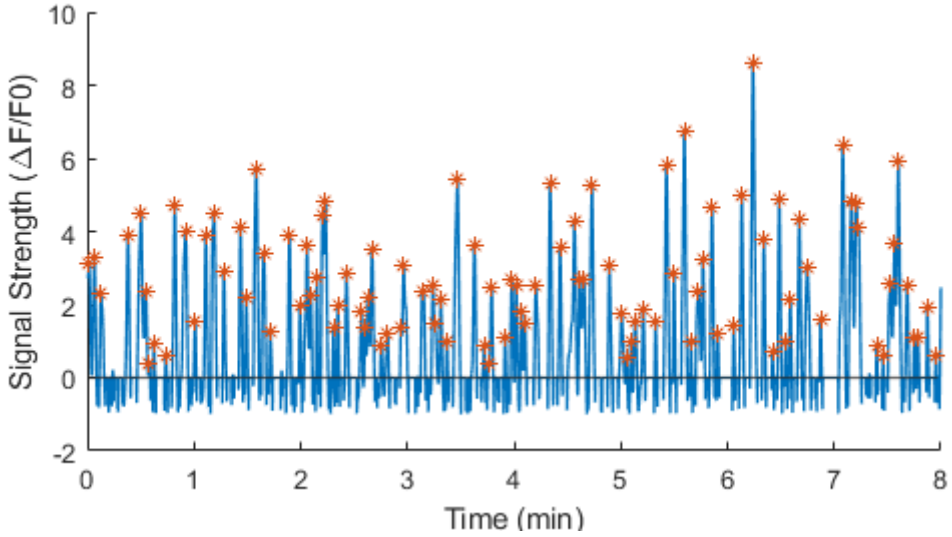
From this time series, we can determine the frequency of brain activity (represented as a peak), the amplitude (the height of the peak) and the duration (the width of a peak). Small peaks cannot be counted because they can arise from noise. Therefore, we put in a threshold condition where peaks above a threshold are counted.

[7]

```
% From the time series, we can now find the total number of
% peaks, the amplitude and the duration of the signal.

threshold = std(trace,1,'omitnan')/2; % threshold
[pk loc w] = findpeaks(trace,'MinPeakHeight',threshold);

figure
set(gcf,'color','w');
plot((scale/60),trace(1:8*60*fps),'LineWidth',1)
hold on
scatter ((loc/fps/ 60),pk, '*');
ylabel('Signal Strength (\DeltaF/F0)')
xlabel('Time (min)')
box off
yline(0,'LineWidth',1)
```



To prepare for analysis, I will now find how frequent a peak was detected in a video file, the average intensity of the signal and the average duration of the signal. These data will be held in the duplicated experimental information we made earlier, before the loops.

[8]

```
        nFreq = ((size(pk,1))/2400)*fps; % peaks detected per second
        avAm = mean(pk);
        avDur = mean(w/fps);

        % Find the column in the duplicated experiment information to replace
        col = varnames{1,iFiles};
        T_frequency{ii,col} = nFreq;
        T_amplitude{ii,col} = avAm;
        T_duration{ii,col} = avDur;
    end
end
>>
>> T_frequency
```

T_frequency =

24×13 table

ID	Genotype	file_1	file_2	file_3	file_4	file_5	file_6	file_7
{'AC12'}	{'WT'}	0.19375	0.24375	0.23333	0.25417	0.23542	0.22917	NaN
{'AC15'}	{'KO'}	0.24792	0.22292	0.24375	0.23125	NaN	NaN	NaN
{'AC17'}	{'WT'}	0.2875	0.3	0.29583	0.30833	0.31667	NaN	NaN
{'AC22'}	{'WT'}	0.09375	0.11667	0.16042	0.21458	0.21875	0.225	0.26458
{'AC23'}	{'WT'}	0.13125	0.15208	0.20208	0.25417	NaN	NaN	NaN
{'AC29'}	{'KO'}	0.19792	0.1875	0.17292	NaN	NaN	NaN	NaN
{'AC30'}	{'WT'}	0.21875	0.21042	0.25	0.23958	0.19792	0.21458	0.14792
{'AC31'}	{'KO'}	0.10417	0.083333	0.1375	0.16667	0.16875	0.14167	0.20417
{'AC32'}	{'KO'}	0.13958	0.13333	0.16875	0.21875	0.20833	NaN	NaN
{'AC35'}	{'WT'}	0.11458	0.091667	0.10208	0.091667	NaN	NaN	NaN
{'AC38'}	{'KO'}	0.22708	0.18958	0.17708	0.17917	0.20208	NaN	NaN
{'AC41'}	{'KO'}	0.072917	0.1625	0.16458	0.17708	0.18333	NaN	NaN
{'AC42'}	{'KO'}	0.22292	0.1875	0.22917	0.17917	0.19583	NaN	NaN
{'AC43'}	{'WT'}	0.12292	0.21458	0.20833	0.17708	NaN	NaN	NaN
{'AC44'}	{'WT'}	0.21875	0.2	0.18958	0.175	0.17917	NaN	NaN
{'AC48'}	{'WT'}	0.27292	0.29375	NaN	NaN	NaN	NaN	NaN
{'AC49'}	{'WT'}	0.31042	0.27917	0.34375	0.34167	NaN	NaN	NaN
{'AC53'}	{'KO'}	0.29583	0.3375	0.30208	NaN	NaN	NaN	NaN
{'AC57'}	{'WT'}	0.32917	0.34375	0.34583	NaN	NaN	NaN	NaN
{'AC58'}	{'WT'}	0.25833	0.26875	0.30417	0.30625	NaN	NaN	NaN
{'AC59'}	{'KO'}	0.25833	0.26875	0.28333	0.30417	0.2875	NaN	NaN
{'AC61'}	{'KO'}	0.11875	0.24375	0.24375	0.24583	0.24792	0.36667	0.36875
{'AC62'}	{'KO'}	0.19583	0.27917	0.27292	0.3	0.30833	NaN	NaN
{'AC63'}	{'WT'}	0.25	0.23333	0.25625	0.21458	NaN	NaN	NaN

Now, we will store all the information in one table for analysis and save this table as a CSV file. We will use this dataset for statistical analysis in part 2.

[9]

```
>> temp_table = [];
>> for rr = 1:size(T_frequency,1)
       ID = {char(T_frequency{rr,'ID'})};
       Genotype = {char(T_frequency{rr,'Genotype'})};

       for cc = 1:size(varnames,2)
           temp_freq = {(T_frequency{rr,(varnames{1,cc})})};
           temp_amplitude = {(T_amplitude{rr,(varnames{1,cc})})};
           temp_duration = {(T_duration{rr,(varnames{1,cc})})};

           temp2(:,1) = ID;
           temp2(:,2) = Genotype;
           temp2(:,3) = temp_freq;
           temp2(:,4) = temp_amplitude;
           temp2(:,5) = temp_duration;

           temp_table  = cat(1, temp_table , temp2);
       end
   end
>>
>> % Save the data set table as a CSV file.
>> signal_characteristics = cell2table (temp_table);
>> signal_characteristics.Properties.VariableNames = {'ID',...
    'Genotype',' Frequency', 'Amplitude', 'Duration'};
>>
>> writetable(signal_characteristics,['Experiment/SignalCharacteristics.csv'])
>>
>> signal_characteristics
```

signal_characteristics =

264×5 table

ID	Genotype	Frequency	Amplitude	Duration
{'AC12'}	{'WT'}	0.19375	9.9501	1.9345
{'AC12'}	{'WT'}	0.24375	6.5289	1.6714
{'AC12'}	{'WT'}	0.23333	6.8631	1.7027
{'AC12'}	{'WT'}	0.25417	6.4024	1.6696
{'AC12'}	{'WT'}	0.23542	6.4834	1.6124
{'AC12'}	{'WT'}	0.22917	7.2542	1.7614
{'AC12'}	{'WT'}	NaN	NaN	NaN
{'AC12'}	{'WT'}	NaN	NaN	NaN
{'AC12'}	{'WT'}	NaN	NaN	NaN
{'AC12'}	{'WT'}	NaN	NaN	NaN
{'AC12'}	{'WT'}	NaN	NaN	NaN
{'AC15'}	{'KO'}	0.24792	1.7429	1.3862
{'AC15'}	{'KO'}	0.22292	1.6338	1.8623
{'AC15'}	{'KO'}	0.24375	1.1604	1.5373
:	:	:	:	:
{'AC62'}	{'KO'}	NaN	NaN	NaN
{'AC62'}	{'KO'}	NaN	NaN	NaN
{'AC62'}	{'KO'}	NaN	NaN	NaN
{'AC63'}	{'WT'}	0.25	1.3891	1.1837
{'AC63'}	{'WT'}	0.23333	1.9103	1.3868
{'AC63'}	{'WT'}	0.25625	1.3527	1.2848
{'AC63'}	{'WT'}	0.21458	1.5679	1.3412
{'AC63'}	{'WT'}	NaN	NaN	NaN
{'AC63'}	{'WT'}	NaN	NaN	NaN
{'AC63'}	{'WT'}	NaN	NaN	NaN
{'AC63'}	{'WT'}	NaN	NaN	NaN
{'AC63'}	{'WT'}	NaN	NaN	NaN
{'AC63'}	{'WT'}	NaN	NaN	NaN
{'AC63'}	{'WT'}	NaN	NaN	NaN

Display all 264 rows.