ANALYSIS OF ELECTROPHYSIOLOGICAL DATA

A Design Project Report

Presented to the School of Electrical and Computer Engineering of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering, Electrical and Computer Engineering

Submitted by:

Wen Fan (wf85), Wentao Li (wl553), Yiqing Li (yl2558)

MEng Field Advisor: Prof. Bruce R. Land

MEng Outside Advisor: Prof. Christiane Linster

Degree Date: May, 2016

ABSTRACT

Master of Electrical Engineering Program
Cornell University
Design Project Report

Project Title: ANALYSIS OF ELECTROPHYSIOLOGICAL DATA

Author: Wen Fan (wf85@cornell.edu)

Wentao Li (wl553@cornell.edu)

Yiqing Li (yl2558@cornell.edu)

Abstract:

Data including respiration and neural activity of mice has been acquired in Spike2. The main task for the MEng project is processing the data with Python to extract relevant information such as number of spikes during each trial (a few seconds of recording), respiration phase and coefficient of variation for each cell. If infusion or stimulation was used, the data should be separated in before and after. The Python program is supposed to produce output, which carries information for each recording file. As a functional module, the program will be applied to electrophysiological research repeatedly.

OVERVIEW

In this project, we desire to analyze the data acquired in Spike2 (A multi-channel continuous data acquisition and analysis package which is applied in many fields including electrophysiology, neurophysiology, cardiovascular and respiratory studies, pharmacology). Spike2 data files can be read using NEO.IO (A platform provided with an exhaustive way of loading and saving several widely used data formats in electrophysiology) and then be processed in Python code. Relevant parameters needed to be analyzed are listed in the KEY ISSUES part. Finally, Python programs we designed will be encapsulated as different functional modules, and there is a friendly GUI also created by Python to give the users an easy access. This is a common and important approach to facilitate electrophysiological research. Our work will provide a powerful tool for later researchers to acquire the data they want.

KEY ISSUES

Below are the key parameters of data we need to analyze. For one cell/single data file, programs are designed to generate these parameters. If more than one cell/data file are analyzed, we generate multiple histogram output in addition to the individual information.

Rates1: Here we should analyze how many spikes each cell fired during each trial and we also want to know the frequency(number/time). If stimulus (such as infusion and odors) were used, the data needs to be sorted by stimulus identity.
Rates2: Same as 1 with the difference that each trial will be subdivided into shorter segments to be analyzed separately.
Rate3: Coefficient of variation: A statistical measure of the dispersion of spikes in a spike series around the mean.
Rate4: Interspike interval distribution: Parameter describing the interval distribution patterns among different spikes. It represents the regularity of spike distribution.
Rate5: Phase of respiration: Parameter reveals how action potentials are distributed along a

respiration cycle.
Rate6: Coherence and co-variation between spike trains and respiration.
Rate7: Pairwise correlations between spike trains recorded at the same time. (undone)

Schedule & Timeline

Semester 1:

• 1. Late Sep. --- early Oct.	Theoretical analysis and software learning
• 2. Mid Oct. --- late Nov.	Single cell data analysis and calculation
• 3. Before mid Dec.	Rate 1~4 Test & term report

Semester 2:

• 1. Mid Feb. --- late March.	Finish Single cells leftover & Multi cells analysis
• 2. Early April. --- before May.	GUI & algorithm improve
• 3. Before mid May.	Evaluation & final report

In the first semester, we generally focused on the processing of the single cell data. we finished tasks from rate 1 to rate 4(single cell), and also analyzed the structure and approach of rate 5 as well as data of multiple cells (basically rate 1 and 2).

In the second semester, we mainly focused on the multiple cell cases and finish rate 5-7. Besides, we built a user-friendly GUI, and improve the algorithm of some previous tasks.

Analysis & Result

Data format:

Using the data from “laura_example”, there are 39 AnnalogSignal channels in which 2 of them are Respiration, and the rest 37 channels are split parts of a single cell signal; three EventArray channels and 1 SpikeTrain channels. The channel.annotations can tell us the information about each channel, like physical_channel_index, channel_index, title etc. The following notes in *italic style* are the information of all 39 channels:

Note: channel.annotations:

```
SpikeTrain:
  {'comment': '', 'title': 'ob25', 'physical_channel_index': 25, 'channel_index': 1, 'ced_unit': 1}
AnalogSignal:
  {'comment': '', 'physical_channel_index': 25, 'title': 'ob25'}
  {'comment': 'No comment', 'physical_channel_index': 7, 'title': 'Respirat'}
EventArray:
  {'comment': 'No comment', 'physical_channel_index': 0, 'extra_labels': array(['Odor Mix
0.01Pa', 'Odor Mix 0.01Pa', 'Odor Mix 0.01Pa',
     'Odor Mix 0.01Pa', 'Odor Mix 0.01Pa', 'Odor Mix 0.01Pa',
     'Odor Mix 0.1Pa', 'Odor Mix 0.1Pa', 'Odor Mix 0.1Pa',
     'Odor Mix 0.1Pa', 'Odor Mix 0.1Pa', 'Odor Mix 0.1Pa',
     'Odor mix 1Pa', 'Odor mix 1Pa', 'Odor mix 1Pa', 'Odor mix 1Pa',
     'Odor mix 1Pa', 'Odor mix 1Pa', 'BLANK', 'Odor Mix 0.01Pa',
     'Odor Mix 0.01Pa', 'Odor Mix 0.01Pa', 'Odor Mix 0.01Pa',
     'Odor Mix 0.01Pa', 'Odor Mix 0.01Pa', 'Odor Mix 0.1Pa',
     'Odor Mix 0.1Pa', 'Odor Mix 0.1Pa', 'Odor Mix 0.1Pa',
     'Odor Mix 0.1Pa', 'Odor Mix 0.1Pa', 'Odor mix 1Pa', 'Odor mix 1Pa',
     'Odor mix 1Pa', 'Odor mix 1Pa', 'Odor mix 1Pa', 'Odor mix 1Pa'],
    dtype='|S200'), 'channel_index': 29, 'title': 'TextMark'}
  {'comment': 'No comment', 'physical_channel_index': 0, 'channel_index': 30, 'title': 'Keyboard'}
  {'comment': 'No comment', 'physical_channel_index': 0, 'channel_index': 31, 'title': 'DigMark'}
```

The physical channel index is the channel index in the real hardware, the channel index is the index assigned in the computer, they can be different. The annotations also show what odor we use and the amount of each odor.

The following pictures come from Spike2:

Figure 1 is the whole picture of “laura_example”, there are 3 signal channel in this picture. The top one in green is the respiratory *AnalogSignal* channel, and the bottom channel which is divided into 37 trials is the actual cell signal of *AnalogSignal* channel. The middle signal in blue is the *SpikeTrain* channel, which is the samples of the cell signal.

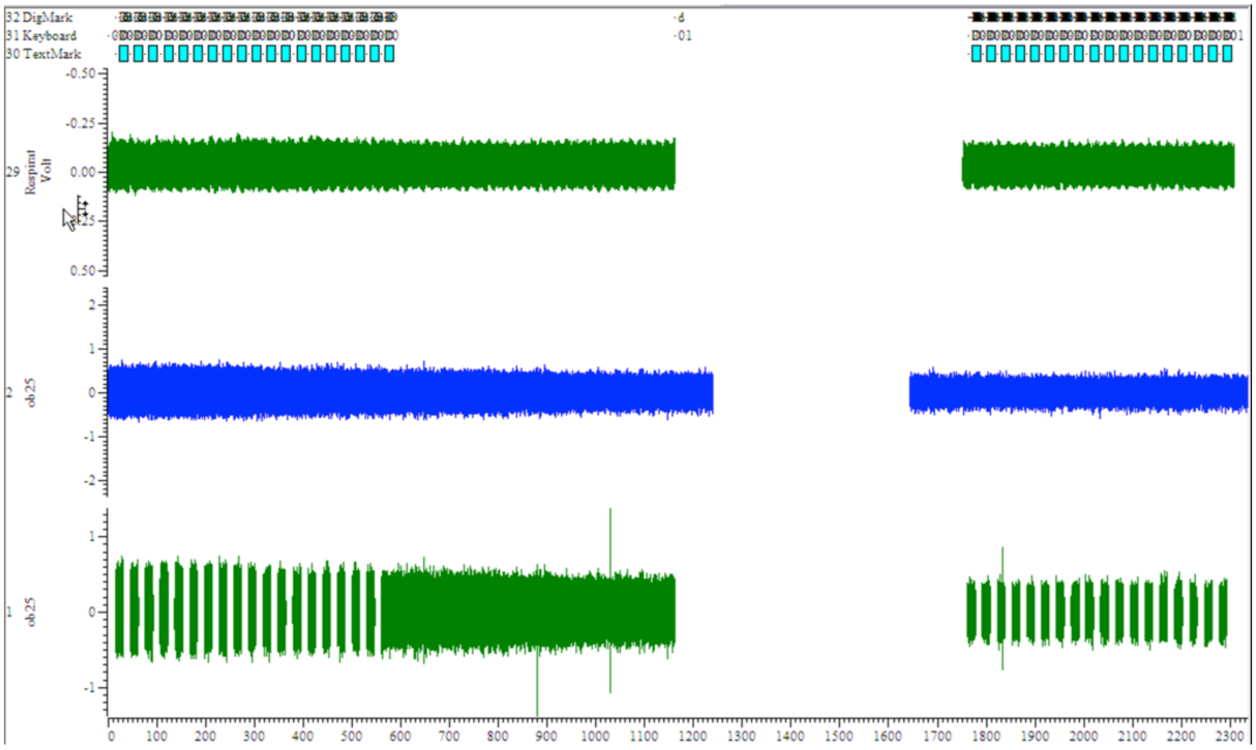


Figure 1.

Figure 2&3, are stretched versions of figure 1, which show more detailed information. Each trial starts at a *DigMark* 'D' and end with a *DigMark* 'd'; and between the *DigMark* 'O' and 'o', there is an odor involved. For every six trials, odors inserted are the same; and there are three different odors in the first 18 trials and last 18 trials, and the 19 trial has no odor, which is denoted as 'Blank' in '*TextMark'* channel of *EventArray*.

Specifically in figure 3, we can see how spikes are distributed in a single respiration event.

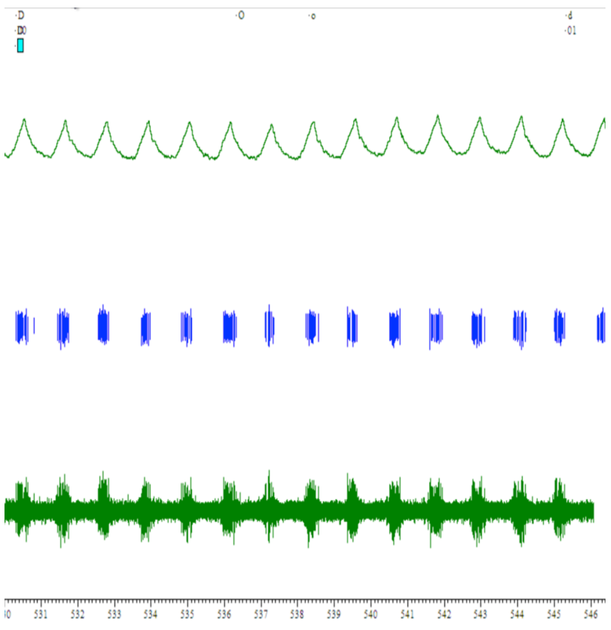


Figure 2.

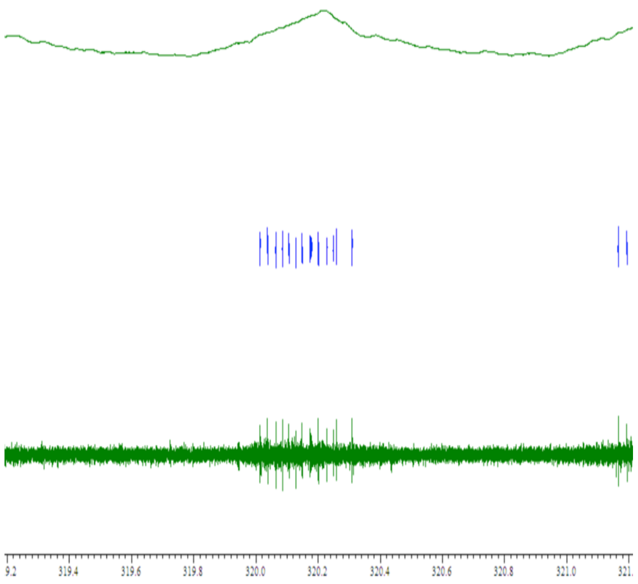


Figure 3.

Rates 1 & 2:

First we want to calculate the trail duration and save the start, end time for each trial in two arrays, array trailDuration[] and array trial[]. We want to use the data from EventArray channel and the channel_index should be 31 which is the DigMark eventArray channel. Using three important build-in functions in Python, we first map the channel.lable into an array codes[] which record the start and end of each trial as ‘D’, ‘d’. For each element in array codes[], we save channel.time in array trial[]. We calculate the trial duration with the help of array trial[]. Then we analysis the SpikeTrain channel, counting the spikes in each trial and the frequency in each trial. In this part, we encounter with a problem. It costs nearly 5 min to get the results which is to slow. We implement an algorithm to reduce the time complexity, and it costs just 10 seconds to get the results. In the end, we plot the frequency array to get the two figures we want.

We now get the Frequency Distribution (x-axis denote the frequency, and y-axis is how many of them in each region) as figure 4 and the Frequency in each trial (37 points in total, and y-axis is the actual frequency figure) as figure 5.

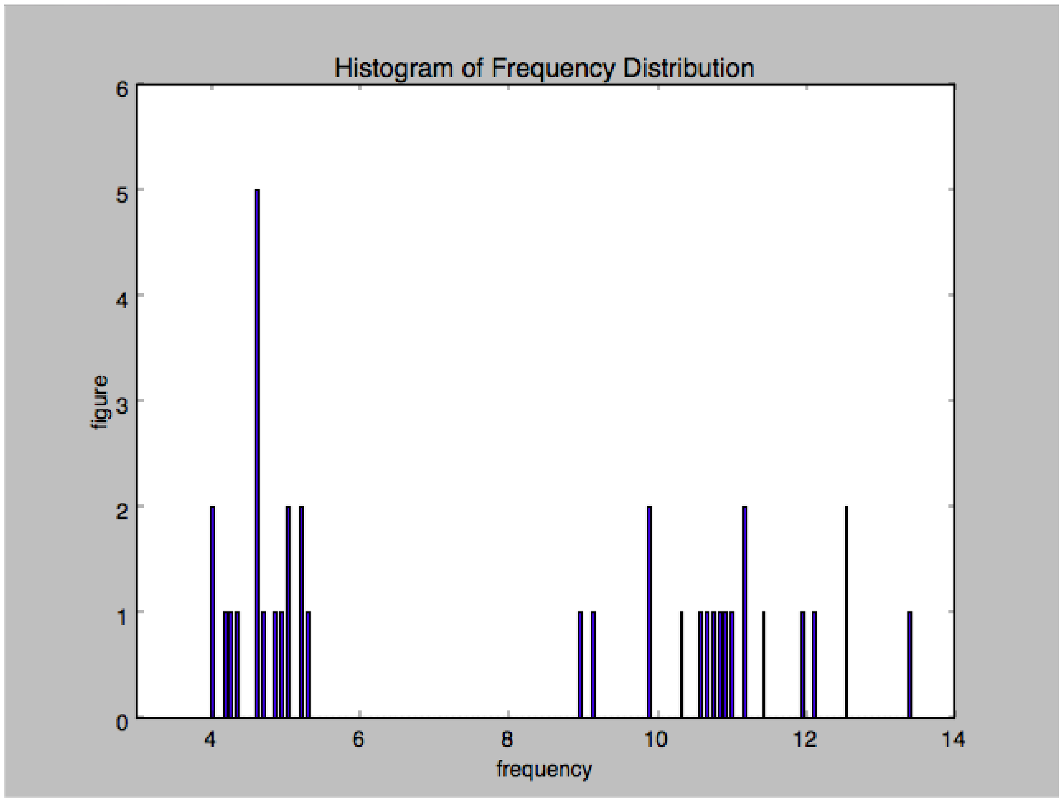


Figure 4.

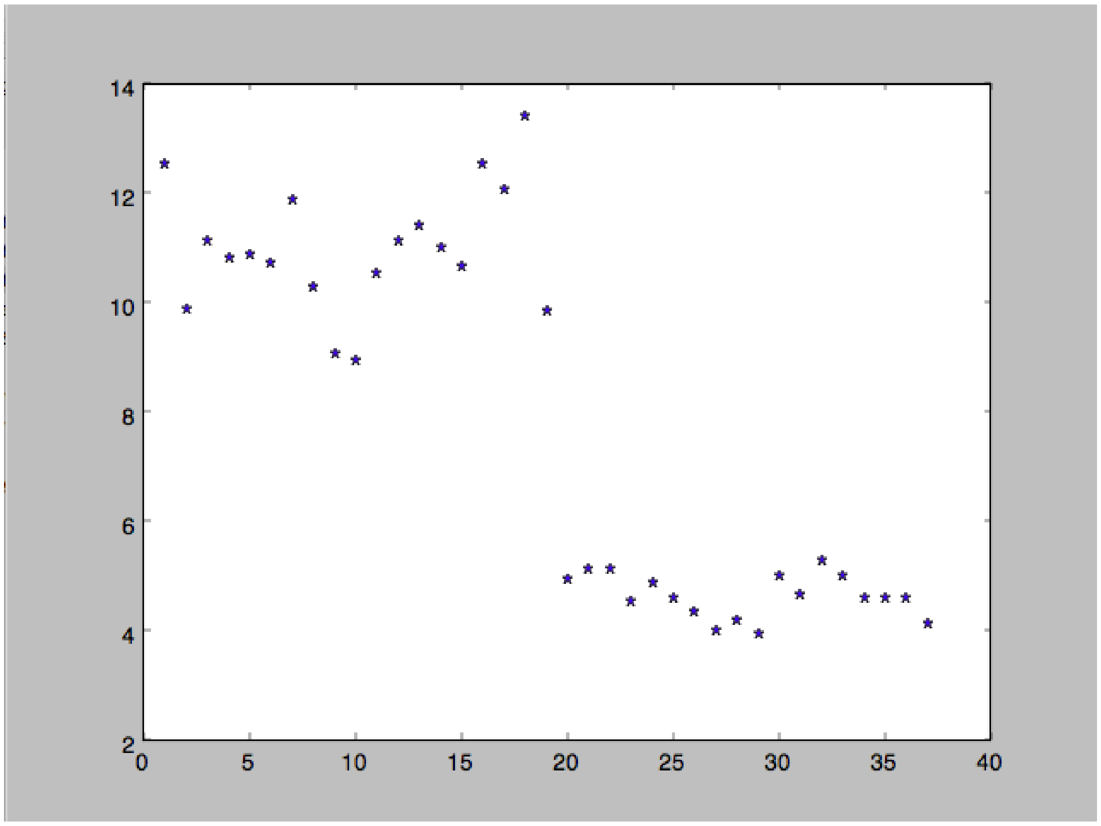


Figure 5.

Rate 3&4:

1). **Coefficient of variation**:
For a random variable X, coefficient of variation is defined as the ratio of the standard deviation σ to the mean μ

$$\mathrm{C_V} = \frac{\sigma}{\mu}$$

It is a measure of dispersion of a probability distribution. In our case, the distribution needs to be measured is interspike interval distribution. A value of $\mathrm{C_V} = 1$ implies that a given spike train is a Poisson process. If $\mathrm{C_V} > 1$, then the given spike train is less regular than a Poisson process with the same firing rate. If $\mathrm{C_V} < 1$, more regular.

In the code we compute the coefficient of variation for the interspike intervals and get the value of 0.9086 (for the laura_example.smr), which means that the interspike interval distribution of the example file is more regular than a Poisson process.

2). **Interspike interval distribution**:
The distance between two spike trains is the value of an interspike interval (ISI). Interspike interval distribution is simply a plot of the distribution of the observed time between multiple spikes. It displays the distribution pattern of spike activities in a neuron cell. For the example file, we record all the interspike intervals under odor stimulation and the plot the ISI distribution histogram. The results are shown in the following histograms, figure 6,7,8. Lateral axis denotes values of interspike intervals. Vertical axis denotes how many pairs of spikes have the values of interspike intervals which are less than a specific threshold. Since there are three different odors, there are three histograms and “coe”s below, which are results of the first 18 trials (three different odors).

```
# 'Odor Mix 0.01Pa'
Coe: 0.543624;
```

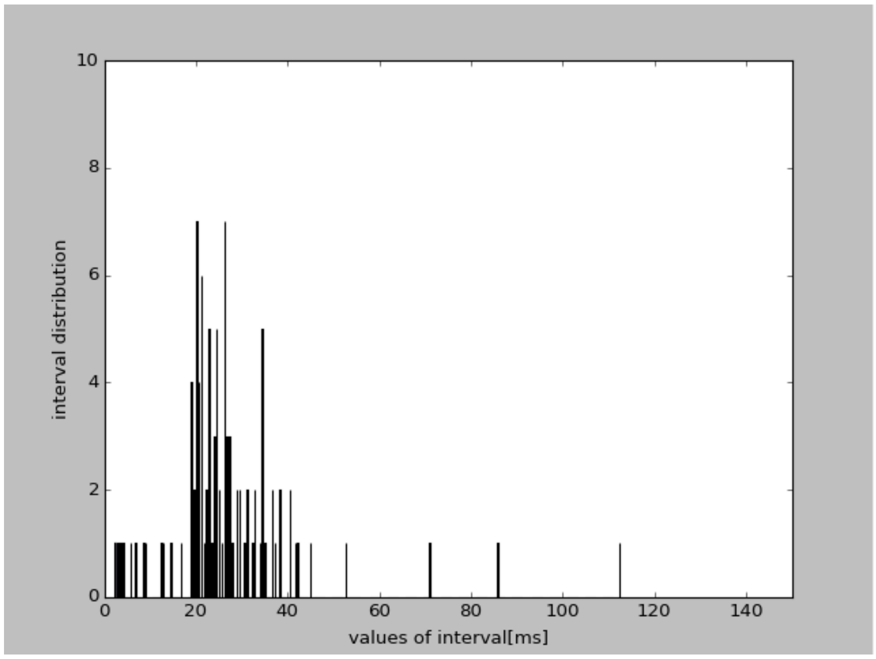


Figure 6.

# 'Odor Mix 0.1Pa'

Coe: 0.472321;

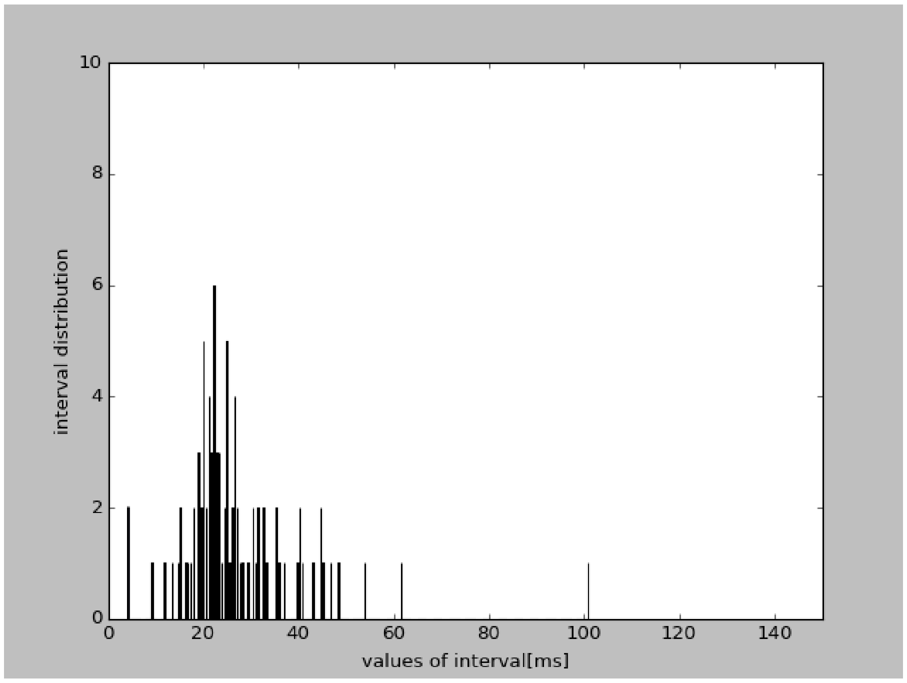


Figure 7.

# 'Odor Mix 1Pa'

Coe: 0.623842;

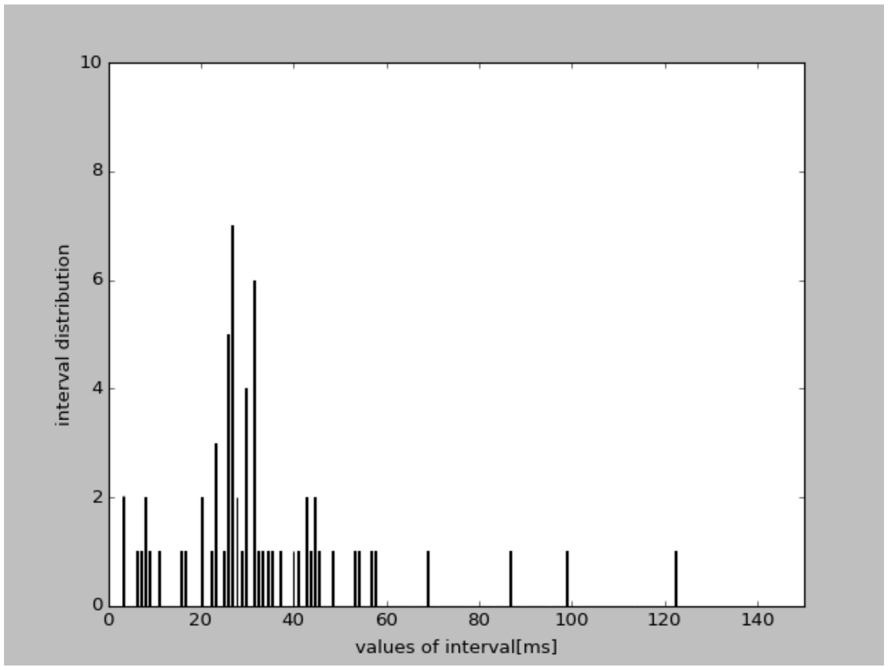


Figure 8.

Figure 6-Figure 8 reveal that under different odors, spike trains hold different patterns of interspike interval distribution. The more dispersed the distribution is, the bigger coefficient of variation is. For spike trains whose distribution is a Poisson process, coe equals to 1. A value of coe less than 1 implies that the spike train is more regular than Poisson process. We set up a Poisson model as an approximation to predict when next spike will occur based on the fact that the last spike occurred at time t. Figure 9 below is the simulation result. It shows that it is nearly impossible that next spike will happen in 35ms or longer, which has discrepancy compared with simulation results in Figure 6-Figure 8.

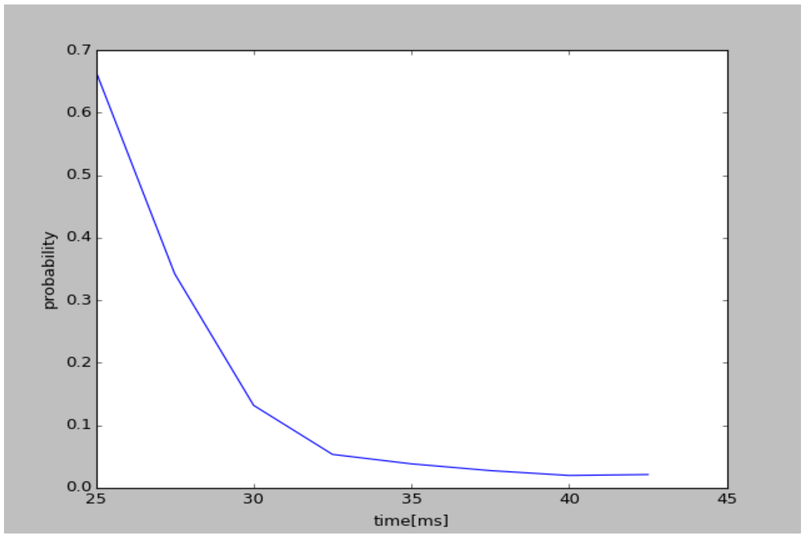


Figure 9.

Rate 5：

With regard to the rate 5 (phase of respiration) of a single cell, it was shown in figure 10 below (how each spike is distributed in a single respiration). Before we calculated the actual number, we used a 6th-order low-pass Butterworth filter with 300Hz cutoff frequency to reduce the disturbance of high-frequency noises as shown in Figure 11 below. So it became easier for us to locate and calculate the min & max value in next step. Afterwards, we found and extracted each local max and min value in the “refined” respiratory signal channel as well as the time of these values, and we store those time values in two separate arrays. Since there is also a time value for each of the spike event, we can have a clear view of how these spikes are distributed, and in which phase of the respiration they tend to appear. For simplicity and efficiency, we chose one of the trial to test this distributing pattern, and it has the same logic for both all other trials or the whole channel together, which has a larger data volume. The result is shown in Figure 12 as a histogram format. In x-axis, there are only 3 mark: 0, PI and 2PI, which means the starting point of a respiration cycle or min value (phase = 0), the max value (phase = π), and the ending point of this cycle or the next starting point (phase = 2π). In other words, it is distributed by phase value instead of timestamp. Y-axis stands for how many spikes shown in a given range. Thus, as the figure shown below, it is obvious that most of the spikes appear close to the max value, which in fact, is the exchange of inhale and exhale.

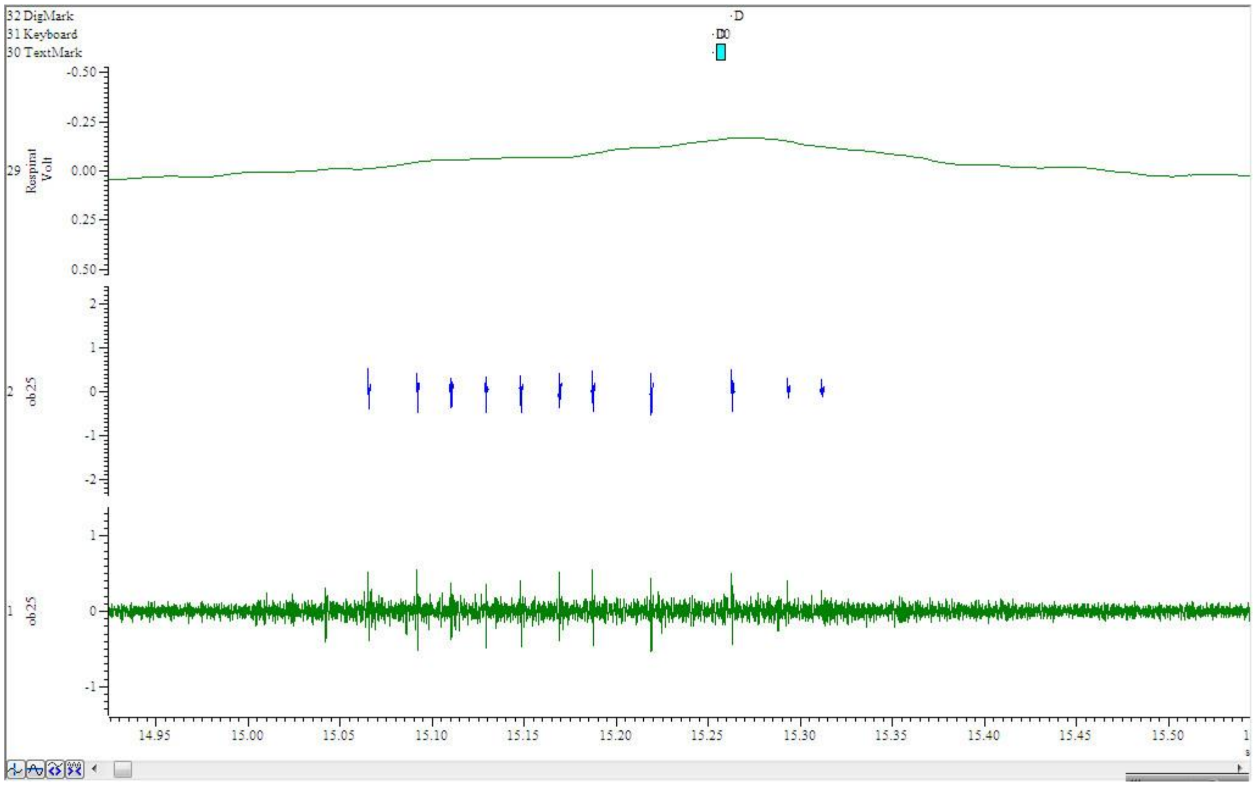


Figure 10.

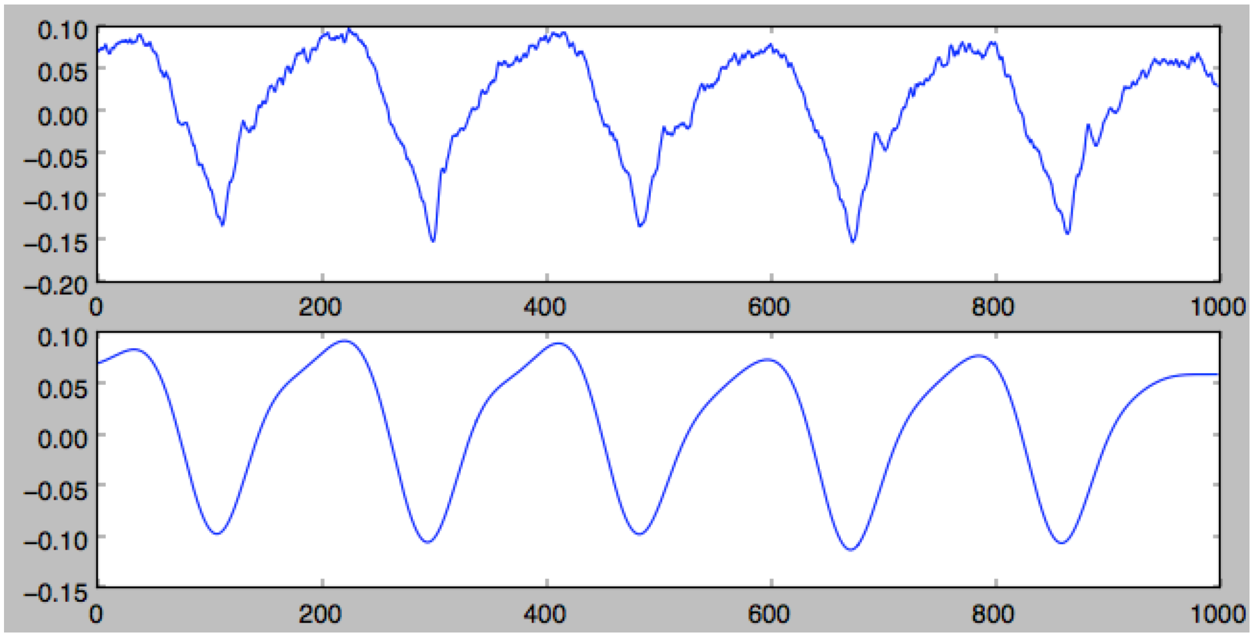


Figure 11.

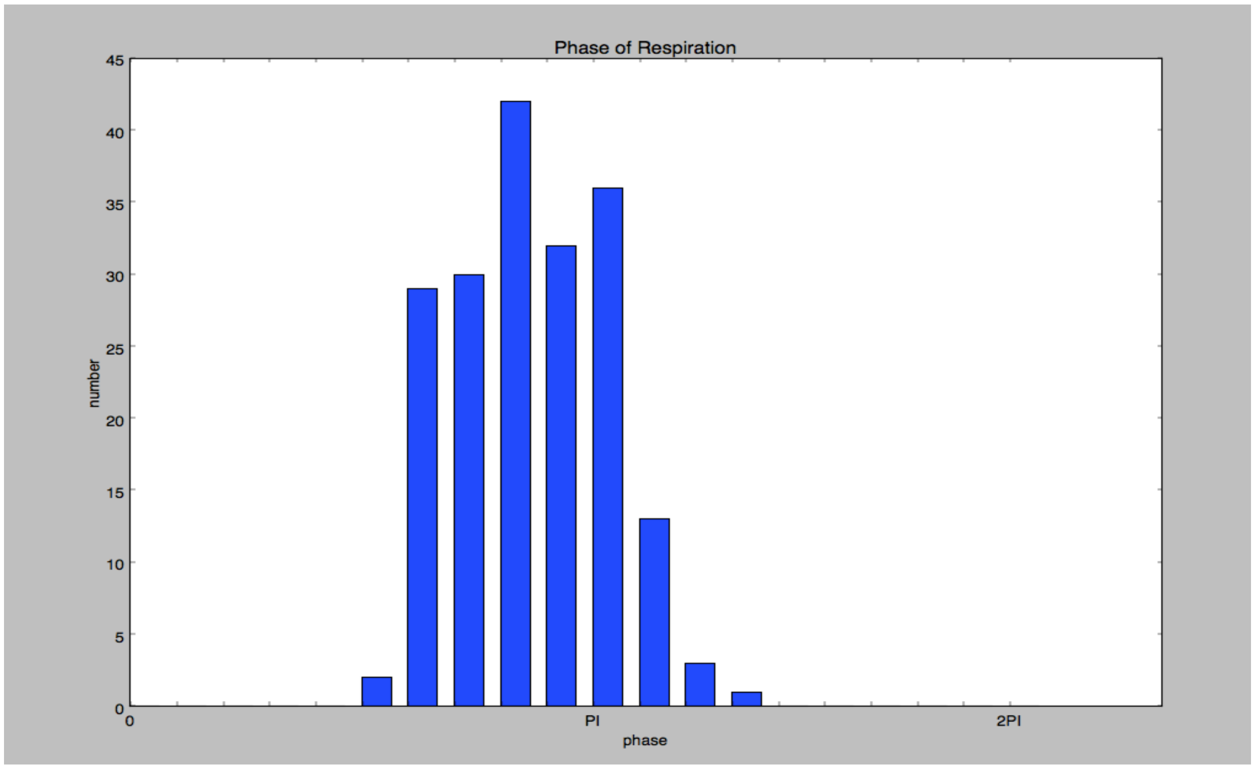


Figure 12.

Rate 6:

In this rate, we examine another two specific statistical properties: coherence and co-variation between spike trains and respiration. By definition: Coherence between two signals or data sets $x(t)$ and $y(t)$ is defined as:

$$C_{xy}(f) = \frac{|G_{xy}(f)|^2}{G_{xx}(f)G_{yy}(f)}$$

It is used to estimate the causality between two signals or data sets. In our example, the two data sets are spike trains and respiration. We extract these two signals from the file and use the function "signal.coherence (signal1, signal2, fs, nperseg)" from Scipy Python library to compute and display their coherence which is shown in Figure 13.

Co-variation indicates the level to which two variables vary together. Suppose there are two variables X and Y, co-variation between them is defined as

$$\mathrm{cov}(X, Y) = \mu([X - \mu(X)][Y - \mu(Y)])$$

In our code we use function "np.cov (signal1,signal2)" from Python package "np" to compute the co-variation. The two signals are the first 100 sample data extracted from spike trains and respiration. The result is a 2 dimensional matrix. It is exactly the covariance matrix.

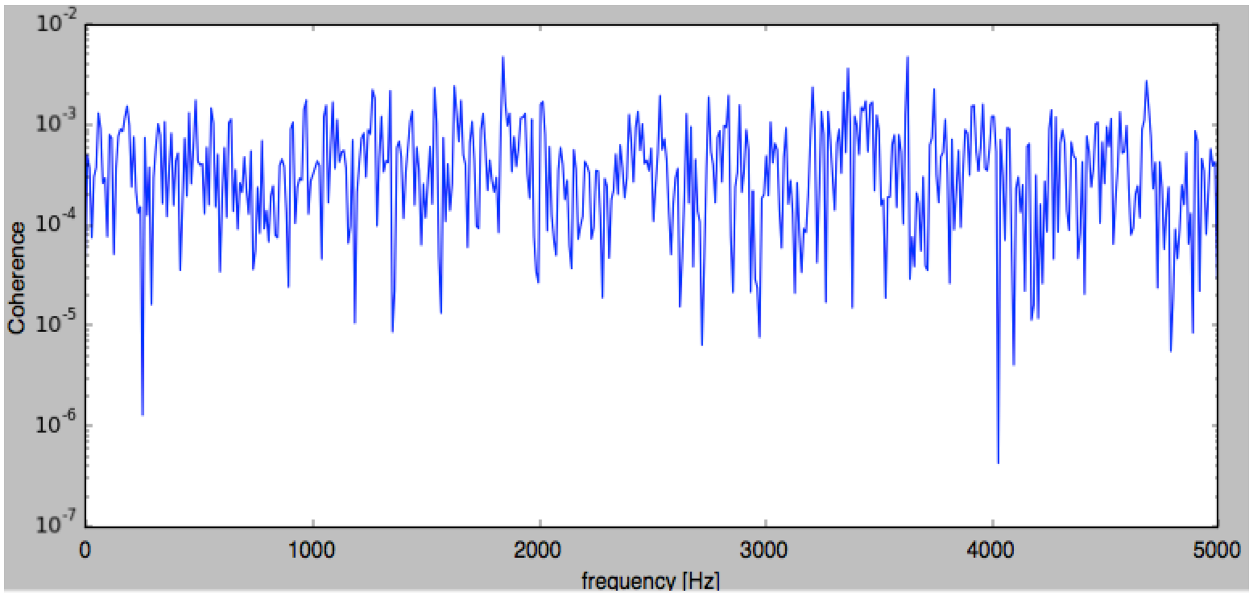


Figure 13.

Multiple cells:

Before we got several new files of multiple cells (e.g. nick_0316615_002). From figure 11&12, we noticed that different cells are shown in different colors, but there are two classifiers which decide which spike comes from which cell and if it is a spike (surpass the threshold value). Different color(cell) can be extract by its identity (different channel of SpikeTrain). Thus, by selecting a specific channel number within the SpikeTrain channels, we can have it go through the same process as a single cell's SpikeTrain.

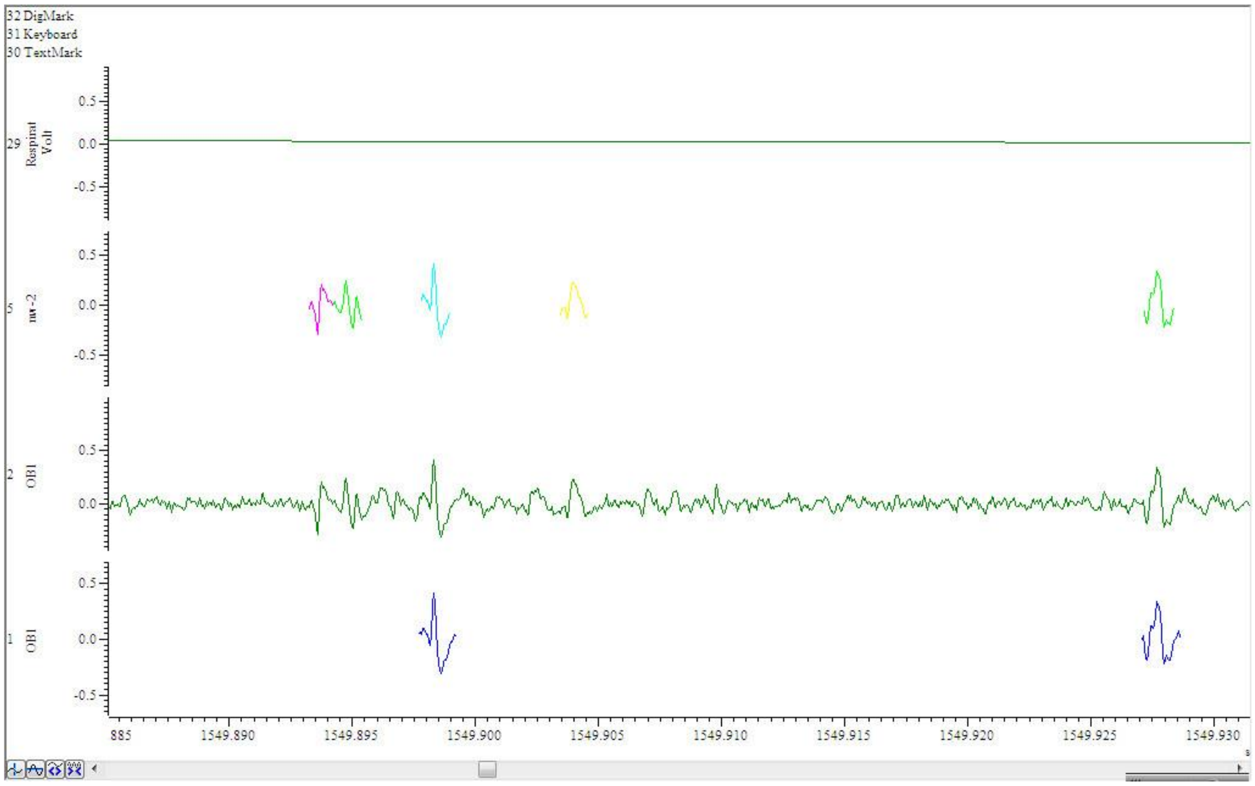


Figure 14.

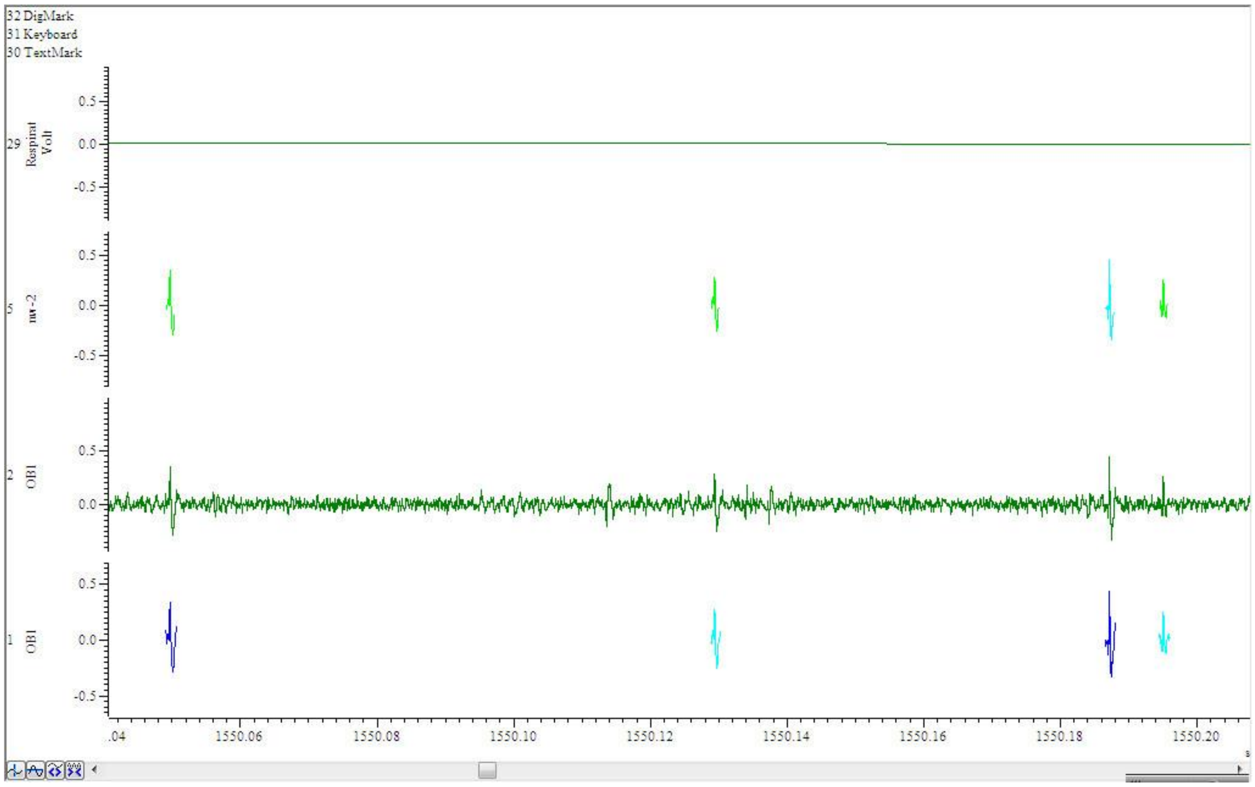


Figure 15.

GUI part:

When you run this project, the GUI will then appear.

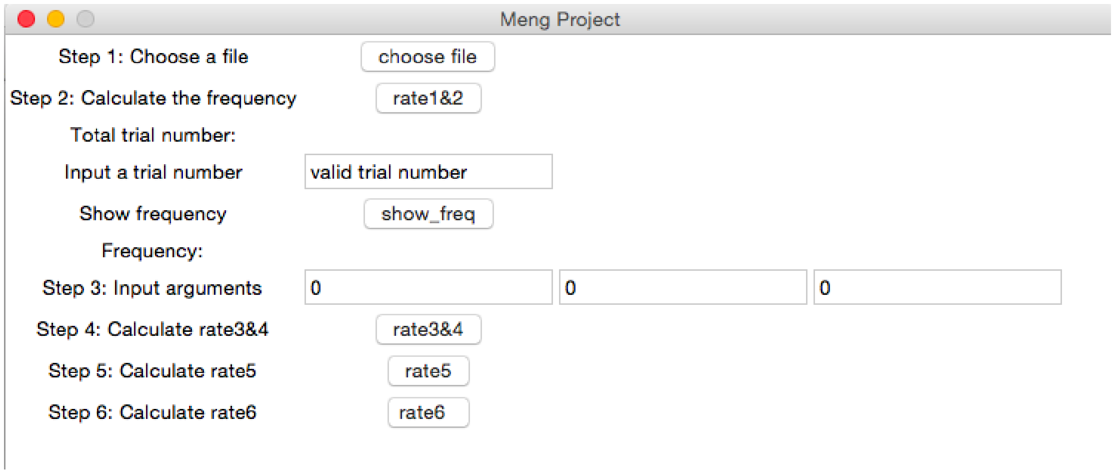


Figure 16.

First, press the ‘choose file’ Button, and direct to the location of the suitable input file. For example, “/Desktop/M.Eng_Doc/laura_exmple.smr”

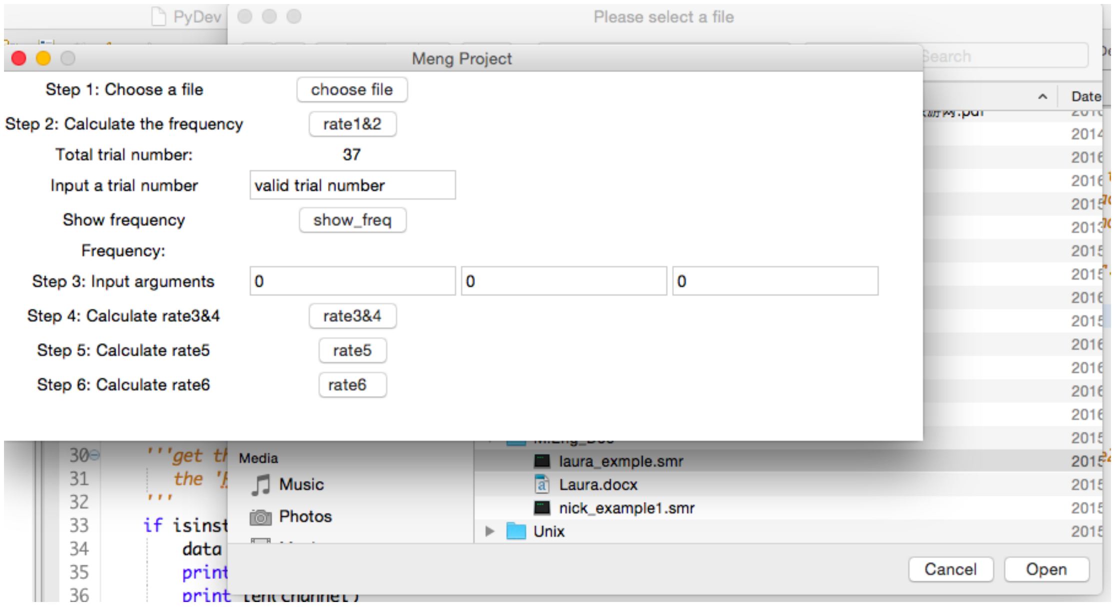


Figure 17.

After choosing file, you can calculate the frequency for this input file.

Please don’t press the ‘show_freq’ Button before you press ‘rate1&2’ Button.

Press ‘rate1&2’ Button, you will get the histogram for the spike frequency for each trail. The figures below is the outcome. And also, the total trail number will appear after the Total trial number Label, with this value, you can choose a valid trail number.

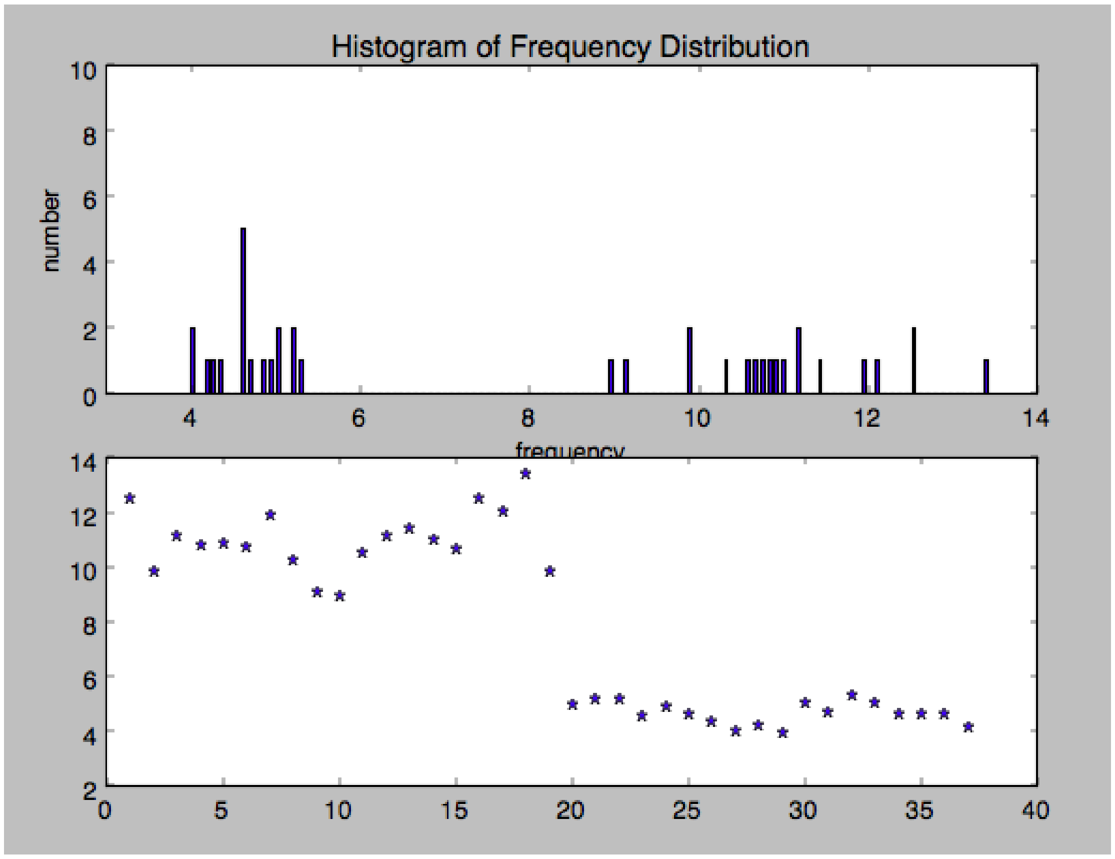
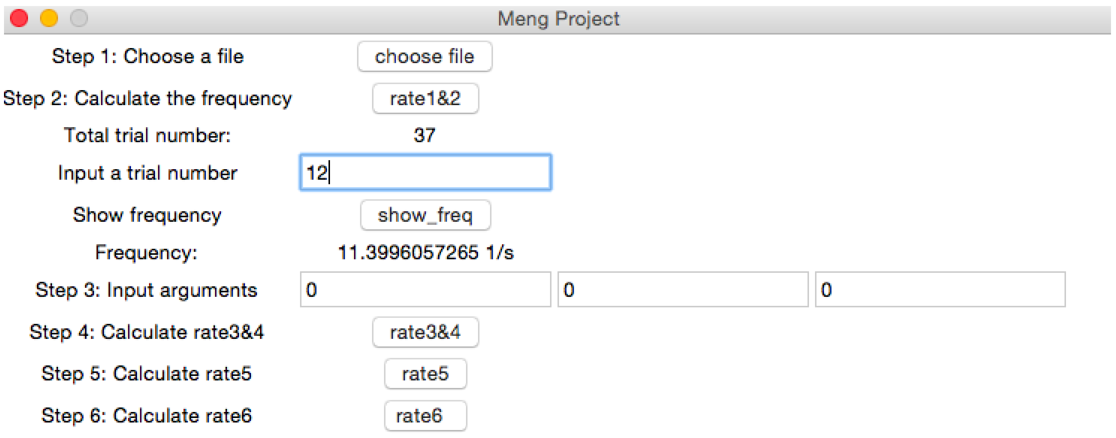


Figure 18.

In this input file, there are 37 trails in total. And you can get the specific frequency value for each of them with a proper input and press the ‘show_freq’ Button.

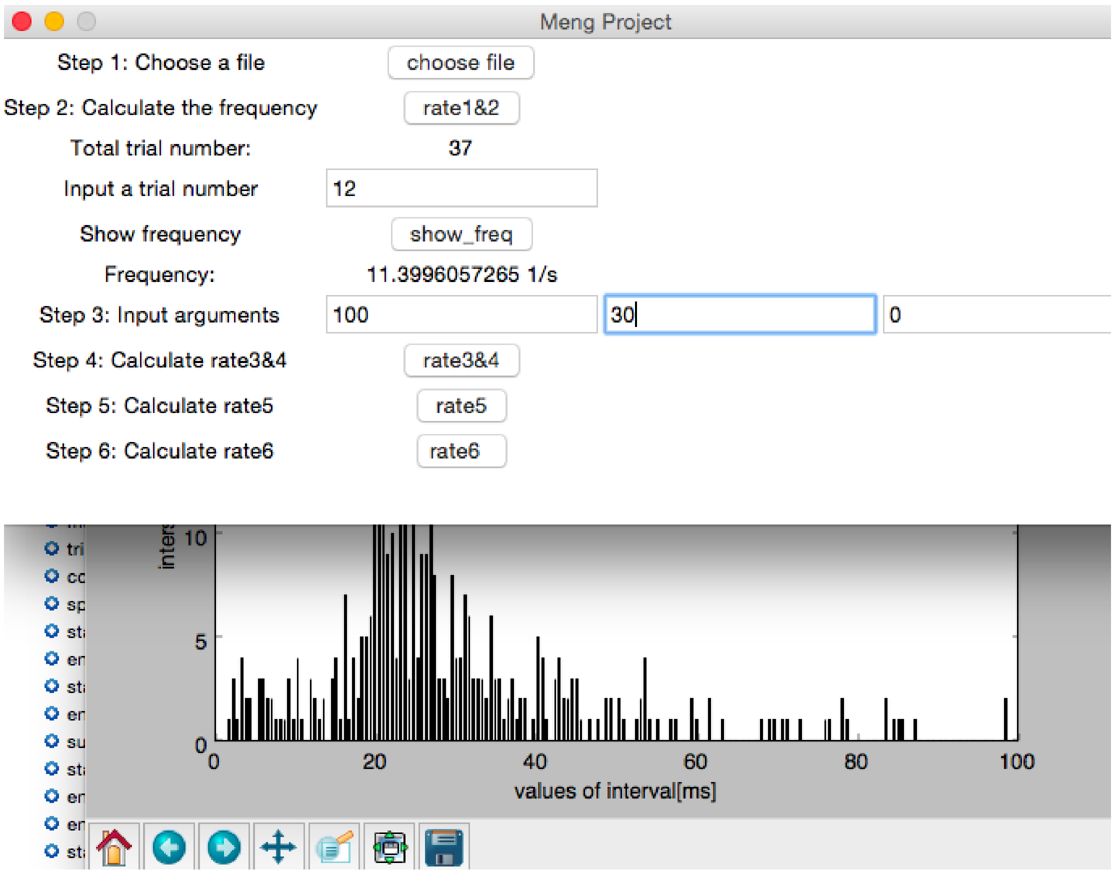


Meng Project

Step 1: Choose a file	choose file		
Step 2: Calculate the frequency	rate1&2		
Total trial number:	37		
Input a trial number	12		
Show frequency	show_freq		
Frequency:	11.3996057265 1/s		
Step 3: Input arguments	0	0	0
Step 4: Calculate rate3&4	rate3&4		
Step 5: Calculate rate5	rate5		
Step 6: Calculate rate6	rate6		

Figure 19.

Before you press 'rate3&4' Button, you should give a proper input for this function, for example "100","30","0" (the parameter choice has been explained in Rate 3&4 section).
Then press the 'rate3&4' Button. You will get the outcome without an error.



Meng Project

Step 1: Choose a file	choose file		
Step 2: Calculate the frequency	rate1&2		
Total trial number:	37		
Input a trial number	12		
Show frequency	show_freq		
Frequency:	11.3996057265 1/s		
Step 3: Input arguments	100	30	0
Step 4: Calculate rate3&4	rate3&4		
Step 5: Calculate rate5	rate5		
Step 6: Calculate rate6	rate6		

Figure 20.

The function for rate5 is partly completed as suggested above, it can give you a test outcome from one of the trials.
When you press ‘rate6’ Button, it will show you the outcome properly.

Problem and Notation:
If you press the ‘show_freq’ without a proper value or before you press ‘rate1&2’ Button, there will be an error.
If you press the ‘rate3&4’ Button without input values, there will be an error.
You must choose an input file before you use any of the functions.
You can press each button again after it shows you the outcome.
There is no specific order for ‘rate1&2’, ‘rate3&4’, or ‘rate6’.

Conclusion

The main task in this project is designing Python modules which can be used to acquire relevant information about data collected from mice. We did a lot of research of specific definition and relevant notes before we started. To achieve all the goals, several analysis methods and algorithm are adopted during both the stage of analysis and coding. As shown in the previous sections, there are method explanation as well as coding strategies involved in each rate of the task. However, the last rate (rate 7) is unfinished yet. Besides, there is also a user friendly GUI specifically designed for this project.

Acknowledgement

We would like to thank our advisors: Bruce R. Land, Christiane Linster for their guidance, and Matthew C. Stockton for his assistance. We were not familiar a lot of ideas and concepts in electrophysiological field at the very first stage, and we have also faced some difficulties during the whole progress. Without their patience and help, we could not easily conquer many obstacles.

Reference

http://ced.co.uk/products/spike2

https://pythonhosted.org/neo/io.html

https://docs.python.org/2/tutorial/

https://en.wikipedia.org/wiki/Coefficient_of_variation

http://neuronaldynamics.epfl.ch/online/Ch7.S3.html

https://en.wikipedia.org/wiki/Coherence_

https://en.wikipedia.org/wiki/Covariance