Clustering Neural Spike Data

Homework 7 Assignment Info Homework #: HW7 Description: GMM, PCA for Spike Sorting Course: EN.553.636 Introduction to Data Science Semester: Spring 2023, Homewood Campus Instructor: Tamas Budavari TA: Matthew Tivnan Date: April 12, 2023

Student Info Name: Amir Hossein Daraie JHED-ID: adaraie1 Email: adaraie1@jhu.edu (mailto:adaraie1@jhu.edu)

1. Introduction:

Extracellular recordings can provide information about neuronal activity in the brain. A recorded signal by the microwire will be amplified and filtered to create waveforms that are a combination of background activity and nearby neuronal firing from a clusters of neurons in the form of spikes. This waveform is further processed (Ex. using amplitude thresholding) to identify the spikes. The next step is to group these spikes into clusters. The assumption is that neurons tend to fire spikes of a particular shape and by exploring diffrences and similarities in the spike shape, one can potentially separate spikes according to their particular population of neurons.

In this activity, we will process sample neuronal voltage recordings and use the Principal Component Analysis (PCA) to separate spikes.

Reference:

http://www.scholarpedia.org/article/Spike_sorting (http://www.scholarpedia.org/article/Spike_sorting)

2.1 Read the data:

You are provided with a matlab file **SpikeSorting.mat** that contains waveform volatage recording from population of neurons. Read this file using **scipy** module. This data has been preprocesed and centered for each spike to coincide with the others. Each spike contains 70 recorings representing 70 ms of data.

In [1]:

```
# read mat files containing the voltage data

import scipy.io
import numpy as np

# note: convert to pandas data frame or numpy
mat = scipy.io.loadmat('SpikeSorting.mat')
```

2.1 Extract and display the data:

The Matlab file contains two sets of arrays:

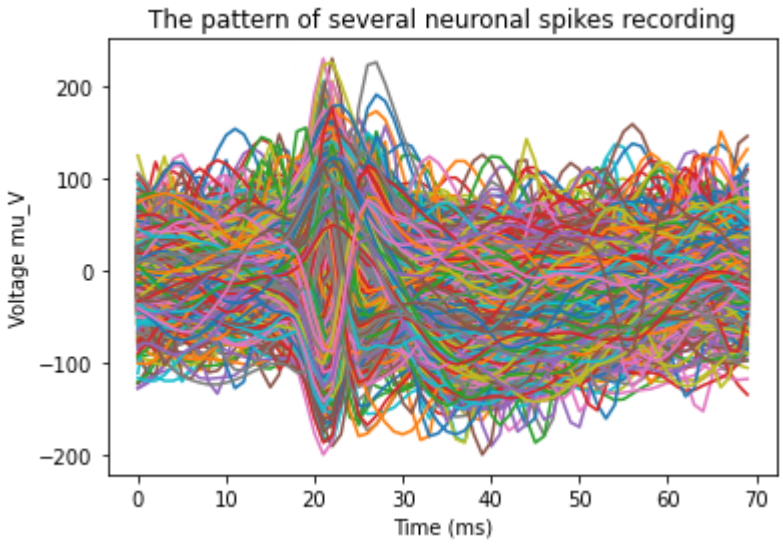
1. spikes : indicies where each spike is detected in the voltage recording.
2. voltage: recording of neuronal firing as voltages. Use the indicies in the **spikes** array to extract recordings for each spike (recall that each spike has 70 recordings).

Use numpy to generate a two dimensional array in which rows are observed spikes and columns are voltage readings for each spike. Display all spikes in one figure.

In [2]:

```
import matplotlib.pyplot as plt
plt.rcParams["figure.figsize"] = (10,5)

# allow using latex in matplotlib
# plt.rcParams['text.usetex'] = True

# Data structure

# spikes array: contains indicies defining the initial location of each spike in the
# The total number of elements in the spikes array defines the total number of spike

# voltage array: recorded voltage waveforms for different spikes. Each spike has 70
# "spikes" array to generate a two dimensional array in which rows are observed spik
# are voltage readings for each spike


spikes = np.squeeze(mat['spikes'])
voltages = np.squeeze(mat['voltage'])

obser_num = np.size(mat['spikes'])
num_samples = 70 # hard coded

data_array = np.zeros((obser_num,num_samples))

print(data_array.shape)

for n in range(obser_num):

    data_array[n,:] = voltages[spikes[n]: spikes[n] + 70]

    plt.plot(np.arange(70),data_array[n,:])

plt.xlabel(r'Time (ms)')
# plt.ylabel(r'Voltage $\displaystyle (\mu V)$')
plt.ylabel(r'Voltage mu_V')
plt.title('The pattern of several neuronal spikes recording')
plt.show()
```

```
(3298, 70)
```

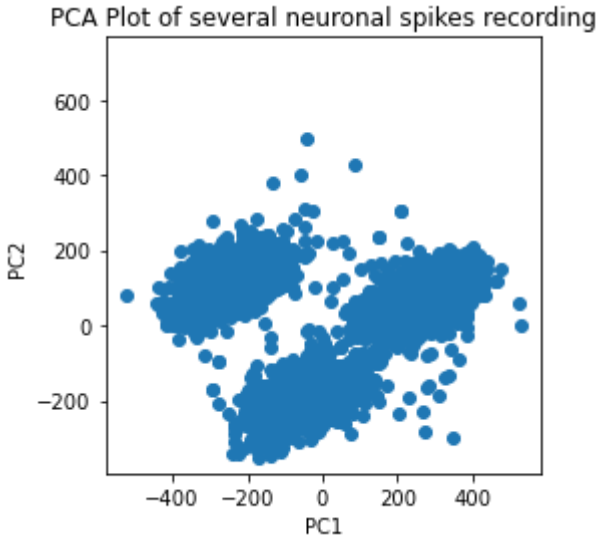


Part 1: Dimensionality Reduction with PCA

[7pts] Apply prinicpal component analysis to reduce the data from 70 raw features (time stamples) down to 2 principal components. Plot a scatterplot showing the first two principal components for the full dataset

In [3]:

```
from sklearn.decomposition import PCA

X = data_array
model_pca = PCA(n_components=2, random_state=42).fit(X)
X_reduced = model_pca.transform(X)

plt.subplot(111)
plt.scatter(X_reduced[:,0], X_reduced[:,1])
plt.axis('square')
plt.xlabel('PC1'), plt.ylabel('PC2')
plt.title("PCA Plot of several neuronal spikes recording")
plt.show()
```

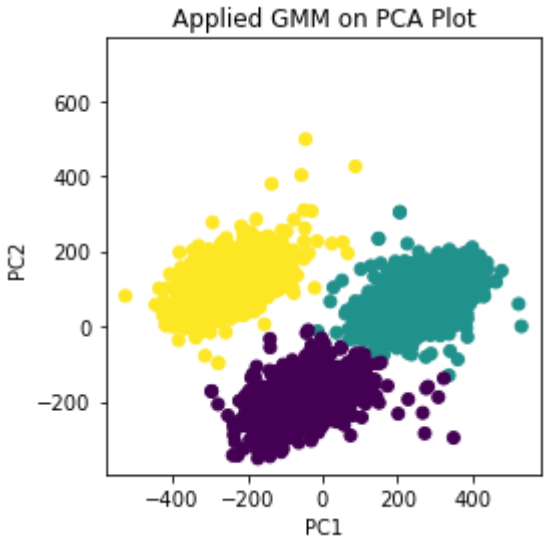


Part 2: Clustering with Gaussian Mixture Models

[7pts] Apply gaussian mixture models with 3 clusters. Train the model on the full dataset.

In [4]:

```
from sklearn.mixture import GaussianMixture

model_gmm = GaussianMixture(n_components=3, covariance_type="full", random_state=42)

classes = model_gmm.predict(X_reduced)

plt.subplot(111)
plt.scatter(X_reduced[:,0], X_reduced[:,1], c=classes)
plt.axis('square')
plt.xlabel('PC1'), plt.ylabel('PC2')
plt.title("Applied GMM on PCA Plot")
plt.show()
```



Part 3: Reconstruct Different Types of Spikes

[5pts] Use the trained gaussian mixture model to sample five new examples of the 2 prinicipal component coefficients from each of the three clusters. Then, use those sampled prinicipal component coefficients (eigen values) together with the corresponding principal components (eigen vectors) to create an approximation of the 70ms signal for each of the fifteen samples (3 clusters, 5 samples per cluster). Create three figures, one for each cluster, with the plots of the five 70ms signals. Add a title to indicate neural spike type A, neural spike type B and neural spike type C.

In [5]:

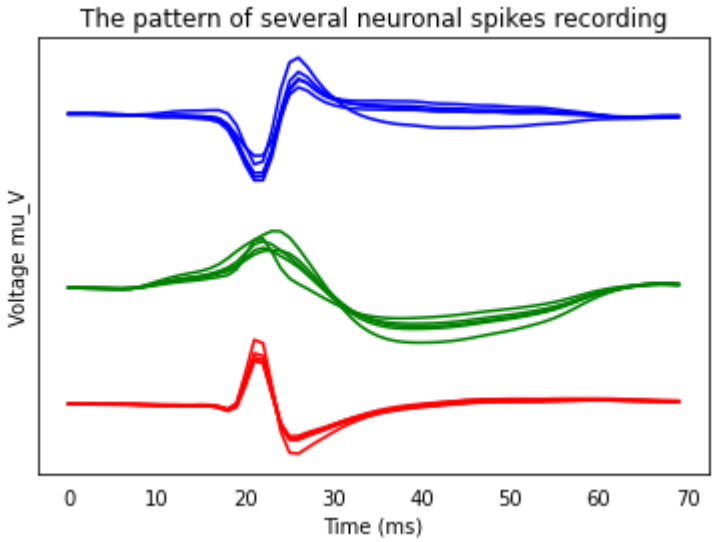
```
X_reduced
```

Out[5]:

```
array([[-147.14476463,  239.47889375],
       [-113.31232773, -294.4298417 ],
       [  42.65143876, -175.68776111],
       ...,
       [  81.56008738, -185.17592532],
       [-216.87798256,  165.51155261],
       [ 307.77546363,   72.87130098]])
```

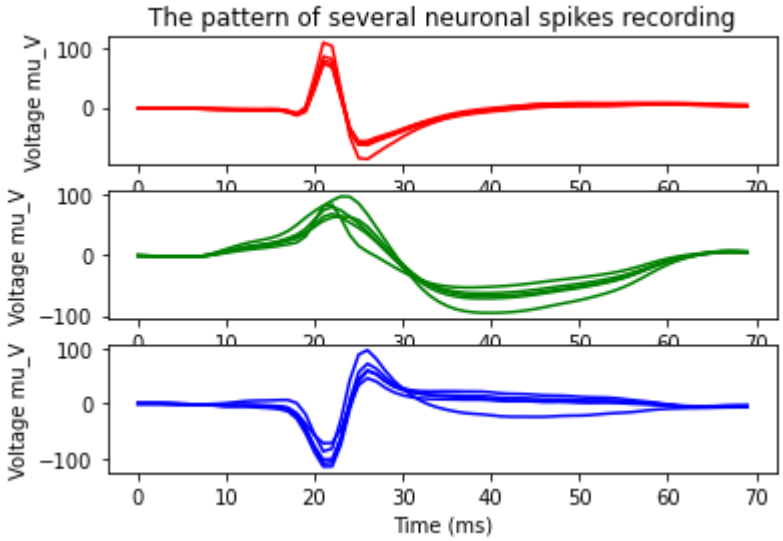
In [6]:

```
idx = np.where(classes==0)[0]
idx = np.random.choice(idx, 5)
X_recon_c0 = model_pca.inverse_transform(X_reduced[idx])

idx = np.where(classes==1)[0]
idx = np.random.choice(idx, 5)
X_recon_c1 = model_pca.inverse_transform(X_reduced[idx])

idx = np.where(classes==2)[0]
idx = np.random.choice(idx, 5)
X_recon_c2 = model_pca.inverse_transform(X_reduced[idx])

plt.subplot(111)
for i in range(5):
    plt.plot(np.arange(70), X_recon_c0[i,:]+0, c='r')
    plt.plot(np.arange(70), X_recon_c1[i,:]+200, c='g')
    plt.plot(np.arange(70), X_recon_c2[i,:]+500, c='b')
plt.xlabel(r'Time (ms)')
plt.ylabel(r'Voltage mu_V')
plt.title('The pattern of several neuronal spikes recording')
frame1 = plt.gca()
for tick in frame1.axes.get_yticklines():
    tick.set_visible(False)
for tick in frame1.axes.get_xticklines():
    tick.set_visible(False)
for xlabel_i in frame1.axes.get_yticklabels():
    xlabel_i.set_fontsize(0.0)
    xlabel_i.set_visible(False)
plt.show()
```



In [7]:

```
plt.subplot(311)

for i in range(5):
    plt.plot(np.arange(70), X_recon_c0[i,:], c='r')
plt.xlabel(r'Time (ms)')
plt.ylabel(r'Voltage mu_V')
plt.title('The pattern of several neuronal spikes recording')

plt.subplot(312)
for i in range(5):
    plt.plot(np.arange(70), X_recon_c1[i,:], c='g')
plt.xlabel(r'Time (ms)')
plt.ylabel(r'Voltage mu_V')

plt.subplot(313)
for i in range(5):
    plt.plot(np.arange(70), X_recon_c2[i,:], c='b')
plt.xlabel(r'Time (ms)')
plt.ylabel(r'Voltage mu_V')

plt.show()
```



Part 4: Provide an Interpretation of the Clusters

[5pts] Write an explanation about how to interpret these three types of neural signals. Why do you think this type of clustering analysis could be useful?

The three different sorts of brain signals might be explained by three separate populations of neurons firing in three different ways. The spike groups in the clusters all have a similar form, indicating that they are part of the same population of neurons. The first cluster can be the brain's background activity or spikes produced by neurons connected to the job under investigation. The activity of several types of neurons or various phases of the same type of neuron activity may be represented by the second and third clusters, respectively. For instance, the second cluster may be an example of an inhibitory neuron, whereas the third cluster might be an example of an excitatory neuron.

Type *Markdown* and LaTeX: α^2