```
# -*- coding: utf-8 -*-
"""
Created on Thu Oct 14 17:31:50 2021

@author: Ben
"""

#Ben McAteer Midterm Project BME 511

import       as
import                    as
import          as
import              as
from                        import
import
#import pandas as pd

#from sklearn.decomposition import PCA
#from scipy import signal
from          import
#from numpy.fft import fft, ifft
#from scipy.signal import find_peaks
#from scipy import linalg

import
import

           6
     2048 #Hz sample rate




def SST                                 # pulls all matlab files for a specific subject in order fr

            '/Users\Ben\OneDrive - purdue.edu\BME 511\MidtermProject\AllSubjsEEG\subject{}\sessio
                        '*.mat'
    #print(mat_files) #troubleshoot
                                                          True


                'data'   str

                                      'data'  #raw data of the current trial
                                        'events'
                                         'stimuli'
                                        'target'
                                                'targets_counted'

    # print('data size', np.shape(Trial['data']))
```

```
    # print('events size', np.shape(Trial['events']))
    # print('stimuli size', np.shape(Trial['stimuli']))
    # print('target ', (Trial['target']))
    # print('targetcounted ', (Trial['targets_counted']))

    # for key, value in data.items():
    #    print(key)


    return

def unpackStamp
                    0
                     1
                    2
                    3
                     4
          5

                                      1e6

    return


def events2samps
                     0 4
                           0

    for   in range
                                                              0
                                                                                  1


    return

#1A
#from numpy.fft import fft, ifft
from       import
import

def sorting_epoch

               'data'   str

    #print(np.shape(data))

    # for key, value in epochs_data.items():
    #     print(key)
```

```
            1 #Hz low frequency of the filter
             12 #Hz high frequency of the filter

                 32       5           33      0 5

    for   in range  0 32
                                       #subtracts the data by the average of the two reference


    #create and filter Data
               int         1   5        #filter length with 0.5 being the transition band
                                                                    False  #compute filter
                                         1                 1  #Filtered Data with baseline filter


                                          #index where event occurs in the session

               #np.zeros(len(Eventsdata))
    #time = np.arange(0,1,1/fs) #0 to 1 second at sample freq
                            len
                               len

    for   in range len              #Eventsdata.astype('int'):
                                   int                int               2048    #events +1000ms of
        if
                              1
        else
                                 1

             #need to create the difference beweteen targets vs non


    for   in range len               #currently, instead of making the target epoch 34x whatever,
        if                    1

        else



##find avg of the first 100 of each epoch, then subtract avg from each epoch



    #subtracting the baseline DC from the target epochs
    for   in range 0 len
```

```
        # targetData[curfile].append(target_epoch[i][:][:] - (np.mean(target_epoch[i][:][0:205])))
        for   in range 0  32

            #epochmean = np.mean(target_epoch[i][j][0:205])
                                                                                    0 205



    for   in range 0 len
        #nontargetData[curfile].append(nontarget_epoch[i][:] - np.mean(nontarget_epoch[i][:][0:205
        for   in range 0  32
             #epochnonmean = np.mean(nontarget_epoch[i][j][0:205])



## REMOVING NOISE OVER 40uV




    for   in range 0 len
                                           abs

                          1
        if                              40
                                                          #should creat a dictionary of
                        1
            #targetData[curfile][i] = 0 #use np.nanmean to avoid nan's in the future
    #filttargetData[curfile] = targetData[~np.isnan(targetData)]    #use np.nanmean to avoid nan's
        #if target_epoch_max[curfile][i] == 0:
            #targetData[curfile][i]= np.delete(targetData[curfile][i],np.nan) #Currently need to r
            #targetData[curfile][i] =targetData[curfile][np.logical_not(np.isnan(targetData[curfil

    for   in range 0 len
                                            abs

        if                                40

                          1
            #nontargetData[curfile][i] = np.nan
        #     nontargetData[curfile][i] = 0 #use np.nanmean to avoid nan's in the future
        # if nontarget_epoch_max[curfile][i] == 0:
        #     nontargetData[curfile].remove(nontargetData[curfile][i],0)


    #set = 0, if x = 0.remove(array, obj)(targetdata, 0)

    #300targetand nontarget epochs per eeg electrode for all trials per one subject

    #print(len(FilterData[1,:])) # two different ways to show either rows or columns length
    #print(FilterData.shape[1])
```

```
    return


           6
         4
       6
             0
                0
       31     1
       32     1
       13     1
       16     1
       1

             0
                0
                  0
```

```
for   in range 0
    print 'youre doing great sweetie'

    for   in range 0

        print 'Session Number'  +1 'Trial Number'  +1
                                                                    +1           #TRIALS START
        #Edata,Eevents,Estimuli,Etarget,Etargets_counted = SST(Subjects,4,j, count) #TRIALS START



                1


def avg_epochs                                      #need to combine all 24 trial epochs, bes

             'data'   str


    #print(len(epoch_targetData[curfile]))
   # for i in range(0,len(epoch_targetData[curfile])):
         #for j in range(0,32):
         #avgtargetepoch[curfile] = np.nanmean(targetData[curfile][:],axis = 0)
                                                                        0  #works but doesnt ign
                                                                           0
```

```
                                                    # converts from a list
                                                         #converts from a list
    #avgnontargetepoch[curfile] = np.nanmean(epoch_nontargetData[curfile][:])

    return                                  #avgtargetepoch,avgnontargetepoch)


def PermutationSetup
    #Need to create target data and non target data OUT of dictionarys like i did withOverall Targ
              'data'   str




    return




     1

for   in range 0
    for   in range 0



        print 'Averaging Epochs Session'  +1 'Trial'  +1
              1
```

```
                        0 32
                          0 32
#for i in range(0,len(avgtarget)):

                                              0
                                                  0
#print(noisytargetcount,'noisy targets')


def Permutation

             1000
                                             0
                                                  0

                                                                        1000

                                             0
                                                  0

                                                                         1000

                                             0
                                                  0

                                                                        1000

                                             0
                                                  0

                                                                        1000


#null hypothesis
             0
             0
             0
             0

                                # To store the peak we get under the null
                                # To store the peak we get under the null
                                # To store the peak we get under the null
                                # To store the peak we get under the null

             len
    #print('length of Nsamps',nsamps)
    for   in range
```

```
                                    len                                 0
                                   len                                  0
                                    len                                 0
                                   len                                  0
                                    len                                 0
                                   len                                  0
                                    len                                 0
                                   len                                  0


        #need to find the index where this happens and dvide by fs and mult by 1000 to get the tim









        if
                        1

        if
                        1

        if
                        1

        if
                        1




    return


# for i in range(0,Session):
#     for j in range(0,Trial):

                                      #Average Filtered target epochs for a single electrode
                                            # Average Filtered non target epochs for a single elect
#goodtargetSingleEEG = Targetlist[:][EEG,:]#Filtered target epochs for a single electrode
#goodnontargetSingleEEG = nonTargetlist[:,EEG,:] #Filtered non target epochs for a single electrod
```

```
print 'B2: The Pvalue of channel Fz for subject'          ' is'
print 'B3: the Values for Cz, Pz, and Oz are'
print 'B3: The peak height occurs at '                'ms for channel Fz,'                'ms for


def PCA_EEG

                    0 1 2 3

    # Do PCA to get 2 dimensions
    for   in range 0 len
        #print('this first loop is running?')
        #for j in range(0,3):

                                                        for     in


                          2


    for   in range 0 len
                                                                for     in


    #pc2 = PCA(n_components=2)
    #pc2.fit(CombnonTargets)


    return
```

```
#need time window of .25-.45 ish


          int  25
           int  5




def avgEpochs
    #instead of 280 total epochs, we would simply have 28 avg epochs to perform PCA on.
           round len
           round len
           round len

           round len
           round len
           round len

    #print(tk10,tk25,tk50,nk10,nk25,nk50)



    for   in range 0 len
        if          0 and     0
                                                          0


        else

        if          0 and     0
                                                          0


        else

        if          0 and     0
                                                          0
```

```
        else


    for   in range 0 len
        if          0 and     0
                                                                0


        else

        if          0 and     0
                                                                0


        else

        if          0 and     0
                                                                0


        else


    return




      10
      25
      50


def PCA_K
```

```
                                2

                                2

                                2

    #k10npc = PCA(n_components=2)
    #k10npc.fit(k10nontargetepoch)


    #k25npc = PCA(n_components=2)
    #k25npc.fit(k25nontargetepoch)


    #k50npc = PCA(n_components=2)
    #k50npc.fit(k50nontargetepoch)



    return









def ROC
    #Thits = 0
    for   in range 0 8
                   130     18
        #print(xbound)
                   0
                 0

        for   in range 0 len
```

```
            # if Target_pck10[i][0] > xbound:
            #     Thits =+1
            if                  0
                          1
            if                      0
                        1
            # if nonTarget_pck10[i][0] > xbound:
            #     Nmisses =+ 1
        # print(Tmisses)
        # print(Nhits)

                    len                             len
                    len                              len                    #x axis

    return

         0




##PLOTS
#time = np.arange(0,len(NewData[1])/fs,1/fs)
              0 1 1

#ROC Curve

                    'o-'
            'Specificity'
            'Sensitivity'
           'ROC curve of K10 Target Detection'



#1st epoch
# plt.figure()
# for i in range(0,data.shape[0]):
#     plt.plot(time[820:820+2048],data[0][820:+2048])
# plt.xlabel('Time (sec)')
# plt.ylabel(u'\u03bcV')
# plt.title('EEG 1st Epoch',fontweight='bold')

# #FilteredData
# plt.figure()
# for i in range(0,NewData.shape[0]):
#     plt.plot(time[820:820+2048],FilterData[i][820:820+2048]) #Plot the signal
# plt.xlabel('Time (sec)')
# plt.ylabel(u'\u03bcV')
# plt.title('EEG Filtered Data',fontweight='bold')

#TARGET AND NONTARGET DATA partA
```

```
                                        "Target ("   str                " trials)"  #Plot the signa
                                           "Non-Target ("   str                    " trials)"  #Plot
           'Time (sec)'
           'Average Response 'u'\u03bcV'
          'PartA2: Subject ' str           ', EEG31 Target vs Nontargets'              'bold'


#Difference of Target and nonTarget P300

#for i in range(0,32):
                          30                   30           "EEG31" #+str(i+1)) #Plot the signal
           'Time (sec)'
           'Average Response 'u'\u03bcV'
          'PartA3: Subject ' str           ', P300 Response'              'bold'



#100 NULLPERMUTATIONS part b2

for   in range 0 1000
                             'k'  #Plot the signal) #Plot the signal
                         'r'             10          "Actual"  #Plot the signal
           'Time (sec)'
           'Average Response 'u'\u03bcV'
          'PartB2: Subject ' str           ', EEG31 Null Examples vs Actual response'


# #Full Value PCA
# plt.figure()
# for i in range(0,len(nonTarget4EEG)):
#     plt.plot(nonTarget_pc[i,0],nonTarget_pc[i,1],'b.')
# for i in range(0,len(Target4EEG)):
#     plt.plot(Target_pc[i,0],Target_pc[i,1],'r.')
# plt.plot([],[],'b.',label = 'NonTarget')
# plt.plot([],[],'r.',label = 'Target')
# plt.xlabel('PC1')
# plt.ylabel('PC2')
# plt.legend()
# plt.title('FullPCA')

# #k10 PCA
# plt.figure()
# for i in range(0,len(nonTarget_pck10)):
#     plt.plot(nonTarget_pck10[i,0],nonTarget_pck10[i,1],'b.')
# for i in range(0,len(Target_pck10)):
#     plt.plot(Target_pck10[i,0],Target_pck10[i,1],'r.')
# plt.plot([],[],'b.',label = 'NonTarget')
# plt.plot([],[],'r.',label = 'Target')
# plt.xlabel('PC1')
# plt.ylabel('PC2')
# plt.legend()
# plt.title('k10 PCA')
```

```
# #k25 PCA
# plt.figure()
# for i in range(0,len(nonTarget_pck25)):
#     plt.plot(nonTarget_pck25[i,0],nonTarget_pck25[i,1],'b.')
# for i in range(0,len(Target_pck25)):
#     plt.plot(Target_pck25[i,0],Target_pck25[i,1],'r.')
# plt.plot([],[],'b.',label = 'NonTarget')
# plt.plot([],[],'r.',label = 'Target')
# plt.xlabel('PC1')
# plt.ylabel('PC2')
# plt.legend()
# plt.title('k25 PCA')

# #k50 PCA
# plt.figure()
# for i in range(0,len(nonTarget_pck50)):
#     plt.plot(nonTarget_pck50[i,0],nonTarget_pck50[i,1],'b.')
# for i in range(0,len(Target_pck50)):
#     plt.plot(Target_pck50[i,0],Target_pck50[i,1],'r.')
# plt.plot([],[],'b.',label = 'NonTarget')
# plt.plot([],[],'r.',label = 'Target')
# plt.xlabel('PC1')
# plt.ylabel('PC2')
# plt.legend()
# plt.title('k50 PCA')

# plt[2].plot(t,constantDifCz, 'r',linewidth = 10, label = "Actual") #Plot the signal
# for i in range(0,100):
#     plt[2].plot(t,difvalsCz[i], 'k') #Plot the signal) #Plot the signal
# #plt[2].xlabel('Time (sec)')
# #plt[2].ylabel('Average Response'u'\u03bcV')
# plt[2].title('PartB1: Subject '+str(Subjects)+', EEG32 Null Examples vs Actual response',fontwei
# plt[2].legend()
# plt[3].plot(t,constantDifPz, 'r',linewidth = 10, label = "Actual") #Plot the signal
# for i in range(0,100):
#     plt[3].plot(t,difvalsPz[i], 'k') #Plot the signal) #Plot the signal
# #plt[3].xlabel('Time (sec)')
# #plt[3].ylabel('Average Response'u'\u03bcV')
# plt[3].title('PartB1: Subject '+str(Subjects)+', EEG13 Null Examples vs Actual response',fontwei
# plt[3].legend()
# plt[4].plot(t,constantDifOz, 'r',linewidth = 10, label = "Actual") #Plot the signal
# for i in range(0,100):
#     plt[4].plot(t,difvalsOz[i], 'k') #Plot the signal) #Plot the signal
# #plt[4].xlabel('Time (sec)')
# #plt[4].ylabel('Average Response'u'\u03bcV')
# plt[4].tile('PartB1: Subject '+str(Subjects)+', EEG16 Null Examples vs Actual response',fontweig
# plt[4].legend()
#print(nontargetcount,'nontarget',targetcount,'target')
# plt.figure()
# for i in range(0,NewData.shape[0]):
#     plt.plot(t,epochs[i]) #Plot the signal
# plt.xlabel('Time (sec)')
# plt.ylabel(u'\u03bcV')
# plt.title('EEG All Epochs',fontweight='bold')
```

```
# plt.figure()
# for i in range(0,NewData.shape[0]):
#     plt.plot(t,targetData[i]) #Plot the signal
# plt.xlabel('Time (sec)')
# plt.ylabel(u'\u03bcV')
# plt.title('Single Trial EEG Epochs Target',fontweight='bold')

# plt.figure()
# for i in range(0,NewData.shape[0]):
#     plt.plot(t,nontargetData[i]) #Plot the signal
# plt.xlabel('Time (sec)')
# plt.ylabel(u'\u03bcV')
# plt.title('Single Trial EEG Non-Target Epochs',fontweight='bold')


# #Filter freq domain
# # [w, hf_filter] = signal.freqz(h_filter,a=1,fs=fs, worN=h_filter.size)
# # plt.figure()
# # plt.plot(w.squeeze(),abs(hf_filter))
# # plt.xlim([0,13])
# # plt.xlabel('Frequency (Hz)',fontsize=12,)
# # plt.ylabel('Magnitude',fontsize=12)
# # plt.title('H(f)',fontsize=15,fontweight='bold')
# # plt.show()

# #for i in range(34): #plots all data lines
# #    plt.plot(data[i,:], '.')

# #plotting Pvalues of all 24 trials
# z = [0.004,0.01,.034,.019,.003,.022,0,0,0,.541,.012,.092,.017,.28,.001,.008,.002,0,0.001,0,0,0,0
# y = [0.986,0.048,0.755,0.678,0.07,0.071,0.529,0.79,0.518,0.794,0.638,0.325,0.157,0.094,0.205,0.4
# x = np.arange(1,25)
# plt.figure()
# plt.plot(x,y)
# plt.title('Pvalues for all 24 trials of Subject 4')
# plt.xlabel('trial number')
# plt.ylabel('p300 Pvalue ')
# avgpval4 = np.mean(y)
# print(avgpval4)
# avgpval6 = np.mean(z)
# print('sub 6',avgpval6)
```