2019ws-BCIIL-Sheet05-Solution

November 27, 2019

1 BCI-IL - Exercise Sheet #05

Sample solution

[1]:
```
import numpy as np
import scipy as sp
import scipy.signal
from matplotlib import pyplot as plt

import bci_minitoolbox as bci
```

1.1 Loading the data

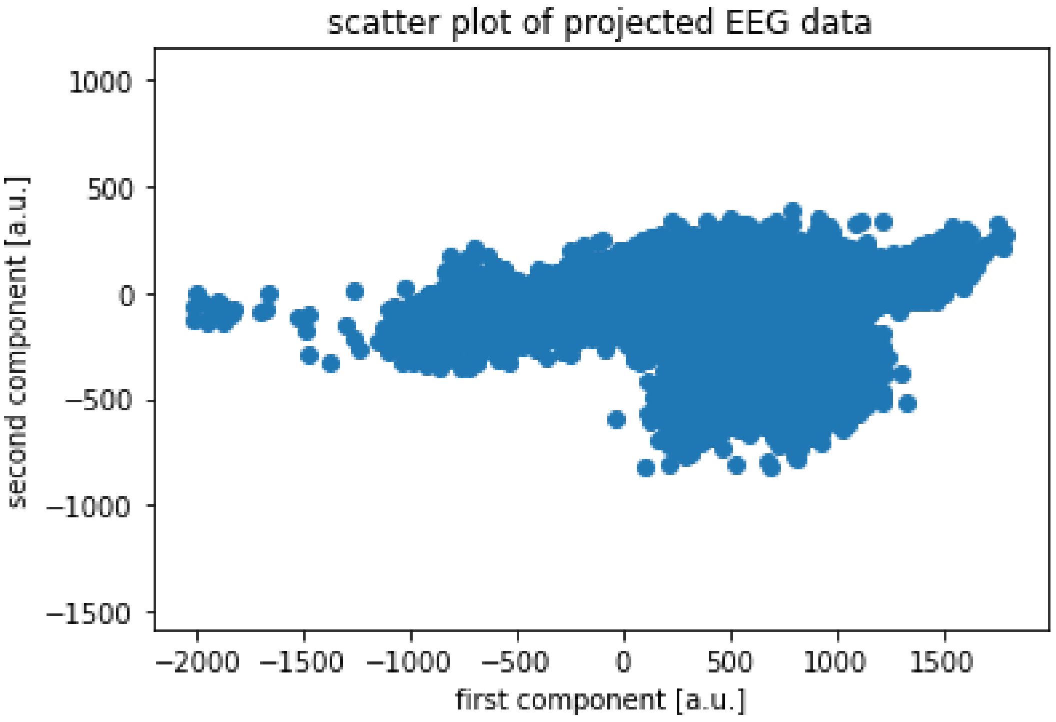
[2]:
```
fname= 'eyes_closed_VPal.npz'
cnt, fs, clab, mnt = bci.load_data(fname)
```

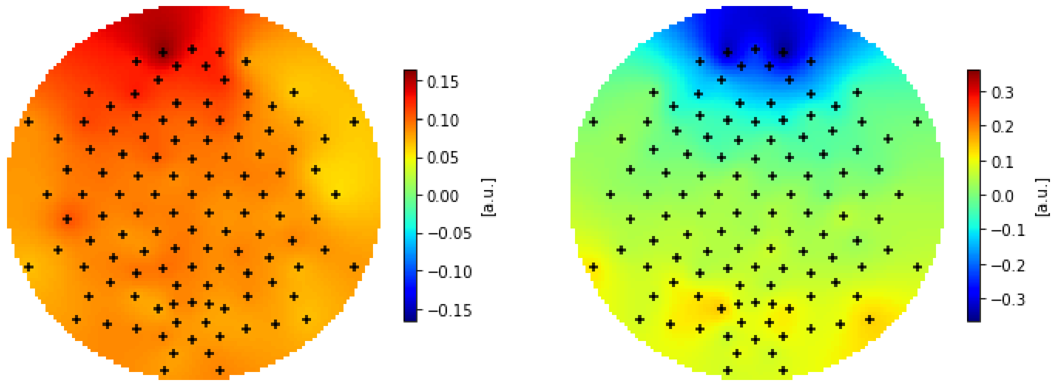
1.2 Exercise 1: PCA on raw data (3 points)

Make a scatter plot of the data with the two directions of largest variance as coordinate axes. Then, depcit the projection vectors of those two components as scalp maps (function `scalpmap` provided in the `bbci_minitoolbox`).

[3]:
```
C= np.cov(cnt)
d, V = np.linalg.eigh(C)

# -> Two components explain most of the variance.
# The std of those is above of what we would expect for brain sources.
plt.figure()
plt.scatter(cnt.T.dot(V[:,-1]),cnt.T.dot(V[:,-2]))
plt.title('scatter plot of projected EEG data')
plt.axis('equal')
plt.xlabel('first component [a.u.]')
plt.ylabel('second component [a.u.]')

plt.figure(figsize=(12, 6))
plt.subplot(1,2,1)
bci.scalpmap(mnt, V[:,-1], clim='sym', cb_label='[a.u.]')
```

```
plt.subplot(1,2,2)
bci.scalpmap(mnt, V[:,-2], clim='sym', cb_label='[a.u.]')
```



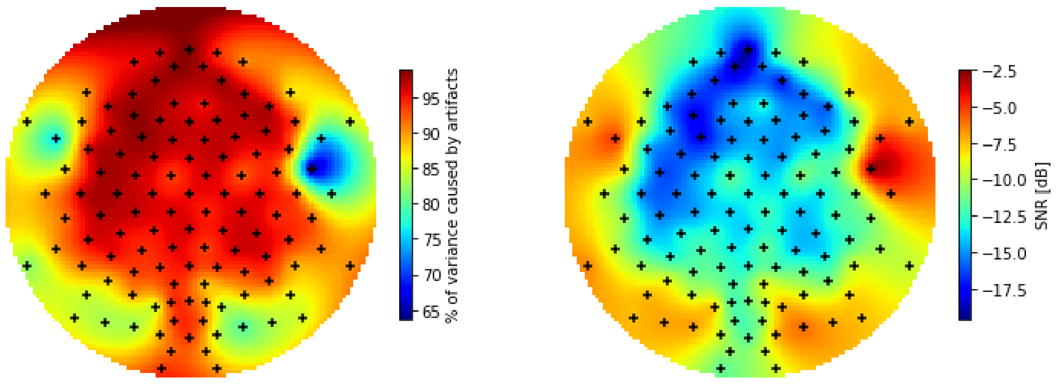


1.3 Exercise 2: Artifact to signal ratio with PCA (5 points)

For this task we assume that the two components from Ex. #01 reflect eye movements, while all other components do not contain artifacts from eye movement. If you did not succeed with Ex. #01, chose an arbitrary component.

Determine for each channel which proportion of the overall variance is caused by eye movements and plot this information as a scalp map. Also, calculate the Signal-To-Noise ratio (SNR) in Decibel (dB).

[4]:
```
D=np.diag(d)
var_rest =np.diag(V[:,:-2]@D[:-2,:-2]@V[:,:-2].T)
var_eyes =np.diag(V[:,-2:]@D[-2:,-2:]@V[:,-2:].T)

plt.figure(figsize=(12, 6))
plt.subplot(1,2,1)
ratio = 100 * var_eyes / np.diag(C)
bci.scalpmap(mnt, ratio, cb_label='% of variance caused by artifacts')

plt.subplot(1,2,2)
ratio = 10 * np.log10(var_rest/var_eyes)
bci.scalpmap(mnt, ratio, cb_label='SNR [dB]')
```



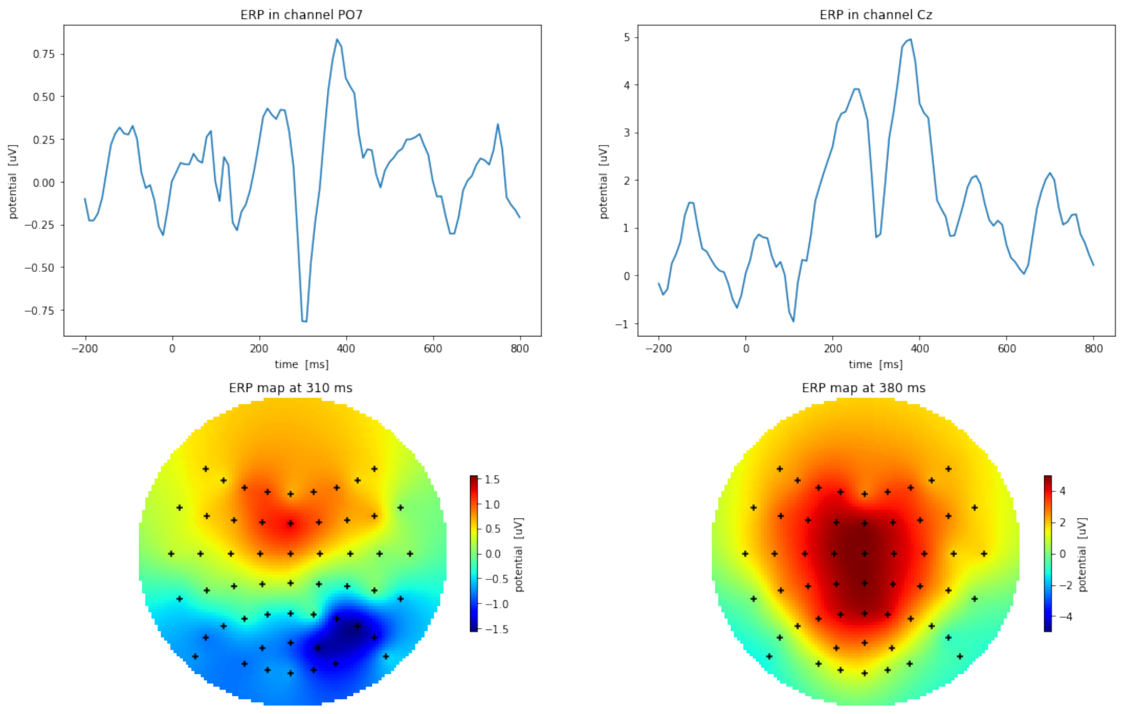
1.4 Loading the data

[5]:
```
fname = 'erp_hexVPsag.npz'
cnt, fs, clab, mnt, mrk_pos, mrk_class, mrk_className = bci.load_data(fname)
```

1.5 Exercise 3: Artificial EEG data (7 points)

Generate one trial of artificial, stereotypical EEG data (1000 ms, 55 channels) out of the data set of sheet #01. The trial should contain a 'clean' target ERP composed of an N2 component (the one negatively peaking at 310 ms in the data on sheet #01) and a P3 component (the one peaking at 380 ms in the data on sheet #01). Both components should have their typical spatial distribution. To this extent, extract the corresponding scalp patterns at the peaks of the average ERPs, calculate the filters, use them to isolate the components from the average ERP and then project them back into the EEG space. Plot the artificial EEG (the backprojected ERP) in channels PO7 and Cz and the scalp patterns correpsonding to the N2 and P3.

[6]:
```python
# Define channels and timepoints to extract N2 and P3
chans = ['PO7', 'Cz']
timepoints = [310, 380]

# Determine target ERP
epo, epo_t = bci.makeepochs(cnt, fs, mrk_pos, [-200, 800])
epo = bci.baseline(epo, epo_t, [-100, 0])
erp = np.mean(epo[:,:,mrk_class==0], axis=2)

chidx = [clab.index(chans[i]) for i in range(len(chans))]
tidx = [np.argmin(np.abs(epo_t-timepoints[i])).astype(int) for i in␣
 ↪range(len(timepoints))]

# Determine patterns and filters (forward and backward model) for N2 and P3␣
 ↪component
# Remark: Here we make the simplifying assumption that the components are␣
 ↪uncorrelated (ignore cov_s)
Cinv = np.linalg.pinv(np.cov(cnt))
N2pattern = erp[tidx[0],:]
N2filter = N2pattern.dot(Cinv)
P3pattern = erp[tidx[1],:]
P3filter = P3pattern.dot(Cinv)

# Compose signal from the two components by backward-forward projection
stereoEEG = N2filter.dot(erp.T)[:,np.newaxis].dot(N2pattern[np.newaxis,:]) +␣
 ↪P3filter.dot(erp.T)[:,np.newaxis].dot(P3pattern[np.newaxis,:]);

plt.figure(figsize=[18, 12])
for i, chan in enumerate(chans):
    plt.subplot(2, len(chans), i+1)
    chidx = clab.index(chan)
    plt.plot(epo_t, stereoEEG[:, chidx])
    plt.title('ERP in channel {}'.format(chan))
    plt.xlabel('time  [ms]')
    plt.ylabel('potential  [uV]')

for i in range(len(tidx)):
    plt.subplot(2, len(chans), i+3)
    scalp = stereoEEG[tidx[i].astype(int), :]
    bci.scalpmap(mnt, scalp, clim='sym', cb_label='potential  [uV]')
    plt.title('ERP map at {} ms'.format(timepoints[i]))
```


ERP in channel PO7
potential [uV]
time [ms]
ERP in channel Cz
potential [uV]
time [ms]
ERP map at 310 ms
potential [uV]
ERP map at 380 ms
potential [uV]