Seminar Hands-On BCI implementation

Session 4: Feature preconditioning and extraction (This session focusses on PCA)

Load data file to workspace (results from session 3)

```
load ecogStruct3.mat
```

Feature preconditioning

Number of trials with finger movement

```
nTrials = size(ecog.periodogram.periodogram,3);
```

Frequency features

```
freqBand = [4:58 62:118 122:178]; % the desired frequencies
freqIdx = unique(nearly(freqBand,ecog.periodogram.centerFrequency)); % finding the closest center frequencies of the periodogram
nFreq = length(freqIdx);
```

Channel features

```
chan = ecog.selectedChannels; % here we make sure that we exclude the bad channels that were identified in Session 2
nChan = length(chan);
```

```
% grab the data
dat = ecog.periodogram.periodogram(freqIdx,chan,:);
```

Task 1 (2 pt):

z-scoring

Prepare data for z-scoring

Reshape to nFreq x nChan * nTrials

dat =

z-score data along the 2nd dimension

dat =

Reshape data to nFreq x nChan x nTrials

dat =

Permute data to nTrials x nChan x nFreq

dat =

Reshape data to nTrials x nChan * nFreq

dat =

```
save zScoredData.mat dat nFreq nChan nTrials
```

Task 2 (2 pt):

Why do we z-score the data?

What does our feature Vector consist of? How many features are there at the moment?

Task 3 (2 pt):

Principal component analysis

(For help, you can check out "PCA-Tutorial-Intuition_jp" that you can find on StudIP in the folder of Presentation 4)

subtract the mean of each feature

```
dat = dat -
```

Calculate the covariance matrix

```
cPCA =
```

calculate the Eigenvalue decomposition

```
[v,d] =
```

extract diagonal of matrix d as vector

```
dVector =
```

sort the results in decreasing order

```
[a, indices] = sort(-1*dVector);
dVector = dVector(indices);
v = v(:,indices);
```

TASK 4 (2 pt):

What to the matrices v and d contain? How can you calculate the proportion of variance explained by each principle component? How much variance does the first / do the first 100 principle components explain?

We can transform the original data to the PC coordinate system by multiplying with v.

```
xPCA = dat * v;
```

In case we want the old data back we do this by multiplying the PC time series representation with the inverse of v.
Remark: Because v is orthonormal inv(v) = v'
Remark: That's the underlying assumption that the observed pattern is generated by an additive superposition of time varying principal patterns (or sources)

```
datReconFull = xPCA * v';
```

In case we want to drop some components we can do this by setting these principal components (i.e. columns in v) to zero.

```
subV = v;
removedPCs = [];
subV(:,removedPCs) = 0;
datReconPartial = xPCA * subV';
```

Visualization of results

```
figure;
shownTrials = 1:50; % some trials (for reasons of better visualization, not
all are shown)
shownFeatures = 1:nFreq*3; % features from three channels (for reasons of
better visualization, not all are shown)
subplot(3,1,1)
imagesc(dat(shownTrials,shownFeatures))
xlabel('Features','Fontsize',18)
ylabel('Trials','Fontsize',18)
title ('original data')
subplot(3,1,2)
imagesc(datReconFull(shownTrials,shownFeatures))
xlabel('Features','Fontsize',18)
ylabel('Trials','Fontsize',18)
title ('reconstructed data, full')
subplot(3,1,3)
imagesc(datReconPartial(shownTrials,shownFeatures))
xlabel('Features','Fontsize',18)
ylabel('Trials','Fontsize',18)
title ('reconstructed data, partial')
```

TASK 5 (1 pt):

How do these three subplots relate to each other at the moment?
Is this expected?

The results of this PCA will be used next session,
so save the results of your PCA for later use.

```
save resultsPCA.mat xPCA v d datReconPartial
```