Conducting Principle Component Analysis (PCA) in Matlab
or
My Journey Towards a Better Understanding of Linear Algebra

Janina Böcher
KAAP-629; Fall 2021

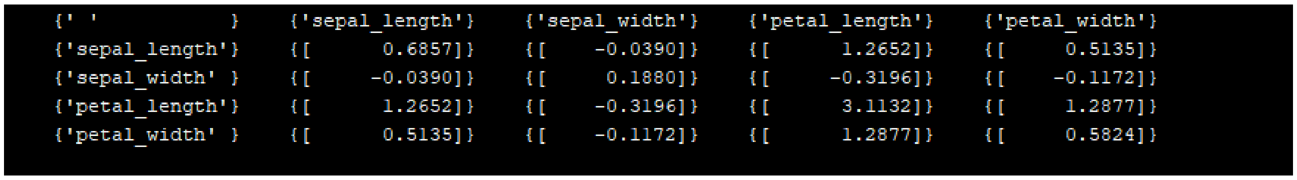
1. Project Motivation

The goal of this project was to gain a deeper understanding of principle component analysis (PCA) by implementing it programmatically in Matlab. PCA is a statistical tool commonly used in the field of electroencephalogram (EEG) research. It is used to identify the electrodes (spatial distribution) and the time points (temporal distribution/time window) that drive the effect of an experimental manipulation. The underlying assumption is that the variable with most variance will be the most significant principle component (PC) and contain the event-related potential (ERP)/experimental effect. ERPs constitute changes in electric current over the scalp in response to a sensory stimulus, such as an auditorily presented speech syllable or a written word flashed onto a screen. EEG measures ERPs by collecting data from sensors (electrodes) that are placed on the scalp. The resulting data consists of voltage measures from all electrodes, at each time point over multiple trials.

2. Covariance Matrices

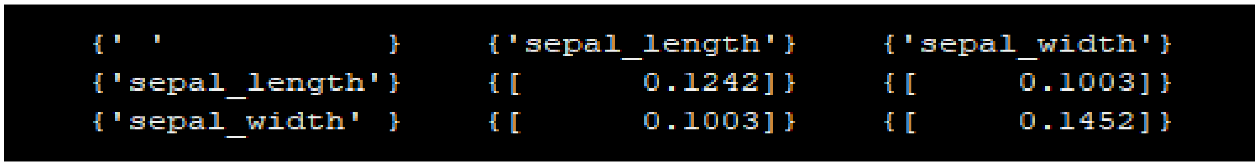
The first step of PCA is to construct a covariance matrix of the raw data. This involves calculating covariance values of all possible variable combinations in a dataset. The covariance of a variable with itself is the variance of that variable. The resulting covariance matrix is symmetric with the variance values along the diagonal. Hence, diagonalizing the covariance matrix results in a representation with no covariance, which assures that the new variables are independent/orthogonal and the ones with the biggest variance are the principle components.

PlantData.m contains a demonstration of a covariance matrix derived from a freely available dataset documenting sepal width and length, and petal width and length of different iris varieties (Fisher, 1936; https://archive.ics.uci.edu/ml/datasets/iris). The data is imported from an excel sheet (plantData.xlsx). Covariances and variances are calculated using the standard formulas.



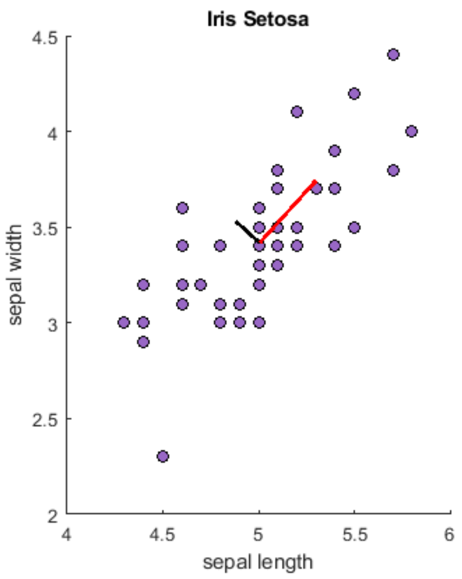
```
{' '           }    {'sepal_length'}    {'sepal_width'}    {'petal_length'}    {'petal_width'}
{'sepal_length'}    {[      0.6857]}    {[   -0.0390]}    {[      1.2652]}    {[     0.5135]}
{'sepal_width' }    {[     -0.0390]}    {[    0.1880]}    {[     -0.3196]}    {[    -0.1172]}
{'petal_length'}    {[      1.2652]}    {[   -0.3196]}    {[      3.1132]}    {[     1.2877]}
{'petal_width' }    {[      0.5135]}    {[   -0.1172]}    {[      1.2877]}    {[     0.5824]}
```

To visualize the covariance matrix in two-dimensional space, two variables (‘sepal length’ and ‘sepal width’) were chosen for plotting. A 2x2 covariance matrix was constructed based on the first two variables of the dataset.



```
{' '               }    {'sepal_length'}    {'sepal_width'}
{'sepal_length'}    {[        0.1242]}    {[      0.1003]}
{'sepal_width' }    {[        0.1003]}    {[      0.1452]}
```

As a sanity check, the result of the manually calculated covariance matrix were compared to the output of the Matlab built-in function cov( ). The eigenvectors that capture this data are obtained using [v,d] = eig( ) and scaled by their respective eigenvalues.



The eigenvalues of a covariance matrix represent the magnitude of the spread of the data along the eigenvectors. The eigenvectors show the direction of the data spread between variable 1 (sepal length) and variable 2 (sepal width). The eigenvectors are the principle components of this 2x2 covariance matrix. The first principle component accounts for most of the variance observed in the data, as indicated by the largest corresponding eigenvalue. The second component is perpendicular to the first one, because they are uncorrelated. Component 1 pulls the data in the direction of eigenvector 1 and variables 1 (x-axis) and 2 (y-axis) contribute as much as indicated by the eigenvector coordinates (normalized).

Because there are currently only two variables that influence the data, it is not possible to have more than two principle components to explain the variance. Datasets with many variables may be represented by fewer principle components than there are variables. This is desirable because it serves the purpose of dimension reduction. The more related variables can be summarized as a single eigenvector/principle component, the more interpretable the data will become.

[*Material for this section in folder “CovarianceMatrix”]

2.2 Calculating Eigenvalues and Eigenvectors

Obtaining eigenvalues and vectors is straightforward using the eig( ) function. To calculate them manually, the characteristic equation |M - $\lambda * I$| can be used. Solving this function for $\lambda_{1,2}$ yields the eigenvalues.

$$|\mathrm{M} - \lambda * I| = \begin{bmatrix} 0.1242 & 0.1003 \\ 0.1003 & 0.1452 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} = 0$$

$$(=) \quad \begin{bmatrix} (0.1242-\lambda) & 0.1003 \\ 0.1003 & (0.1452-\lambda) \end{bmatrix} = \lambda^2 - 0.6693\,\lambda + 0.0892952 = 0$$

$$\lambda_{1,2} = 0.1840, 0.4853$$

Eigenvectors are not rotated as a result of multiplication with their eigenvalue λ. Rather, they remain on the same 'line' and are either stretched or squished by some factor. In other words, these vectors end up being a multiplication of themselves after the eigenvalue has been applied. For eigenvectors, the following is true:

$$\vec{v} \xrightarrow{transformation} \overrightarrow{v'} \qquad = \qquad \vec{v} \xrightarrow{transformation} \lambda\vec{v}$$

λ is the factor by which the vector is stretched. The factor by which an eigenvector is stretched as a result of the transformation matrix is called the eigenvalue of the eigenvector. So, the formula above corresponds to: M$\vec{v}$ = $\lambda\vec{v}$. This formula can be used to find the eigenvectors.

$$\mathrm{M}\vec{v} = \lambda\vec{v}$$

$$(M - \overline{v}\,) * \vec{v} = 0$$

$$\begin{bmatrix} (0.1242-\lambda) & 0.1003 \\ 0.1003 & (0.1452-\lambda) \end{bmatrix} * \vec{v} = 0$$

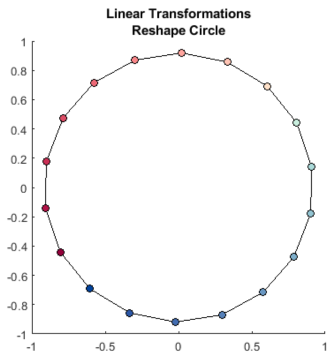
$$\begin{bmatrix} (0.1242-0.1840) & 0.1003 \\ 0.1003 & (0.1452-0.1840) \end{bmatrix} * \begin{bmatrix} \vec{v}_{1,1} \\ \vec{v}_{1,2} \end{bmatrix} = 0$$

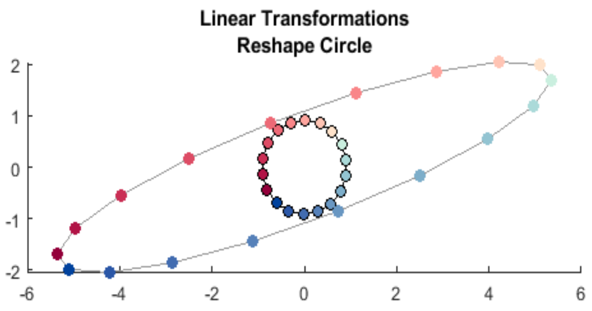
$$v_1 = \lambda_1 * \begin{bmatrix} -0.7429 \\ 0.6694 \end{bmatrix}$$

3. Linear Transformations

In order to express the data in a new way that minimizes redundancy and optimizes dimension reduction, PCA takes a data set (R^n) and transforms it to linear functions of the original data (R^m). Crucially, the features that are derived from the variables of the dataset are orthogonal. The covariance between orthogonal features is zero meaning that any redundancy (information that is captured by more than one variable) has been removed from the dataset. The projection into the new space involves multiplication of the covariance matrix of the data and the eigenvectors of the covariance matrix. The original data points and the transformed data points are the same, however, the x and y axes no longer represent variable 1 and 2, but principle component 1 and 2.

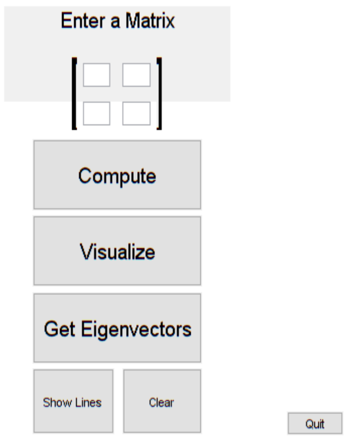
A GUI was created to illustrate the concept of linear transformations using 2x2 matrices. The initial points that make up a circle represent the original raw dataset. The points are vector heads that are plotted from the origin. The user enters a matrix that projects the points onto a different space.



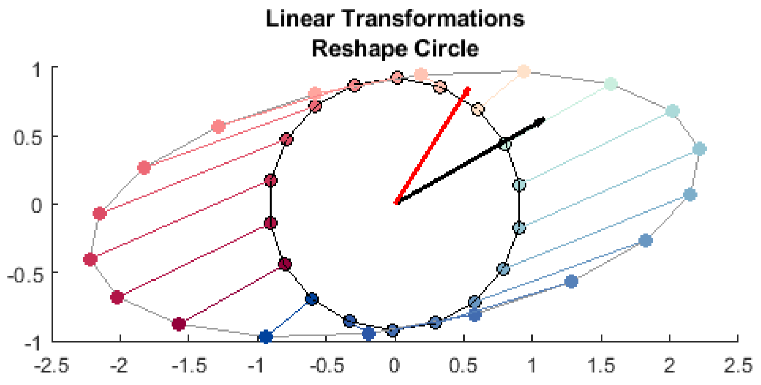


Transformation Matrix = [5 3; 1 2]

GUI Functionality:



Enter a Matrix: The user enters any 2x2 matrix to transform the points of the circle.
Compute: Multiplies the original points (p) with the entered matrix to obtain the transformed points (p'). A new window "SUCCESS" will pop up once the code has run.
Visualize: Plots the transformed points into the first figure.
Get Eigenvectors: Draws scaled eigenvectors into the first figure. A new window will pop up that contains the eigenvalues and eigenvectors.
Show Lines: Plots lines between the original data point and its transformed version.
Clear: Clears axes and displays original circle of points.
Quit: Closes GUI and figures.



As mentioned in section 2, eigenvectors are those vectors that do not get rotated as a result of multiplication with a transformation matrix. The image on the left illustrates that the light greenish point corresponds to the eigenvector. Its transformed counterpart is stretched by the eigenvalue, but not rotated.

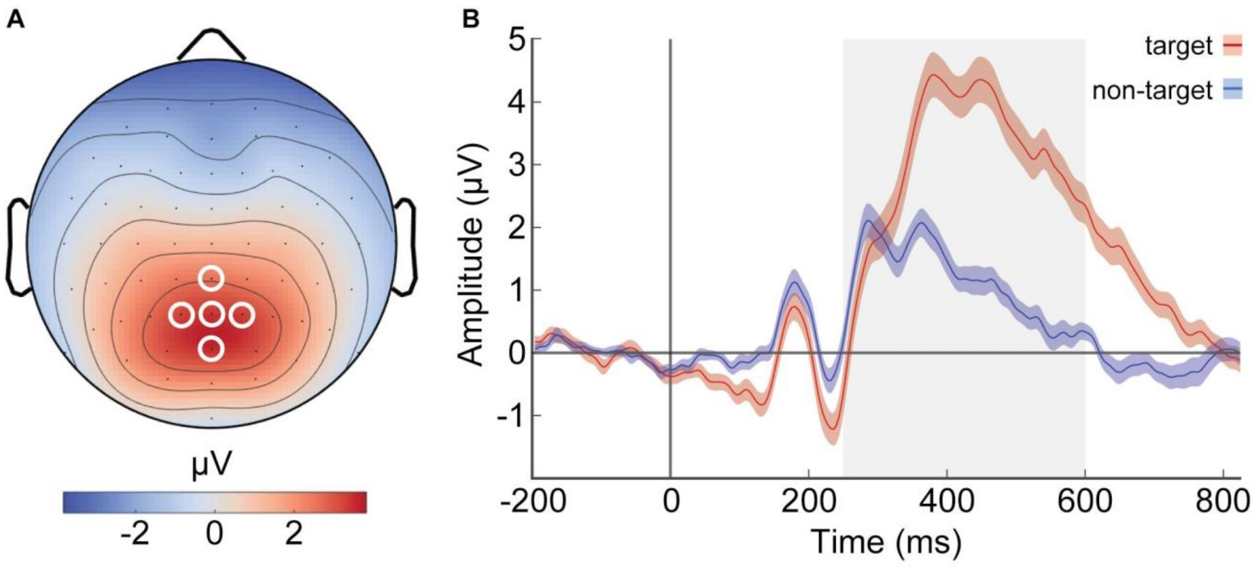
[*Material for this section in folder "LinearTransformation"]

4. Principle Component Analysis

The previous sections laid out the groundwork of PCA. Now I will try to apply these concepts to a real-world problem that I may encounter in my area of research. To this end, an example EEG dataset was obtained and used for analysis (https://hal.archives-ouvertes.fr/hal-02173958/document). The experiment involved subjects playing a Brain-Computer Interface game named Brain Invaders. The goal of this game is to detect a rare alien amongst common aliens. The structure of the game/experiment follows an oddball paradigm, such that the rare alien (target) and the frequent aliens (non-target) are presented at a 1:35 ratio. The stimuli were pseudo-randomized to elicit a P300 response, an ERP appearing at about 300ms post stimulus-onset. Data was recorded from 32 active electrodes.

The purpose of using PCA here is to identify the electrodes and the time points that drive the experimental effect. The a priori hypothesis is that the expected ERP will occur during a time window around 300 ms (B) and will have a central-posterior topographical distribution (A).

[Figure taken from Herzog, N. D., Steinfath, T. P., Tarrasch, R. (2021). Critical Dynamics in Spontaneous Resting-State Oscillations Are Associated With the Attention-Related P300 ERP in a Go/Nogo Task. *Frontiers in Neuroscience*, 15.]

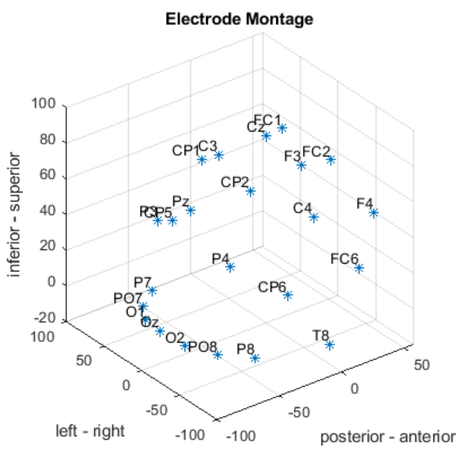


https://www.frontiersin.org/article/10.3389/fnins.2021.632922

4.2 Plotting the Electrode Montage

I began by plotting the electrode montage in 3D space to illustrate how the sensors were distributed across the scalp. Since the dataset did not contain coordinates of the channel locations, I assigned them based on data I've used previously. Unfortunately, I did not find the *x,y,z*-coordinates for all electrodes used in the current study. Hence, I plotted only 23 out of 32 channels. The channels that were used were FP1, FP2, AFz, F7, F3, F4, F8, FC5, FC1, FC2, FC6, T7, C3, Cz, C4, T8, CP5, CP1, CP2, CP6, P7, P3, Pz, P4, P8, PO7, O1, Oz, O2, PO8, PO9 and PO10. I saved the electrode labels with their coordinates in a .mat file (chan_locs.mat).

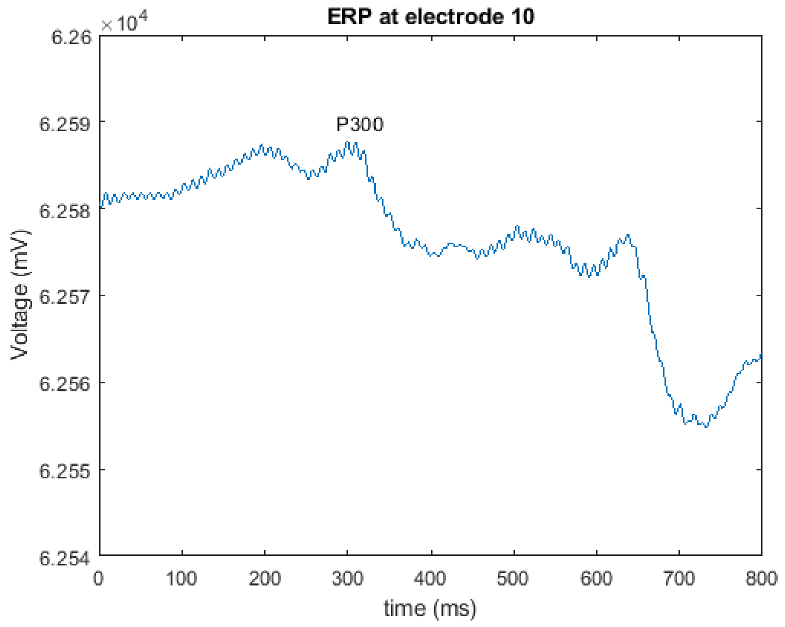
Next, I extracted a subset of the data provided by the website to conduct further analysis. I decided to focus on just one subject and one condition. According to the data, the sampling rate was 512 Hz and the total duration of the experimental block was 218 seconds (~ 3.5 min).



4.3 Calculating ERPs

ERPs are made up of the average across all time points and all trials at a single electrode (or the average across all electrodes and all trials at a single time point). I proceeded to assign the time points (rows in the datasheet) to their respective trials. A new trial was marked by the occurrence of a stimulus (number between 20 and 85) in column 35. I created a new column that assigns all time points of a trial to the same trial number. Since the presentation of the stimuli occurred non-regularly, the trials do not have equal length.

My goal is to create an average for each time point across all trials. To achieve that, I numbered the rows of each trial in such a way that, for example, the 5th time point of trial 1 and the 5th time point of trial 17 would have the same number despite occurring at totally different times of the overall experiment. This information was assigned to column 37 of data. Based on this step, I created a variable *avg_elec_time* that contains the average of each column of the rows that represent the same time point.



I plotted an example of an ERP to get a first impression of the data. The figure shows the average voltage measured at electrode 10 for the time from 0 (stim) to 800 ms. The P300 is a positive peak in the ERP waveform at around 300 ms. I guess we can kind of see that.

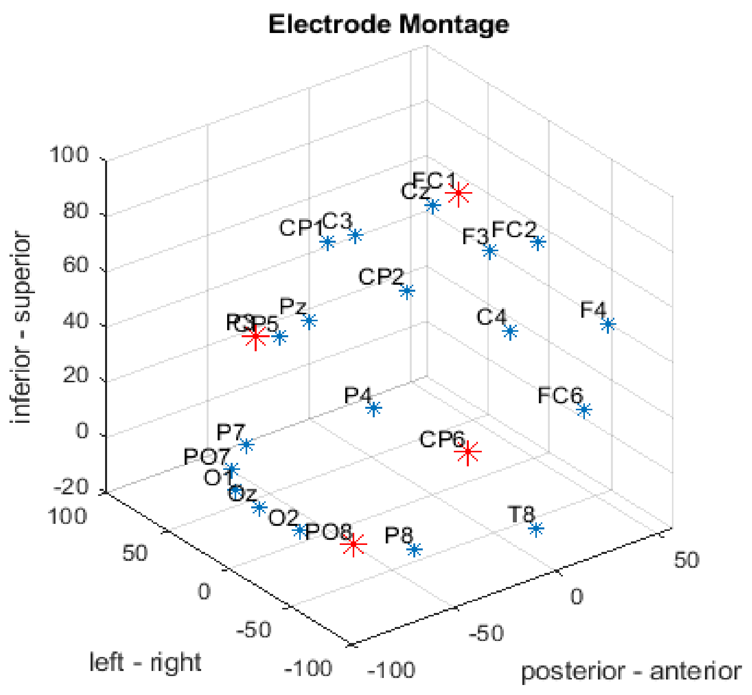
4.4 Conducting PCA

I suspect that the positive deflection of the waveform around 300 ms contains the effect of the experiment and that this effect will be most pronounced at central-posterior electrodes. However, eye-balling increases Type-II error and is therefore unreliable. Instead, I will use PCA to identify the electrodes that drive the effect.

First, I used the functions cov( ) and eig( ) to obtain the principle components of a small subset of the data and their coefficients. To double-check my understanding that the eigenvectors of the covariance matrix are the same as the principle components, I compared these results to the pca( ) output. PCVEC was the same as the eigenvectors and PCCOEFF was the same as the eigenvalues. I then proceeded to conduct PCA on all the data.

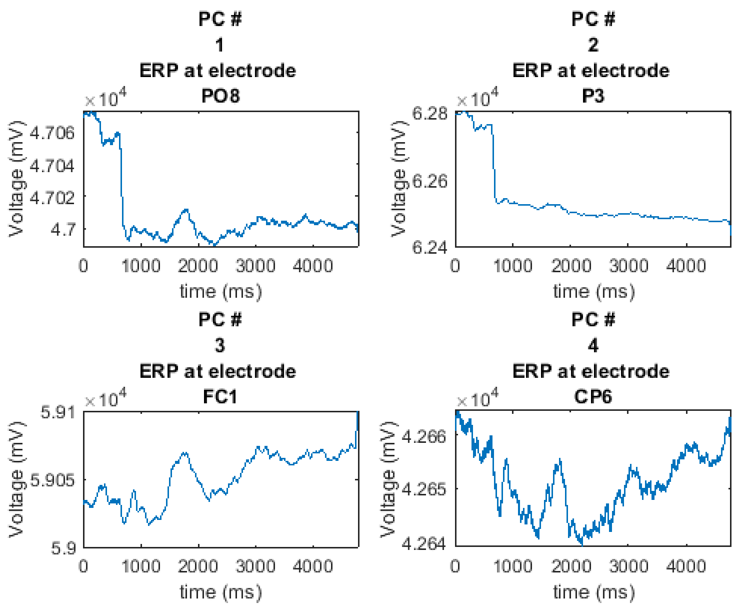
Lastly, it must be determined which electrodes (variables) contribute most to the largest principle components. Principle components are sorted by their eigenvalues. The principle component with the largest eigenvalue accounts for most of the variability in the data. To trace back which variables contributed the most to a pc, we again have to calculate covariances. This time, covariances are calculated between the principle component and each variable. The variable that is most like the pc (highest correlation) drives the effect the most.

The results indicate that the electrode contributing most to the first principle component is electrode PO8 (‘parietal-occipital’). Similarly, the second principle component is mostly driven by electrode P3 and pc3, and pc4 are made up most of electrodes FC1 (‘fronto-central’) and CP6 (‘central-parietal’). Those electrodes are highlighted in red in the figure below.



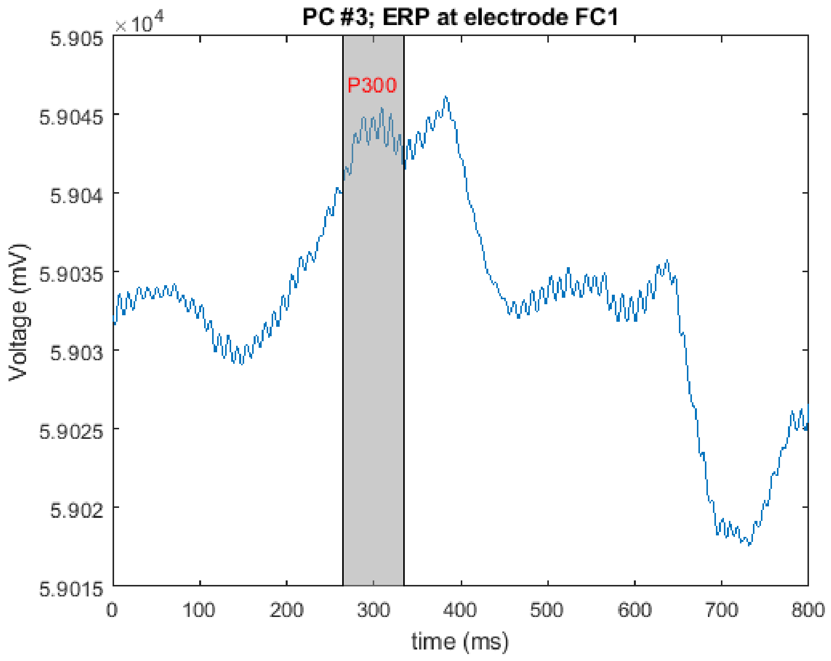
4.5 Identify the Relevant Principle Component

The final step is to figure out which of the principle components actually contains the P300 that we expected. Below are the waveforms associated with the electrodes that correspond to the four largest principle components.



Looking at the first figure, it seems that PCs 1 and 2 might mostly contain noise. There's a sharp drop at around 800 ms, which is probably what is represented in the PCs. I'm not sure what might have introduced noise immediately following stimulus presentation. Perhaps this is overlap from the preceding trial.

PC 3 and 4 seem to be more promising candidates. I adjusted the x-axis to show only the first 800 ms, as the effect we care about should occur at around 300 ms. There is a peak in the time between 200 and 400 ms in the waveform plot of FC1. FC1 is the electrode that correlates most with PC 3.



[*Material for this section in folder "PCA"]