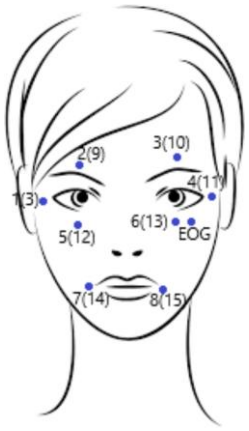
Coding exercise: EEG properties, artifacts and artifact removal

Goals

This exercise is designed to:

- Provide practical experience with the loading and interpretation of stored EEG and event data, as well as with the trial extraction procedure which is a prerequisite for any further analysis.
- Enable students to visualize sound and artifact-contaminated raw EEG signals and their spectra, and assess the effects of spatial filtering and artifact removal methods.

Description



You are given a dataset of a single subject consisting of 8 EEG, EMG, EOG and gyroscope files stored in BDF format. Each file corresponds to a single run/block of a protocol allowing the synchronous collection of the above types of data while the subject was idling or performing a number of "artifact" tasks, i.e., tasks that are known to generate activity that contaminates the EEG signal with artifactual components. Signals were collected at a sampling rate of 500 Hz. The EEG data were recorded on 31 locations on the scalp according to the 10-20 system. Additionally, data from 15 sensors were synchronously acquired; these include 1 EOG and 8 EMG sensors placed on the subject's face as shown in the figure on the right. Another 6 channels correspond to the x, y and z dimensions of an accelerator and a gyroscope sensor. The information of the label and index of each of these channels in the data matrix can be found after loading the headers of the BDF files in the struct field variable header.Channel.Label. All EEG signals were referenced to electrode CPz. The subject executed sequentially, in randomized order, several trials of the following artifact-generating tasks (event code in parenthesis): Resting/idling/baseline (783), Press electrode (261), Blinking (1081), Eye movements: Left-Right (1077), Up-Down (1078), Circular (1079), Random (1082), Head movements: Left-Right (1111), Up-Down (1113), Circular (1114), Random (1116), Speaking (1092), Swallowing (1093), Clenching teeth (1094), Frowning (1099), Eyebrow raise (1100). Each trial lasts 5 seconds, except for Resting/baseline trials that last 2 seconds. Each trial starts with code 768 and the trial task starts with a visual cue associated to the event code of the respective task (numbers in parentheses above). A trial-end event is also present and corresponds to the number 32768 + cue code of the respective trial. All events are delivered as instantaneous pulses (Dirac deltas) sampled synchronously to the EEG and

stored in the last Trigger/Status channel of the data matrix. Furthermore, an additional EEG file (in GDF format) is provided, where EEG data are recorded while a professional Formula E race driver is driving in a racing simulator. All material (neurophysiological data, auxiliary MATLAB scripts and toolboxes) can be downloaded from this link: Zenodo link

Tasks

1. Load one of the provided BDF files using the bdfmatlab toolbox. An EEG BDF file can be loaded with the MATLAB commands:

   header = readbdfheader(FILEPATH); % REPLACE "FILEPATH" WITH TRUE PATH STRING

   data = readbdfdata(header);

   Inspect the header and the data matrix and try to interpret all the variables that appear. What are the dimensions of the data matrix and why? How is this related to the sampling rate and EEG system used? Plot the trigger channel (how will you find its position in the data matrix?) and attempt to interpret the plot given the information about the experiment given above. Use the function gettriger to get a list of the positions of all events that have occurred:

   ```
   POS = gettrigger(data(:,end),1);
   ```

   How can you get, additionally, the type of each of these events?

2. Having solved the previous task, try now to proceed with a code snippet (hint: based on for-loops) that performs the "trial extraction" procedure. Your code should iterate through all available BDF files, segment the periods of interest (i.e., the resting and artifact trials/epochs) and store these in a single "Trials" variable (feel free to use naming conventions of your own) by concatenating the trial segmentation result of each run/file with the already built Trials variable. You can choose to use MATLAB cell arrays for the "Trials" variable or 3-dimensional matrices NxTxC, where N the number of retrieved trials, T the amount of time samples in a trial and C the number of channels. In the latter case, use different such variables for the Resting and the Artifact trials, as they will have different duration (2 s vs 5 s) and thus different number of samples (i.e., they cannot be stored in the same Trials 3D array). In any case, make sure you also use the event information in the trigger channel to save a 1D Labels array which points to the type of each of the N trials segmented.
3. Treat all trials with a high-pass Butterwirth filter with cut-off at 1 Hz and/or DC removal (i.e., for each channel, remove form all time samples the average potential of the trial on this channel). To set up and apply a high-pass filter, check the use and documentation of MATLAB commands **butter** and **filtfilt**. It is advisable to perform spectral filtering immediately after you load the data throughout the whole file, rather than individually for each trial, to avoid edge effects. You can perform DC removal either using MATLAB's **mean**, **repmat** and **permute** commands on the extracted trials, or by embedding this procedure in the trial extraction (i.e, store the trials without DC component directly using **mean**). DC removal will remove the DC frequency component ("bandpower at 0 Hz") and thus center the trials' waveform around 0, thus making comparisons of

trials in the raw signal more intuitive. For one or more artifact types and channels of your choice (hint: choose a channel that you estimate it could be seriously affected, e.g., a frontal channel for the "Blinking" artifact), plot the DC-removed signals of a single resting and a single "artifactual" trial. Can you tell the presence of artifacts in the raw signal? Is this true for all artifact types?

4. Repeat the previous test in a more systematic way, by computing grand averages of Resting and Artifact trials (separately for each artifact type). Plot these grand averages in a single figure. For which artifact classes and channels are the artifacts visible? How do they compare to the Resting class? Are the effects on all channels the same?

5. Using the grand average of each artifact class (and for the Resting class), acquire a single potential value per channel by further averaging across time. In order to not lose information of the magnitude of oscillations, take the absolute value of the signal before averaging (i.e., rectify with MATLAB command **abs**). Keeping only EEG channels, this process will result in a single 31-long vector (as many as the EEG channels recorded with this system), whose values show the average magnitude of the amplitude on this channel for the artifact type in question. You can visualize these vectors for each artifact to reveal the distribution of the influence of this artifact on the scalp channels with EEGLAB's topoplots with:

   load('chanlocs31Elvira.mat');

   topoplot(ARTIFACTVECTOR, locs, 'maplimits', [0 50]); colorbar;

   Inspect the resulting topographical distributions and infer whether all artifact tasks affect the EEG signal, and for which ones the implications seem to be more serious.

6. For a channel of your choice, compute the grand average spectrum for Resting and for each type of artifact. You can compute the EEG signal spectrum with a large variety of methods, the most classic of which is the Fast Fourier Transform (**fft** MATLAB command). For a more intuitive output, you are advised to use the Welch periodogram method to acquire the signal's power spectral density as:

   [psd, bands] = pwelch(signal, windowsize, overlapsize, DesiredFrequencies, SamplingRate);

   Type "help pwelch" in the MATLAB command prompt to access details for each of these arguments.

   Does the Resting class obey the "Power low" (1/f relationship between power and frequency band) of EEG? Can you detect the characteristic alpha-rhythms peak? Do the average spectra of the artifact-contaminated data violate these properties, and how? Do all artifacts affect the signal spectrum in the same way, and why?

7.Process the trial-extracted signal with the following methods:

- Common Average Reference spatial filtering. Implement this using MATLAB commands **mean** and **repmat**.
- Small Surface Laplacian spatial filtering. You can implement this filter with auxiliary functions given as follows:

  montage = channels2montage(ChannelLabels); % ChannelLabels is a cell array with the 31 % EEG channel labels in the order they appear in the data matrix

  laplacianMatrix = montage2laplacian(montage, 'all');

  FilteredTrial = Trial*laplacianMatrix;

  Attempt to interpret the different variables involved (montage, laplacianMatrix) according to what was discussed in class.
- FORCe ICA-based artifact removal. You can apply FORCe with command:

  cleanEEG = FORCe(datawindow, SamplingRate, locs, 0);

  datawindow is a channels x time data segment of 1 sec (i.e., trials must be cleaned in windows of 1 sec). locs is the same variable as used above for topoplot.
- Regression-based artifact removal. To implement this method you need to define "predictor variables" from the EOG/EMG/accelerometer/gyroscope channels recorded synchronously to the EEG in the given dataset. It is best to define bipolar channels. Horizontal (difference of channels between the left and the right eye canthus) and vertical (difference of channels above and below each eye) EOG bipoles are customarily used to remove eye-related artifacts with this approach. In this dataset, we enjoy additional such bipoles capturing muscular activity on other locations on the face as shown in the picture in the Description. Acc/gyro can be added (as monopolar channels) to the predictors. Predictors should become a matrix X (time samples x predictor variables). X is usually augmented with a first column of ones (1) to implement the intercept of the regressions' hyperplanes. Define also an output variable Y (time samples x EEG channels). X and Y can be curved out from the whole file or, ideally, constructed by concatenation of the previously extracted trials. Once X and Y are defined, we can compute a set of regressors b for each EEG channel as:

  for ch=1:NChEEG

  b = [b regress(Y(:,ch), X)];

  end

  The cleaned signal can then be estimated as:

  cleanEEG = noisyEEG – Predictors*b;

Inspect raw signal grand averages, topoplots and spectra after cleaning the signal with each of these methods. Are all methods able to remove articacts? Does their effectiveness depend on the artifact type? Is there any method that is substantially more effective and more generic (i.e., applicable to all kinds of artifacts) than the others? Besides effectiveness in removing artifacts, can you think of other pros and cons of each method that could influence the decision to employ it or not?

8. Load the GDF EEG file FormulaE.gdf with the biosig toolbox:

[data, header] = sload('FILEPATH');

This file has been recorded with 512 Hz sampling rate over 61 10-20 locations as shown in header.Label. The trigger channel is at the last position (position 67 of the overall channel array). Save the trigger channel in a separate variable and remove from the data matrix and the channel label list channels 13, 19, 32, 65, 66, 67 which do not correspond to useful EEG locations. Process the data with Butterworth filtering in [1, 60] Hz and extract the period of the data between triggers/events 2 and 4 (first lap of simulated driving). Keeping the latest version of the data, produce a second version after CAR spatial filtering has been applied. Isolate a single channel and compare its spectrum between the raw and CAR-processed version. What do you observe in the raw spectrum? What is the source of this type of noise? Remember that the EEG system in this experiment was very close to the driving simulator which includes many active electronic components. What is the frequency of this type of noise? Did it appear in the first dataset given, too? If so, was it as strong as in this file, and why? Is CAR able to cope with this type of noise?