EEG Course: Semesterproject

# Project description

## Project pipeline architecture:

The pipeline with several single python scripts which can be execute together or individually by using the package **Pydoit**. In general this work tries to follow the seven quick tips given by Marijn van Vliet [1]. Also some recommendations and best practices were taken over from his conpy example project like the filenames class which conveniently manages file paths.

Using the pipeline with Pydoit has several advantages:

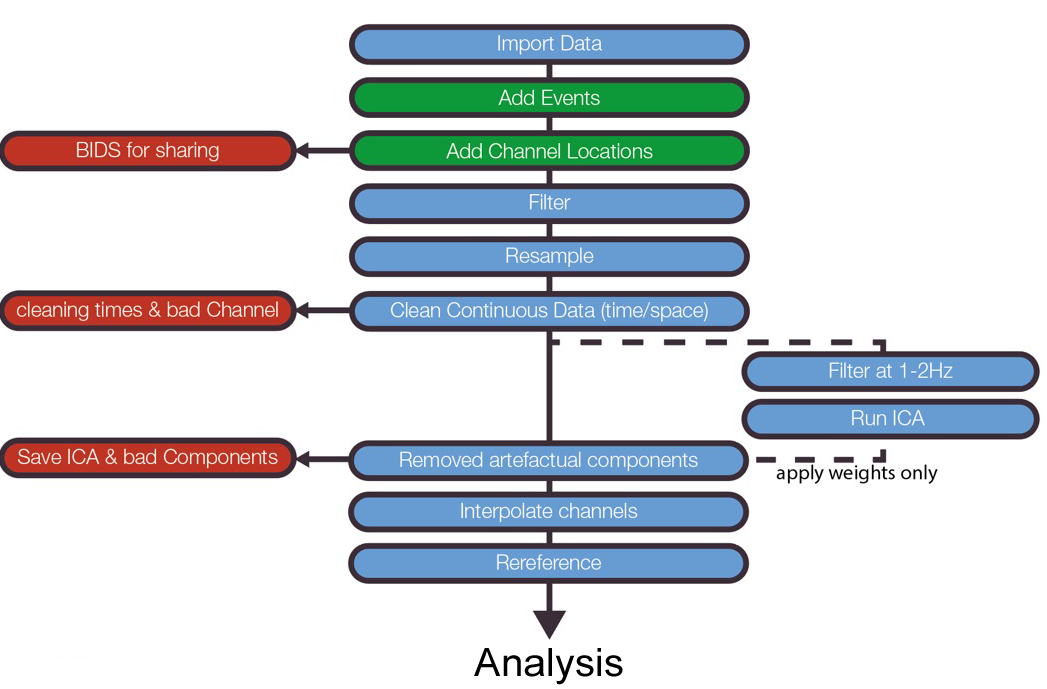
* **Data consistency/ Step-wise artefacts:**

Each step takes as input the artefact of the previous step. Then it performs its operation and produces a new result. All the artefacts can be individually inspected and one could restart from any point in the pipeline

* **Parallelization:**

Pydoit takes fully care of parallelization of the specified jobs and uses the provided computing resource therefore very efficiently which saves time especially computing results for multiple subjects.

# Preprocessing



This pipeline by Cohen shows the basic preprocessing and cleaning steps for the ERP Core data.  
or the whole preprocessing part I follow this procedure. In the next subchapter the steps are explained in more detail including parameter choices.

## Filter (01\_preprocess.py)

Filtering the EEG Core data with a bandpass filter provided in MNE. Filter choices:

* **Bandpass-Min:** 0.5 Hz

Recommended by Cohan and others for EEG data

* **Bandpass-Max:** 50 Hz

Typical choice for EEG data, Pitfalls of these choices as described in [2]

* **Filter design:** Firwin

## Cleaning ( 02\_cleaning.py)

This step performs cleaning of bad segments and bad channels. It can be executed either using precomputed annotations and loading them or in an interactive way which let you choose with MNE plot functions which segments/channels are bad.

* Bad segments:
  + Are saved in the raw file instance as annotations
* Bad channels:
  + Saves them in raw.info[“bads”]
  + Interpolates bad channels if there are any

## ICA (03\_ica.py)

The Independent component analysis (ICA) allows to split into individual components which then can be analyzed regarding their effect and meaningfulness on the scenario. The ICA operates the following steps:

1. Load cleaned raw object from step 02
2. Use ICA specific Highpass Filter as described by Cohen and others.

This shall remove slow drifts in the data before computing the ICA.

Therefore, I choose a Highpass with Cutoff-Frequency 1Hz.

1. Fit the ICA and remove bad components to the initial raw data object

## Rereference (04\_rereference.py)

Selects the reference electrodes for the signal. Having a look in the literature e.g. ERP Core Paper [3] and several Tutorials  
<https://mne.tools/stable/auto_tutorials/preprocessing/plot_55_setting_eeg_reference.html>   
<https://www.fieldtriptoolbox.org/faq/why_should_i_use_an_average_reference_for_eeg_source_reconstruction/>

There seems to be several possibilities to do this. Modern approaches recommend to go with the Average of the electrodes as Reference, whereas the ERP Core Paper uses the average of the mastoid sites (P9, P10). We also learned that besides these two approaches the REST (Reference Electrode Standardization Techniques) can be used.

For this purpose I stayed close to the ERP Core paper by using the mastoids (average P9/P10) as reference.