Preprocessing pipeline and utility tools

Makoto Miyakoshi

25th EEGLAB Workshop in Tokyo

Sep 29 2017

Motivation for building a processing pipeline

- When I process the multiple data sets from one study, I want to make sure that:
  - I use objective and consistent processes to all the subjects (*scientific requirement for replicatability and sharability*)
  - If I need to change parameters later, re-computation is easy (*efficiency in work flow*)
- Imagine you have n = 952 data sets. Do you still want to manually process them one by one? Is it even adequate?
  - How many post docs will you need? :-)

An example of preprocessing pipeline

Log in

Page | Discussion | Read | View source | View history | Search | Go | Search

Makoto's preprocessing pipeline

Contents [hide]

https://sccn.ucsd.edu/wiki/Makoto's_preprocessing_pipeline

A template batch code

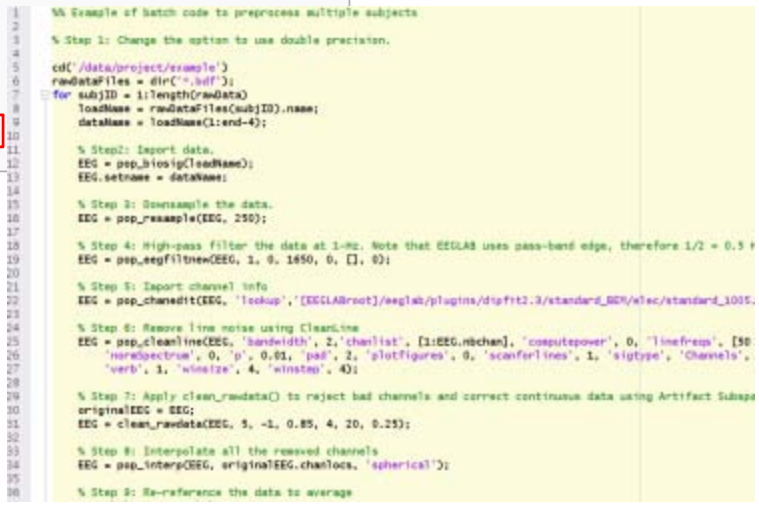
Swartz Center for Computational Neuroscience

Log in

Page | Discussion | Read | View source | View history | Search | Go | Search

Makoto's useful EEGLAB code

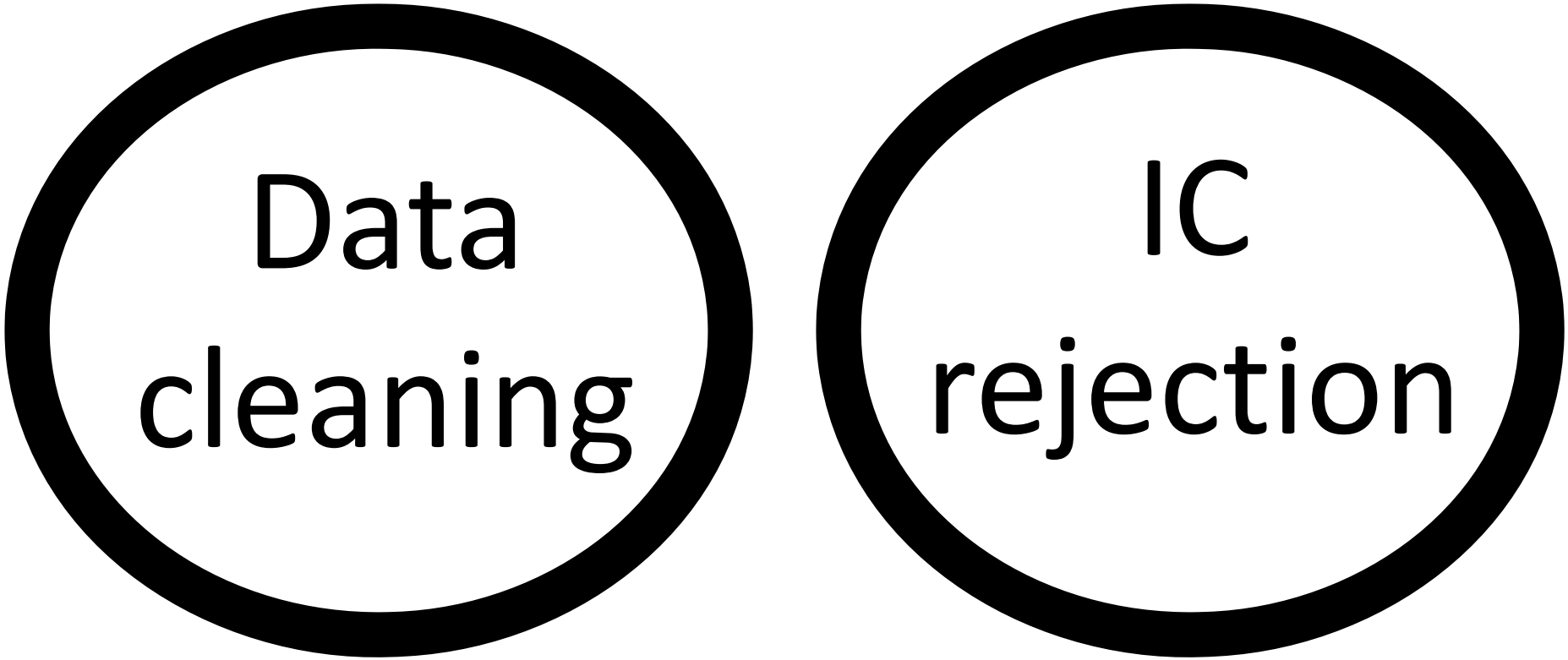
Contents [hide]



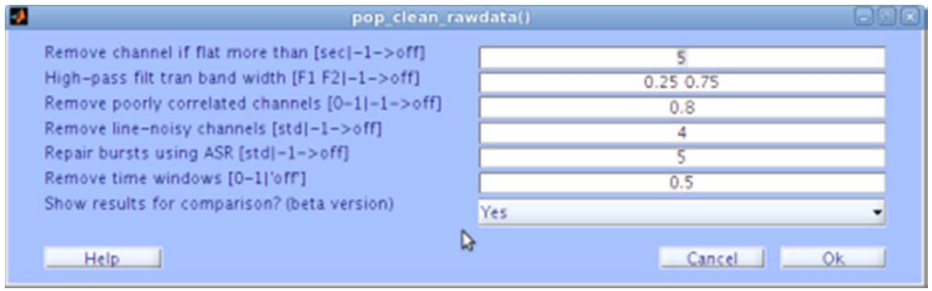
```
%% Example of batch code to preprocess multiple subjects

% Step 1: Change the option to use double precision.

cd('/data/project/example')
rawDataFiles = dir('*.bdf');
for subjID = 1:length(rawData)
    loadName = rawDataFiles(subjID).name;
    dataName = loadName(1:end-4);

    % Step2: Import data.
    EEG = pop_biosig(loadName);
    EEG.setname = dataName;

    % Step 3: Downsample the data.
    EEG = pop_resample(EEG, 250);

    % Step 4: High-pass filter the data at 1-Hz. Note that EEGLAB uses pass-band edge, therefore 1/2 = 0.5 H
    EEG = pop_eegfiltnew(EEG, 1, 0, 1650, 0, [], 0);

    % Step 5: Import channel info
    EEG = pop_chanedit(EEG, 'lookup','[EEGLABroot]/eeglab/plugins/dipfit2.3/standard_BEM/elec/standard_1005.

    % Step 6: Remove line noise using CleanLine
    EEG = pop_cleanline(EEG, 'bandwidth', 2,'chanlist', [1:EEG.nbchan], 'computepower', 0, 'linefreqs', [50
        'normSpectrum', 0, 'p', 0.01, 'pad', 2, 'plotfigures', 0, 'scanforlines', 1, 'sigtype', 'Channels',
        'verb', 1, 'winsize', 4, 'winstep', 4);

    % Step 7: Apply clean_rawdata() to reject bad channels and correct continuous data using Artifact Subspa
    originalEEG = EEG;
    EEG = clean_rawdata(EEG, 5, -1, 0.85, 4, 20, 0.25);

    % Step 8: Interpolate all the removed channels
    EEG = pop_interp(EEG, originalEEG.chanlocs, 'spherical');

    % Step 9: Re-reference the data to average
```

https://sccn.ucsd.edu/wiki/Makoto%27s_useful_EEGLAB_code

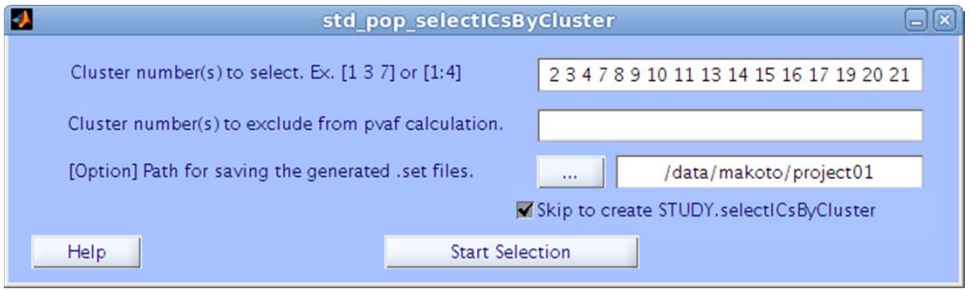
Two obstacles in automated preprocessing



Two key plugins for the solutions
(both installable from EEGLAB extension manager)



clean_rawdata()



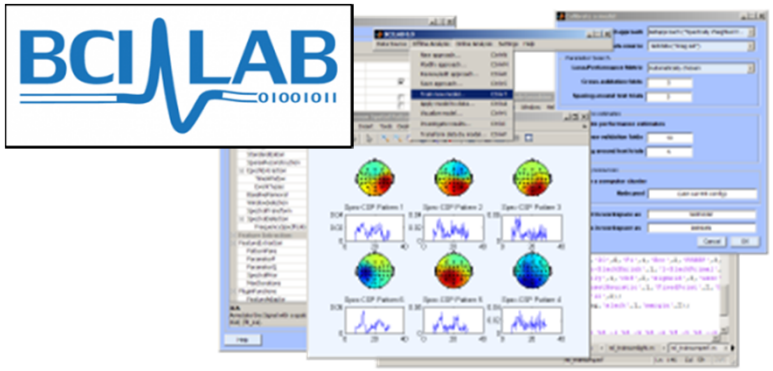
std_selectICsByCluster()

clean_rawdata()

taken and modified from BCILAB written by *Christian Kothe*

Christian Kothe, the author of [Simulation and Neuroscience Application (SNAP)] which uses an game engine Panda3D that includes graphics, audio, I/O, collision detection, and other abilities relevant to the creation of 3D games.

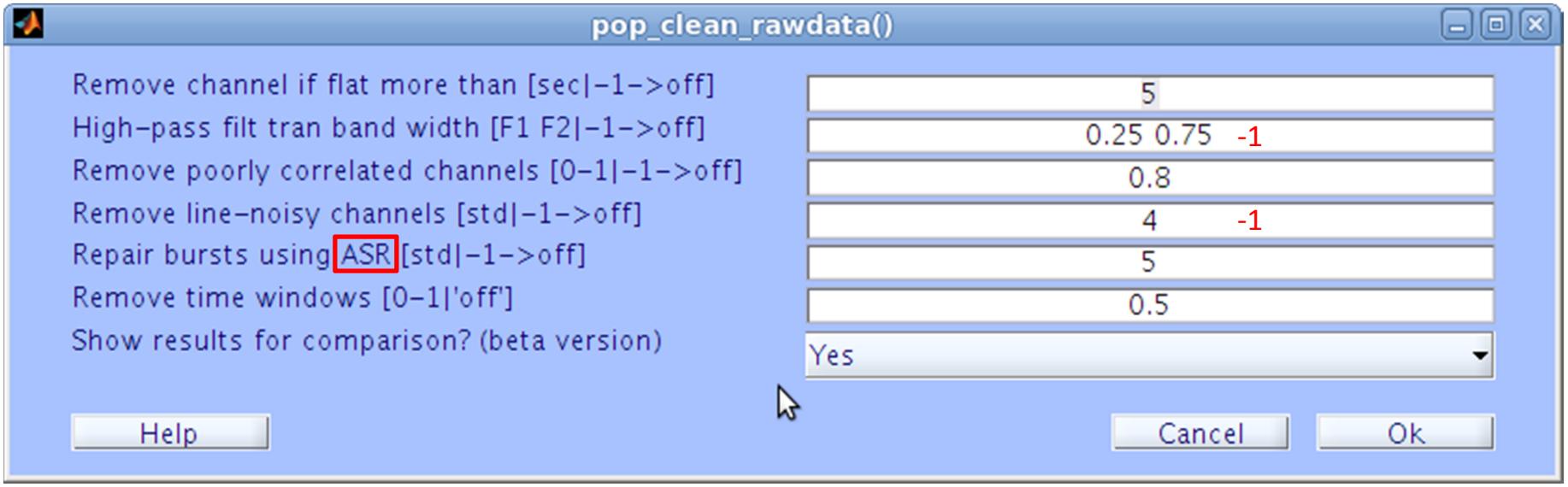
Christian Kothe, the author of [Brain-Computer Interfacec LAB (BCILAB)]



Christian Kothe, the author of [Lab-streaming Layer (LSL)]

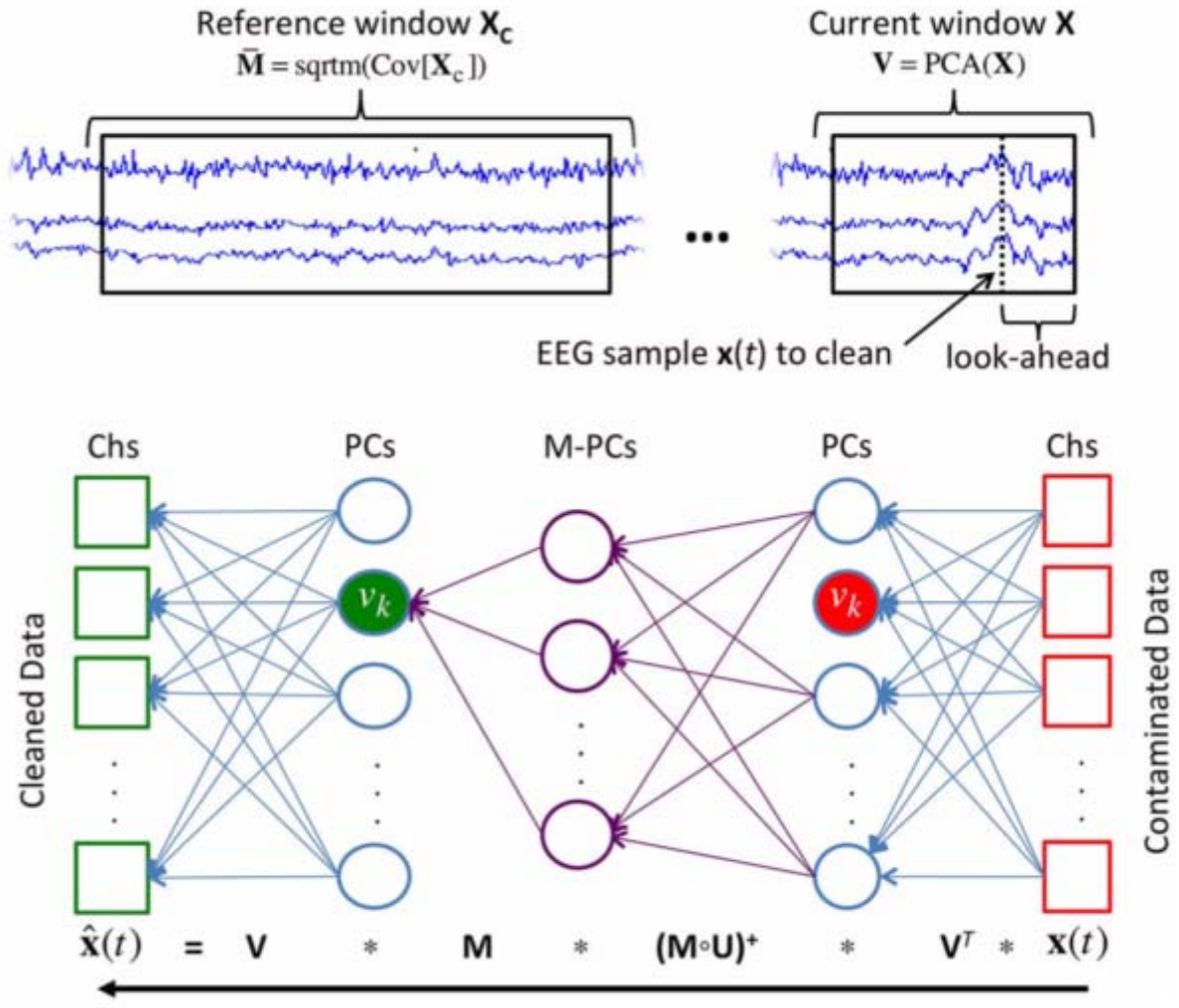
- ***The 1st International Workshop on LSL*** was held in Delmenhorst, Germany in Dec19-20, 2016.

clean_rawdata()



- *clean_rawdata()* takes <u>continuous</u> channel data.
- Apply this <u>after importing channel locations</u> because it benefits bad channel rejection.

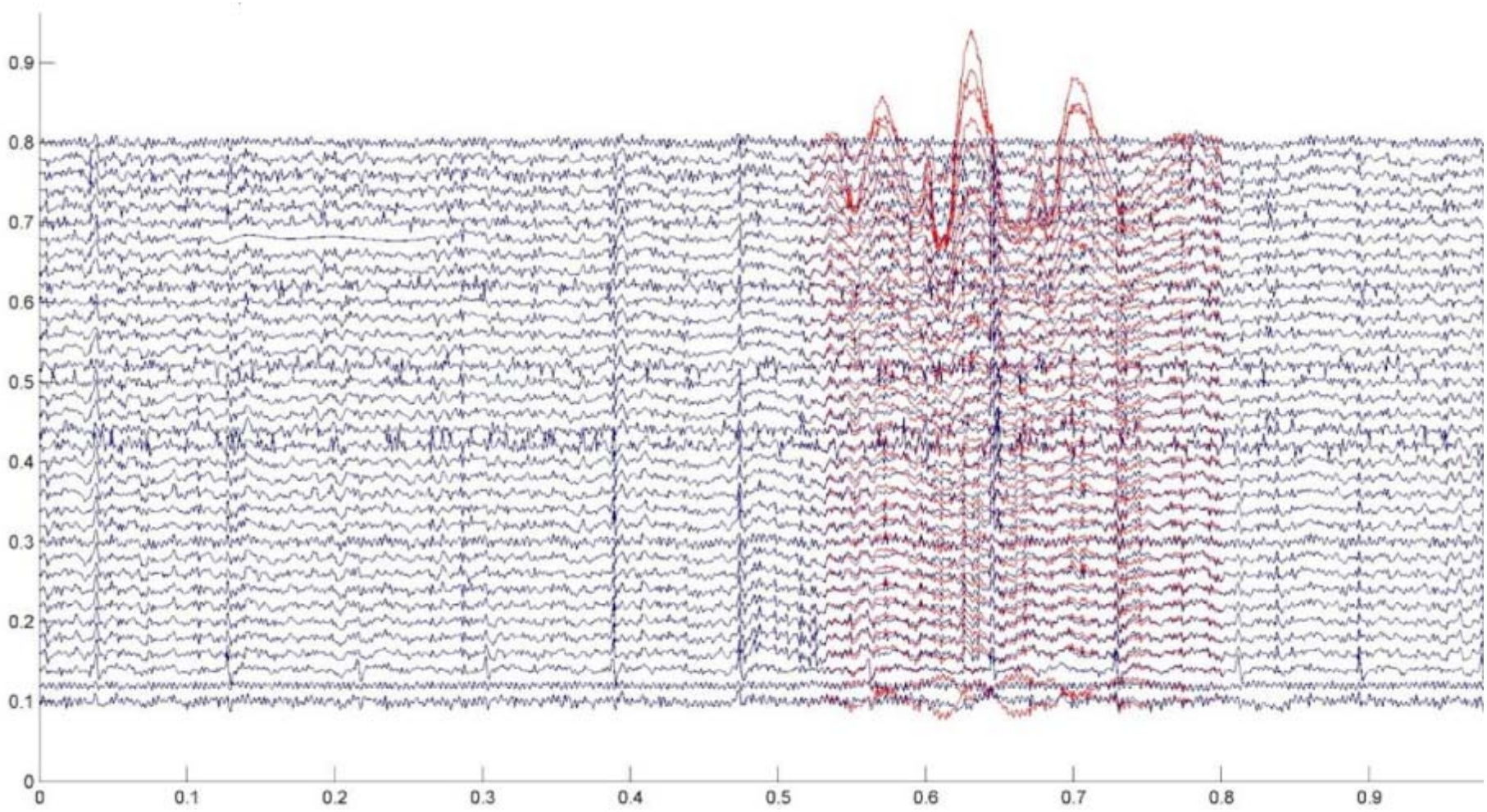
Artifact Subspace Reconsutruction (ASR)



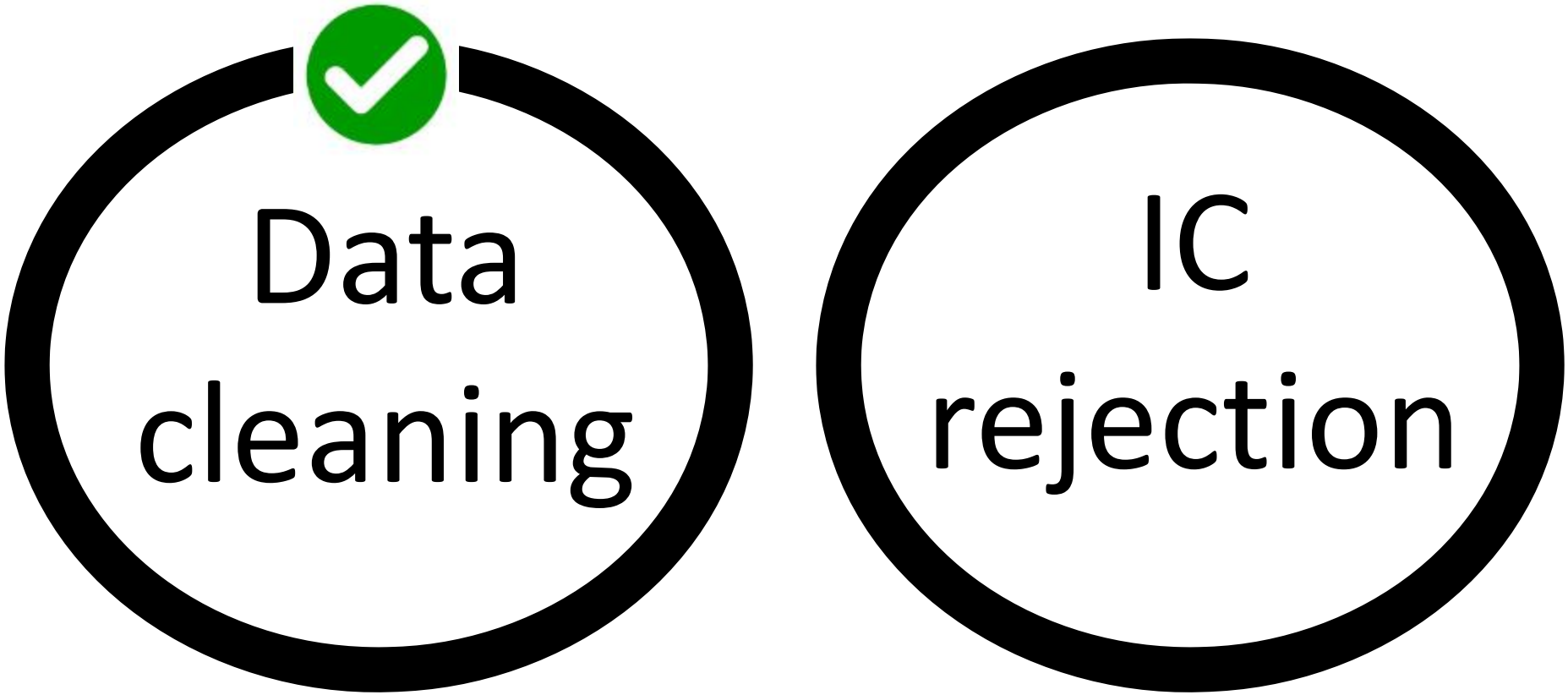
1. Finds the cleanest part of data.
2. Learns robust covariance matrix.
3. Using 1-s sliding window, perform Principal Component Analysis (PCA)
4. If PCs of a given window goes beyond 5 SD of cleanest part of data (i.e. *artifact subspace*), reject them.
5. *Reconstruct* the rejected PCs using the learned covariance matrix.
6. Recover the channel data.

Mullen et al. (2015). *IEEE Trans. Biomed. Eng.*
See also http://sccn.ucsd.edu/eeglab/plugins/ASR.pdf for ppt slides by Christian.

ASR result example



Two obstacles in automated preprocessing

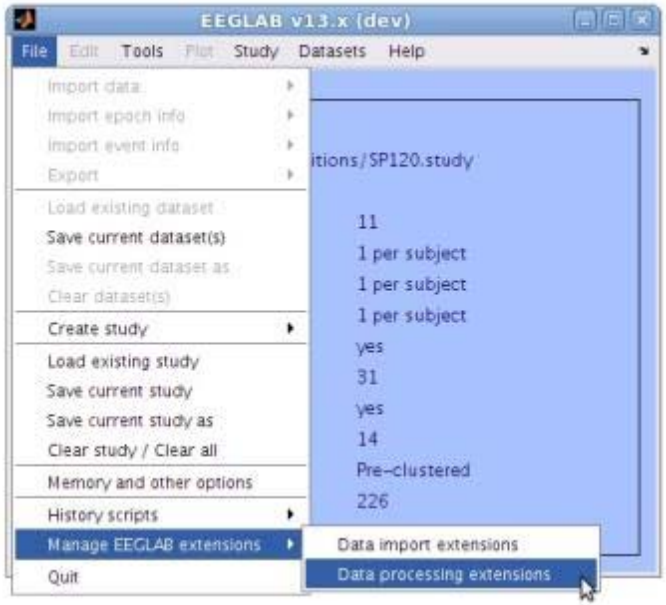


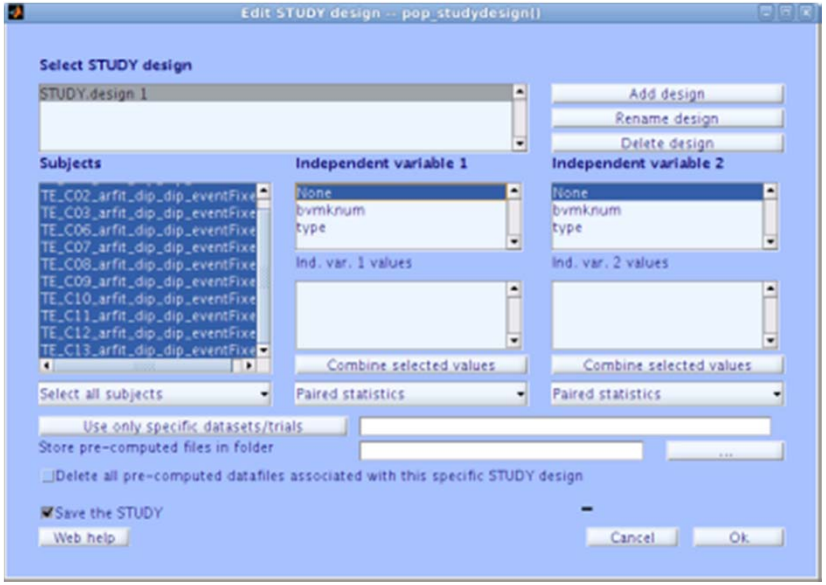
std_selectICsByCluster()

Group-level IC cluster selection

- If you have hundreds of subjects, how do you select/reject ICs?
- STUDY provides IC rejection for 1) outside the brain, 2) high r.v. dipoles. But, EOG and MEG always sneak into final results (shown later).
- Use std_selectICsByCluster() plugin.

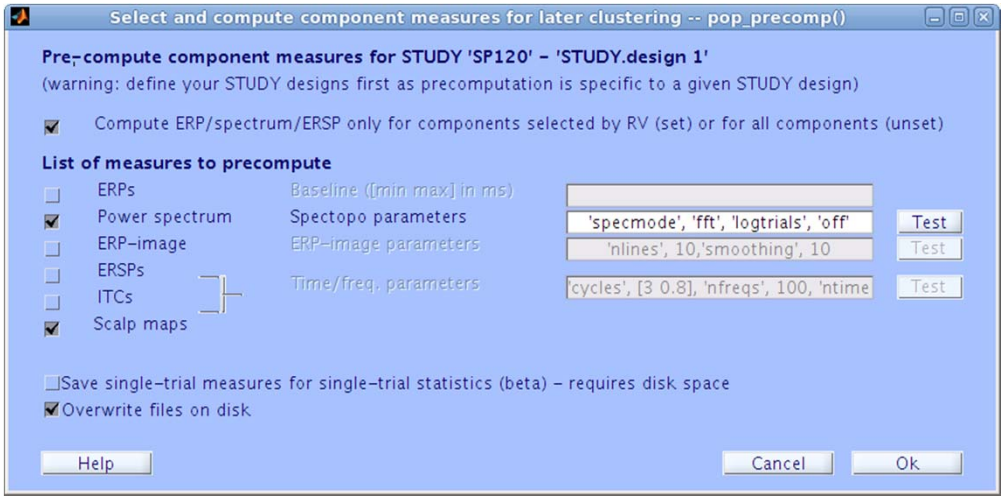
  https://sccn.ucsd.edu/wiki/Std_selectICsByCluster

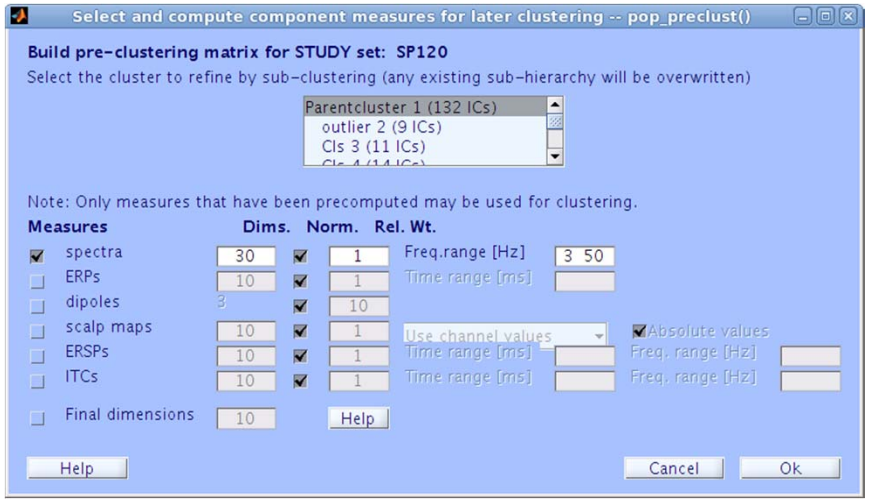




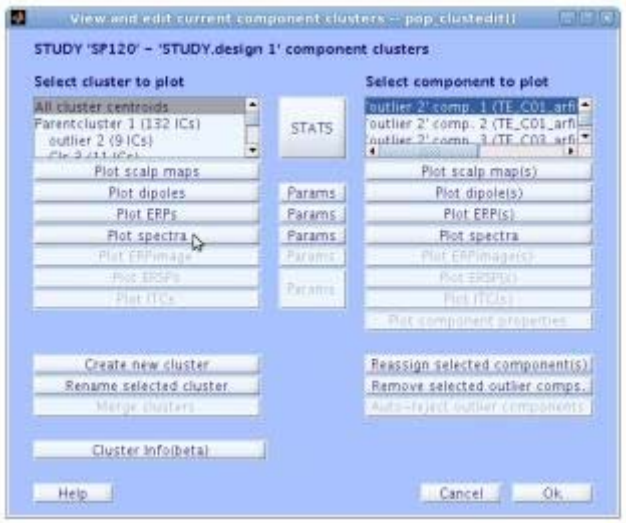
1. Install the plugin. 2. Load a STUDY. 3. Set STUDY.design to be 'None' for variable 1 and 2.

GUI tutorial (continued)



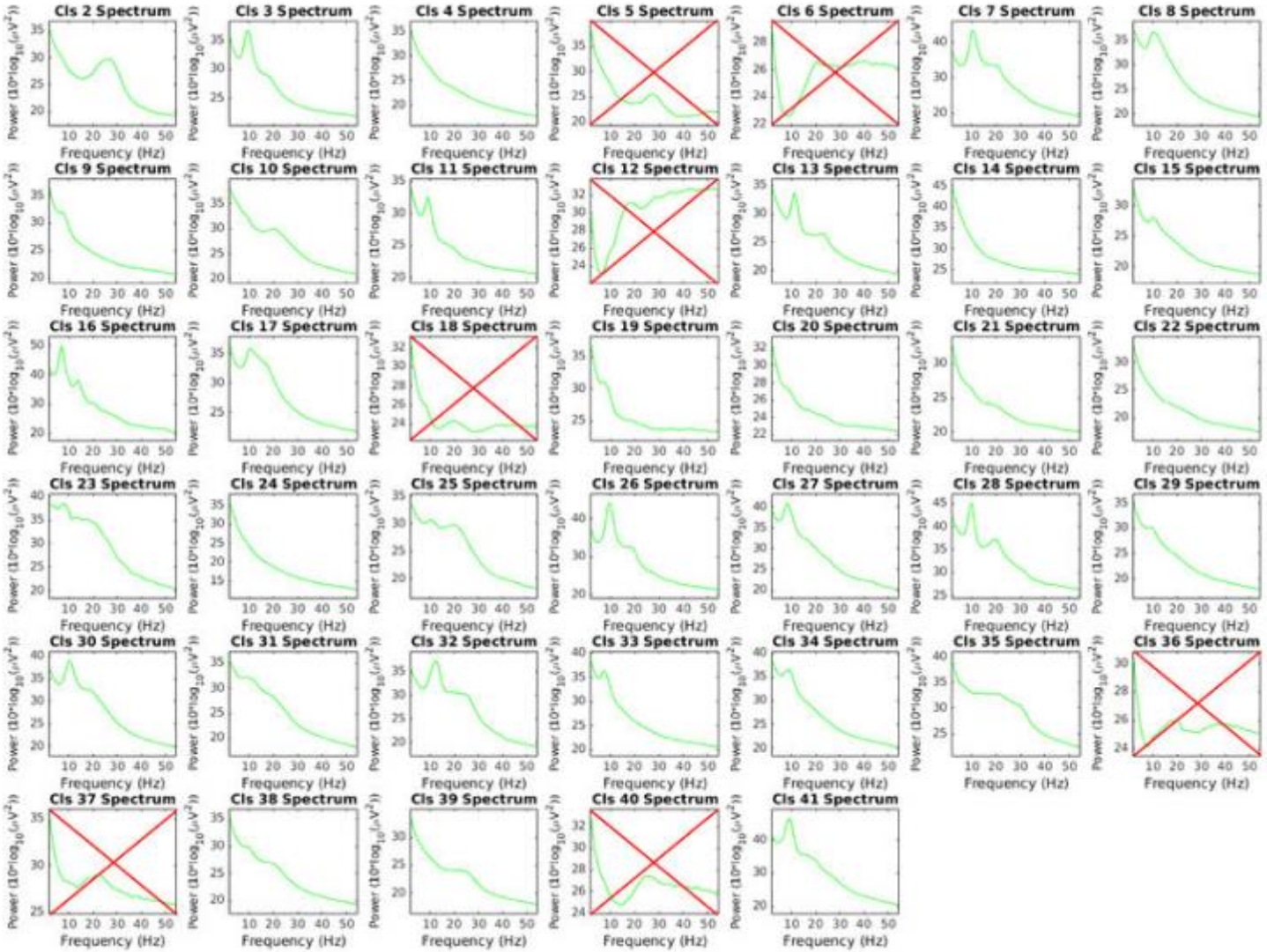


4. Precompute spectrum (optionally with scalp maps) 5. Cluster all ICs just by using spectra



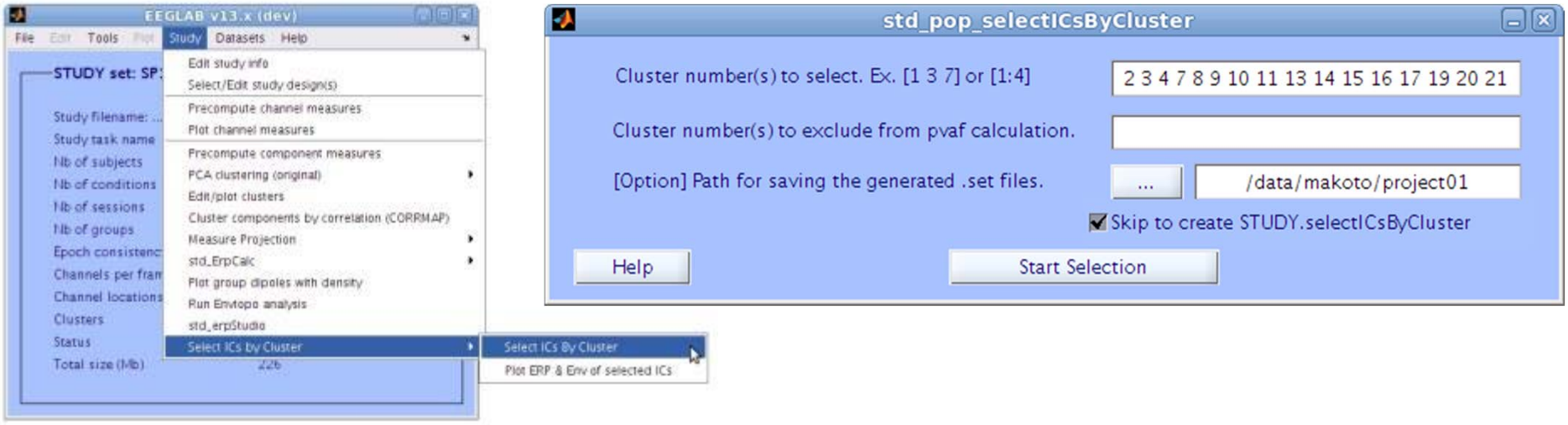
6. Create the same number of clusters and 'Plot spectra'

Reject *clusters* with non-brain power spectrum density!



7. Write down which cluster shows bad spectrum!

Use the plugin to back-project good IC clusters only

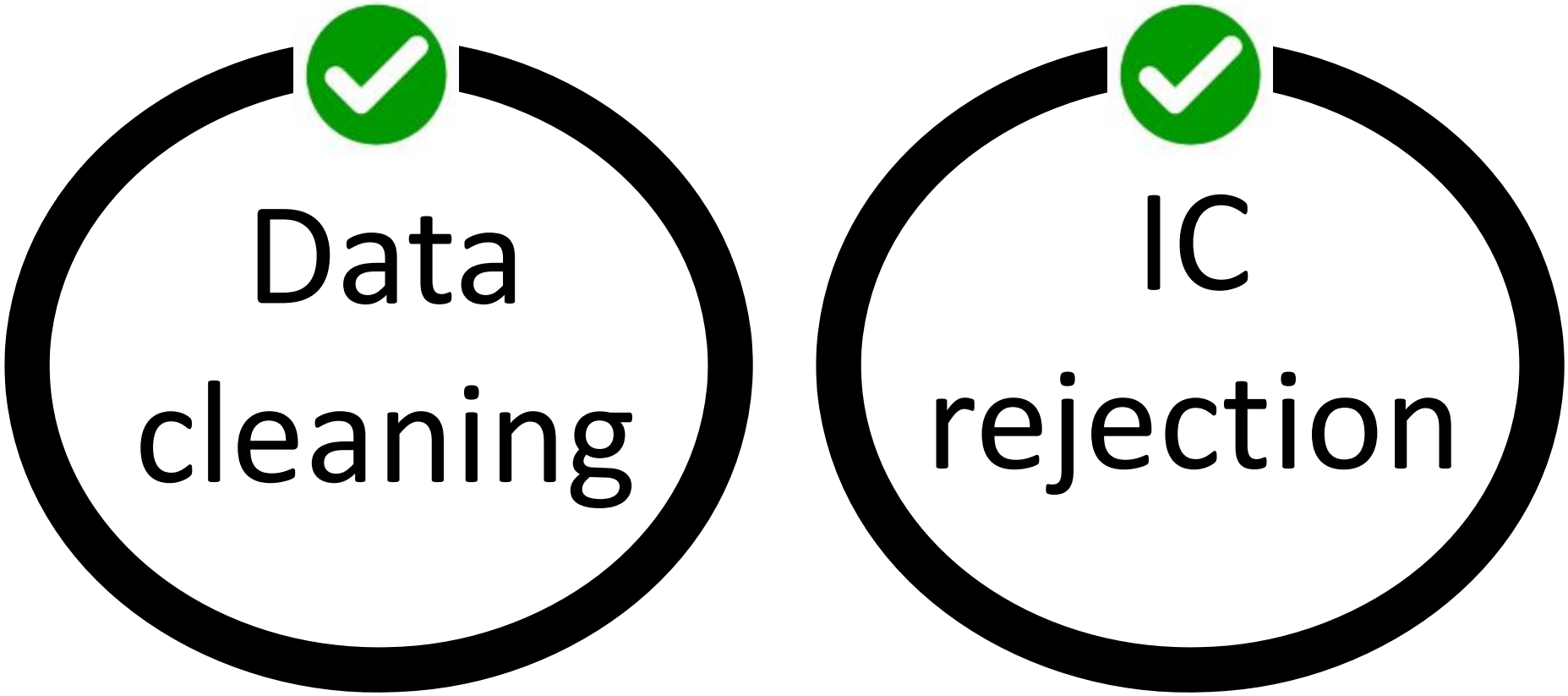


8. Launch the *std_selectICsByCluster()* plugin. 9. Enter good cluster indices and path to a folder.

10. It'll generate the same number of .set files but with only good ICs. Create another STUDY using all of the back-projected ICs since they are clean (even if they are noisy, their PSD follows 1/f curve so beneign.

Note that **advanced analysis tools, such as SIFT and Measure Projection Toolbox, take ALL ICs**. Hence pre-selecting ICs is mandatory, otherwise you'll include all noises and junks!

Two obstacles in the automated and objective preprocessing



You are ready to go! Enjoy EEG data mining.

Thank you!

‘Dad, what do you do in your job?’
‘I build and clean pipelines.’