

# Preprocessing pipeline and utility tools

Makoto Miyakoshi

25<sup>th</sup> 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](#)

## Makoto's preprocessing pipeline

### Contents [\[hide\]](#)

- 1 Change the option to use double precision
- 2 Import data
- 3 Downsample if necessary
- 4 High-pass filter the data at 1-Hz (for ICA, ASR, and CleanLine)(03/29/2017 updated)
- 5 Import channel info
- 6 Remove bad channels
- 7 Interpolate all the removed channels
- 8 Re-reference the data to average
- 9 Remove line noise using CleanLine
- 10 Epoch data to -1 to 2 sec (12/07/2016 updated)
- 11 Reject epochs for cleaning
- 12 [Adjust data rank for ICA \(12/26/2016 updated\)](#)
- 13 Run ICA (07/25/2017 updated)
  - 13.1 To learn how to evaluate EEG and artifact ICs (01/24/2017 updated)
  - 13.2 Unofficial instruction to install AMICA (08/03/2017 updated)
- 14 Estimate single equivalent current dipoles
- 15 Search for and estimate symmetrically constrained bilateral dipoles
- 16 Create STUDY (01/05/2017 updated)
  - 16.1 A tip to compute time-frequency transform (i.e. ERSP & ITC) (01/11/2017 updated)
- 17 Alternatively, cleaning continuous data using ASR (12/23/2016 updated)
  - 17.1 Example of batch code to preprocess multiple subjects (01/12/2017 updated)

[https://sccn.ucsd.edu/wiki/Makoto's\\_preprocessing\\_pipeline](https://sccn.ucsd.edu/wiki/Makoto's_preprocessing_pipeline)

# A template batch code



Page Discussion

Read

[View source](#)

[View history](#)

Search

Go

Search

[Log in](#)

## Makoto's useful EEGLAB code

### Contents [\[hide\]](#)

- 1 How to obtain executed code with input parameters by operating graphical user interface (GUI)
- 2 How to extract subjects and independent components from STUDY structure
- 3 How to extract EEG power of frequency bands
- 4 How to build EEG structure (05/17/2017 updated)
- 5 Event types
  - 5.1 How to obtain unique event types
  - 5.2 How to obtain event indices whose event type exactly matches a keyword
  - 5.3 How to obtain event indices whose event types contain a part of the keyword (12/13/2016 updated)
  - 5.4 How to change event type names
- 6 Event latency
  - 6.1 How to change event latency
  - 6.2 How to adjust event latency by adding or subtracting values
- 7 Example of batch code to preprocess multiple subjects (06/27/2017 updated)

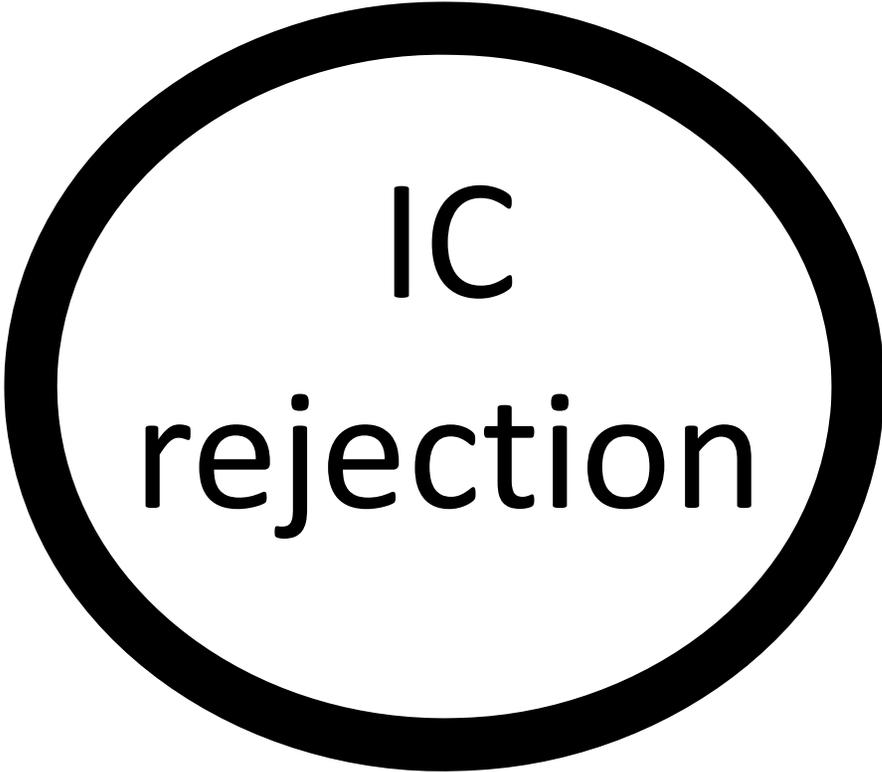
```
1 % Example of batch code to preprocess multiple subjects
2
3 % Step 1: Change the option to use double precision.
4
5 cd('/data/project/example')
6 rawDataFiles = dir('*.sdF');
7 for subjID = 1:length(rawData)
8     loadName = rawDataFiles(subjID).name;
9     dataName = loadName(1:end-4);
10
11 % Step2: Import data.
12 EEG = pop_loadset(loadName);
13 EEG.setname = dataName;
14
15 % Step 3: Downsample the data.
16 EEG = pop_resample(EEG, 250);
17
18 % Step 4: High-pass filter the data at 1-Hz. Note that EEGLAB uses pass-band edge, therefore 1/2 = 0.5 +
19 EEG = pop_eegfiltnew(EEG, 1, 0.1850, 0, [], 0);
20
21 % Step 5: Export channel info
22 EEG = pop_chanedit(EEG, 'lookup', '[EEGLABroot]/eeglab/plugins/dspfilt2.3/standard_00R/wlec/standard_1005.
23
24 % Step 6: Remove line noise using Cleanline
25 EEG = pop_cleanline(EEG, 'bandwidth', 2, 'chanlist', [1:EEG.nbchan], 'computePower', 0, 'linefreq', [50
26 'nordSpectrum', 0, 'p', 0.01, 'pad', 2, 'plotfigures', 0, 'scatterlines', 1, 'sigtype', 'Channels',
27 'verb', 1, 'winsize', 4, 'winstep', 4);
28
29 % Step 7: Apply clean_rawdata() to reject bad channels and correct continuous data using Artifact Subspace
30 originalEEG = EEG;
31 EEG = clean_rawdata(EEG, 5, -1, 0.85, 4, 20, 0.25);
32
33 % Step 8: Interpolate all the removed channels
34 EEG = pop_interp(EEG, originalEEG.chanlocs, 'spherical');
35
36 % Step 9: Re-reference the data to average
```

[https://scn.ucsd.edu/wiki/Makoto%27s\\_useful\\_EEGLAB\\_code](https://scn.ucsd.edu/wiki/Makoto%27s_useful_EEGLAB_code)

# Two obstacles in automated preprocessing

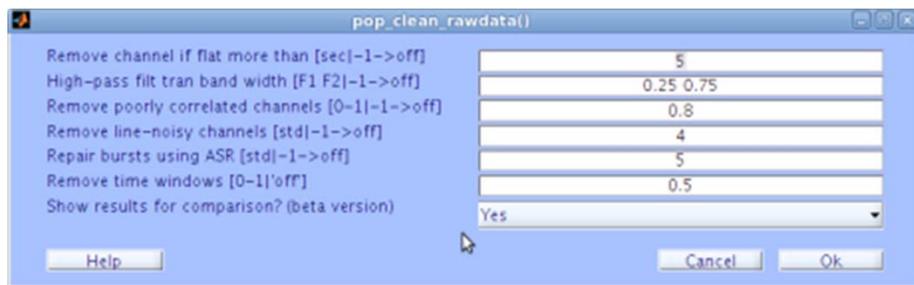


Data  
cleaning

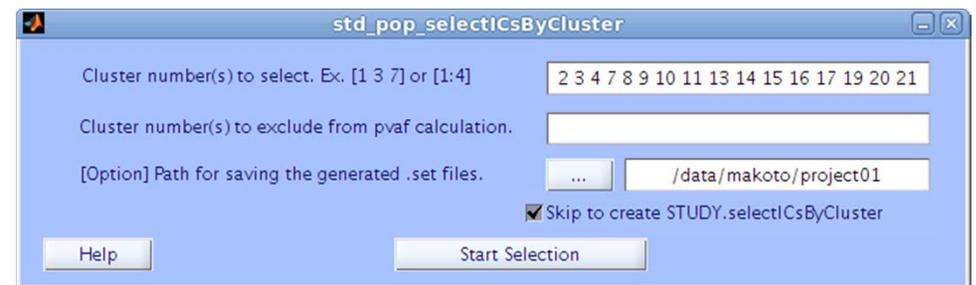


IC  
rejection

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



`clean_rawdata()`



`std_selectICsByCluster()`

# clean\_rawdata()

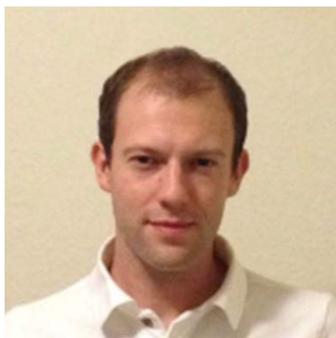
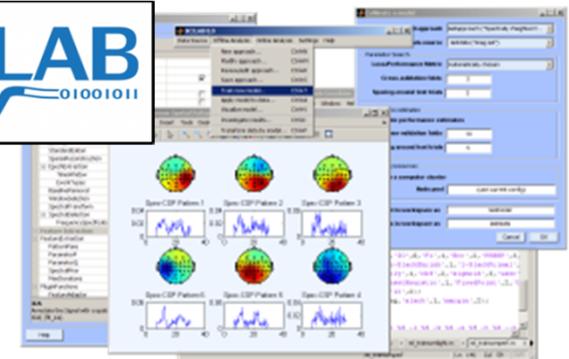
taken and modified from BCILAB written by *Christian Kothe*



**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 \(BCILAB\)](#)

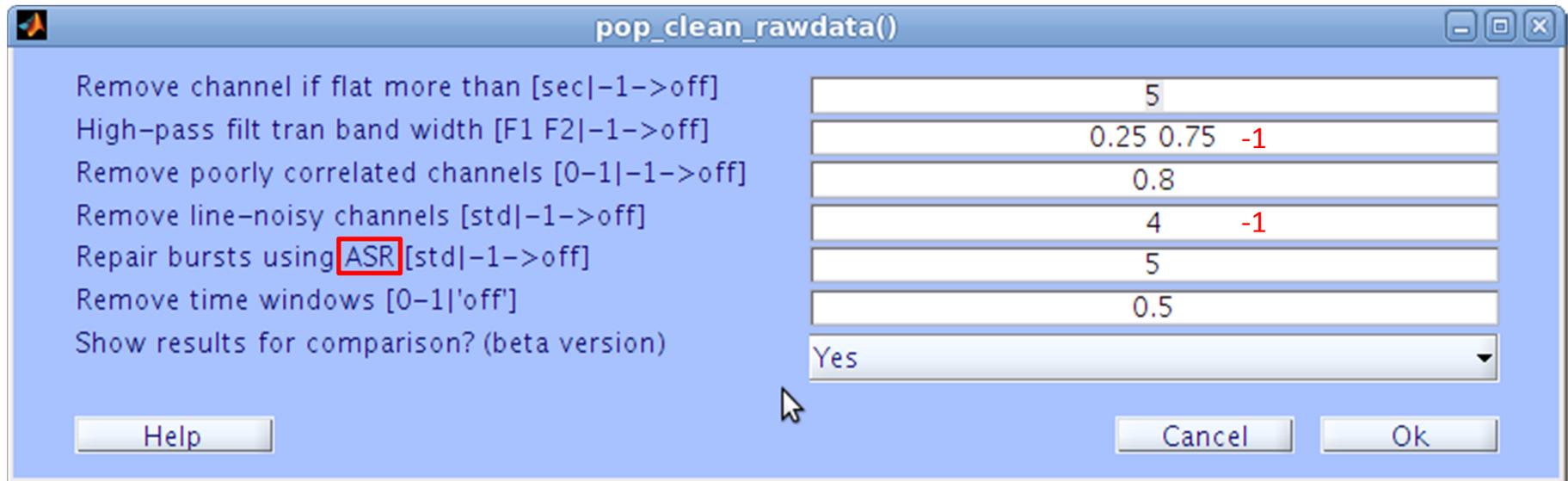


**Christian Kothe**, the author of [Lab-streaming Layer \(LSL\)](#)

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

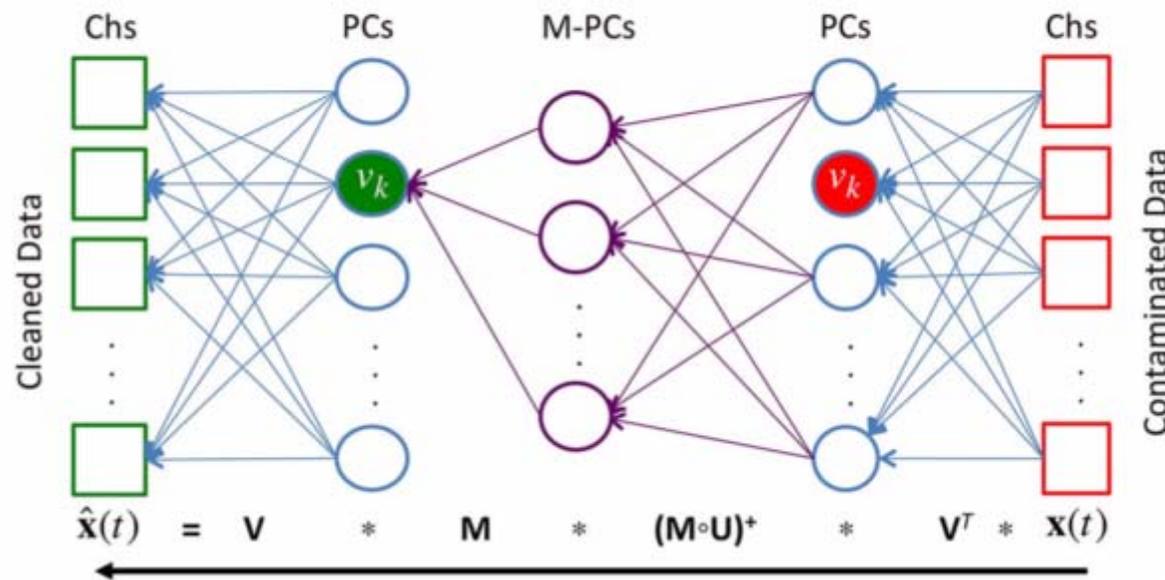
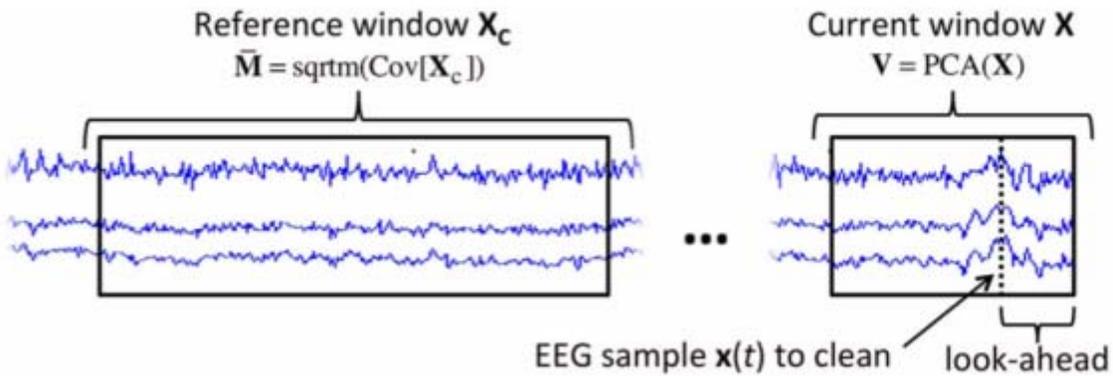


# clean\_rawdata()



- *clean\_rawdata()* takes continuous channel data.
- Apply this after importing channel locations because it benefits bad channel rejection.

# Artifact Subspace Reconstruction (ASR)

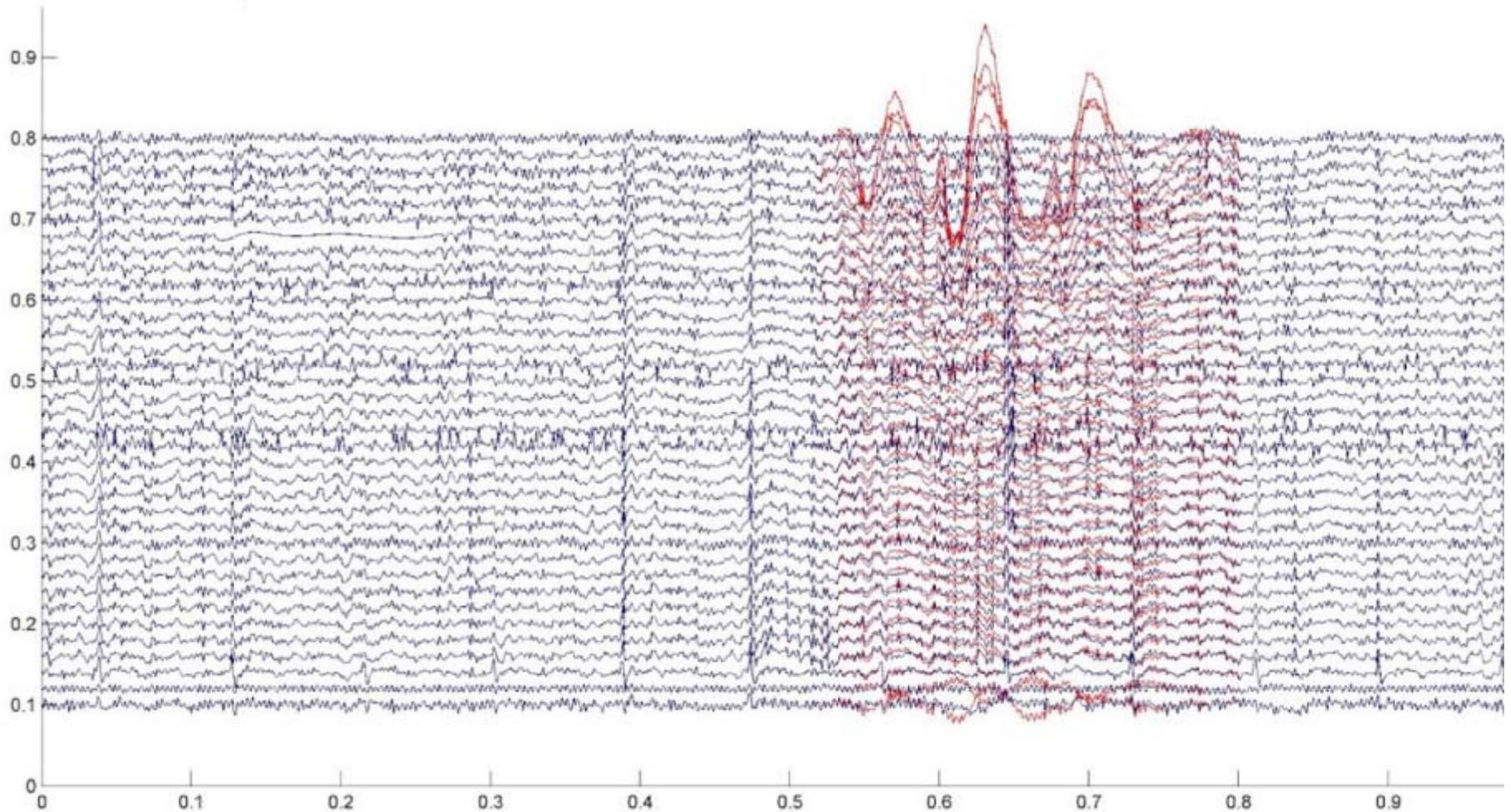


1. Finds the cleanest part of data.
2. Learns robust covariance matrix.
3. Using 1-s sliding window, perform Principal Component Analysis (PCA)
4. If PCs of a given window goes beyond 5 SD of cleanest part of data (i.e. artifact subspace), reject them.
5. Reconstruct the rejected PCs using the learned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



Data  
cleaning

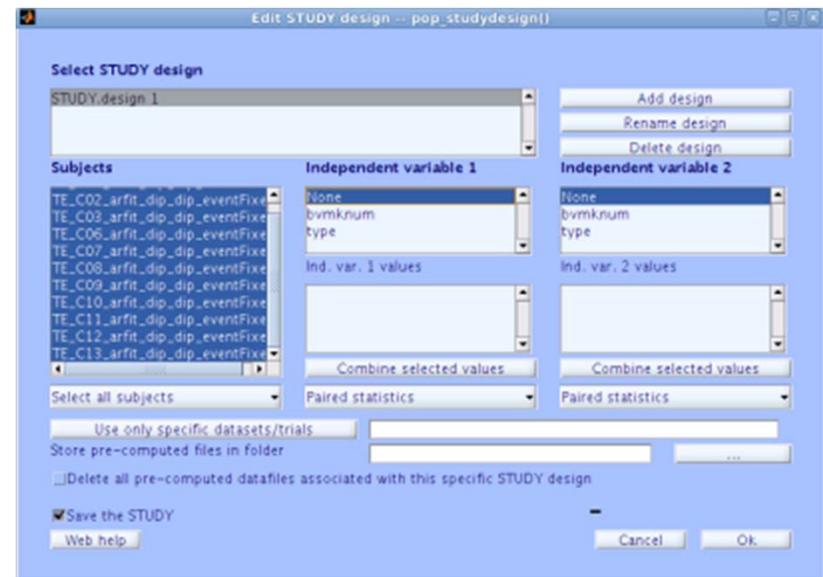
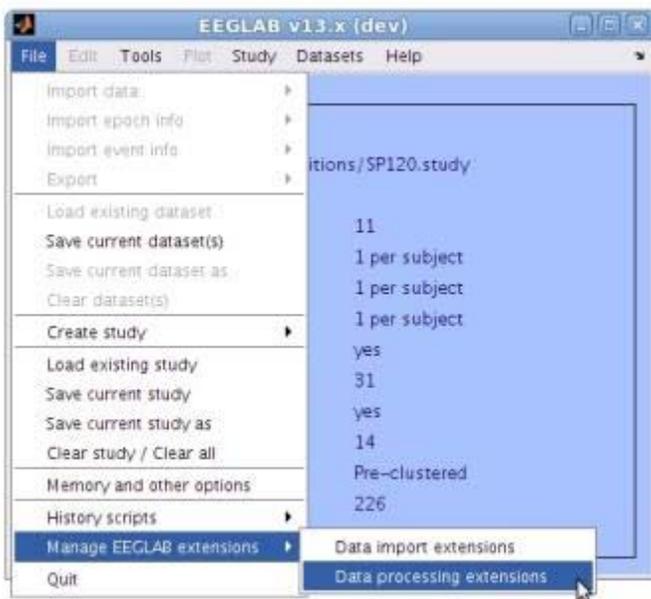
IC  
rejection

`std_selectICsByCluster()`

# Group-level IC cluster selection

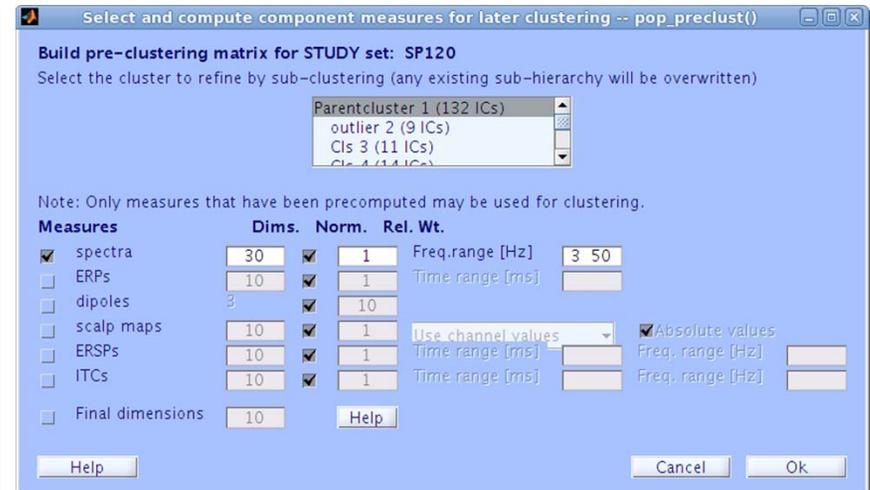
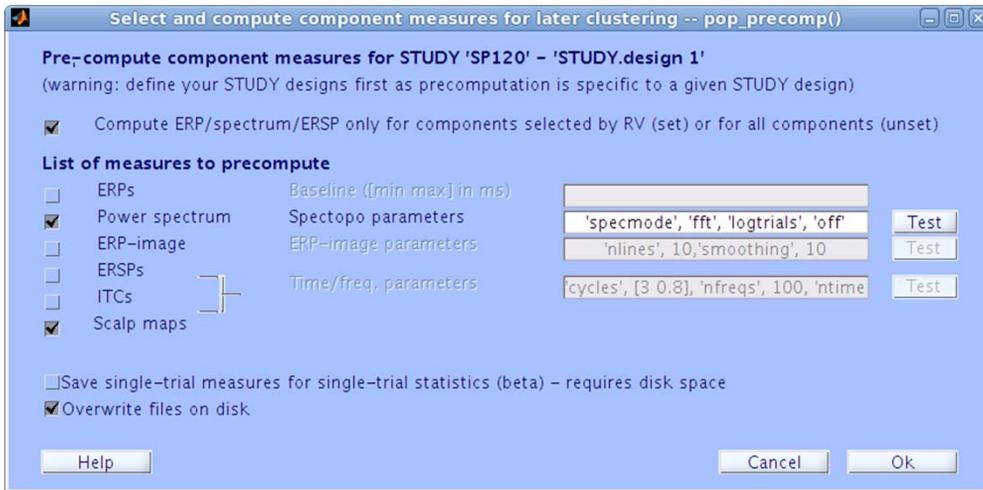
- If you have hundreds of subjects, how do you select/reject ICs?
- STUDY provides IC rejection for 1) outside the brain, 2) high r.v. dipoles. But, EOG and MEG always sneak into final results (shown later).
- Use `std_selectICsByCluster()` plugin.

[https://sccn.ucsd.edu/wiki/Std\\_selectICsByCluster](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)

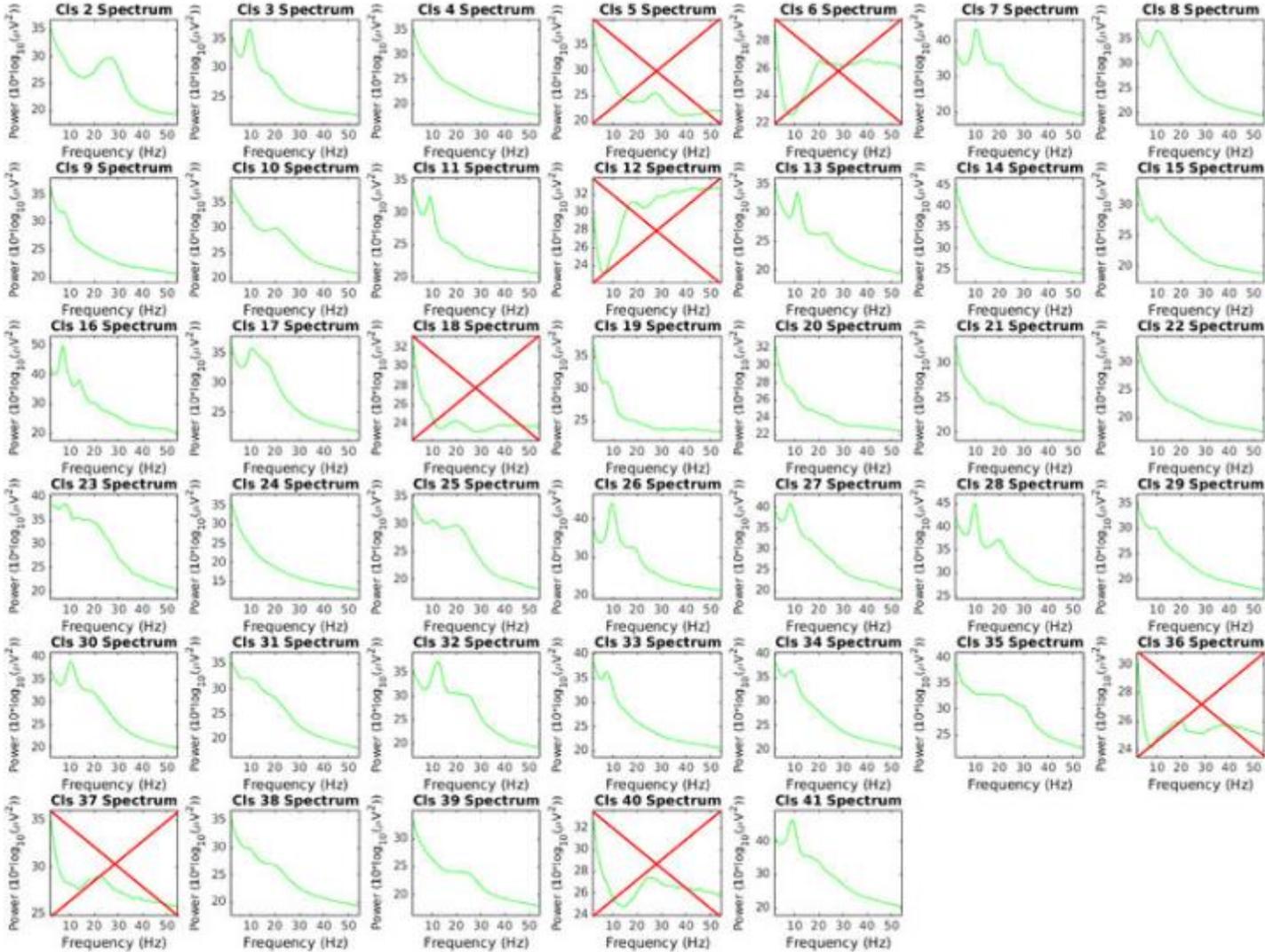


4. Precompute spectrum (optionally with scalp maps) 5. Cluster all ICs just by using spectra



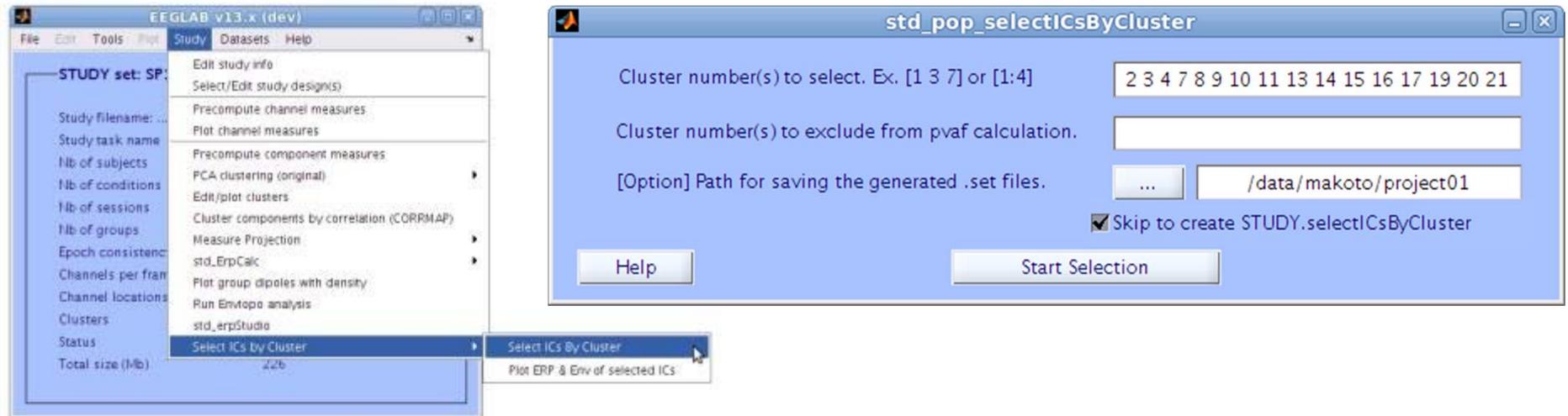
6. Create the same number of clusters and 'Plot spectra'

# 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

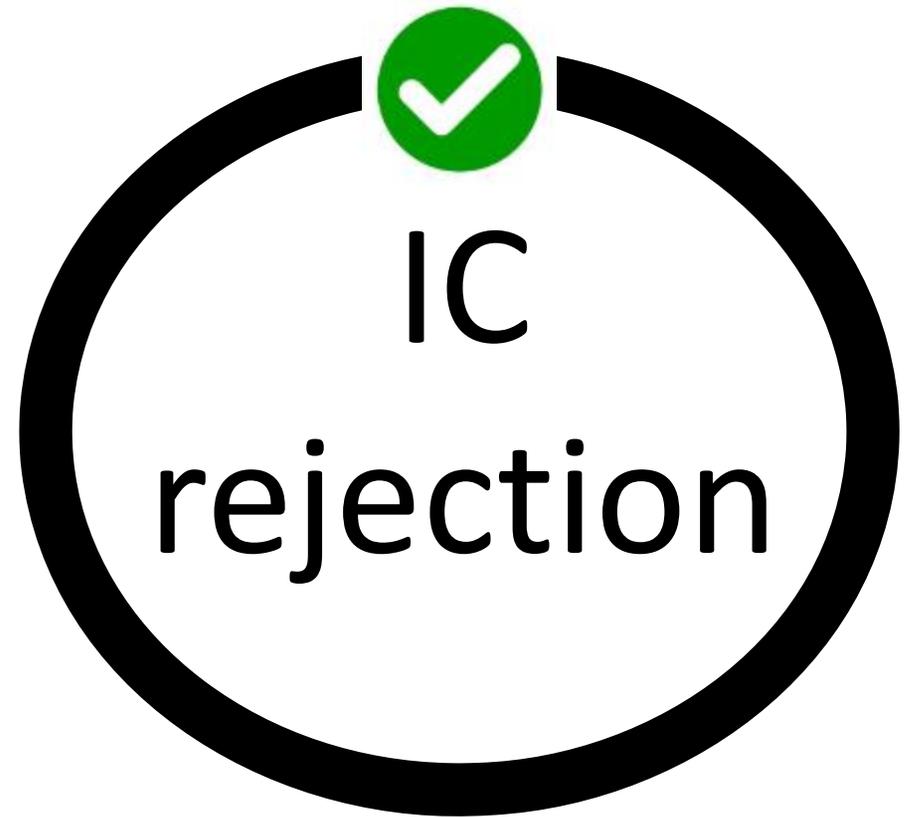


8. Launch the `std_selectICsByCluster()` plugin. 9. Enter good cluster indices and path to a folder.

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 benign).

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.’