Multivariate analysis

Romain Grandchamp, PhD
Arnaud Delorme, PhD

Practical MEEG 2025, Aix-Marseille Université, France, October, 2025 frontiers in Neuroscience

METHODS
published: 03 July 2018
doi: 10.3389/fnins.2018.00368

Check for updates

From ERPs to MVPA Using the Amsterdam Decoding and Modeling Toolbox (ADAM)

Johannes J. Fahrenfort[1,2,3], Joram van Driel[1], Simon van Gaal[2,3] and Christian N. L. Olivers[1]*

[1] Department of Experimental and Applied Psychology, Institute Brain and Behavior Amsterdam (iBBA), VU University Amsterdam, Amsterdam, Netherlands, [2] Department of Psychology, University of Amsterdam, Amsterdam, Netherlands, [3] Amsterdam Brain and Cognition (ABC), University of Amsterdam, Amsterdam, Netherlands

Adam Features

- **Decoding (backward decoding, BDM):** Classifiers can be trained to discriminate between different experimental conditions based on EEG/MEG patterns.
- **Encoding modeling (forward encoding, FEM):** Allows reconstruction of “Channel Tuning Functions” (CTFs) or spatial maps of transformed weights (using the Haufe method) from the encoding model.
- **Temporal and generalization analyses (temporal generalization):** ADAM can compute “train vs. test at different times” matrices to examine the stability of neural representations over time.
- **Time–frequency domain decoding:** Before decoding, one can apply a time–frequency decomposition (TFR) and then perform decoding within separate frequency bands.
- **Statistics and multiple-comparison correction:** ADAM provides permutation testing, FDR correction, and group-level analyses to assess the statistical significance of results.
- **Visualization and publication-ready figures:** Generation of classifier performance time courses, topographical maps of weights, channel tuning functions, generalization matrices, and more.
- **Automatic control of potential biases (balanced design, AUC computation, etc.):** To prevent biases in the analyses, ADAM applies several default controls, such as balancing training classes and using AUC instead of raw accuracy.

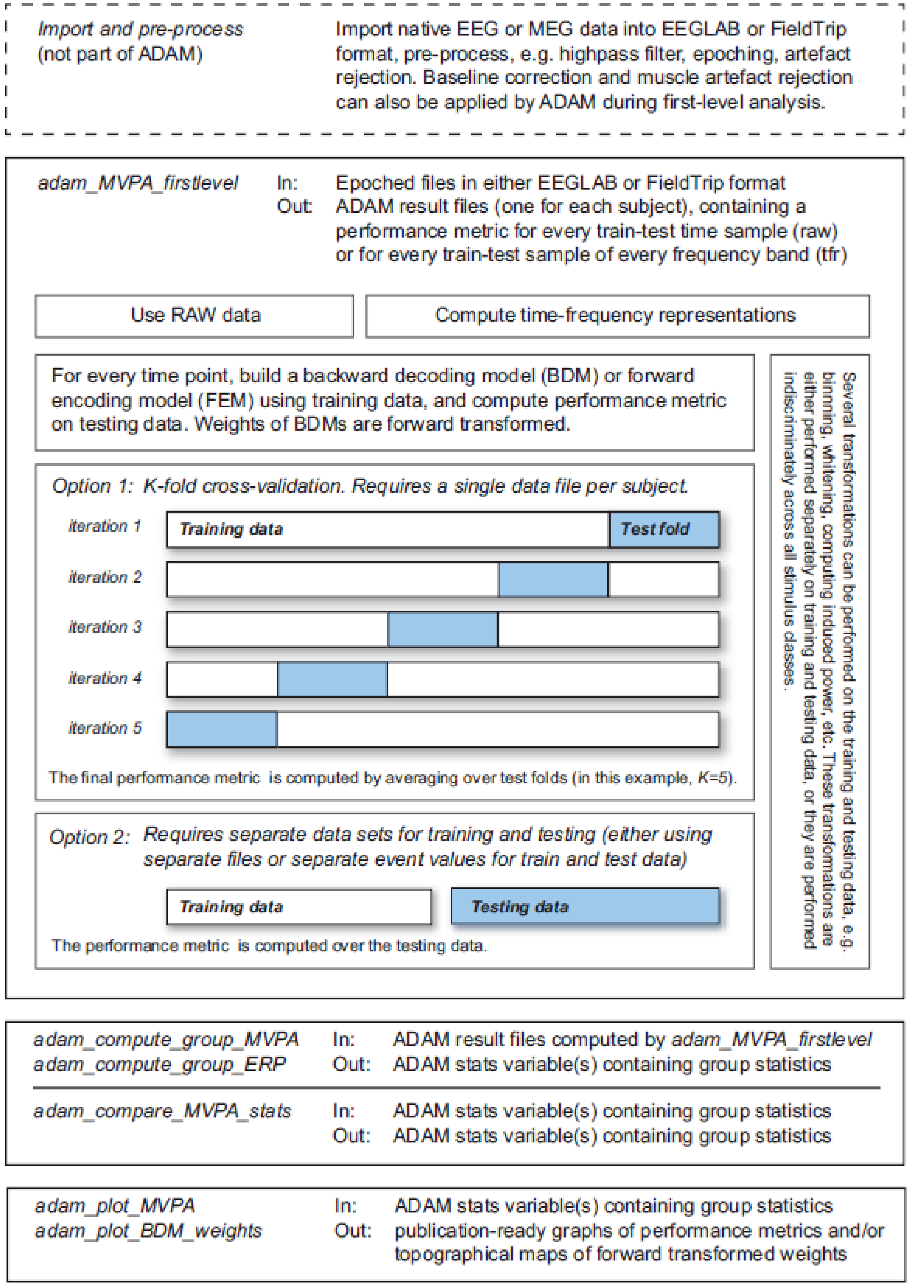


FIGURE 1 | ADAM processing pipeline, from top to bottom. The top left corner of each box states the ADAM function that performs the transformations that are described in the box. The top right describes the input-output transformation that the function performs. The output of each function serves as input for the function described in the box below it. The adam_MVPA_firstlevel box contains more detailed information about train-test algorithms. Further details about functionality and how to execute functions are provided in the main text.

Main functions

- adam_MVPA_firstlevel (computes and stores first level / single subject results)
- adam_compute_group_ERP (reads single subject ERPs and computes group statistics which can be plotted using adam_plot_MVPA)
- adam_compute_group_MVPA (reads single subject classification performance and computes group statistics which can be plotted using adam_plot_MVPA)
- adam_compare_MVPA_stats (compares outcomes of group analyses, which can be plotted using adam_plot_MVPA)
- adam_plot_MVPA (plots the outcome of the adam_compute_group_ or the adam_compare_MVPA_stats functions)
- adam_plot_BDM_weights (plots topomaps of the classifier weights or forward transformed weights, the latter of which are equivalent to univariate difference maps and are interpretable as neural sources, see (Haufe et al., 2014).

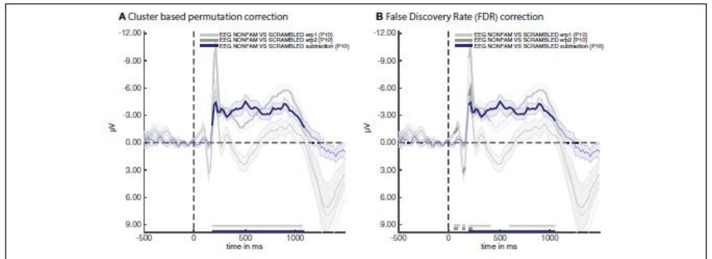


FIGURE 2 | ERPs for the non-familiar (erp1) and scrambled (erp2) condition as well as their difference (subtraction). (A) Thick lines denote $p < 0.05$ under two-sided cluster-based permutation (Maris and Oostenveld, 2007) (B) Thick lines denote $q < 0.05$ for dependent observations under FDR correction (Benjamini and Yekutieli, 2001).

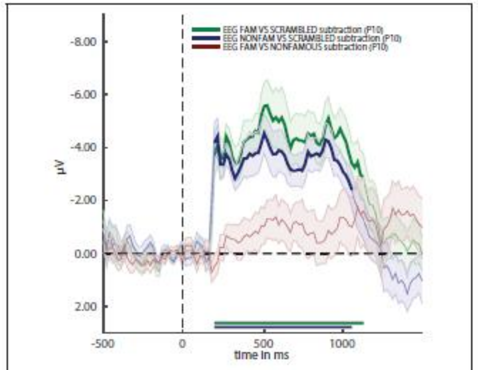


FIGURE 3 | ERP difference waves for the three comparisons in the experiment. Thick lines denote $p < 0.05$ under two-sided cluster-based permutation (Maris and Oostenveld, 2007).

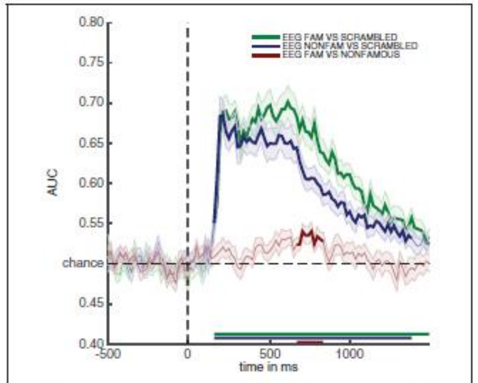


FIGURE 4 | Classification performance across time for the three comparisons in the experiment. Thick lines denote $p < 0.05$ under two-sided cluster-based permutation (Maris and Oostenveld, 2007).

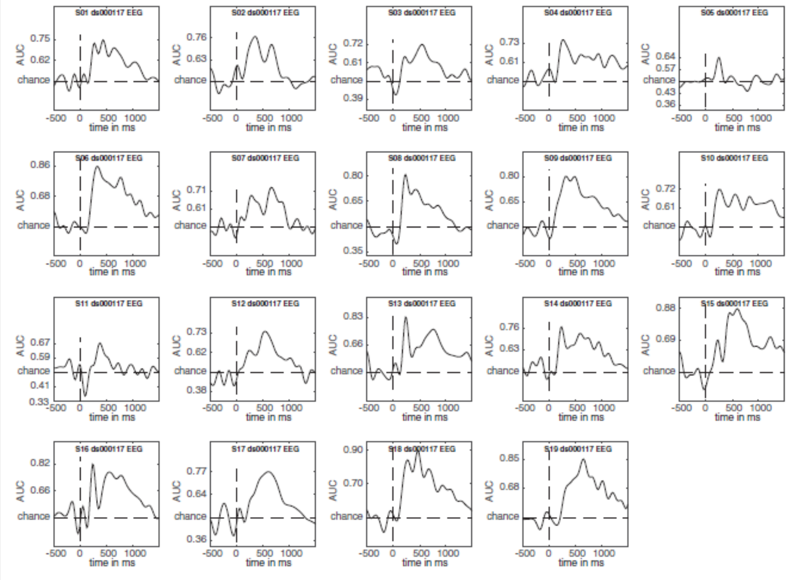


FIGURE 5 | Decoding results of individual subjects for the famous vs. scrambled faces comparison.

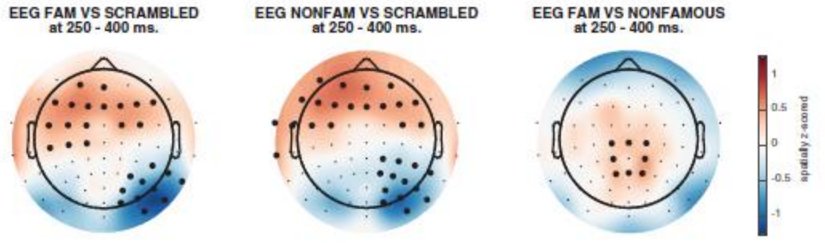


FIGURE 6 | Activation patterns from 250 to 400 ms, spatially normalized (z-scored) for every subject. Thick electrodes denote $p < 0.05$ under two-sided cluster-based permutation (Maris and Oostenveld, 2007).

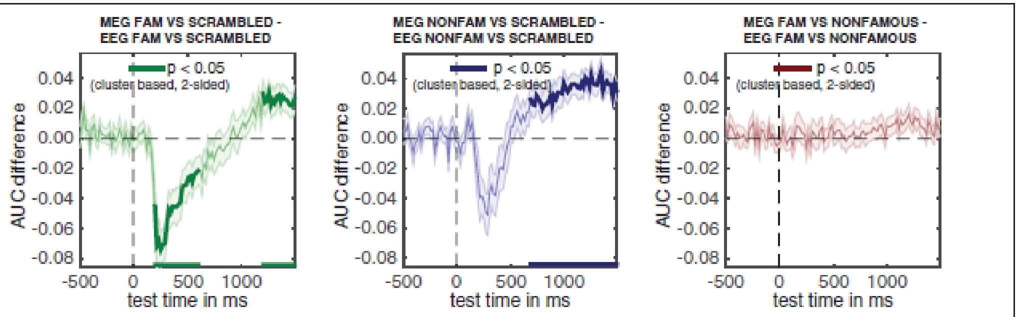


FIGURE 9 | Directly testing the difference in temporal generalization between MEG and EEG given the 250–400 ms training window.

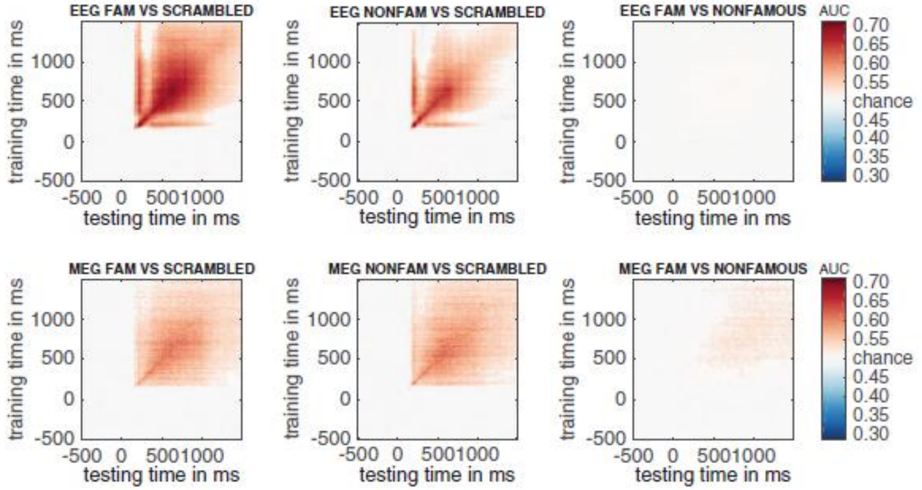


FIGURE 7 | Temporal generalization plots for all six analyses. The plots show the degree to which the classifier when trained on a given time point (on the y-axis), generalizes to other time points in the trial (on the x-axis). Color indicates classifier performance using AUC. The diagonal (from the left bottom to the top right) shows classification performance when the classifier is trained and tested on the same time point. More off-diagonal activity indicates stronger temporal generalization. (top row: EEG, bottom row: MEG).

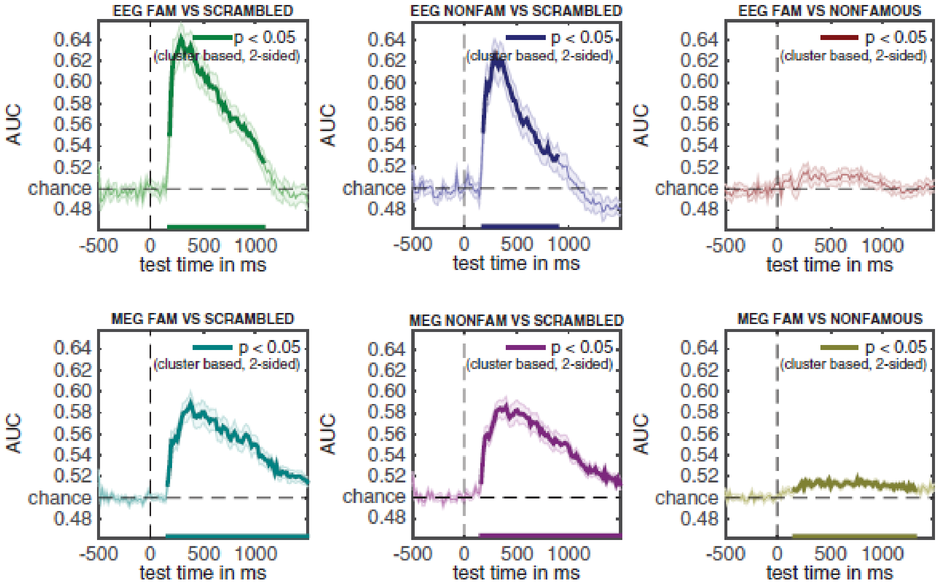


FIGURE 8 | Assessing cortical stability in EEG and MEG through generalization across time given a 250–400 ms training window.

```matlab
%% GENERAL SPECIFICATIONS OF THE EXPERIMENT
filenames = {
              'S01_ds000117_EEG' 'S01_ds000117_MEG_grad' ...
              'S02_ds000117_EEG' 'S02_ds000117_MEG_grad' ...
              'S03_ds000117_EEG' 'S03_ds000117_MEG_grad' ...
              'S04_ds000117_EEG' 'S04_ds000117_MEG_grad' ...
              'S05_ds000117_EEG' 'S05_ds000117_MEG_grad' ...
              'S06_ds000117_EEG' 'S06_ds000117_MEG_grad' ...
              'S07_ds000117_EEG' 'S07_ds000117_MEG_grad' ...
              'S08_ds000117_EEG' 'S08_ds000117_MEG_grad' ...
              'S09_ds000117_EEG' 'S09_ds000117_MEG_grad' ...
              'S10_ds000117_EEG' 'S10_ds000117_MEG_grad' ...
              'S11_ds000117_EEG' 'S11_ds000117_MEG_grad' ...
              'S12_ds000117_EEG' 'S12_ds000117_MEG_grad' ...
              'S13_ds000117_EEG' 'S13_ds000117_MEG_grad' ...
              'S14_ds000117_EEG' 'S14_ds000117_MEG_grad' ...
              'S15_ds000117_EEG' 'S15_ds000117_MEG_grad' ...
              'S16_ds000117_EEG' 'S16_ds000117_MEG_grad' ...
              'S17_ds000117_EEG' 'S17_ds000117_MEG_grad' ...
              'S18_ds000117_EEG' 'S18_ds000117_MEG_grad' ...
              'S19_ds000117_EEG' 'S19_ds000117_MEG_grad' ...
              };
eeg_filenames = file_list_restrict(filenames,'EEG'); % only EEG files
meg_filenames = file_list_restrict(filenames,'MEG'); % only MEG files

% event code specifications for factor stimulus type
famous_faces = [5 6 7];               % specifies event codes of all famous faces
nonfamous_faces = [13 14 15];         % specifies event codes of all non-famous faces
scrambled_faces = [17 18 19];         % specifies event codes of all scrambled faces

% event code specifications for factor stimulus repetition
first_presentation = [5 13 17];       % specifies event codes of all first presentations
immediate_repeat = [6 14 18];         % specifies event codes of all immediate repeats
delayed_repeat = [7 15 19];           % specifies event codes of all delayed repeats

% GENERAL ANALYSIS CONFIGURATION SETTINGS
cfg = [];                             % clear the config variable
cfg.datadir = 'C:\MY_EXP\DATA';       % this is where the data files are
cfg.model = 'BDM';                    % backward decoding ('BDM') or forward encoding ('FEM')
cfg.raw_or_tfr = 'raw';               % classify raw or time frequency representations ('tfr')
cfg.nfolds = 5;                       % the number of folds to use
cfg.class_method = 'AUC';             % the performance measure to use
cfg.crossclass = 'yes';               % whether to compute temporal generalization
cfg.channelpool = 'ALL_NOSELECTION'; % the channel selection to use
cfg.resample = 55;                    % downsample (useful for temporal generalization)
cfg.erp_baseline = [-.1,0];           % baseline correction in sec. ('no' for no correction)

% SPECIFIC SETTINGS: EEG NONFAMOUS VERSUS SCRAMBLED FACES
cfg.filenames = eeg_filenames;        % data filenames (EEG in this case)
cfg.class_spec{1} = cond_string(nonfamous_faces,first_presentation); % the first stimulus class
cfg.class_spec{2} = cond_string(scrambled_faces,first_presentation); % the second stimulus class
cfg.outputdir = 'C:\MY_EXP\RESULTS\EEG_RAW\EEG_NONFAM_VS_SCRAMBLED'; % output location
adam_MVPA_firstlevel(cfg);            % run first level analysis
```

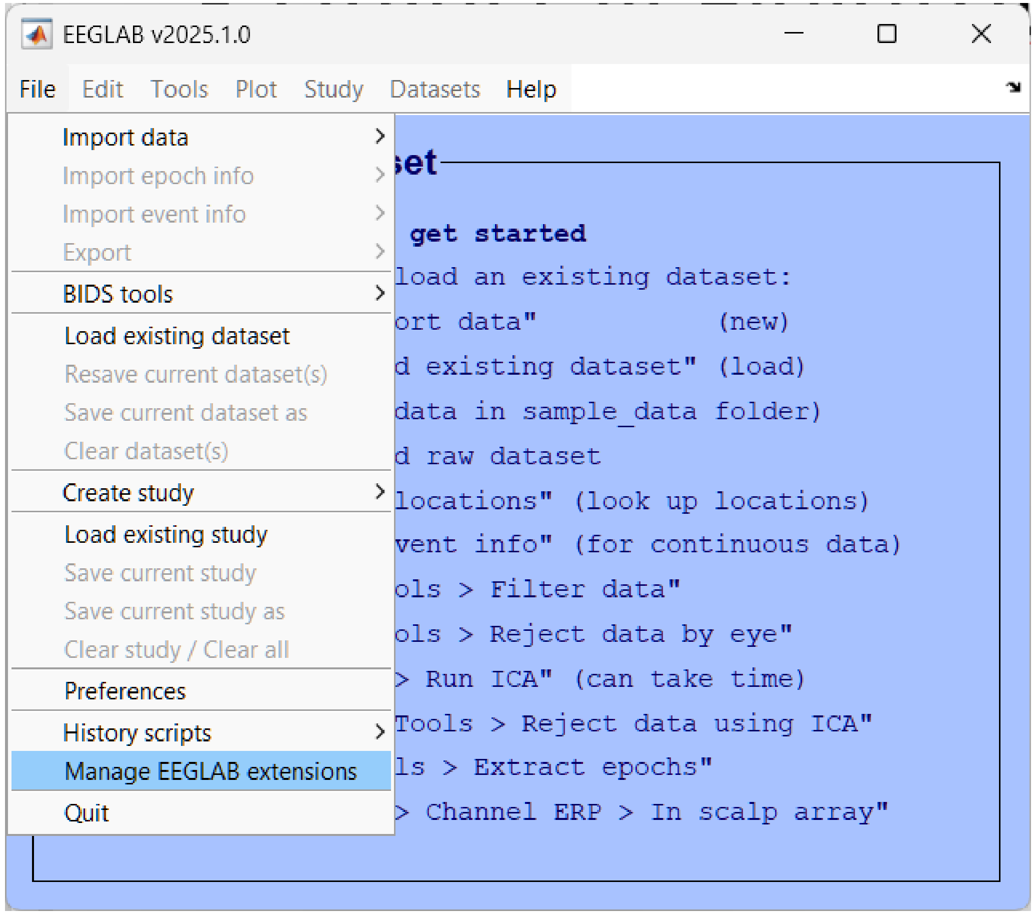
eeg_adam plugin

Manually install eeglab_adam plugin

- Unzip eeglab_adam.zip in eeglab/plugins folder
- Contains:
  - Fieldtrip special brew
  - ADAM toolbox
  - EEGLAB Study
  - Paper results

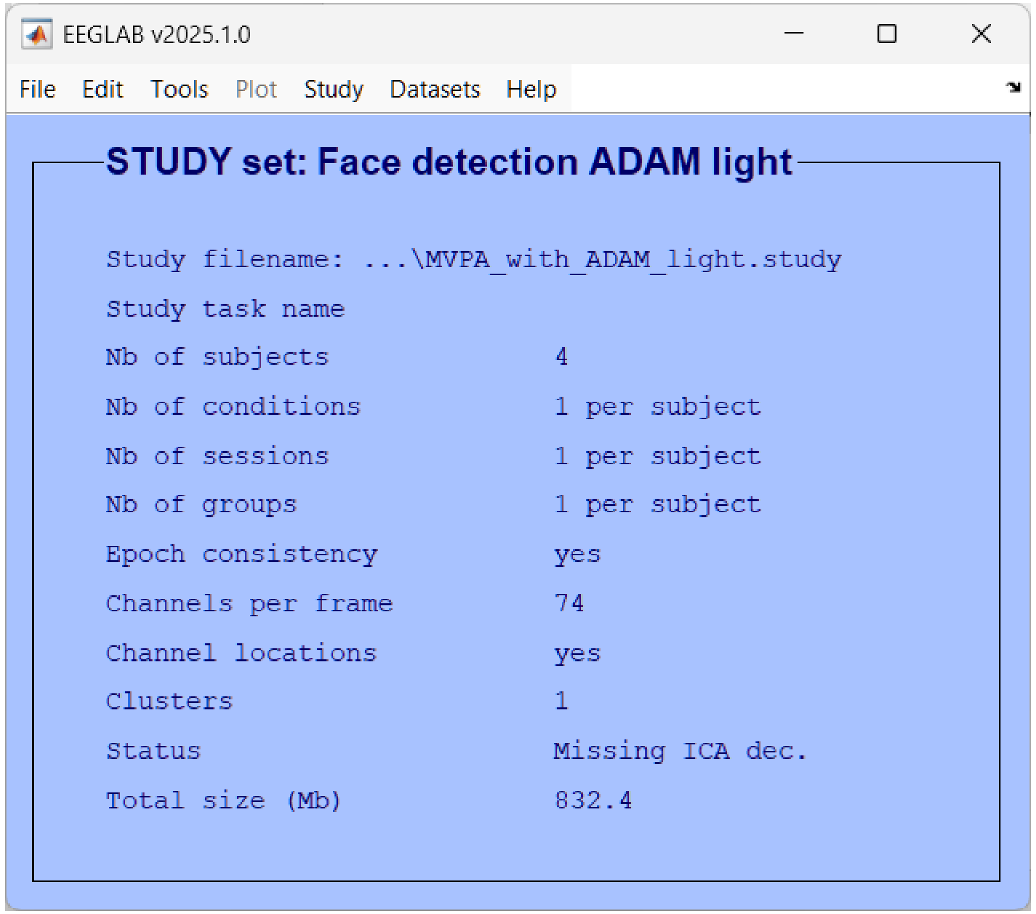
Setup : Step 1

- Uninstall Fieldtrip plugin using EEGLAB Extengion Manager



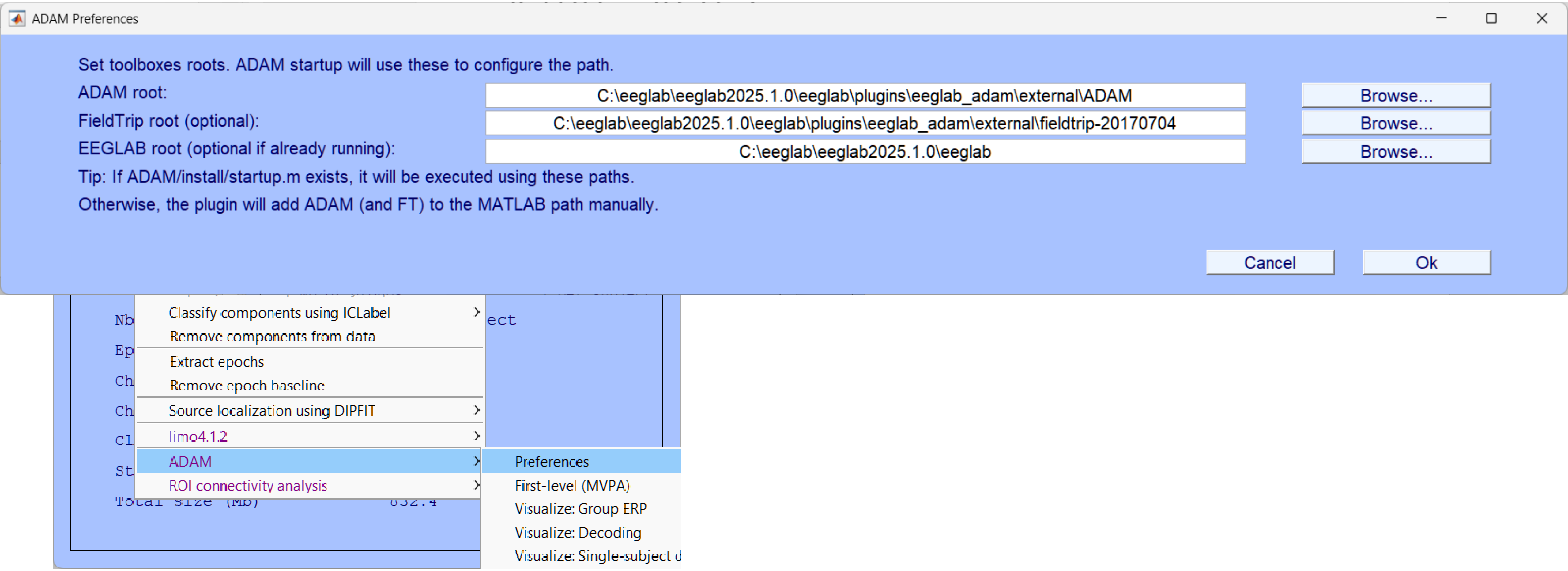
Setup : Step 2

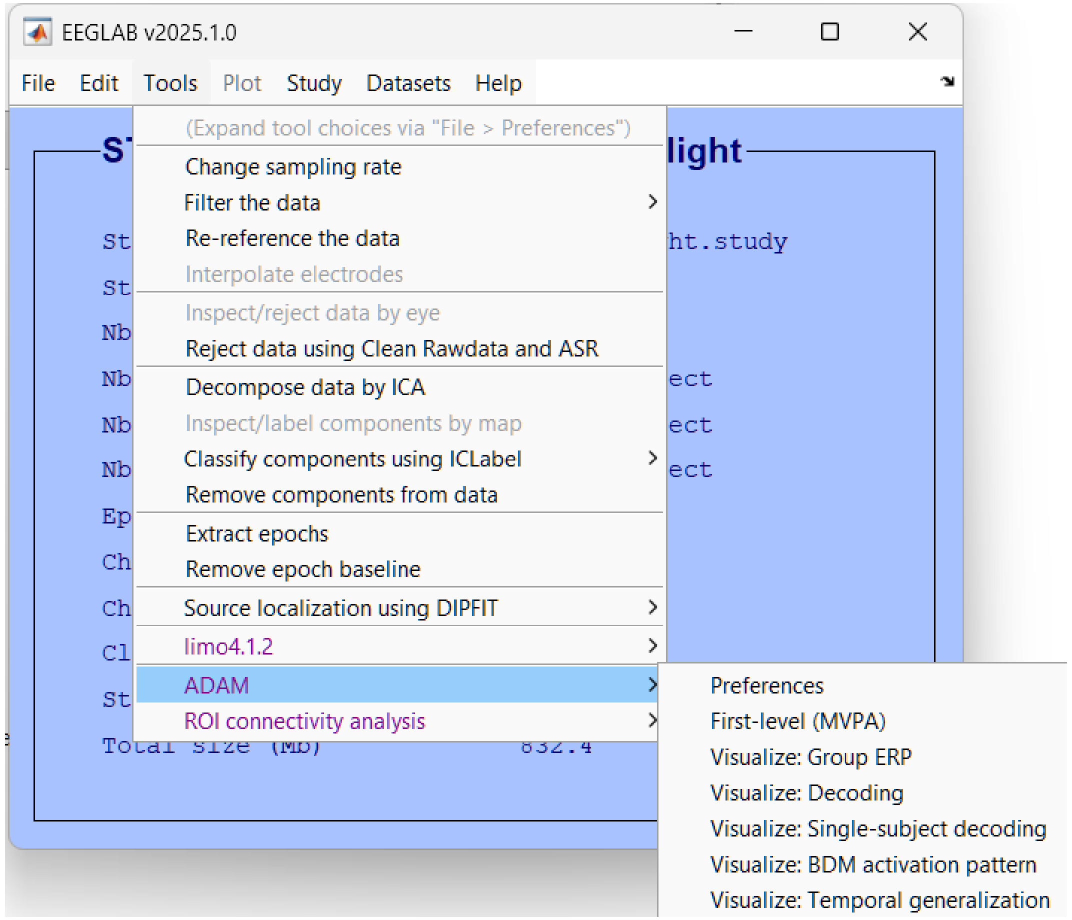
- Load STUDY eeglab_adam\ds002718\MVPA_with_ADAM_light.study



Setup : Step 3

- Set eeglab_adam plugin Preferences to setup proper paths to required toolboxes




EEGLAB v2025.1.0
File
Edit
Tools
Plot
Study
Datasets
Help
(Expand tool choices via "File > Preferences")
Change sampling rate
Filter the data
Re-reference the data
Interpolate electrodes
Inspect/reject data by eye
Reject data using Clean Rawdata and ASR
Decompose data by ICA
Inspect/label components by map
Classify components using ICLabel
Remove components from data
Extract epochs
Remove epoch baseline
Source localization using DIPFIT
limo4.1.2
ADAM
ROI connectivity analysis
Preferences
First-level (MVPA)
Visualize: Group ERP
Visualize: Decoding
Visualize: Single-subject decoding
Visualize: BDM activation pattern
Visualize: Temporal generalization