
CM-graph Documentation

Release 0.1.0

Gansheng TAN

Jul 06, 2022

CONTENTS:

1 Authors: 1

2 Introduction 3

2.1 1. To start with 3

2.2 2. Installation 3

3 Examples 5

4 The description of jupyter notebook examples 7

4.1 a) preprocessing and alignment of EEG-EMG co-registration 7

5 prep 9

6 Example gallery for CM-graph 13

6.1 Introductory example - Plotting sin 13

7 Indices and tables 15

Python Module Index 17

Index 19

CHAPTER
ONE

AUTHORS:

Gansheng TAN

Thanks also to Jinbiao Liu for providing test data.

INTRODUCTION

Hint: If you would like to contribute or share your insight about multichannel brain-muscle graph analysis, join us by contacting Gansheng TAN and upload your codes (see the contributing [advices](#)).

2.1 1. To start with

2.1.1 Sample data

Data architecture

Data clarification

2.1.2 Script functions

2.2 2. Installation

At current stage, please download the .zip file from Github.

CHAPTER
THREE

EXAMPLES

THE DESCRIPTION OF JUPYTER NOTEBOOK EXAMPLES

Please download the notebooks in `CM-graph/example_notebook`

4.1 a) preprocessing and alignment of EEG-EMG co-registration

`prep_RJ_paradigm_ready4package.ipynb`

The py examples are generated in the gallery

A module for io and preprocessing

```
cmgraph.prep.eeg_emg_alignment(eeg_fName, emg_df, sfreq_final, emg_freq, report_fName=None,  
                               start_marker=True, fir=[1, None], PREP=True, montage='standard_1020')
```

This function takes .set format for eeg and txt format for emg.

Parameters

eeg_fName

[string] eeg_fName should be in .set form

emg_fName

[string] emg_fName should be in .txt form

report_fName: string

optional, default to None which will not generate a report for the reprocessing result. When report_fName is specified, two reports will be generated at the suggested directory including one in HTML, and one in .h5 format. The .h5 format is an edittable report.

montage: string

The default montage is set to standard 1020 montage

start_marker

[boolean] if start_marker is true, the segment before the first marker will be cropped, defaults to True

fir: list

the lower and upper boundary of filter. If the boundary is set to None, then the filter become high-pass or low-pass filter.

PREP: boolean

the EEG preprocessing pipeline process, defaults to True. It can be deactivated when set to false.

emg_chs_selected

[list of int] the index of emg channels (columns) that are supposed to be used in further analysis, defaults to 'all'

Returns

mne raw object containing aligned eeg and emg data

Notes

Make sure the eeg recording are no less than the emg recording. The current version of this function crop emg signal with respect to emg in order to keep these data aligned.

Examples

```
>>>eeg_fName = r'D:DataRuiJinFirstStroke11JanEEGsubj1_healthy_session1.set' >>>emg_fName =
r'D:DataRuiJinFirstStroke11JanEMGsubj1_healthy_session1.txt' >>>emg_df = pd.read_csv(emg_fName,
header = None, skiprows=3,
sep = ' ',engine = 'python')
>>>eeg_emg_alignment(eeg_fName,emg_df,emg_freq=1000,sfreq_final=500,report_fName=None,PREP=False)
cmgraph.prep.emg_io(emg_fName, skiprows, sep=' ', emg_chs_selected='all')
this function takes .txt format emg file and renders to a pandas dataframe for further processing.
```

Parameters

emg_fName

[string] emg_fName should be in .txt or in .csv form.

emg_chs_selected

[list of int] the index of emg channels (columns) that are supposed to be used in further analysis, defaults to 'all'

sep

[string] the separator to be specified when reading the txt file with pandas.read_csv

skiprows

[int] rows to be skipped. This applies for data containing emg information at the beginning of the data file.

Returns

pandas's dataframe

Notes

The .txt file or .csv file should not have headers. If so, please use skiprow to trim them. The first column should be 0 in the emg_chs_selected parameter

Examples

```
>>>emg_fName = r'D:DataRuiJinFirstStroke11JanEMGsubj1_healthy_session1.txt' >>>emg_io(emg_fName,
skiprows = 3, emg_chs_selected='all')
cmgraph.prep.epochs_basedon_emg(raw_hybrid, ref_emg, windowLen, step=100, threshold=0.5,
report_fName=None, add_to_existed_report=False, reject_criteria={'eeg':
0.0003, 'emg': 10000000000.0}, flat_criteria={'eeg': 5e-07}, tmin=0.0,
tmax=3.0, save_fName=None)
```

this function takes .set format for eeg and txt format for emg.

Parameters

raw_hybrid: mne raw object

The raw object containing aligned eeg and emg

ref_emg: list of string

the emg channels that the algorithm refers to. They should be channels in input raw_hybrid. The algorithm will consider 'the length of the string' as number of movements.

report_fName: string

optional, default to None which will not generate a report for the epoching result. When reproty_fName is specified, two reports will be generated at the suggested directory including one in HTML, and one in .h5 format. The .h5 format is an edittable report.

add_to_existed_report: boolean

It is supposed to be true when one wants to add the epoching report into an existed report.

windowLen: int

rough duration estimation of the movement

step: int

equivalent to resolution, defaults to 100

threshold: float

the threshold should be in [0,1], represnting the quantile of the energy for all windows

reject_criteria: dict

epochs containing eeg and emg that exceed rejection criteria will be rejected. defaults to dict(eeg=30e-5,emg=1e10)

flat_criteria: dict

epochs containing eeg and emg whose amplitude are lower than flat criteria will be rejected. Defaults to dict(eeg=1e-6). If reject_criteria and flat_criteria are both set to None, then no epochs rejection would take place

tmin: float

the begining of the epochs with respect to the movement onsets

tmax: float

the end of the epochs with respect to the movement onsets

savefName: string

the path and filename of the saving epochs. It defaults to None, that is the epochs would not be saved

Returns

mne epochs

See also:

[*eeg_emg_alignment*](#)

cmgraph.prep.**firstOnsetD**(*possibleOnsets*)

an auxilary function that identify true movement onsets for all the possible onsets identified based on energy threshold.

Parameters**possibleOnsets: list**

list of all the possible onsets

Returns

true movement onsets

See also:

eeg_emg_alignment

EXAMPLE GALLERY FOR CM-GRAPH

Below is a gallery of examples

6.1 Introductory example - Plotting sin

This is a general example demonstrating a Matplotlib plot output, embedded rST, the use of math notation and cross-linking to other examples. It would be useful to compare the source Python file with the output below. Source files for gallery examples should start with a triple-quoted header docstring. Anything before the docstring is ignored by Sphinx-Gallery and will not appear in the rendered output, nor will it be executed. This docstring requires a rST header, which is used as the title of the example and to correctly build cross-referencing links. Code and embedded rST text blocks follow the docstring. The first block immediately after the docstring is deemed a code block, by default, unless you specify it to be a text block using a line of `#'s` or `###` (see below). All code blocks get executed by Sphinx-Gallery and any output, including plots will be captured. Typically, code and text blocks are interspersed to provide narrative explanations of what the code is doing or interpretations of code output. Mathematical expressions can be included as LaTeX, and will be rendered with MathJax. To include displayed math notation, use the directive `.. math::`. To include inline math notation use the `:math:` role. For example, we are about to plot the following function: `.. math:`

```
x \rightarrow \sin(x)
```

Here the function `sin` is evaluated at each point the variable x is defined. When including LaTeX in a Python string, ensure that you escape the backslashes or use a raw docstring. You do not need to do this in text blocks (see below).

```
# Code source: Óscar Nájera
# License: BSD 3 clause

import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 2 * np.pi, 100)
y = np.sin(x)

plt.plot(x, y)
plt.xlabel(r'$x$')
plt.ylabel(r'$\sin(x)$')
# To avoid matplotlib text output
plt.show()
```

To include embedded rST, use a line of `>= 20 #'s` or `###` between your rST and your code (see `embedding_rst`). This separates your example into distinct text and code blocks. You can continue writing code below the embedded rST text block:

```
print('This example shows a sin plot!')
```

LaTeX syntax in the text blocks does not require backslashes to be escaped:

sin

6.1.1 Cross referencing

You can refer to an example from any part of the documentation, including from other examples. Sphinx-Gallery automatically creates reference labels for each example. The label consists of the `.py` file name, prefixed with `sphx_glr_` and the name of the folder(s) the example is in. In this case, the example we want to cross-reference is in `auto_examples` (the `gallery_dirs`; see `configure_and_use_sphinx_gallery`), then the subdirectory `no_output` (since the example is within a sub-gallery). The file name of the example is `plot_syntaxerror.py`. We can thus cross-link to the example ‘SyntaxError’ using: `:ref:`sphx_glr_auto_examples_no_output_plot_syntaxerror.py``.

See also:

`sphx_glr_auto_examples_no_output_plot_syntaxerror.py` for a an example with an error.

Total running time of the script: (0 minutes 0.000 seconds)

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

C

`cmgraph.prep`, [7](#)

INDEX

C

`cmgraph.prep`
module, [7](#)

E

`eeg_emg_alignment()` (*in module cmgraph.prep*), [9](#)
`emg_io()` (*in module cmgraph.prep*), [10](#)
`epochs_basedon_emg()` (*in module cmgraph.prep*), [10](#)

F

`firstOnsetD()` (*in module cmgraph.prep*), [11](#)

M

module
`cmgraph.prep`, [7](#)