Open Software in Open Science

Dr. Britta Westner

Open Science and Reproducibility Workshop
March 12, 2019

AARHUS UNIVERSITY

✉ britta@cfin.au.dk @britta_wstnr britta-wstnr

Outline

Let's find answers to the following questions:

- Why is open source **essential for open science**?
- What are **best practices** for open tools?
- How does all this facilitate **reproducibility**?
- Is there an **open source crisis**?

What is open source?

Software is *open source* if the source code

- is **freely available**
- may be **modified**
- may be **redistributed**

Why does open science need open source?

What I cannot create, I do not understand.

Richard Feynman, 1988

Black boxes do not belong in science.

Fernando Pérez, 2017

Open source science

For **reproducibility** of results, the following things need to be considered:

- **computational tools**: your scripts, toolboxes, programming language, operating system, . . .
- the **data**
- **sharing** of the work
- **communicating** the work

Open source science

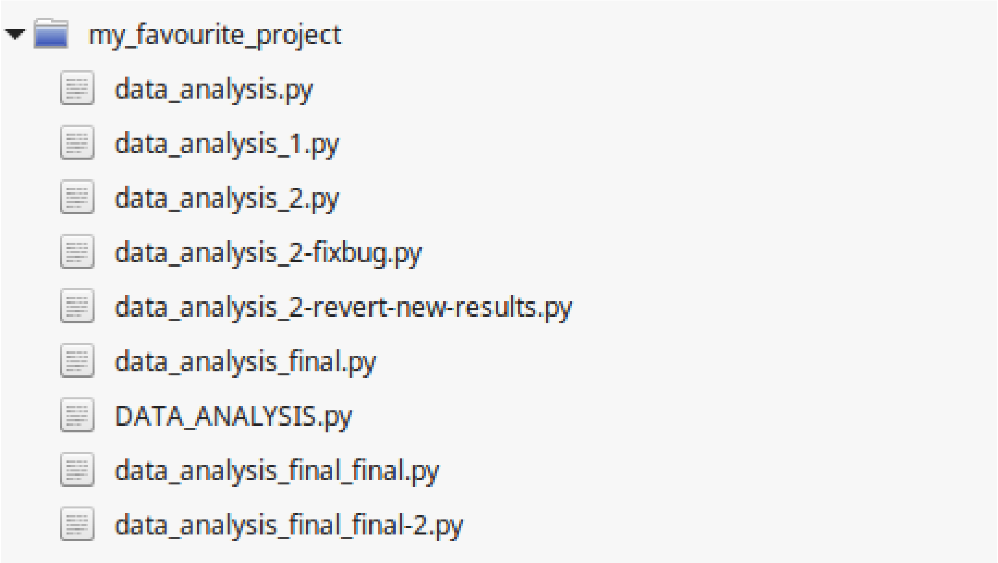
For **reproducibility** of results, the following things need to be considered:

- **computational tools**: **use open tools and share your code**
- the **data**: **share**
- **sharing** of the work: **in an easily accessible manner**
- **communicating** the work: **publish, tweet, . . . – and include links to code and data!**

Making data analyses reproducible

Reproducibility starts with you.

Looks familiar?



my_favourite_project
data_analysis.py
data_analysis_1.py
data_analysis_2.py
data_analysis_2-fixbug.py
data_analysis_2-revert-new-results.py
data_analysis_final.py
DATA_ANALYSIS.py
data_analysis_final_final.py
data_analysis_final_final-2.py

- can you reproduce your own results at a later stage?
- use **version control**
- **document** your code

A word about version control

Using version control provides you with your own **time machine**.

Principle:

- you are responsbile for time stamps
- file only exists in most recent version
- log of changes
- recommendation: **git**

POLICE PUBLIC CALL BOX
POLICE TELEPHONE
FREE
FOR USE OF
PUBLIC
ADVICE & ASSISTANCE OBTAINABLE IMMEDIATELY
OFFICERS & CARS RE POND TO ALL CALL
PULL TO OPEN

photograph by Babbel1996 / CC-BY-2.5

Making code public

Where?

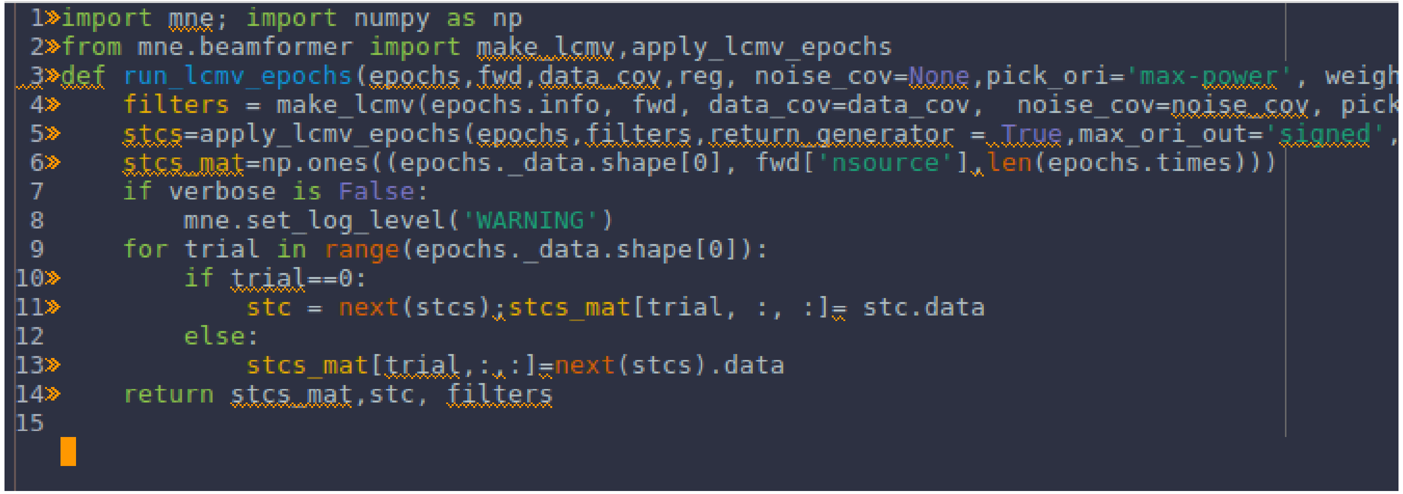
GitHub GitLab GitBucket

How? Etiquette for sharing code.

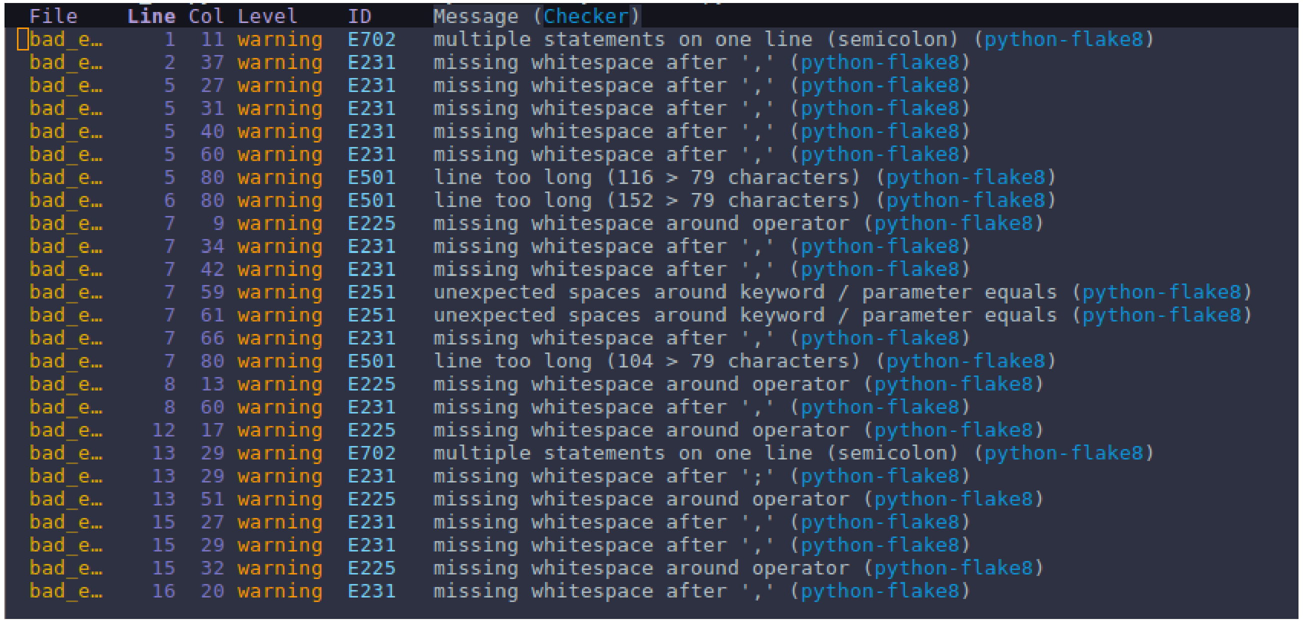
- include a **license**
- share your code **formatted**: line width, coding stlyes (linters)
- **document** your code: comments, docstrings, project description
- note down **dependencies** and versions

Got style?

A demonstration how coding styles make things easier.



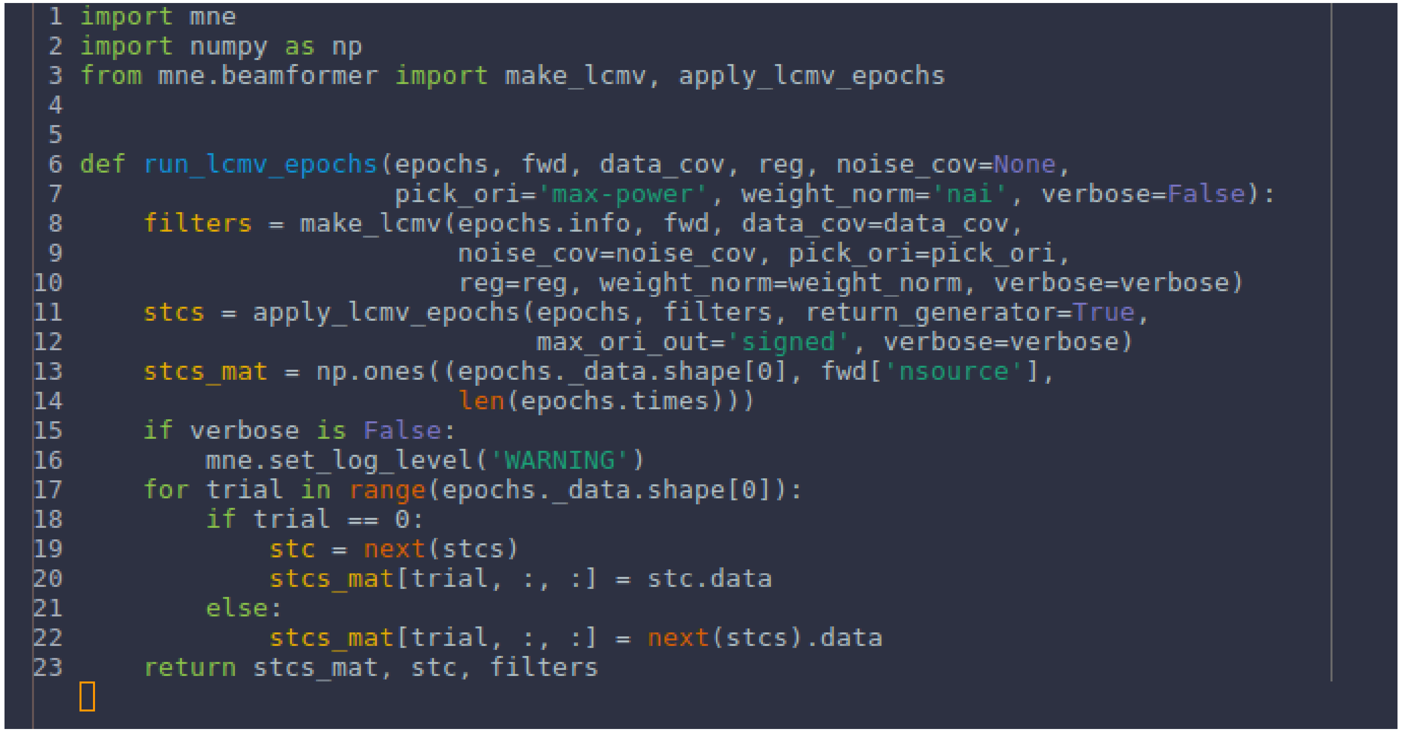
```
 1»import mne; import numpy as np
 2»from mne.beamformer import make_lcmv,apply_lcmv_epochs
 3»def run_lcmv_epochs(epochs,fwd,data_cov,reg, noise_cov=None,pick_ori='max-power', weigh
 4»    filters = make_lcmv(epochs.info, fwd, data_cov=data_cov,  noise_cov=noise_cov, pick
 5»    stcs=apply_lcmv_epochs(epochs,filters,return_generator = True,max_ori_out='signed',
 6»    stcs_mat=np.ones((epochs._data.shape[0], fwd['nsource'],len(epochs.times)))
 7      if verbose is False:
 8          mne.set_log_level('WARNING')
 9      for trial in range(epochs._data.shape[0]):
10»        if trial==0:
11»            stc = next(stcs);stcs_mat[trial, :, :]= stc.data
12          else:
13»            stcs_mat[trial,:,:]=next(stcs).data
14»    return stcs_mat,stc, filters
15
```



```
 File     Line Col Level    ID     Message (Checker)
 bad_e…      1  11 warning  E702   multiple statements on one line (semicolon) (python-flake8)
 bad_e…      2  37 warning  E231   missing whitespace after ',' (python-flake8)
 bad_e…      5  27 warning  E231   missing whitespace after ',' (python-flake8)
 bad_e…      5  31 warning  E231   missing whitespace after ',' (python-flake8)
 bad_e…      5  40 warning  E231   missing whitespace after ',' (python-flake8)
 bad_e…      5  60 warning  E231   missing whitespace after ',' (python-flake8)
 bad_e…      5  80 warning  E501   line too long (116 > 79 characters) (python-flake8)
 bad_e…      6  80 warning  E501   line too long (152 > 79 characters) (python-flake8)
 bad_e…      7   9 warning  E225   missing whitespace around operator (python-flake8)
 bad_e…      7  34 warning  E231   missing whitespace after ',' (python-flake8)
 bad_e…      7  42 warning  E231   missing whitespace after ',' (python-flake8)
 bad_e…      7  59 warning  E251   unexpected spaces around keyword / parameter equals (python-flake8)
 bad_e…      7  61 warning  E251   unexpected spaces around keyword / parameter equals (python-flake8)
 bad_e…      7  66 warning  E231   missing whitespace after ',' (python-flake8)
 bad_e…      7  80 warning  E501   line too long (104 > 79 characters) (python-flake8)
 bad_e…      8  13 warning  E225   missing whitespace around operator (python-flake8)
 bad_e…      8  60 warning  E231   missing whitespace after ',' (python-flake8)
 bad_e…     12  17 warning  E225   missing whitespace around operator (python-flake8)
 bad_e…     13  29 warning  E702   multiple statements on one line (semicolon) (python-flake8)
 bad_e…     13  29 warning  E231   missing whitespace after ';' (python-flake8)
 bad_e…     13  51 warning  E225   missing whitespace around operator (python-flake8)
 bad_e…     15  27 warning  E231   missing whitespace after ',' (python-flake8)
 bad_e…     15  29 warning  E231   missing whitespace after ',' (python-flake8)
 bad_e…     15  32 warning  E225   missing whitespace around operator (python-flake8)
 bad_e…     16  20 warning  E231   missing whitespace after ',' (python-flake8)
```

Got style?

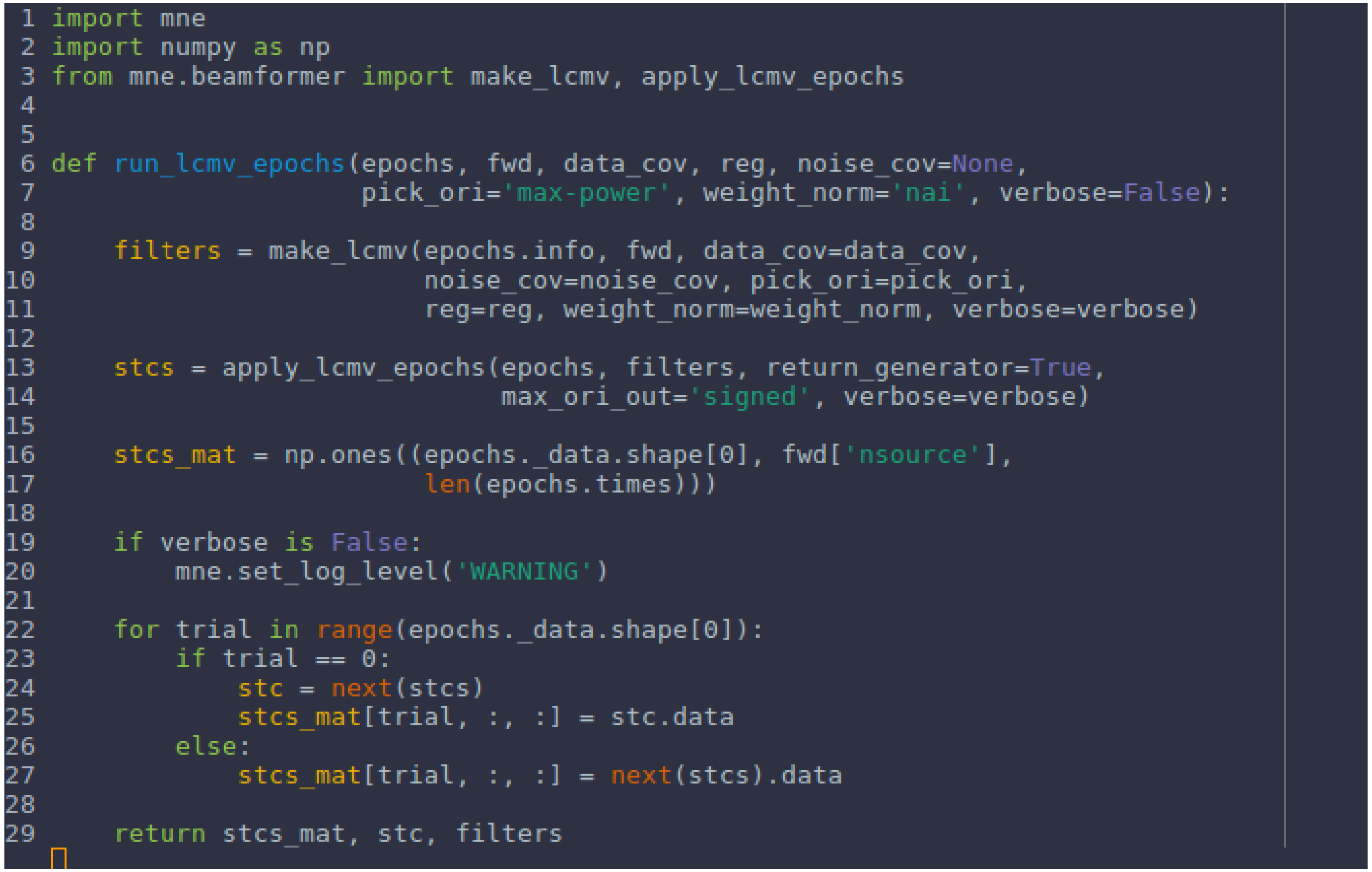
A demonstration how coding styles make things easier.



```python
import mne
import numpy as np
from mne.beamformer import make_lcmv, apply_lcmv_epochs


def run_lcmv_epochs(epochs, fwd, data_cov, reg, noise_cov=None,
                    pick_ori='max-power', weight_norm='nai', verbose=False):
    filters = make_lcmv(epochs.info, fwd, data_cov=data_cov,
                        noise_cov=noise_cov, pick_ori=pick_ori,
                        reg=reg, weight_norm=weight_norm, verbose=verbose)
    stcs = apply_lcmv_epochs(epochs, filters, return_generator=True,
                             max_ori_out='signed', verbose=verbose)
    stcs_mat = np.ones((epochs._data.shape[0], fwd['nsource'],
                        len(epochs.times)))
    if verbose is False:
        mne.set_log_level('WARNING')
    for trial in range(epochs._data.shape[0]):
        if trial == 0:
            stc = next(stcs)
            stcs_mat[trial, :, :] = stc.data
        else:
            stcs_mat[trial, :, :] = next(stcs).data
    return stcs_mat, stc, filters
```

Got style?

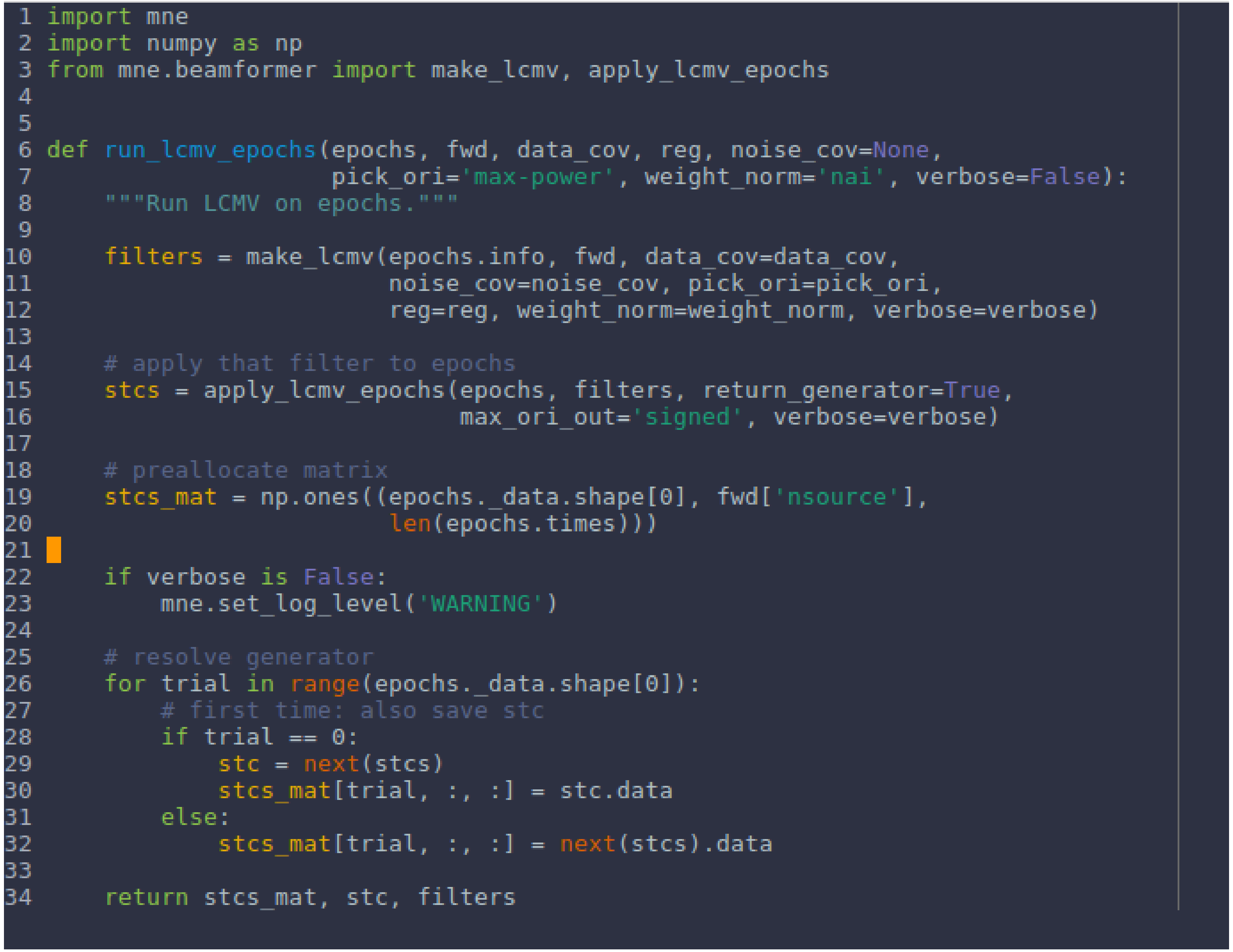
A demonstration how coding styles make things easier.



```python
import mne
import numpy as np
from mne.beamformer import make_lcmv, apply_lcmv_epochs


def run_lcmv_epochs(epochs, fwd, data_cov, reg, noise_cov=None,
                    pick_ori='max-power', weight_norm='nai', verbose=False):

    filters = make_lcmv(epochs.info, fwd, data_cov=data_cov,
                        noise_cov=noise_cov, pick_ori=pick_ori,
                        reg=reg, weight_norm=weight_norm, verbose=verbose)

    stcs = apply_lcmv_epochs(epochs, filters, return_generator=True,
                             max_ori_out='signed', verbose=verbose)

    stcs_mat = np.ones((epochs._data.shape[0], fwd['nsource'],
                        len(epochs.times)))

    if verbose is False:
        mne.set_log_level('WARNING')

    for trial in range(epochs._data.shape[0]):
        if trial == 0:
            stc = next(stcs)
            stcs_mat[trial, :, :] = stc.data
        else:
            stcs_mat[trial, :, :] = next(stcs).data

    return stcs_mat, stc, filters
```

Got style?

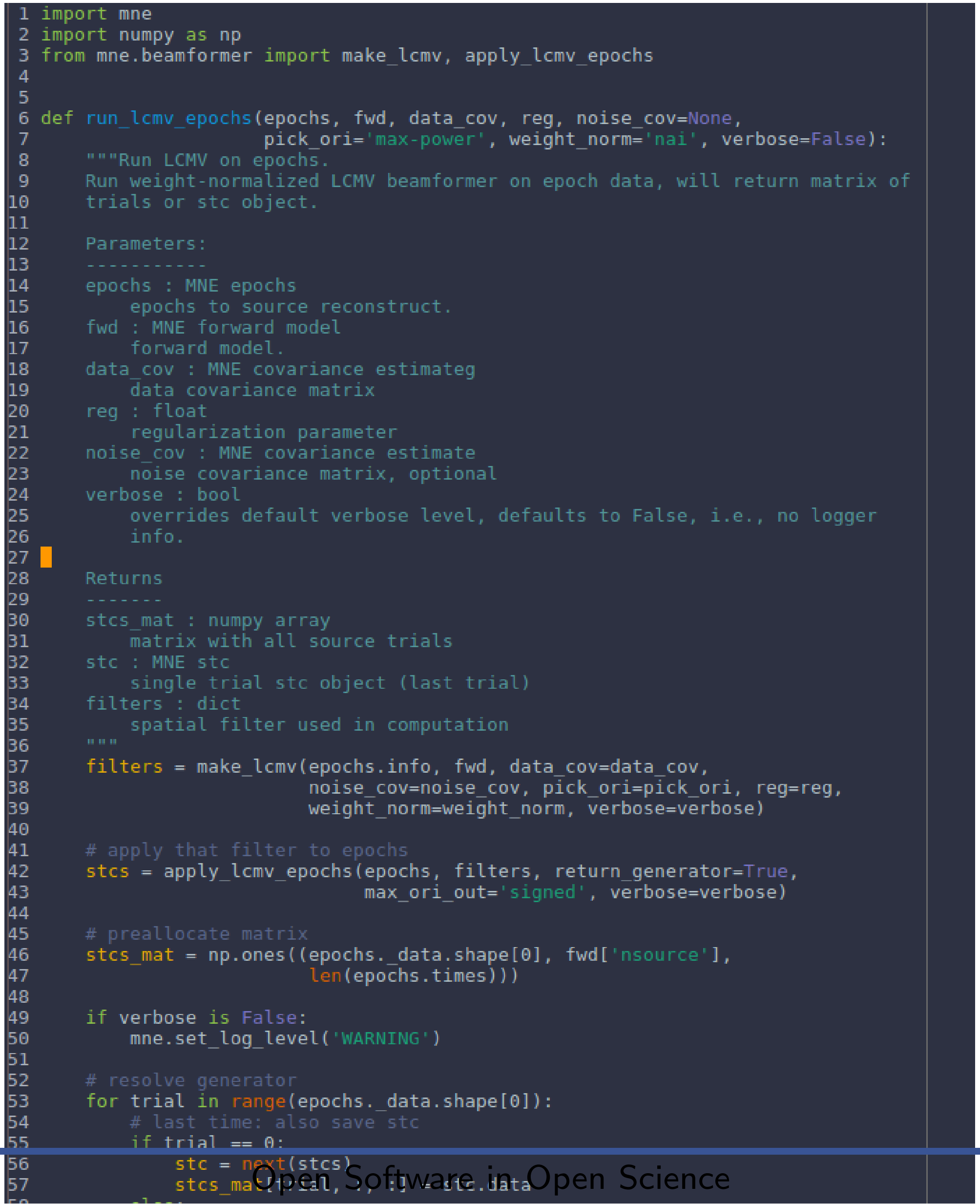
A demonstration how coding styles make things easier.



```python
import mne
import numpy as np
from mne.beamformer import make_lcmv, apply_lcmv_epochs


def run_lcmv_epochs(epochs, fwd, data_cov, reg, noise_cov=None,
                    pick_ori='max-power', weight_norm='nai', verbose=False):
    """Run LCMV on epochs."""

    filters = make_lcmv(epochs.info, fwd, data_cov=data_cov,
                        noise_cov=noise_cov, pick_ori=pick_ori,
                        reg=reg, weight_norm=weight_norm, verbose=verbose)

    # apply that filter to epochs
    stcs = apply_lcmv_epochs(epochs, filters, return_generator=True,
                             max_ori_out='signed', verbose=verbose)

    # preallocate matrix
    stcs_mat = np.ones((epochs._data.shape[0], fwd['nsource'],
                        len(epochs.times)))

    if verbose is False:
        mne.set_log_level('WARNING')

    # resolve generator
    for trial in range(epochs._data.shape[0]):
        # first time: also save stc
        if trial == 0:
            stc = next(stcs)
            stcs_mat[trial, :, :] = stc.data
        else:
            stcs_mat[trial, :, :] = next(stcs).data

    return stcs_mat, stc, filters
```

Got style?

A demonstration how coding styles make things easier.



```python
import mne
import numpy as np
from mne.beamformer import make_lcmv, apply_lcmv_epochs


def run_lcmv_epochs(epochs, fwd, data_cov, reg, noise_cov=None,
                    pick_ori='max-power', weight_norm='nai', verbose=False):
    """Run LCMV on epochs.
    Run weight-normalized LCMV beamformer on epoch data, will return matrix of
    trials or stc object.

    Parameters:
    -----------
    epochs : MNE epochs
        epochs to source reconstruct.
    fwd : MNE forward model
        forward model.
    data_cov : MNE covariance estimateg
        data covariance matrix
    reg : float
        regularization parameter
    noise_cov : MNE covariance estimate
        noise covariance matrix, optional
    verbose : bool
        overrides default verbose level, defaults to False, i.e., no logger
        info.

    Returns
    -------
    stcs_mat : numpy array
        matrix with all source trials
    stc : MNE stc
        single trial stc object (last trial)
    filters : dict
        spatial filter used in computation
    """
    filters = make_lcmv(epochs.info, fwd, data_cov=data_cov,
                        noise_cov=noise_cov, pick_ori=pick_ori, reg=reg,
                        weight_norm=weight_norm, verbose=verbose)

    # apply that filter to epochs
    stcs = apply_lcmv_epochs(epochs, filters, return_generator=True,
                             max_ori_out='signed', verbose=verbose)

    # preallocate matrix
    stcs_mat = np.ones((epochs._data.shape[0], fwd['nsource'],
                        len(epochs.times)))

    if verbose is False:
        mne.set_log_level('WARNING')

    # resolve generator
    for trial in range(epochs._data.shape[0]):
        # last time: also save stc
        if trial == 0:
            stc = next(stcs)
            stcs_mat[trial, :, :] = stc.data
        else:
```

How GitHub facilitates open science

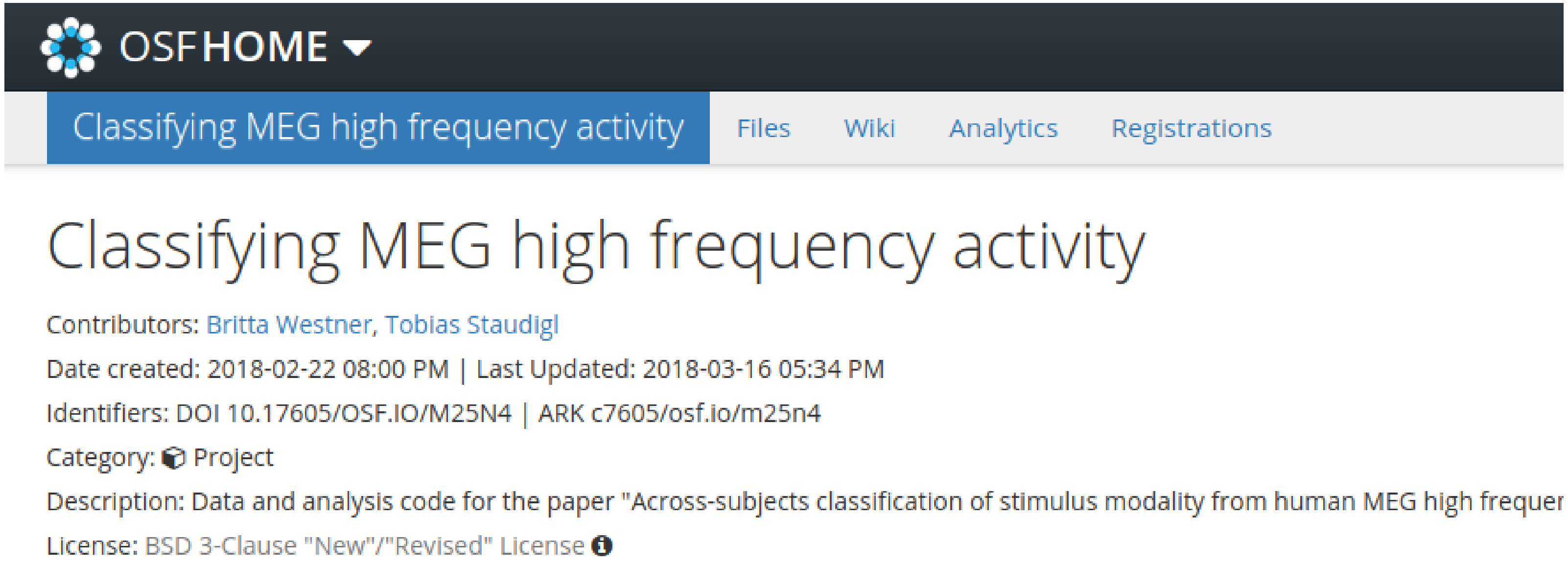
On GitHub*/Lab/Bucket you can:

- **share** code
- **follow** researchers and toolboxes to stay up-to-date
- **collaborate** on projects
- **fork** projects to make your own version of them
- **contribute** to projects, e.g., open source toolboxes

* GitHub itself is **not** open source!

Making data public

For full reproducibility, **data** is needed.
One possibility for sharing data: **The Open Science Framework**



Classifying MEG high frequency activity

Contributors: Britta Westner, Tobias Staudigl

Date created: 2018-02-22 08:00 PM | Last Updated: 2018-03-16 05:34 PM

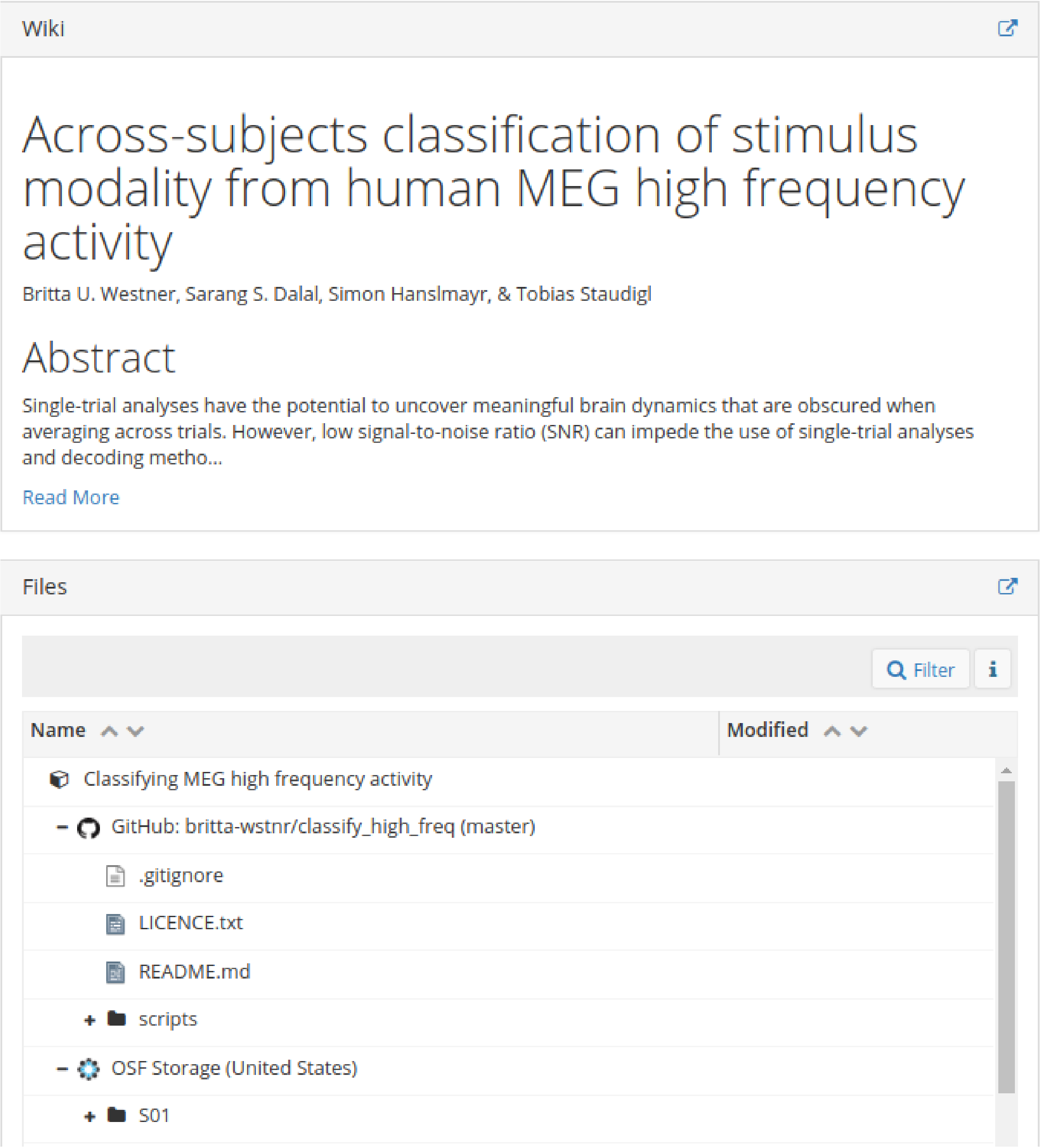
Identifiers: DOI 10.17605/OSF.IO/M25N4 | ARK c7605/osf.io/m25n4

Category: Project

Description: Data and analysis code for the paper "Across-subjects classification of stimulus modality from human MEG high frequer

License: BSD 3-Clause "New"/"Revised" License

OSF: Keeping data and code together



Wiki

Across-subjects classification of stimulus modality from human MEG high frequency activity

Britta U. Westner, Sarang S. Dalal, Simon Hanslmayr, & Tobias Staudigl

Abstract

Single-trial analyses have the potential to uncover meaningful brain dynamics that are obscured when averaging across trials. However, low signal-to-noise ratio (SNR) can impede the use of single-trial analyses and decoding metho...

Read More

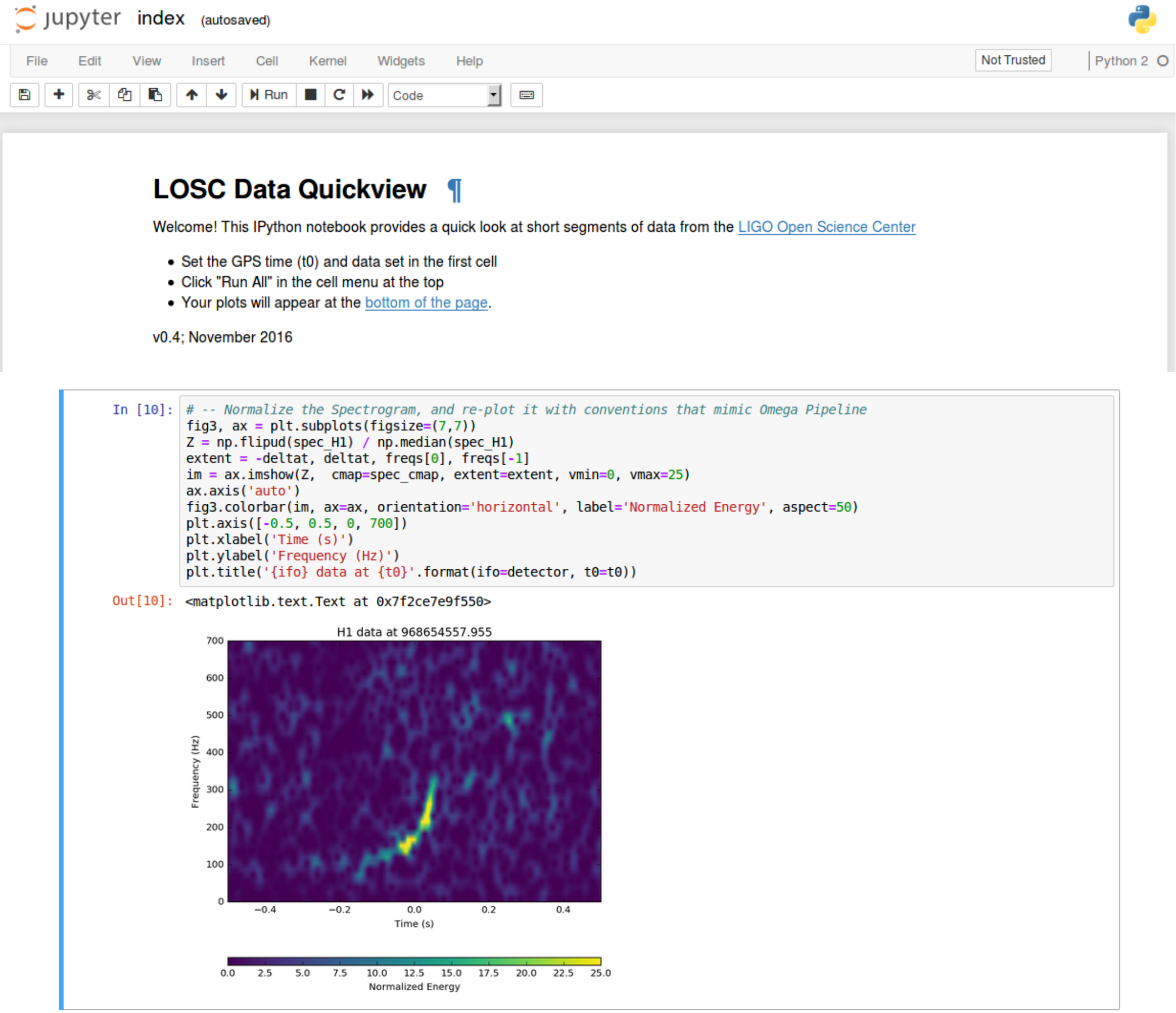
Files

Classifying MEG high frequency activity

- GitHub: britta-wstnr/classify_high_freq (master)
  - .gitignore
  - LICENCE.txt
  - README.md
  - + scripts
- OSF Storage (United States)
  - + S01

Technical vs. practical reproducibility

How easy is it to re-run your analysis?



LOSC Data Quickview ¶

Welcome! This IPython notebook provides a quick look at short segments of data from the LIGO Open Science Center

- Set the GPS time (t0) and data set in the first cell
- Click "Run All" in the cell menu at the top
- Your plots will appear at the bottom of the page.

v0.4; November 2016

In [10]:
```
# -- Normalize the Spectrogram, and re-plot it with conventions that mimic Omega Pipeline
fig3, ax = plt.subplots(figsize=(7,7))
Z = np.flipud(spec_H1) / np.median(spec_H1)
extent = -deltat, deltat, freqs[0], freqs[-1]
im = ax.imshow(Z,  cmap=spec_cmap, extent=extent, vmin=0, vmax=25)
ax.axis('auto')
fig3.colorbar(im, ax=ax, orientation='horizontal', label='Normalized Energy', aspect=50)
plt.axis([-0.5, 0.5, 0, 700])
plt.xlabel('Time (s)')
plt.ylabel('Frequency (Hz)')
plt.title('{ifo} data at {t0}'.format(ifo=detector, t0=t0))
```

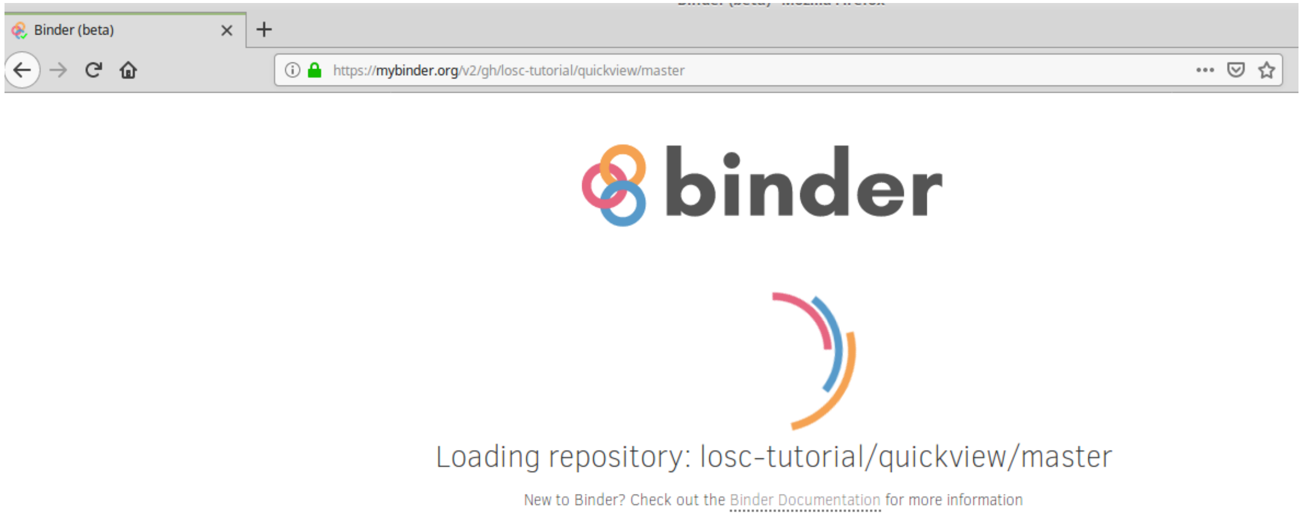
Out[10]:
```
<matplotlib.text.Text at 0x7f2ce7e9f550>
```

https://www.gw-openscience.org/tutorials/

Practical reproducibility: Binder

Notebooks are great, but:

- still need to download the **data**
- still need to create the right **environment** (software versions, operating system)



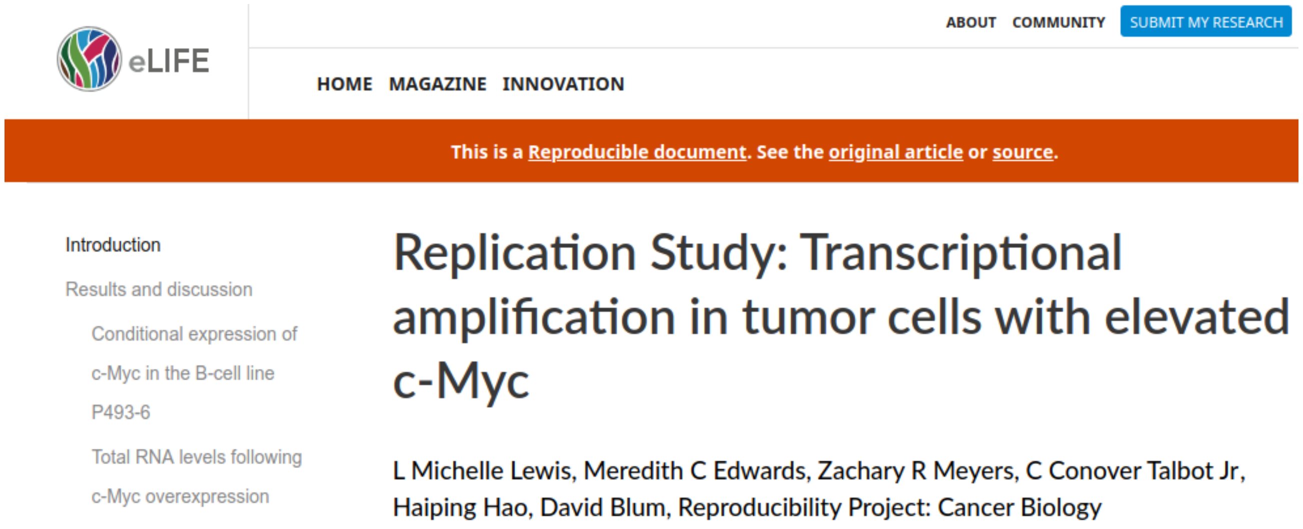
Loading repository: losc-tutorial/quickview/master

New to Binder? Check out the Binder Documentation for more information

https://www.gw-openscience.org/tutorials/

Wait, couldn't we write whole papers like this?

Practical reproducibility: eLife



This is a Reproducible document. See the original article or source.

Introduction

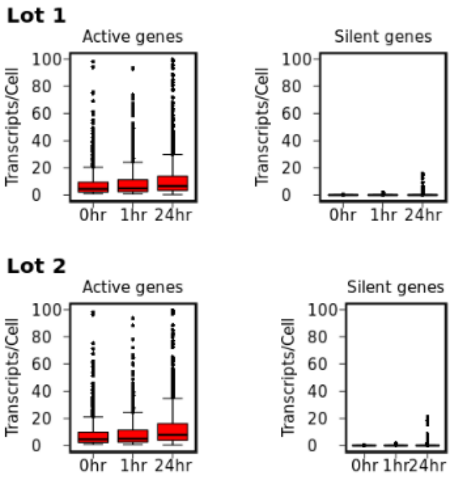
Results and discussion

Conditional expression of c-Myc in the B-cell line P493-6

Total RNA levels following c-Myc overexpression

Replication Study: Transcriptional amplification in tumor cells with elevated c-Myc

L Michelle Lewis, Meredith C Edwards, Zachary R Meyers, C Conover Talbot Jr, Haiping Hao, David Blum, Reproducibility Project: Cancer Biology



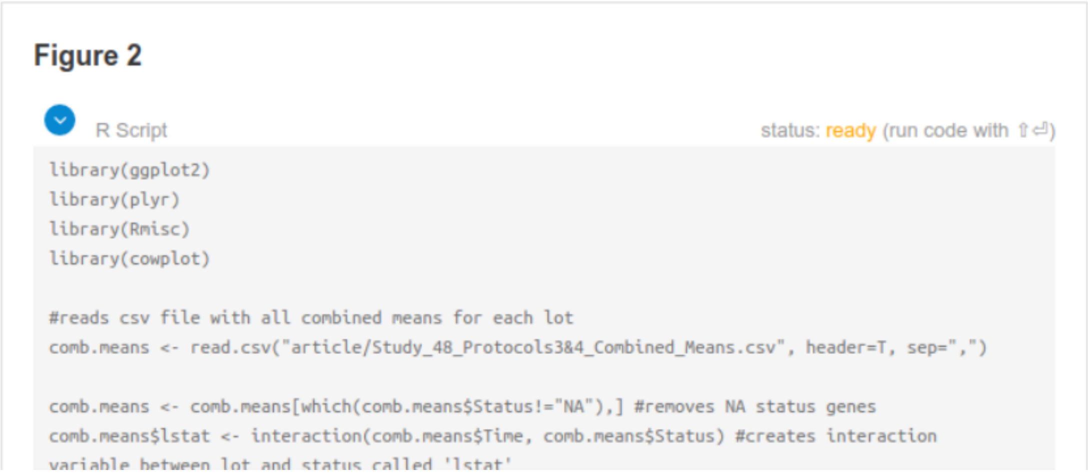


Figure 2

R Script

status: ready (run code with ⇧⏎)

```
library(ggplot2)
library(plyr)
library(Rmisc)
library(cowplot)

#reads csv file with all combined means for each lot
comb.means <- read.csv("article/Study_48_Protocols3&4_Combined_Means.csv", header=T, sep=",")

comb.means <- comb.means[which(comb.means$Status!="NA"),] #removes NA status genes
comb.means$lstat <- interaction(comb.means$Time, comb.means$Status) #creates interaction
variable between lot and status called 'lstat'
```

Lewis et al. 2018

Level up: Contributing to open source

Why should I contribute to open source?

- **solve a problem**
  1/2 of Github contributors contribute only once Eghbal 2017
- for the **reputation**
- for the **community**
  Came for the language, stayed for the community. Brett Cannon

Contributing to open source: getting started

- Annoyed by that one **bug** in the toolbox? **Open an issue.**
- Know how to **fix** it? **Open a PR.**
- Most communities have a **how to contribute** wiki page.
- Most communities are very **welcoming**!

Recap: open source in science

- Open source is **essential** for open science.
- Spans from **sharing code** to **using open source** toolboxes and software.
- **Practical reproducibility** is important.
- **Contributing** to open source toolboxes is fun!

Is there an open source crisis?

OpenSSL

The toolkit for internet connection security was used on 66% of all web servers worldwide (2014).
Prior to "Heartbleed", it was maintained by only a handful of volunteers.

Eghbal 2016; Klug & Miller 2018

NumPy and scientific Python

Being one of the pillars of scientific Python, NumPy only secured stable funding in 2017.
The scientific Python world relied on an estimated 30 people in 2011.

NumFOCUS 2017; Pérez 2011

Open source crisis — toolbox maintenance

2/3 of top projects on GitHub are maintained by only one or two people.

Avelino et al. 2017

The Truck Factor of toolboxes:
minimal number of developers that have to be hit by a truck before a project is lost.

Project	Truck Factor
git	12
scikit-learn	7
IPython	4
pandas	2

Avelino et al. 2017

Open source crisis — other factors

- **funding**
- **needs of maintainers**: traditionally not considered in open source

> *Our goal should be to spread freedom and then defend it. That is more important than making our software popular, which would just be catering to our egos.*
>
> *Richard Stallman, 2005*

- **burning out on projects**: workload and toxic feedback

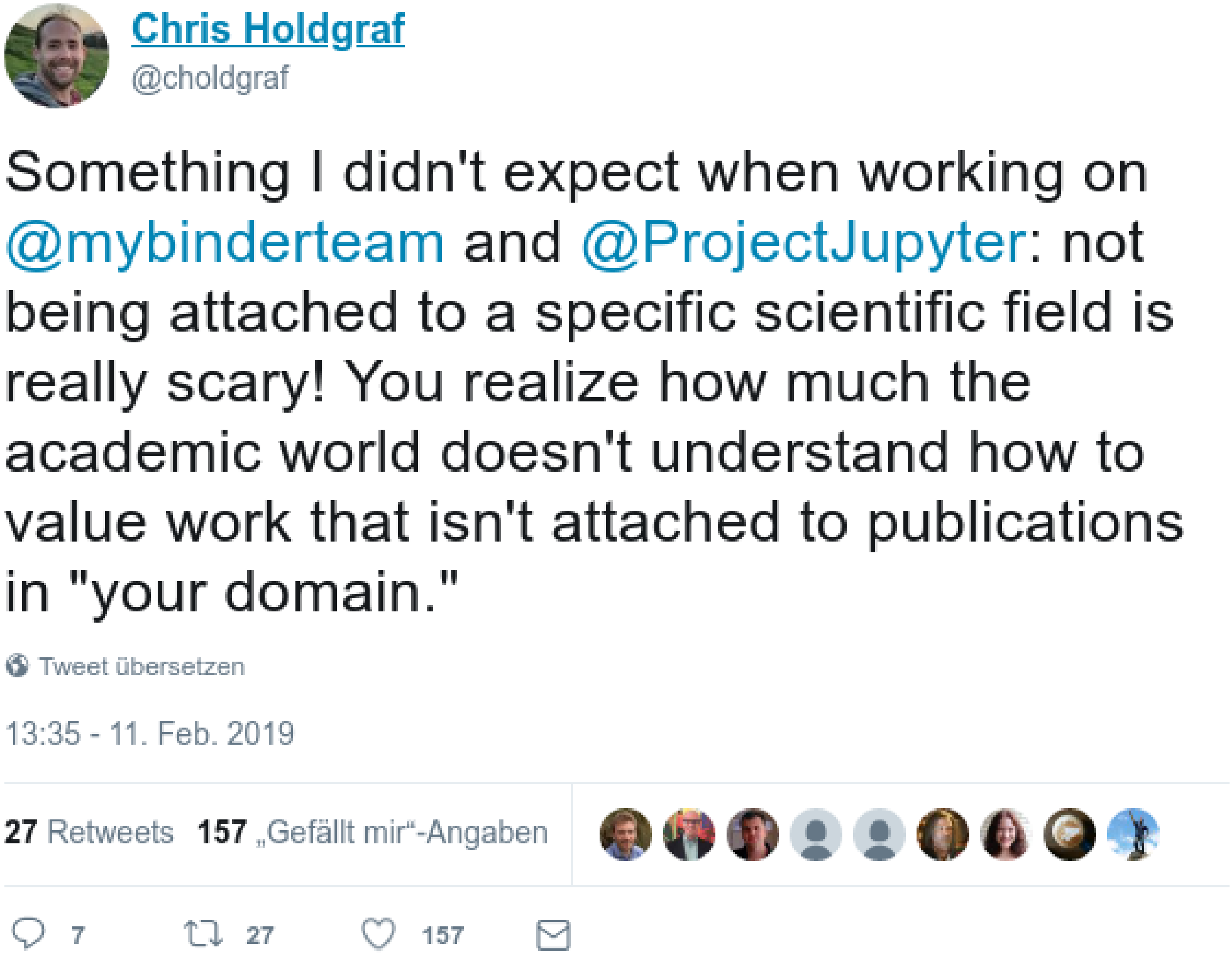
> *[T]he angry response has been overwhelming. Every single day I'm reading someone else rant about how awful of a job we're doing. It's been hard to stay motivated.*
>
> *James Kyle, 2016*

Open source crisis — academia

Software work in science can be career suicide.

Fernando Pérez, 2011

Open source crisis — academia



Chris Holdgraf
@choldgraf

Something I didn't expect when working on @mybinderteam and @ProjectJupyter: not being attached to a specific scientific field is really scary! You realize how much the academic world doesn't understand how to value work that isn't attached to publications in "your domain."

Tweet übersetzen

13:35 - 11. Feb. 2019

27 Retweets 157 „Gefällt mir"-Angaben

7 27 157

What can we do about it?

The problems:

- incentive structure of **modern academia** fits poorly with developers: *contributions instead of publications*
- **tradeoff**: expertise vs. time

Possible solutions:

- **critical mass**: sharing and contributing
- consider open source in **teaching** and supervising
- consider open source “sacrifices” in **hiring decisions** and with **grants**

Conclusions

- Open source is **essential** for open science.
- Ways towards **higher reproducibility**.
- Ways towards **contributing to open source**.
- Awareness of the **open source dilemmas** and **ideas how to cope**.

Acknowledgements

CFIN @ Aarhus university
Sarang Dalal

Google Summer of Code

MNE
MEG + EEG ANALYSIS & VISUALIZATION

MNE-Python
Alexandre Gramfort
Denis A. Engemann
Eric Larson