Open Science: How to organize and share research data and code

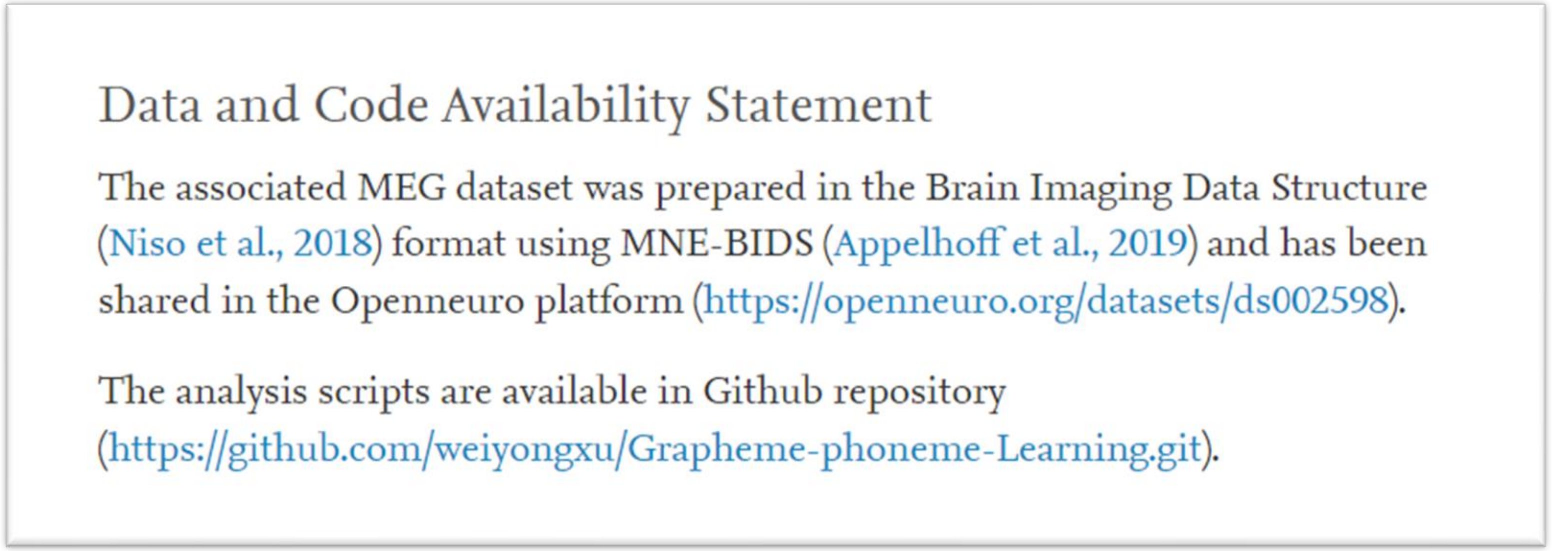
CNSJ1202

Weiyong Xu

https://weiyongxu.github.io/

Outline

- How to organize your data (in BIDS)
- How to share your data (in OpenNeuro)
- Data anonymization
- How to organize your code
- How to share your code (on GitHub)



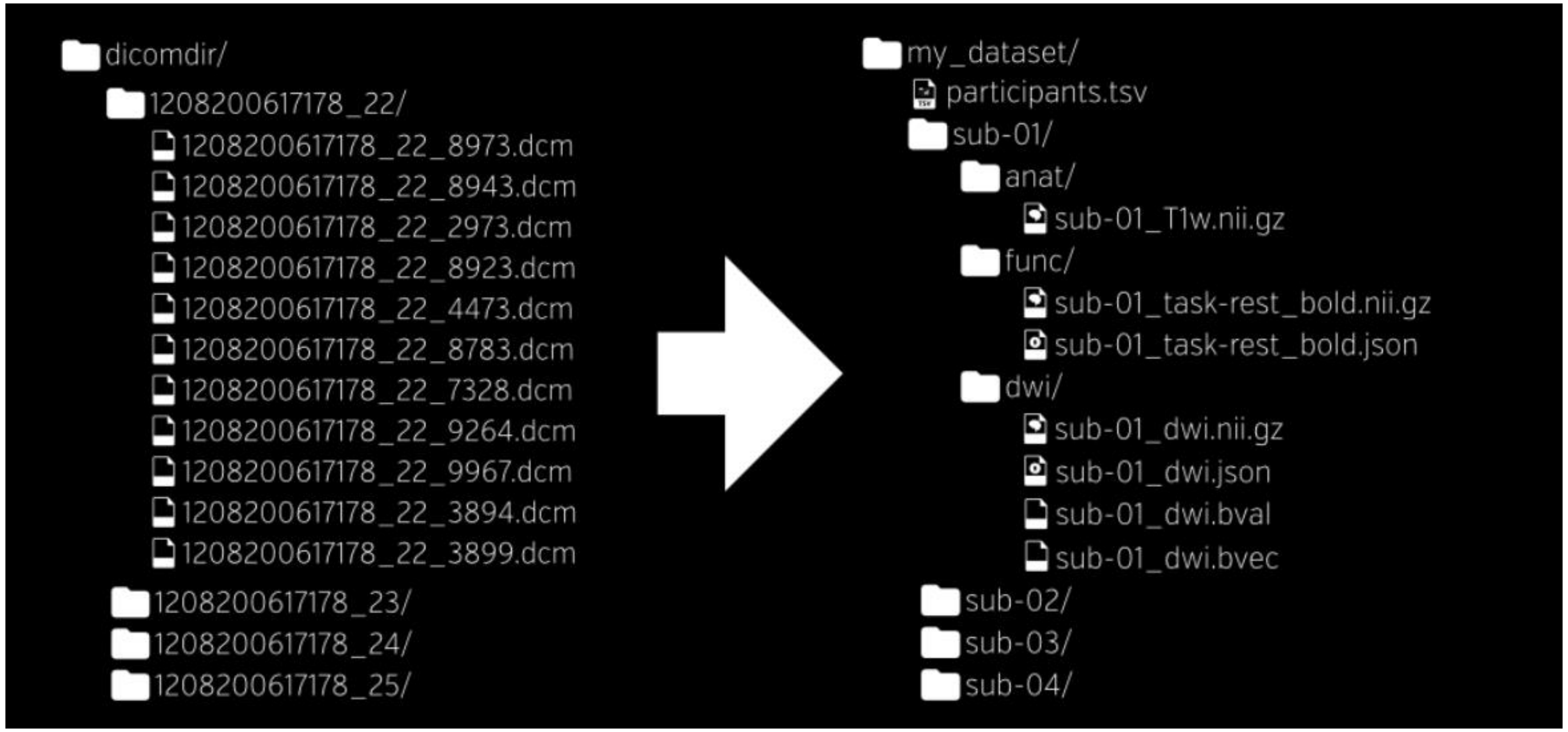
Data and Code Availability Statement

The associated MEG dataset was prepared in the Brain Imaging Data Structure (Niso et al., 2018) format using MNE-BIDS (Appelhoff et al., 2019) and has been shared in the Openneuro platform (https://openneuro.org/datasets/ds002598).

The analysis scripts are available in Github repository (https://github.com/weiyongxu/Grapheme-phoneme-Learning.git).

Brain Imaging Data Structure(BIDS)

- BIDS is an emerging standard for the organization of neuroimaging data



```
dicomdir/
    1208200617178_22/
        1208200617178_22_8973.dcm
        1208200617178_22_8943.dcm
        1208200617178_22_2973.dcm
        1208200617178_22_8923.dcm
        1208200617178_22_4473.dcm
        1208200617178_22_8783.dcm
        1208200617178_22_7328.dcm
        1208200617178_22_9264.dcm
        1208200617178_22_9967.dcm
        1208200617178_22_3894.dcm
        1208200617178_22_3899.dcm
    1208200617178_23/
    1208200617178_24/
    1208200617178_25/
```

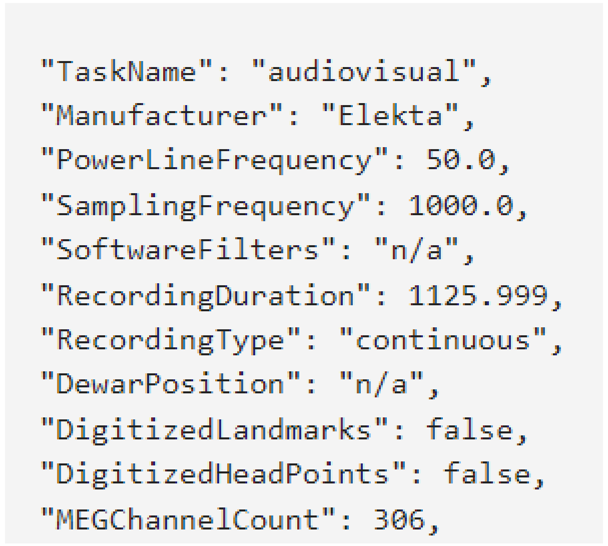
```
my_dataset/
    participants.tsv
    sub-01/
        anat/
            sub-01_T1w.nii.gz
        func/
            sub-01_task-rest_bold.nii.gz
            sub-01_task-rest_bold.json
        dwi/
            sub-01_dwi.nii.gz
            sub-01_dwi.json
            sub-01_dwi.bval
            sub-01_dwi.bvec
    sub-02/
    sub-03/
    sub-04/
```

File types in BIDS

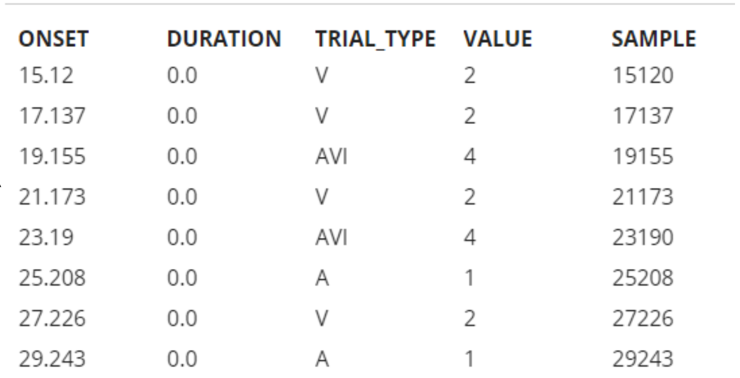
- These are the three main types of files

1. **.json** files that contain ***key: value*** metadata
2. **.tsv** files that contain ***tables*** of metadata
3. Raw data files (for example: .fif files for MEG or .nii.gz files for MRI data.)

These three types of files are organized into a hierarchy of **folders** that have specific naming conventions.



```
"TaskName": "audiovisual",
"Manufacturer": "Elekta",
"PowerLineFrequency": 50.0,
"SamplingFrequency": 1000.0,
"SoftwareFilters": "n/a",
"RecordingDuration": 1125.999,
"RecordingType": "continuous",
"DewarPosition": "n/a",
"DigitizedLandmarks": false,
"DigitizedHeadPoints": false,
"MEGChannelCount": 306,
```



ONSET	DURATION	TRIAL_TYPE	VALUE	SAMPLE
15.12	0.0	V	2	15120
17.137	0.0	V	2	17137
19.155	0.0	AVI	4	19155
21.173	0.0	V	2	21173
23.19	0.0	AVI	4	23190
25.208	0.0	A	1	25208
27.226	0.0	V	2	27226
29.243	0.0	A	1	29243

Metadata are stored in **.json** and **.tsv** files. They are language-agnostic, meaning you can work with them in, for example: Python, Matlab, or R.

Folder in BIDS

- There are four main levels of the folder hierarchy:

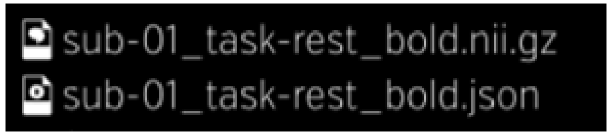
• project/	Can be any name but should be descriptive for the dataset contained in the folder.
• └── subject	**sub-<participant label>** One folder per subject in this dataset. Labels should be unique for each subject.
• └── session	**ses-<session label>** Represents a recording session. You might have multiple sessions per subject if you collected data on different days. If there is only a single session per subject, this level of the hierarchy may be omitted.
• └── datatype	Represents different types of data. Must be one of: *func, dwi, fmap, anat, meg, eeg, ieeg, beh, pet, micr*. The name for the datatype depends on the recording modality.

BIDS filenames

BIDS has a standardized way of naming files that tries to implement the following **principles**:

- Do not white spaces in file names
- Use only letters, numbers, hyphens, and underscores.
- Do not rely on letter case (UPPERCASE and lowercase)
- Use separators and case in a systematic and meaningful way.

key1 - value1 _ key2 - value2 _ suffix .extension


sub-01_task-rest_bold.nii.gz
sub-01_task-rest_bold.json

- Suffixes are preceded by an underscore
- Entities are composed of key-value pairs separated by underscores
- For a given suffix, some entities are required and some others are [optional].
- Entity key-value pairs have a specific order in which they must appear in filename.

Required BIDS annotation files

Annotation refers to metadata that is directly associated with data.

Required BIDS annotation files:

- Dataset sourcing: ***dataset_description.json***
- Dataset description: ***README***
- Subject annotations: ***participants.tsv*** and ***participants.json***

Session annotations (optional)

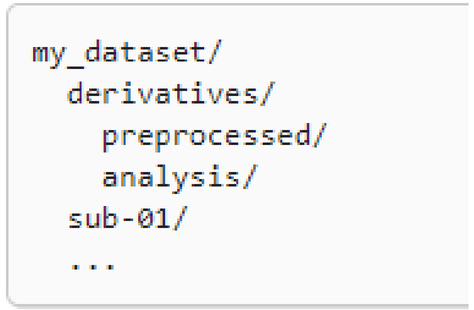
Scans/run annotations (optional)

Event annotations: ***events.tsv***

BIDS Derivatives

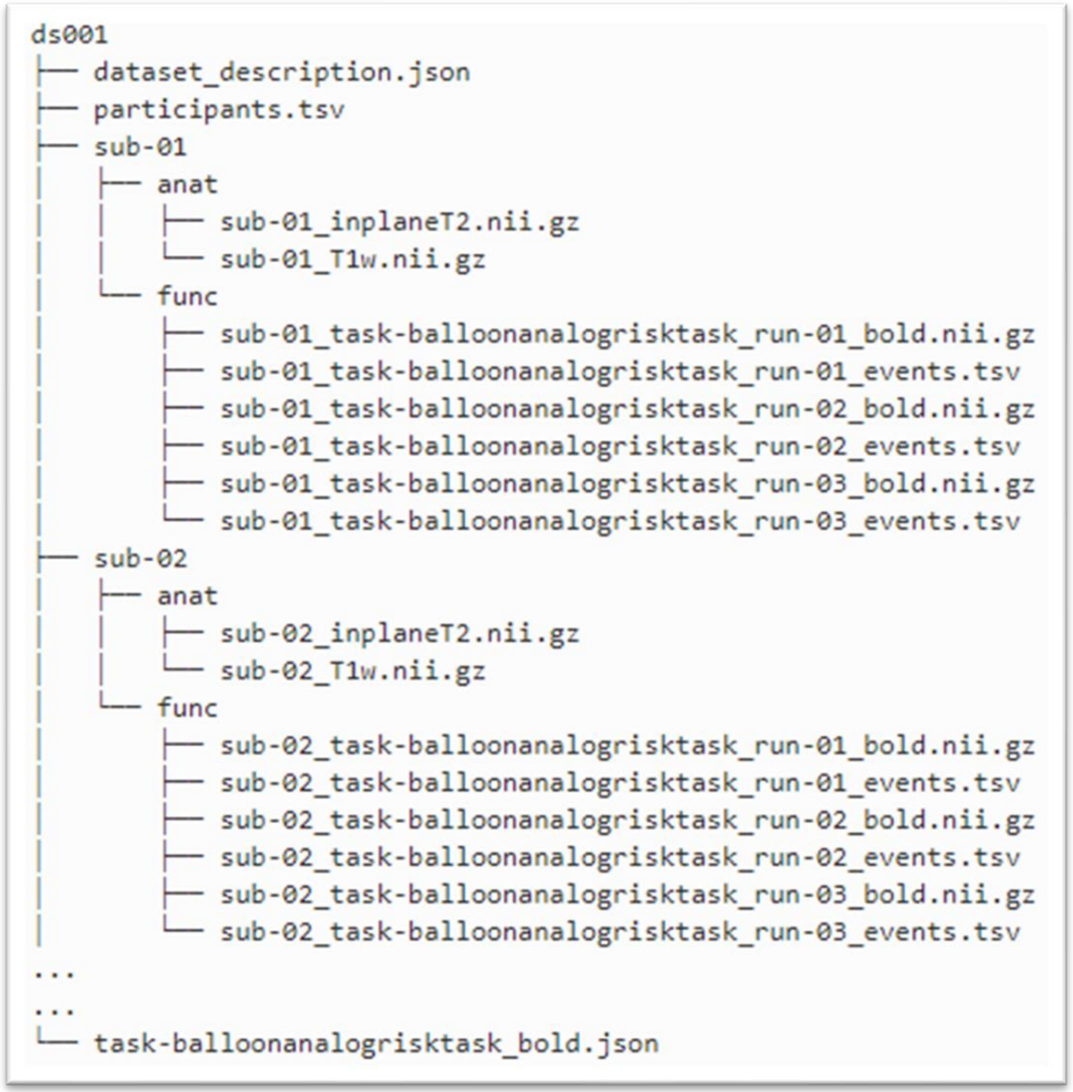
- Derivatives are outputs of (pre-)processing pipelines, capturing data and meta-data sufficient for a researcher to understand and (critically) reuse those outputs in subsequent processing.
    - Preprocessed data(e.g., Maxfilter data)
    - Derivative data types (processed)
    - Unspecified data types (e.g., Freesurfer output)

- BIDS Derivatives datasets
- A collection of derivative datasets may be stored in the ***derivatives/*** subdirectory of a BIDS dataset

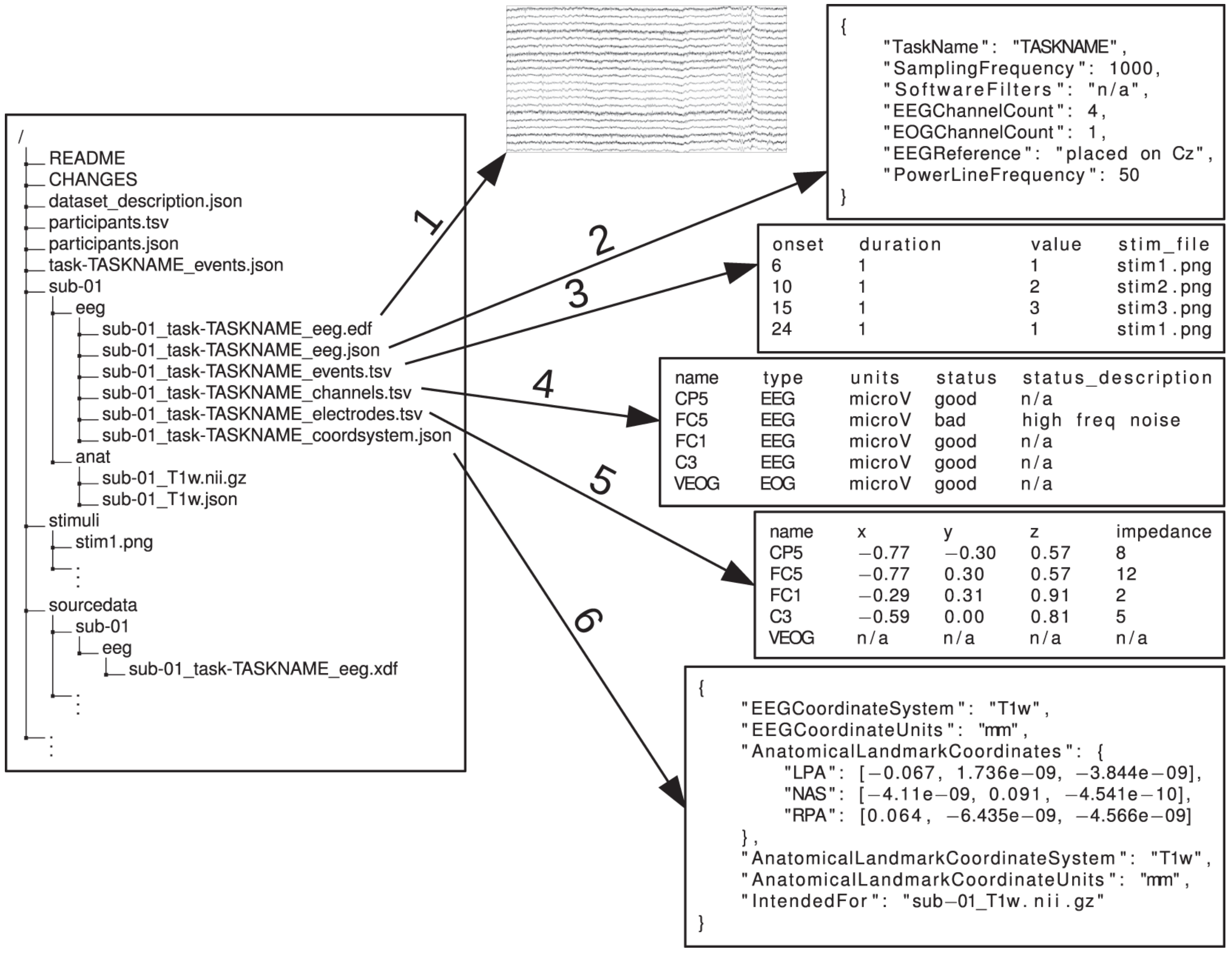


```
my_dataset/
  derivatives/
    preprocessed/
    analysis/
  sub-01/
  ...
```

BIDS examples



```
ds001
├── dataset_description.json
├── participants.tsv
├── sub-01
│   ├── anat
│   │   ├── sub-01_inplaneT2.nii.gz
│   │   └── sub-01_T1w.nii.gz
│   └── func
│       ├── sub-01_task-balloonanalogrisktask_run-01_bold.nii.gz
│       ├── sub-01_task-balloonanalogrisktask_run-01_events.tsv
│       ├── sub-01_task-balloonanalogrisktask_run-02_bold.nii.gz
│       ├── sub-01_task-balloonanalogrisktask_run-02_events.tsv
│       ├── sub-01_task-balloonanalogrisktask_run-03_bold.nii.gz
│       └── sub-01_task-balloonanalogrisktask_run-03_events.tsv
├── sub-02
│   ├── anat
│   │   ├── sub-02_inplaneT2.nii.gz
│   │   └── sub-02_T1w.nii.gz
│   └── func
│       ├── sub-02_task-balloonanalogrisktask_run-01_bold.nii.gz
│       ├── sub-02_task-balloonanalogrisktask_run-01_events.tsv
│       ├── sub-02_task-balloonanalogrisktask_run-02_bold.nii.gz
│       ├── sub-02_task-balloonanalogrisktask_run-02_events.tsv
│       ├── sub-02_task-balloonanalogrisktask_run-03_bold.nii.gz
│       └── sub-02_task-balloonanalogrisktask_run-03_events.tsv
...
...
└── task-balloonanalogrisktask_bold.json
```



```
/
├─ README
├─ CHANGES
├─ dataset_description.json
├─ participants.tsv
├─ participants.json
├─ task-TASKNAME_events.json
├─ sub-01
│  ├─ eeg
│  │  ├─ sub-01_task-TASKNAME_eeg.edf
│  │  ├─ sub-01_task-TASKNAME_eeg.json
│  │  ├─ sub-01_task-TASKNAME_events.tsv
│  │  ├─ sub-01_task-TASKNAME_channels.tsv
│  │  ├─ sub-01_task-TASKNAME_electrodes.tsv
│  │  └─ sub-01_task-TASKNAME_coordsystem.json
│  └─ anat
│     ├─ sub-01_T1w.nii.gz
│     └─ sub-01_T1w.json
├─ stimuli
│  ├─ stim1.png
│  └─ :
├─ sourcedata
│  ├─ sub-01
│  │  └─ eeg
│  │     └─ sub-01_task-TASKNAME_eeg.xdf
│  └─ :
└─ :
```

```
{
    "TaskName": "TASKNAME",
    "SamplingFrequency": 1000,
    "SoftwareFilters": "n/a",
    "EEGChannelCount": 4,
    "EOGChannelCount": 1,
    "EEGReference": "placed on Cz",
    "PowerLineFrequency": 50
}
```

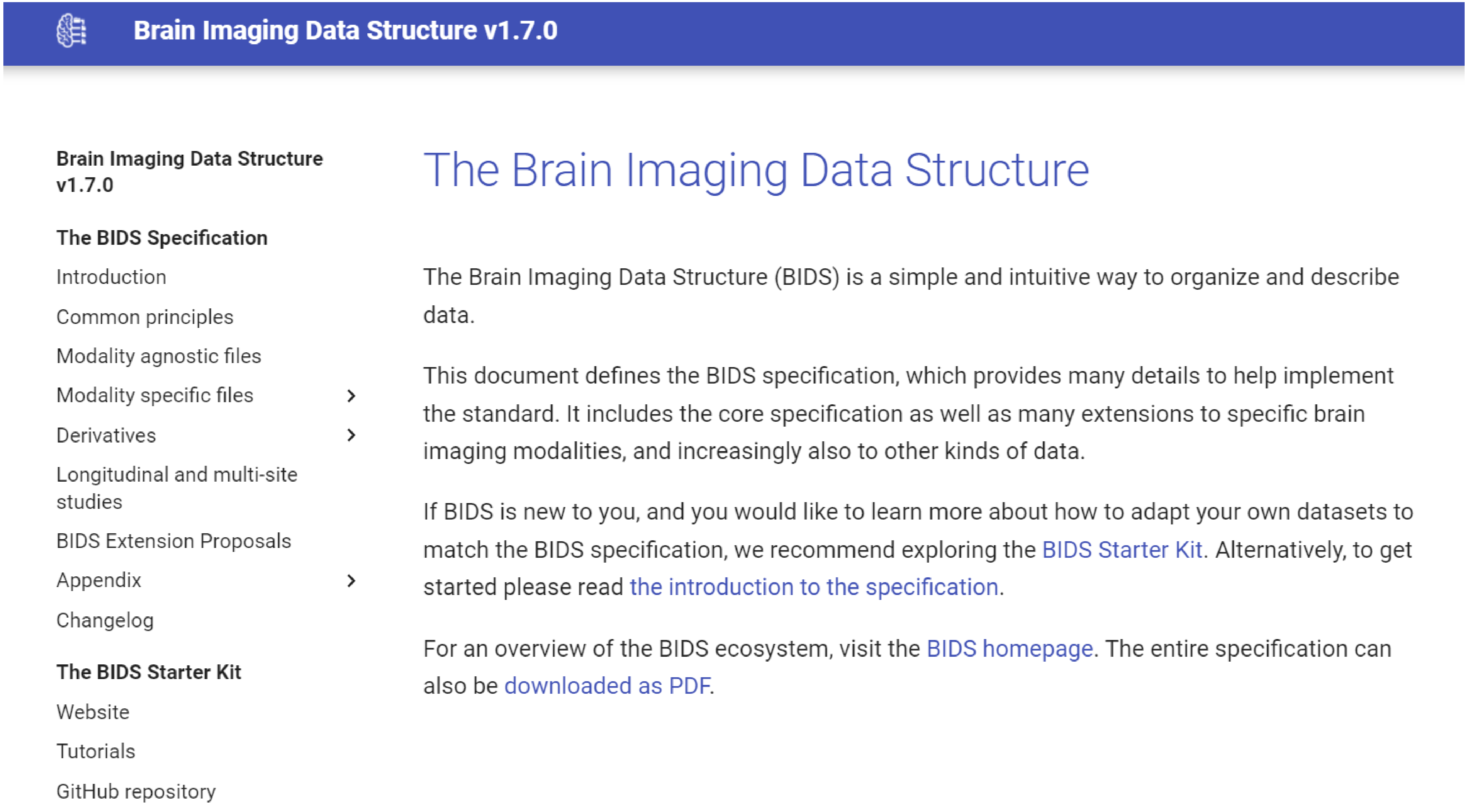
onset	duration	value	stim_file
6	1	1	stim1.png
10	1	2	stim2.png
15	1	3	stim3.png
24	1	1	stim1.png

name	type	units	status	status_description
CP5	EEG	microV	good	n/a
FC5	EEG	microV	bad	high freq noise
FC1	EEG	microV	good	n/a
C3	EEG	microV	good	n/a
VEOG	EOG	microV	good	n/a

name	x	y	z	impedance
CP5	−0.77	−0.30	0.57	8
FC5	−0.77	0.30	0.57	12
FC1	−0.29	0.31	0.91	2
C3	−0.59	0.00	0.81	5
VEOG	n/a	n/a	n/a	n/a

```
{
    "EEGCoordinateSystem": "T1w",
    "EEGCoordinateUnits": "mm",
    "AnatomicalLandmarkCoordinates": {
        "LPA": [−0.067, 1.736e−09, −3.844e−09],
        "NAS": [−4.11e−09, 0.091, −4.541e−10],
        "RPA": [0.064, −6.435e−09, −4.566e−09]
    },
    "AnatomicalLandmarkCoordinateSystem": "T1w",
    "AnatomicalLandmarkCoordinateUnits": "mm",
    "IntendedFor": "sub−01_T1w.nii.gz"
}
```

Pernet, C.R., Appelhoff, S., Gorgolewski, K.J. *et al.* EEG-BIDS, an extension to the brain imaging data structure for electroencephalography. *Sci Data* **6,** 103 (2019).



Brain Imaging Data Structure v1.7.0

Brain Imaging Data Structure v1.7.0

The BIDS Specification

Introduction

Common principles

Modality agnostic files

Modality specific files

Derivatives

Longitudinal and multi-site studies

BIDS Extension Proposals

Appendix

Changelog

The BIDS Starter Kit

Website

Tutorials

GitHub repository

The Brain Imaging Data Structure

The Brain Imaging Data Structure (BIDS) is a simple and intuitive way to organize and describe data.

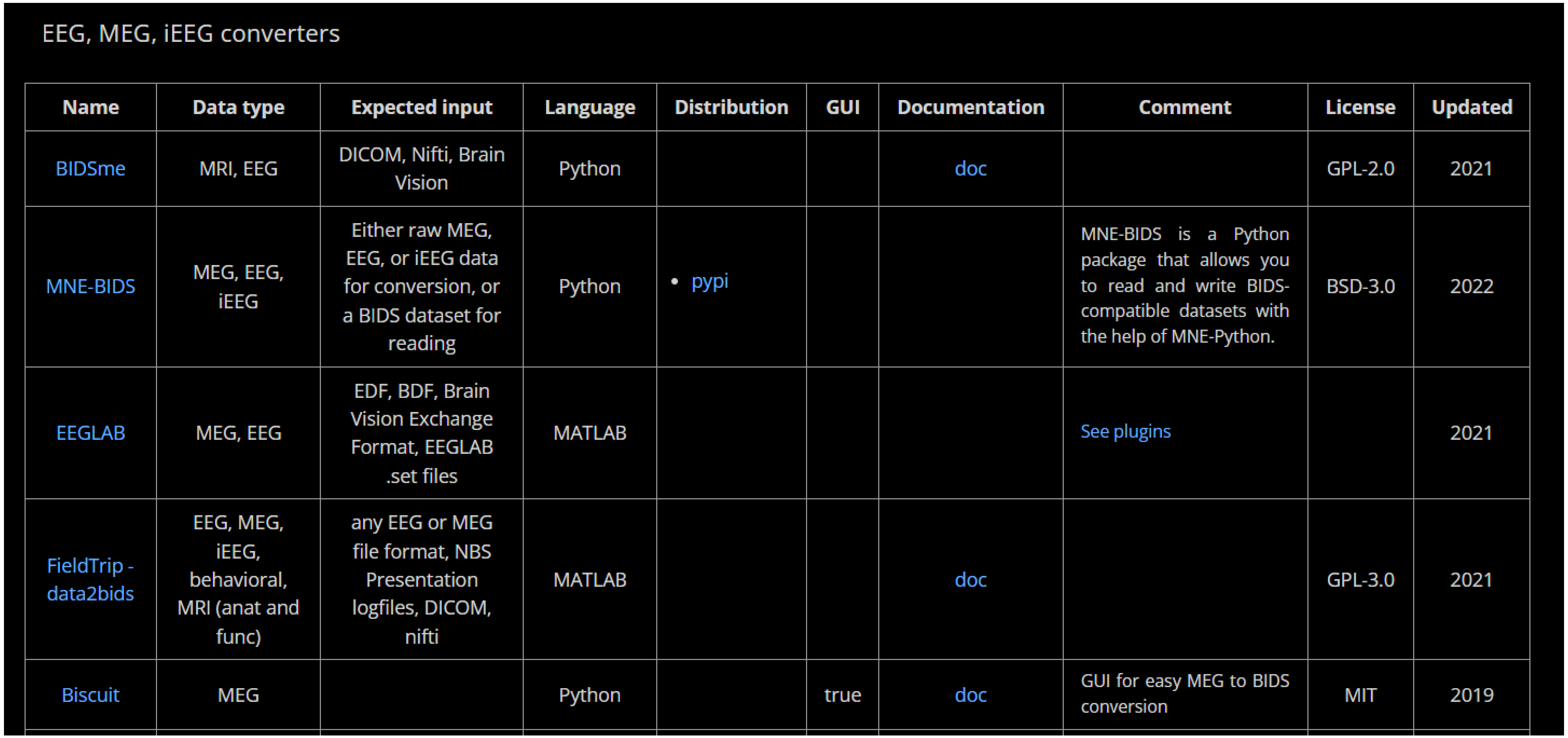
This document defines the BIDS specification, which provides many details to help implement the standard. It includes the core specification as well as many extensions to specific brain imaging modalities, and increasingly also to other kinds of data.

If BIDS is new to you, and you would like to learn more about how to adapt your own datasets to match the BIDS specification, we recommend exploring the BIDS Starter Kit. Alternatively, to get started please read the introduction to the specification.

For an overview of the BIDS ecosystem, visit the BIDS homepage. The entire specification can also be downloaded as PDF.

Tools to create BIDS datasets for M/EEG

https://bids.neuroimaging.io/benefits.html#converters

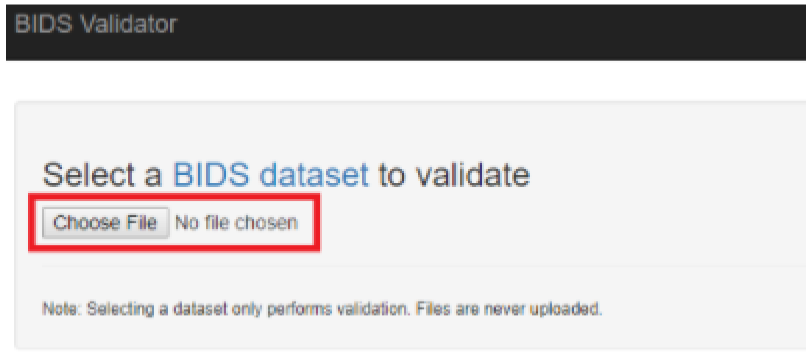


EEG, MEG, iEEG converters

Name	Data type	Expected input	Language	Distribution	GUI	Documentation	Comment	License	Updated
BIDSme	MRI, EEG	DICOM, Nifti, Brain Vision	Python			doc		GPL-2.0	2021
MNE-BIDS	MEG, EEG, iEEG	Either raw MEG, EEG, or iEEG data for conversion, or a BIDS dataset for reading	Python	• pypi			MNE-BIDS is a Python package that allows you to read and write BIDS-compatible datasets with the help of MNE-Python.	BSD-3.0	2022
EEGLAB	MEG, EEG	EDF, BDF, Brain Vision Exchange Format, EEGLAB .set files	MATLAB				See plugins		2021
FieldTrip - data2bids	EEG, MEG, iEEG, behavioral, MRI (anat and func)	any EEG or MEG file format, NBS Presentation logfiles, DICOM, nifti	MATLAB			doc		GPL-3.0	2021
Biscuit	MEG		Python		true	doc	GUI for easy MEG to BIDS conversion	MIT	2019

BIDS validation

- The BIDS Validator is a tool that checks if a dataset is compliant with the BIDS standard. The validator is available for use within several different environments to best suit individual user preferences and use cases, those versions are:
- A web browser-based version
- Command line version
- Docker based version
- A python library installable via pip



Note: Selecting a dataset only performs validation. Files are never uploaded.

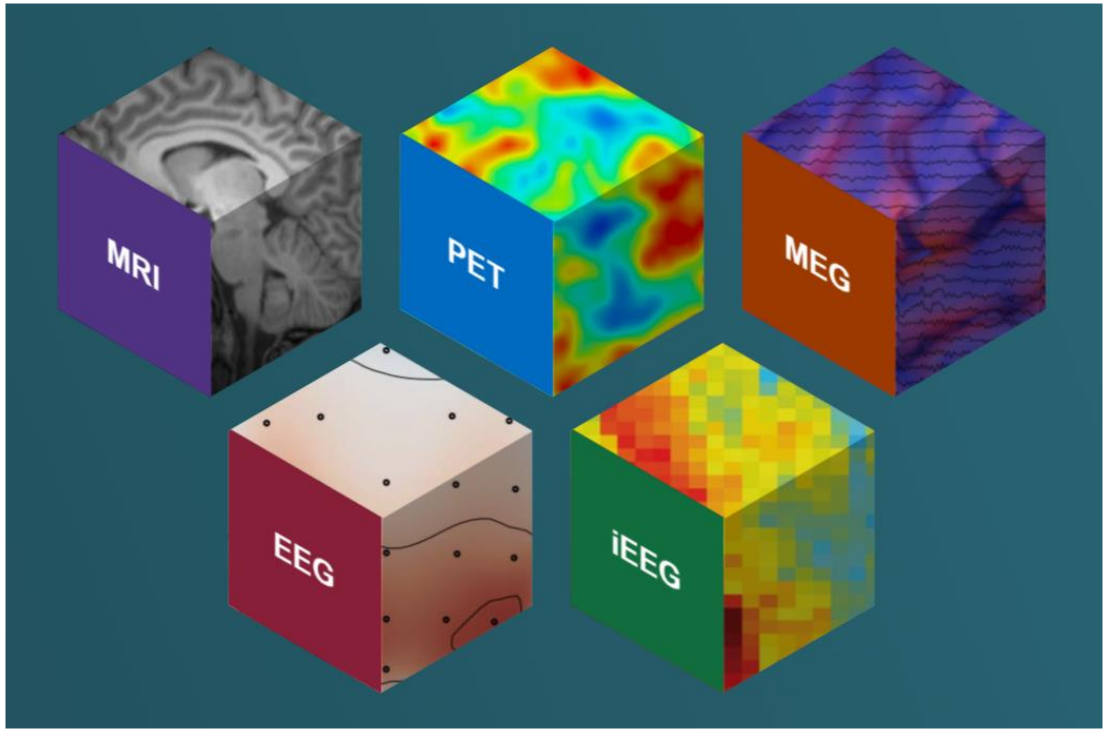
Benefits of using BIDS

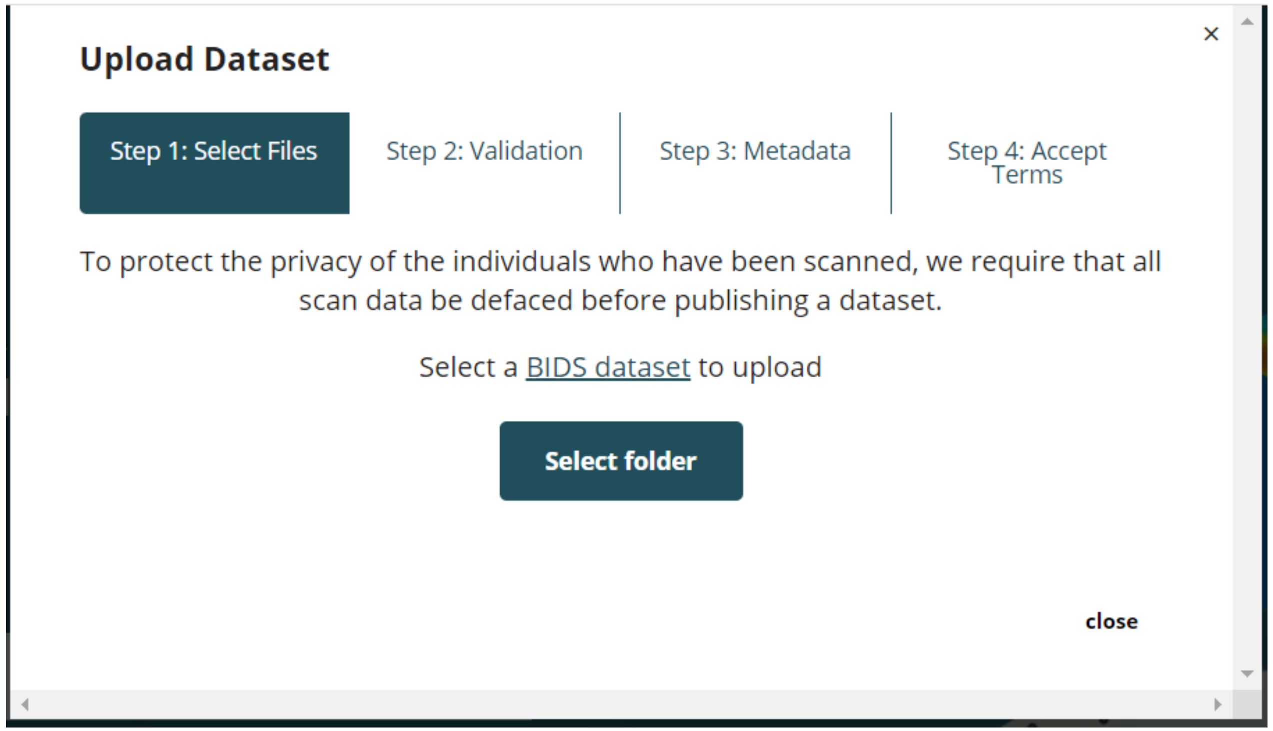
- It will be easy for another researcher to work on your data.
- There is a growing number of data analysis software packages that can understand data organized according to BIDS.
- Databases such as OpenNeuro will accept, and export datasets organized according to BIDS (open data sharing).
- Validation tools that can check your dataset integrity and let you easily spot missing values.

https://bids-specification.readthedocs.io/en/stable/01-introduction.html

Sharing BIDS dataset: OpenNeuro

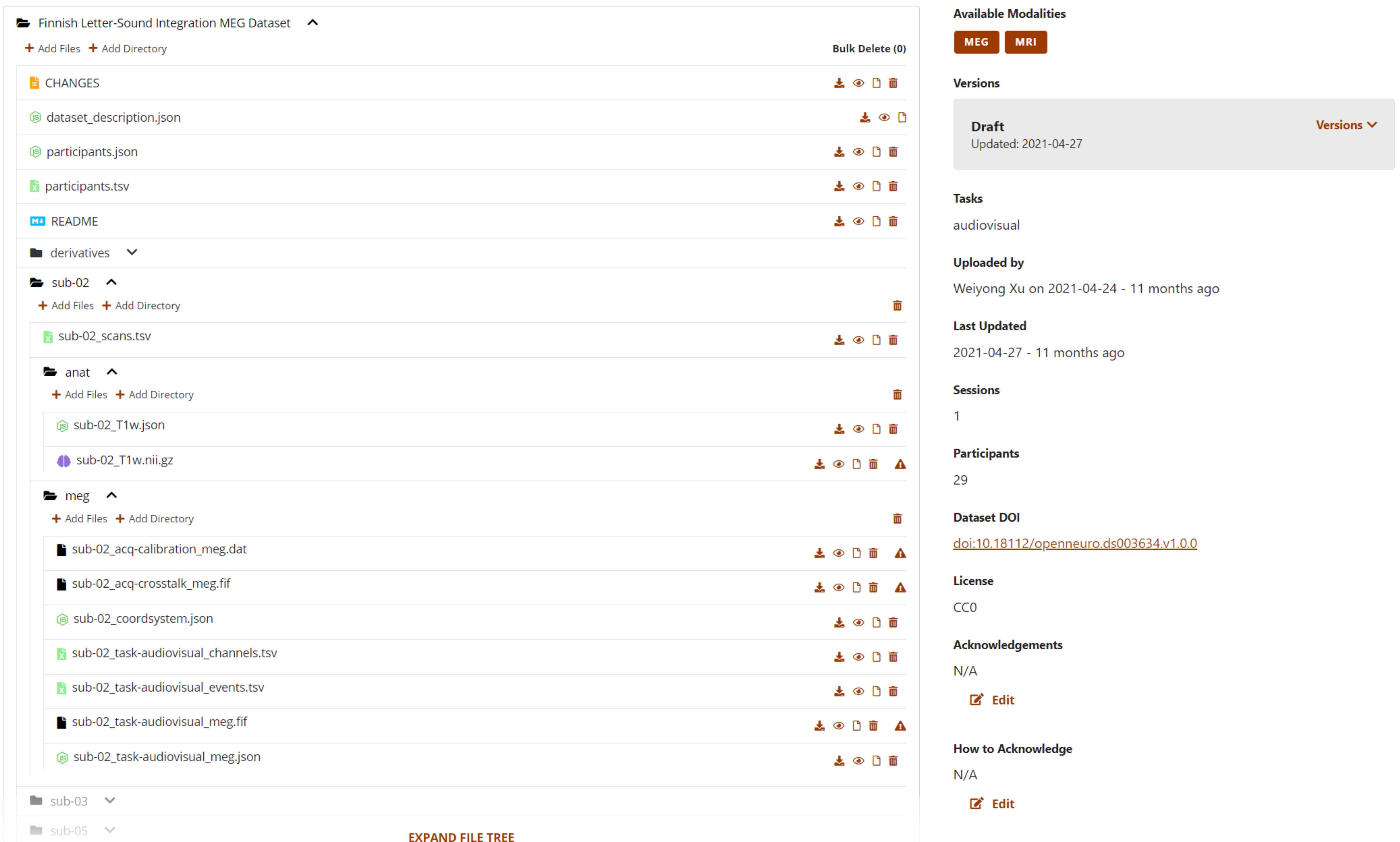
- A free and open platform for validating and sharing BIDS-compliant MRI, PET, MEG, EEG, and iEEG data





To protect the privacy of the individuals who have been scanned, we require that all scan data be defaced before publishing a dataset.

Select a BIDS dataset to upload



Finnish Letter-Sound Integration MEG Dataset

+ Add Files + Add Directory | Bulk Delete (0)

- CHANGES
- dataset_description.json
- participants.json
- participants.tsv
- README
- derivatives
- sub-02
  - + Add Files + Add Directory
  - sub-02_scans.tsv
  - anat
    - + Add Files + Add Directory
    - sub-02_T1w.json
    - sub-02_T1w.nii.gz
  - meg
    - + Add Files + Add Directory
    - sub-02_acq-calibration_meg.dat
    - sub-02_acq-crosstalk_meg.fif
    - sub-02_coordsystem.json
    - sub-02_task-audiovisual_channels.tsv
    - sub-02_task-audiovisual_events.tsv
    - sub-02_task-audiovisual_meg.fif
    - sub-02_task-audiovisual_meg.json
- sub-03
- sub-05

EXPAND FILE TREE

Available Modalities

MEG MRI

Versions

Draft
Updated: 2021-04-27
Versions

Tasks

audiovisual

Uploaded by

Weiyong Xu on 2021-04-24 - 11 months ago

Last Updated

2021-04-27 - 11 months ago

Sessions

1

Participants

29

Dataset DOI

doi:10.18112/openneuro.ds003634.v1.0.0

License

CC0

Acknowledgements

N/A

Edit

How to Acknowledge

N/A

Edit

Data anonymization

- All personally identifiable information should be removed before sharing the dataset publicly.

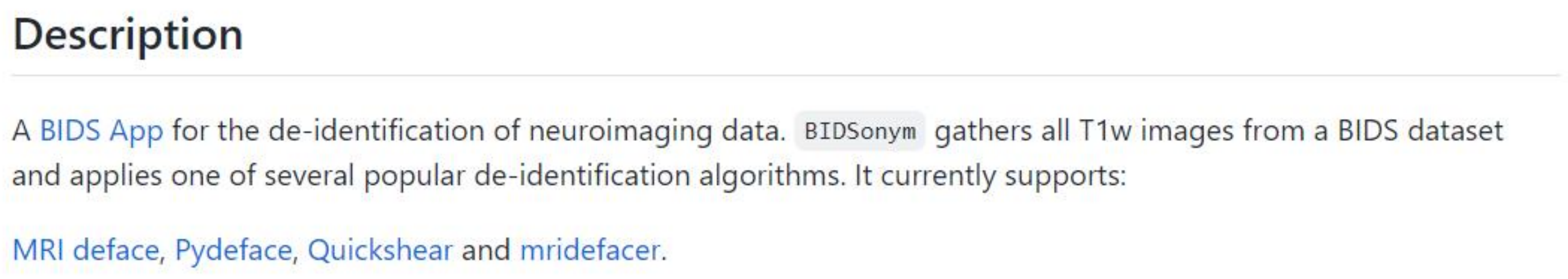
- MNE-BIDS provides a dedicated function for anonymizing a BIDS dataset: **mne_bids.anonymize_dataset()**

- You can use the **ft_anonymizedata** function to scrub unwanted information from the provenance in FieldTrip.

A BIDS App is an analysis pipeline that takes a BIDS formatted dataset as input and has all of its dependencies packaged within a container (Docker or Singularity)

MRI defacing

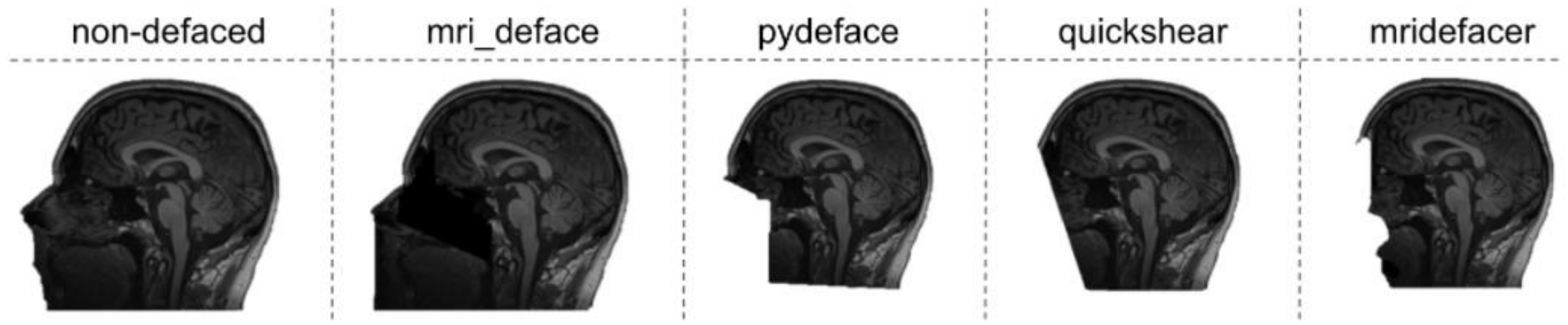
- **BIDSonym**



Description

A BIDS App for the de-identification of neuroimaging data. `BIDSonym` gathers all T1w images from a BIDS dataset and applies one of several popular de-identification algorithms. It currently supports:

MRI deface, Pydeface, Quickshear and mridefacer.



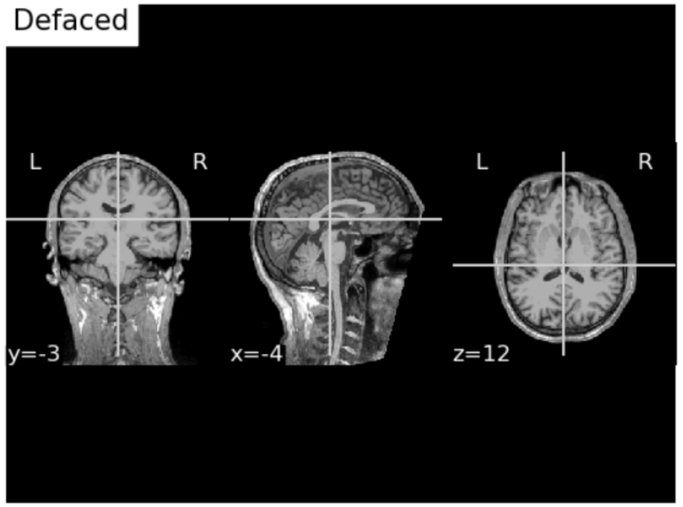
Writing defaced and anonymized T1 image

We can deface the MRI for anonymization by passing `deface=True`.

```
t1w_bids_path = write_anat(
    image=t1_fname,  # path to the MRI scan
    bids_path=bids_path,
    landmarks=landmarks,
    deface=True,
    overwrite=True,
    verbose=True  # this will print out the sidecar file
)
anat_dir = t1w_bids_path.directory

# Our MRI written to BIDS, we got `anat_dir` from our `write_anat` function
t1_nii_fname = op.join(anat_dir, 'sub-01_ses-01_T1w.nii.gz')

# Plot it
fig, ax = plt.subplots()
plot_anat(t1_nii_fname, axes=ax, title='Defaced', vmax=160)
plt.show()
```



MNE BIDS

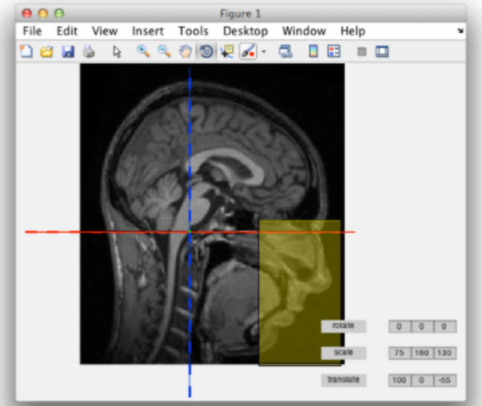
How can I anonymize or deidentify an anatomical MRI?

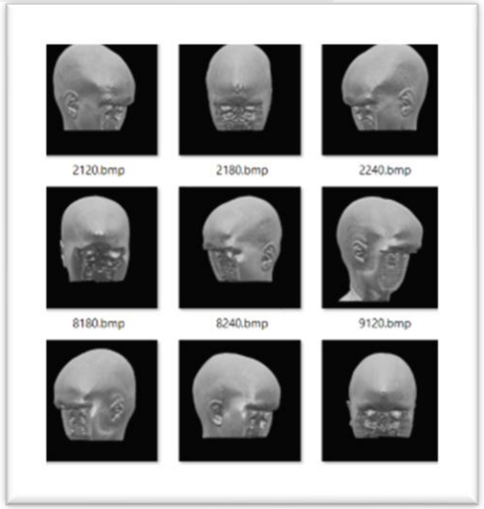
This is something that in general you will want to do **after** the coregistration of the anatomical MRI with the MEG data (using **ft_volumerealign**), as the coregistration often relies on facial landmarks.

You can deface an anatomical MRI using the FieldTrip **ft_defacevolume** function. The default is to show a graphical user interface that allows you to scale, rotate and translate a box, such that it overlaps with the facial details that you want to be removed. Alternatively, you can use the cfg.method='spm' option to use an automated defacing procedure.

```
mri = ft_read_mri('oostenveld_r.mri');

cfg = [];
mri_anon = ft_defacevolume(cfg, mri);
```



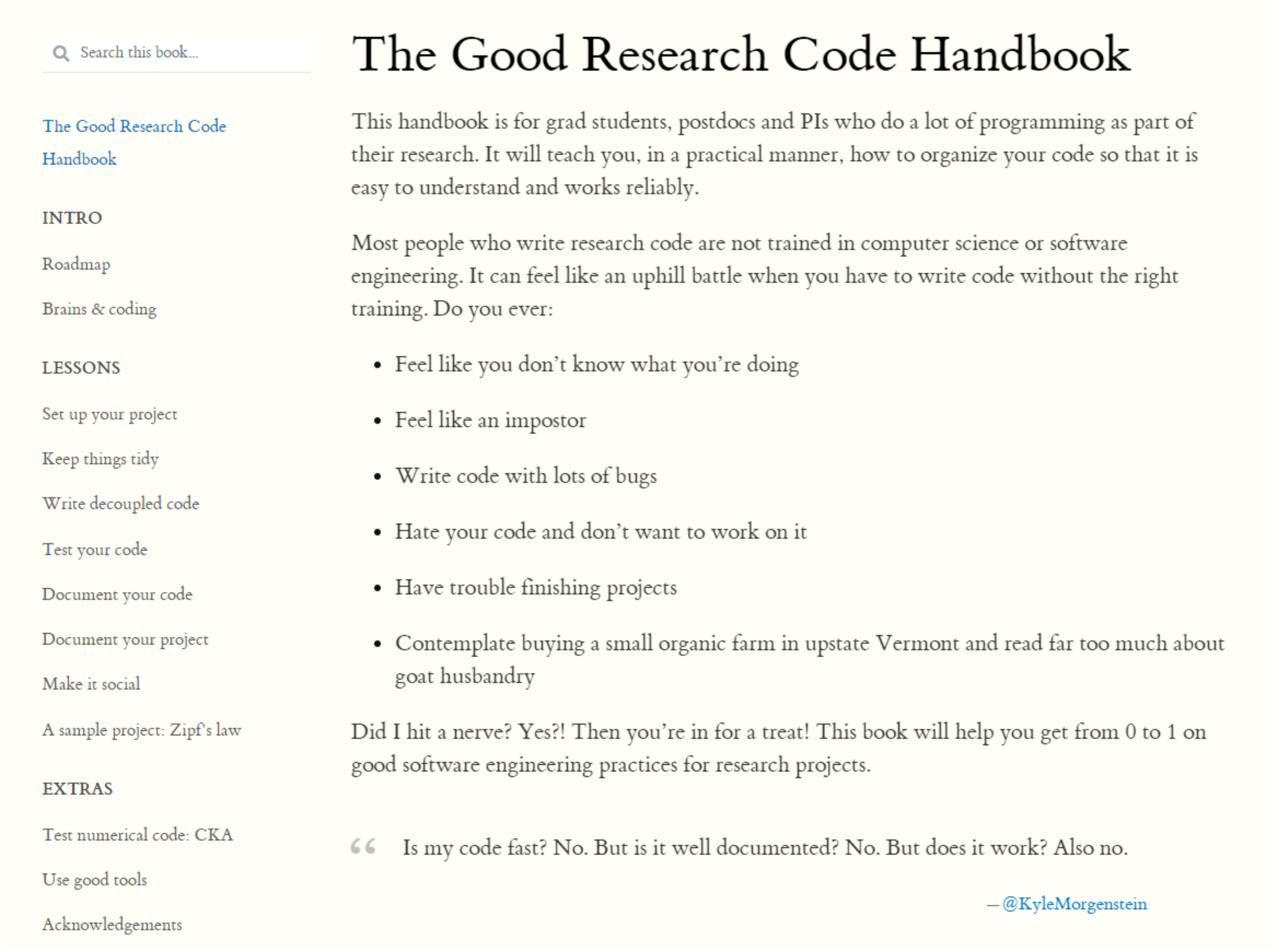


You can use the standard MATLAB figure rotate button to look at the MRI from different angles.

FieldTrip

Organize code

- Document the code
- Use a version control system – Git



Search this book...

The Good Research Code Handbook

INTRO

Roadmap

Brains & coding

LESSONS

Set up your project

Keep things tidy

Write decoupled code

Test your code

Document your code

Document your project

Make it social

A sample project: Zipf's law

EXTRAS

Test numerical code: CKA

Use good tools

Acknowledgements

The Good Research Code Handbook

This handbook is for grad students, postdocs and PIs who do a lot of programming as part of their research. It will teach you, in a practical manner, how to organize your code so that it is easy to understand and works reliably.

Most people who write research code are not trained in computer science or software engineering. It can feel like an uphill battle when you have to write code without the right training. Do you ever:

- Feel like you don't know what you're doing
- Feel like an impostor
- Write code with lots of bugs
- Hate your code and don't want to work on it
- Have trouble finishing projects
- Contemplate buying a small organic farm in upstate Vermont and read far too much about goat husbandry

Did I hit a nerve? Yes?! Then you're in for a treat! This book will help you get from 0 to 1 on good software engineering practices for research projects.

> Is my code fast? No. But is it well documented? No. But does it work? Also no.
>
> —@KyleMorgenstein

https://goodresearch.dev/

Share code in GitHub

- Step 1: Set up git and GitHub
- Step 2: Choose a license for your code (https://choosealicense.com/)
- Step 3: Create a new repository on GitHub, and clone it to your computers
- Step 4: Organize your code
- Step 5: Commit your code to the local repository
- Step 6: Push your changes to GitHub

- Other platforms: OSF (The Open Science Framework)

https://poldrack.github.io/psych-open-science-guide/2_codesharing.html

Thank you for listening!