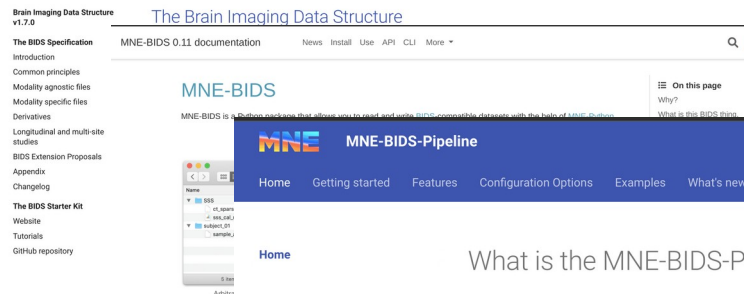


# BIDS / MNE Bids / MNE-Bids-Pipeline / Biowulf Modules



Why?

MNE-BIDS link  
and code share

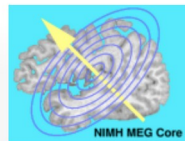
## What is the MNE-BIDS-Pipeline?

The MNE-BIDS-Pipeline is a full-fledged processing pipeline for your MEG and EEG data.

It operates on raw data stored according to the Brain Imaging Data Structure (BIDS). Processing is controlled using a simple human-readable configuration file.

[Learn more](#)

[Get started](#)



### General Information:

[Main page](#)  
[MEG Overview](#)  
[Facility Description](#)  
[Staff](#)  
[Lab Status](#)  
[Empty Room](#)  
[Recordings](#)

### For Users:

[User Info and Policies](#)  
[COVID-19 Safety](#)  
[Protocols](#)  
[MEG Software and](#)  
[Analysis](#)  
[Schedule](#)  
[Request Staff Support](#)  
[Troubleshooting](#)  
[Brainlight](#)  
[Coregistration](#)  
[Helpful Links](#)

[Talks, Trainings,](#)  
[Workshops, and](#)  
[Groups:](#)

[Club MEG](#)  
[Machine Learning SIG](#)  
[MEG North America](#)  
[Workshop](#)  
[ENIGMA MEG Working](#)  
[Group](#)  
[MEG Hackathon 2022](#)

Page

[Discussion](#)

## Mne bids pipeline

### Contents [hide]

- 1 [Skip Background](#)
- 2 [Intro](#)
  - 2.1 [BIDS website](#)
  - 2.2 [Bids data organization](#)
- 3 [MNE Bids](#)
  - 3.1 [MNE Bids website](#)
- 4 [MNE bids pipeline background](#)
  - 4.1 [Processing](#)
  - 4.2 [Example Config](#)
- 5 [Use on biowulf](#)
  - 5.1 [Make MEG modules accessible](#)
  - 5.2 [Start interactive session with scratch to render visualization offscreen](#)
  - 5.3 [Create BIDS data from MEG data](#)
  - 5.4 [Freesurfer Processing](#)
    - 5.4.1 [Already Processed - Copy Data](#)
    - 5.4.2 [Process Using MNE-BIDS-Pipeline](#)
  - 5.5 [Process data using MNE Bids Pipeline](#)
- 6 [TEST DATA for biowulf](#)
  - 6.1 [Start Interactive Session](#)
  - 6.2 [Copy and untar the data into your folder](#)
  - 6.3 [Load module and process the data](#)

## Skip Background

[Use on Biowulf](#)

## Intro

# Outline (with arrows)

BIDS	→	Data Structure/Descriptors
MNE bids	→	Locate / Read / Write BIDS
MNE-bids-pipeline	→	Processing Pipeline for BIDS
Biowulf modules	→	Helper functions specific to NIH

# Describing (meg) data is hard

- Which direction is X,Y,Z in the MEG scanner
  - Which direction is positive
- How are the fiducials localized on the scalp
  - Headshape / polhemus
  - MRI localized
    - Vitamin E Capsule
    - Measured from anatomical fiducial
  - Brainsight Localized
- Which direction is X,Y,Z in the MRI scanner
  - And which direction is increasing pixel count
    - Afni differs from Freesurfer
- How is the MRI coregistered to the MEG
- What is the electrical mains frequency
- How are the events described
- What is the acquisition frequency
- Multiple session?
- Multiple runs?

# BIDS – (MRI/fMRI/MEG/EEG...)

Describe all the attributes of data for analysis

📄 bids-specification.readthedocs.io/en/stable/

g...

📄 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 neuroimaging data.

This document defines the BIDS specification, which provides many details about the standard. It includes the core specification as well as many extensions for different 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 data to match the BIDS specification, we recommend exploring the [BIDS Starter Kit](#). If you have started please read [the introduction to the specification](#).

For an overview of the BIDS ecosystem, visit the [BIDS homepage](#). The specification can also be [downloaded as PDF](#).

📄 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

Magnetic Resonance Imaging

Magnetoencephalography

Electroencephalography

Intracranial Electroencephalography

Task events

Physiological and other continuous recordings

Behavioral experiments (with no neural recordings)

Genetic Descriptor

Positron Emission Tomography

Microscopy

Magnetoencephalography

Support for Magnetoencephalography (MEG) was developed as a [BIDS Extension Proposal](#). Please see [Citing BIDS](#) on how to appropriately credit this extension when referring to it in the context of the academic literature.

The following example MEG datasets have been formatted using this specification and can be used for practical guidance when curating a new dataset.

- [multimodal MEG and MRI](#)

Further datasets are available from the [BIDS examples repository](#).

MEG recording data

Template:

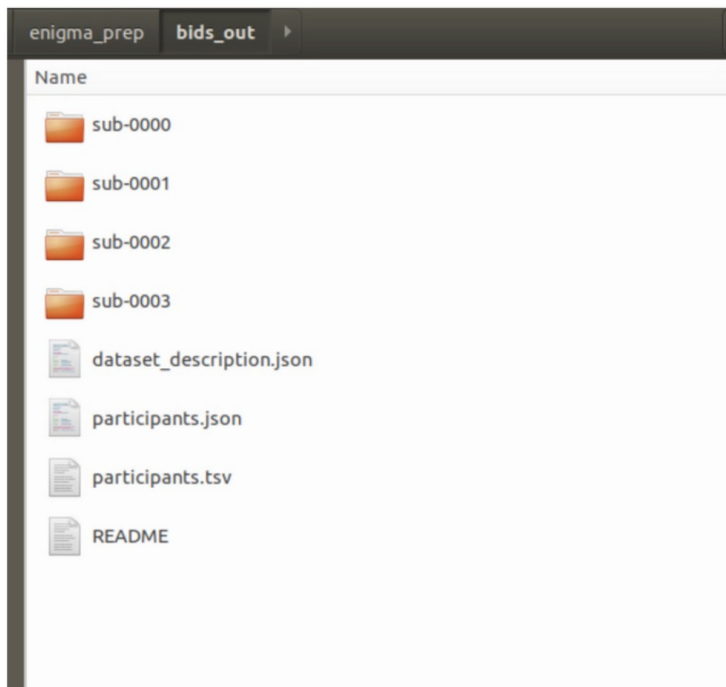
```
sub-<label>/
[ses-<label>/]
meg/
sub-<label>[_ses-<label>][_task-<label>][_acq-<label>][_run-<index>][_prc-<label>]
sub-<label>[_ses-<label>][_task-<label>][_acq-<label>][_run-<index>][_prc-<label>]
sub-<label>[_ses-<label>][_acq-<calibration>].meg.dat
sub-<label>[_ses-<label>][_acq-<crosstalk>].meg.fif
```

# BIDS

- IS
  - The Brain Imaging Data Structure (BIDS) is a simple and intuitive way to **organize and describe data**.
  - Codified structure for any software to locate info on data
  - Describes RAW data (currently)
- **IS NOT**
  - (not) A way to process your data
  - (not) Platform dependent / Software dependent
  - (not) A data reader

# What does MEG bids look like

## BIDS Directory Organization



**Subject Data Folders** at the top level

Dataset description (if distributing or archiving)

Participants.json – Describe column headers in participants.tsv

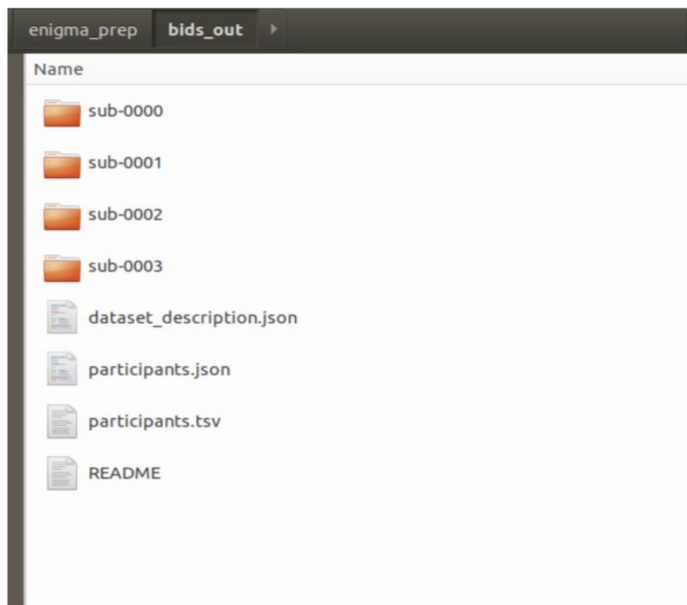
Participants.tsv – Demographic info [sex, age, handedness.....whatever else]

README – version of bids output etc

# What does bids look like:

## Cascading Directories

### BIDS Directory Organization



```
(enigma) bids_out$ tree -L 4
.
├── dataset_description.json
├── participants.json
├── participants.tsv
├── README
├── sub-0000
│   └── ses-01
│       ├── anat
│       │   ├── sub-0000_ses-01_T1w.json
│       │   └── sub-0000_ses-01_T1w.nii.gz
│       └── meg
│           ├── sub-0000_ses-01_coordsystem.json
│           ├── sub-0000_ses-01_task-rest_run-01_channels.tsv
│           ├── sub-0000_ses-01_task-rest_run-01_meg.ds
│           ├── sub-0000_ses-01_task-rest_run-01_meg.json
│           └── sub-0000_ses-01_scans.tsv
├── sub-0001
│   └── ses-01
│       ├── anat
│       │   ├── sub-0001_ses-01_T1w.json
│       │   └── sub-0001_ses-01_T1w.nii.gz
│       └── meg
│           ├── sub-0001_ses-01_coordsystem.json
│           ├── sub-0001_ses-01_task-rest_run-01_channels.tsv
│           ├── sub-0001_ses-01_task-rest_run-01_meg.ds
│           ├── sub-0001_ses-01_task-rest_run-01_meg.json
│           └── sub-0001_ses-01_scans.tsv
├── sub-0002
│   └── ses-01
│       ├── anat
│       │   ├── sub-0002_ses-01_T1w.json
│       │   └── sub-0002_ses-01_T1w.nii.gz
│       └── meg
│           ├── sub-0002_ses-01_coordsystem.json
│           ├── sub-0002_ses-01_task-rest_run-01_channels.tsv
│           ├── sub-0002_ses-01_task-rest_run-01_meg.ds
│           ├── sub-0002_ses-01_task-rest_run-01_meg.json
│           └── sub-0002_ses-01_scans.tsv
└── sub-0003
    └── ses-01
        ├── anat
        │   ├── sub-0003_ses-01_T1w.json
        │   └── sub-0003_ses-01_T1w.nii.gz
        └── meg
            ├── sub-0003_ses-01_coordsystem.json
            ├── sub-0003_ses-01_task-rest_run-01_channels.tsv
            ├── sub-0003_ses-01_task-rest_run-01_meg.ds
            ├── sub-0003_ses-01_task-rest_run-01_meg.json
            └── sub-0003_ses-01_scans.tsv
```

# Raw Bids Tags

- Root (your top level bids directory)
- Subject
- Session
- Datatype (MEG/Anat/fMRI....)
- Task
- Run



# Raw Bids Tags: tagID-VALUE\_

- **Sub-0000**     *{Subject}*
  - **Ses-1**       *{Session}*
    - **Meg**       *{Datatype}*
      - sub-0000\_ses-1\_task-airpuff\_run-01\_meg.ds
      - sub-0000\_ses-1\_task-airpuff\_run-01\_meg.json
      - sub-0000\_ses-1\_task-airpuff\_run-02\_meg.ds
      - ...
      - sub-0000\_ses-1\_task-rest\_run-01\_meg.ds
      - ...
    - **Anat**       *{DataType}*
      - sub-0000-ses-1\_T1w.nii
      - sub-0000-ses-1\_T1w.json
  - **Ses-2**
    - ....

# MNE-BIDS

- Software for interacting with and creating BIDS
- Locate files using the python object `BIDSPath`
- Coregister data
  - Fids must be localized
- Write/Read/Update MEG/EEG bids structure

# MNE Bids website - w/examples

MNE-BIDS 0.11 documentation

[News](#) [Install](#) [Use](#) [API](#) [CLI](#) [More](#) ▼

## Section Navigation

- 01. Read BIDS datasets
- 02. Convert MNE sample data to BIDS format
- 03. Interactive data inspection and bad channel selection
- 04. Convert EEG data to BIDS format
- 05. BIDS conversion for group studies
- 06. Rename BrainVision EEG data files
- 07. Save and load T1-weighted MRI scan along with anatomical landmarks in BIDS
- 08. Convert iEEG data to BIDS format
- 09. Manually storing empty room data
- 10. An introduction to BIDSPATH
- 11. Creating BIDS-compatible folder names and filenames
- 12. Updating BIDS datasets
- 13. Anonymizing a BIDS dataset
- 13. Convert NIRS data to BIDS format

## Using MNE-BIDS

### Quickstart

#### Python

```
>>> import mne
>>> from mne_bids import BIDSPATH, write_raw_bids
>>> raw = mne.io.read_raw_fif('my_old_file.fif')
>>> bids_path = BIDSPATH(subject='01', session='01', run='05',
                        datatype='meg', root='./bids_dataset')
>>> write_raw_bids(raw, bids_path=bids_path)
```

#### Command Line Interface

Simply type `mne_bids` in your command line and press enter to see a list of accepted commands. Then type `mne_bids <command> --help`, where `<command>` is one of the accepted commands, to get more information about it.

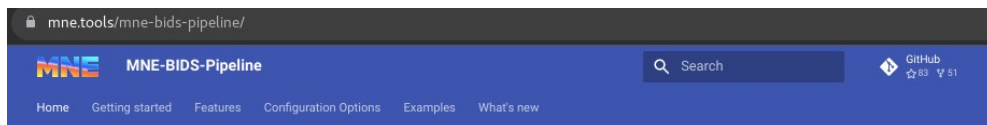
Example:

```
$ mne_bids raw_to_bids --subject_id sub01 --task rest --raw data.edf --bids_root new
```

# Jupyter notebook

- Note to self - you are ranting
  - just show them the code

# MNE-BIDS-PIPELINE



mne.tools/mne-bids-pipeline/examples/ds000117.html#demonstrated-features

## Faces dataset

Search

MEG processing	✓
EEG processing	✗
Maxwell filter	✓
Frequency filter	✓
SSP	✗
ICA	✗
Evoked contrasts	✓
Time-by-time decoding	✓
Time-generalization decoding	✓
Time-frequency analysis	✗
BEM surface creation	✗
Template MRI	✗

**Examples**

- Examples Gallery
- hMT+ Localizer
- Single-subject infant dataset for testing maxwell\_filter with movecomp.
- Brainstorm - Auditory Dataset.
- OMEGA Resting State Sample Data
- MNE Sample Data: M/EEG combined processing
- MNE Sample Data: ICA
- MNE Sample Data: Using the 'fsaverage' template MRI
- Somato
- Matching pennies EEG experiment
- tDCS EEG

**Faces dataset**

- SRM Resting-state EEG
- MIND DATA
- Mobile brain body imaging (MoBi) gait adaptation experiment.
- ERP CORE

## Dataset source

This dataset was acquired from <https://openneuro.org/datasets/ds000117>

**How to download this dataset**

## Configuration

```
study_name = 'ds000117'
bids_root = '~/mne_data/ds000117'
deriv_root = '~/mne_data/derivatives/mne-bids-pipeline/ds000117'

task = 'facerecognition'
ch_types = ['meg']
runs = ['01', '02']
sessions = ['meg']
interactive = False
acq = None
subjects = ['01']

resample_sfreq = 125.
crop_runs = (0, 350) # Reduce memory usage on CI system
```

# Config File

Mne-bids-pipeline-run.py --config=MYCONFIG.py (Run all subjects)

***Or ... select options***

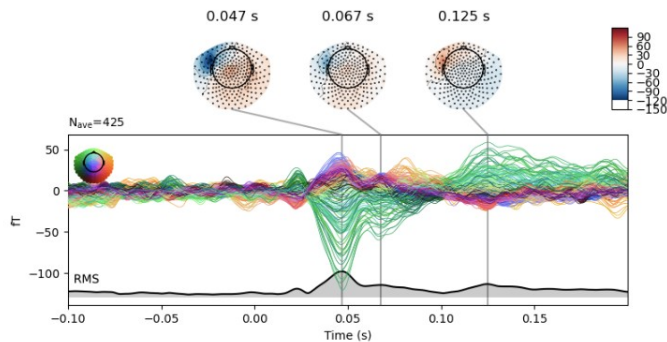
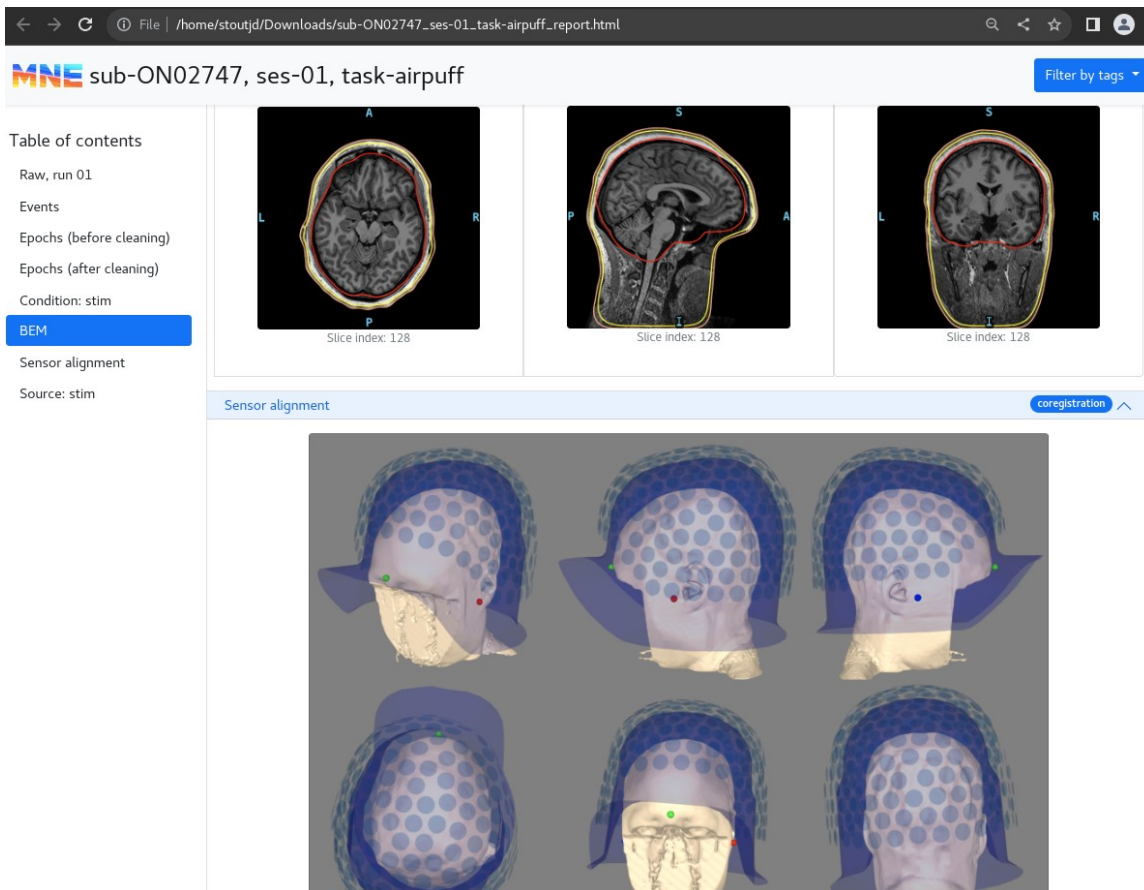
Mne-bids-pipeline-run.py --config=MYCONFIG.py --subject=ON02747 --steps=source,report  
--task=airpuff

```
study_name = 'TESTSTudy'
bids_root = 'YOUR_BIDS_DIR' #<< Modify
l_freq = 1.0
h_freq = 100.
epochs_tmin = -0.1
epochs_tmax = 0.2
baseline = (-0.1, 0.0)
resample_sfreq = 300.0
ch_types = ['meg']
conditions = ['stim'] #<< list of conditions
N_JOBS=4
```

# Run Example

- Code please

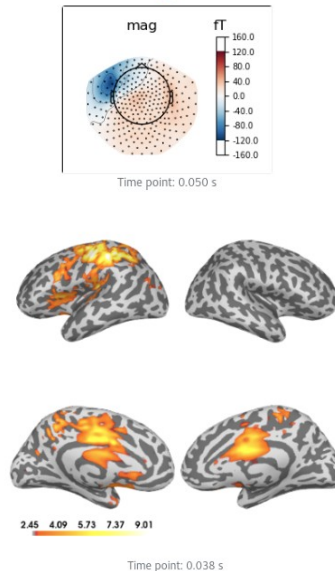
# Automatically Generated Interactive QA Report



Topographies

stim


















Move slider to change view



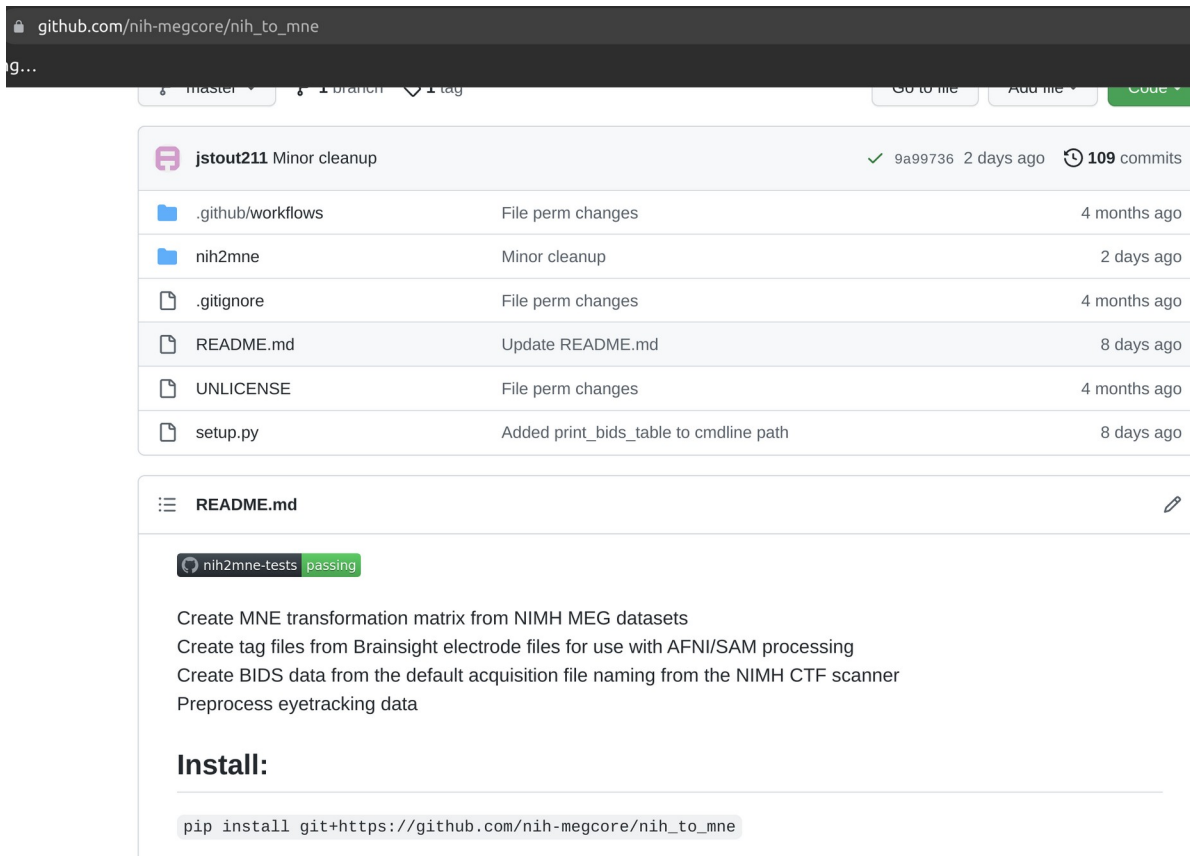


# Also saves all intermediary steps

# These can be loaded for additional analysis

data	BIDS_ZOOM	derivatives	mne-bids-pipeline	sub-ON02747	ses-01	meg
Name						
	sub-ON02747_ses-01_task-airpuff_ave.fif					
	sub-ON02747_ses-01_task-airpuff_cov.fif					
	sub-ON02747_ses-01_task-airpuff_epo.fif					
	sub-ON02747_ses-01_task-airpuff_fwd.fif					
	sub-ON02747_ses-01_task-airpuff_inv.fif					
	sub-ON02747_ses-01_task-airpuff_missingstim+dSPM+hemi-lh.stc					
	sub-ON02747_ses-01_task-airpuff_missingstim+dSPM+hemi-rh.stc					
	sub-ON02747_ses-01_task-airpuff_missingstim+dSPM+morph2fsaverage+hemi-lh.stc					
	sub-ON02747_ses-01_task-airpuff_missingstim+dSPM+morph2fsaverage+hemi-rh.stc					
	sub-ON02747_ses-01_task-airpuff_proc-clean_epo.fif					
	sub-ON02747_ses-01_task-airpuff_report.html					
	sub-ON02747_ses-01_task-airpuff_run-01_proc-filt_raw.fif					
	sub-ON02747_ses-01_task-airpuff_stim+dSPM+hemi-lh.stc					
	sub-ON02747_ses-01_task-airpuff_stim+dSPM+hemi-rh.stc					
	sub-ON02747_ses-01_task-airpuff_stim+dSPM+morph2fsaverage+hemi-lh.stc					
	sub-ON02747_ses-01_task-airpuff_stim+dSPM+morph2fsaverage+hemi-rh.stc					
	sub-ON02747_ses-01_task-airpuff_trans.fif					

# Create BIDS from NIMH data



The screenshot shows the GitHub repository page for `nih-megcore/nih_to_mne`. The repository is owned by `jstout211` and has a "Minor cleanup" commit 2 days ago. The repository contains several files and folders, including `.github/workflows`, `nih2mne`, `.gitignore`, `README.md`, `UNLICENSE`, and `setup.py`. The `README.md` file is expanded, showing the project's purpose and installation instructions.

Repository: `jstout211` Minor cleanup ✓ 9a99736 2 days ago 109 commits

File/Folder	Change	Time
<code>.github/workflows</code>	File perm changes	4 months ago
<code>nih2mne</code>	Minor cleanup	2 days ago
<code>.gitignore</code>	File perm changes	4 months ago
<code>README.md</code>	Update README.md	8 days ago
<code>UNLICENSE</code>	File perm changes	4 months ago
<code>setup.py</code>	Added print_bids_table to cmdline path	8 days ago

**README.md**

`nih2mne-tests` passing

Create MNE transformation matrix from NIMH MEG datasets  
Create tag files from Brainsight electrode files for use with AFNI/SAM processing  
Create BIDS data from the default acquisition file naming from the NIMH CTF scanner  
Preprocess eyetracking data

**Install:**

```
pip install git+https://github.com/nih-megcore/nih_to_mne
```

Install on your system or use biowulf module

Install script adds mne, mne\_bids, and nih\_to\_mne to python/conda environment

# Biowulf Modules

- Add to .bashrc
  - module use –append /data/MEGmodules/modulefiles

```
[stoutjd@cn4303 ~]$ module spider mne
```

mne:

Versions:

mne/0.24.1

mne/1.0

mne/1.0.3

Other possible modules matches:

MNE mne\_bids\_pipeline mne\_hcp mne\_scripts mne\_spyder

# Biowulf Module

# mne\_scripts

- Loads python env, freesurfer, afni
- Given a meg\_input\_dir and a brik/(or bsight),
  - this will automatically identify subject ID and task IDs

```
[stoutjd@cn4303 ~]$ module load mne_scripts
[+] Loading freesurfer 7.1.1 on cn4303
[+] Loading AFNI current-openmp ...
AFNI/current-openmp last updated 2022-10-21
```

```
[+] Loading mne 1.0 ...
[+] Loading mne_scripts 0.1_dev ...
Available:
    spatiotemporal_clustering_stats.py
    make_meg_bids.py
    bstags.py
    calc_mnetrans.py
[stoutjd@cn4303 ~]$ make_meg_bids.py -h
usage:
```

Convert MEG dataset to default Bids format using the MEG hash ID or entered subject ID as the bids ID.

WARNING: This does NOT anonymize the data!!!

```
[ -h ] [ -bids_dir BIDS_DIR ] [ -meg_input_dir MEG_INPUT_DIR ] [ -mri_brik MRI_BRIK ]  
[ -mri_bsight MRI_BSIGHT ] [ -mri_bsight_elec MRI_BSIGHT_ELEC ]  
[ -bids_session BIDS_SESSION ] [ -subj_id SUBJ_ID ]
```

```
options:
```

[illegible]

# MNE bids pipeline

Configures environment and enables virtualized 3D rendering for saving images to the report (requires the lscratch)

Set the cpus-per-task to be the same the number of parallel computations

```
[stoutjd@cn4303 ~]$ module load mne_bids_pipeline
[+] Loading freesurfer 7.1.1 on cn4303
[+] Loading Xvfb 1.19.6 on cn4303
[+] Loading singularity 3.10.3 on cn4303
[+] Loading mesa 17.0.0 ...
[+] Loading mne_bids_pipeline 0.2dev ...
[+]
[+] !! Make sure you have an lscratch if performing report gen!!
[+] (prior to loading this module you should have initiated like below)
[+] e.g.:
[+] sinteractive --mem=6G --cpus-per-task=4 --gres=lscratch:50
[+]
[+] Available Scripts
[+] mne-bids-pipeline-run.py
```

## TEST DATA for biowulf

---

This section provides some test data to analyze using mne-bids-pipeline. Feel free to adjust parameters in the config.py after you run through the analysis the first time. All of the analysis parameters are defined in the config.py provided. Additional config.py parameters can be found above in [Processing](#).

### Start Interactive Session

```
sinteractive --mem=6G --cpus-per-task=4 --gres=lscratch:50
```

### Copy and untar the data into your folder

```
cp -R /vf/users/MEGmodules/modules/bids_example_data_airpuff.tar ./
tar -xvf bids_example_data_airpuff.tar

#Add the bids_root to your config file
echo bids_root='\$(pwd)/bids_example_data_airpuff\' >>
$(pwd)/bids_example_data_airpuff/config.py
```

### Load module and process the data

```
module load mne_bids_pipeline
mne-bids-pipeline-run.py --config=$(pwd)/bids_example_data_airpuff/config.py
```

```
#Copy this path for the next step
echo $(pwd)/bids_example_data_airpuff/derivatives/mne-bids-pipeline/sub-ON02747/ses-01/meg/sub-ON02747_ses-01_task-airpuff_report.html
```

Thats All Folks!