

The Brain Imaging Data Structure (BIDS) Ecosystem and fMRIPrep

Saren Seeley
02/17/2020

sarenseeley@email.arizona.edu



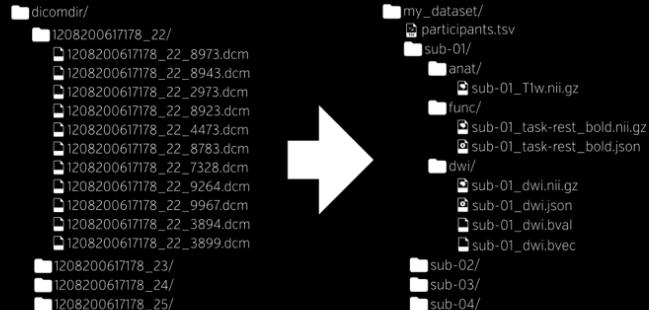
Brain Imaging Data Structure

A simple and intuitive way to organize and describe your neuroimaging and behavioral data.

ABOUT NEWS BENEFITS THE SPECIFICATION GETTING STARTED GET INVOLVED
GOVERNANCE ACKNOWLEDGMENTS

About BIDS

Neuroimaging experiments result in complicated data that can be arranged in many different ways. So far there is no consensus how to organize and share data obtained in neuroimaging experiments. Even two researchers working in the same lab can opt to arrange their data in a different way. Lack of consensus (or a standard) leads to misunderstandings and time wasted on rearranging data or rewriting scripts expecting certain structure. Here we describe a simple and easy to adopt way of organizing neuroimaging and behavioral data.



BIDS is heavily inspired by the format used internally by the OpenfMRI repository (now known as OpenNeuro.org). While working on BIDS we consulted many neuroscientists to make sure it covers most common experiments, but at the same time is intuitive and easy to adopt. The specification is intentionally based on simple file formats and folder structures to reflect current lab practices and

<http://bids.neuroimaging.io>

Consensus standard for organizing and describing MRI datasets:

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/

<https://bids-specification.readthedocs.io>

Getting started with BIDS

- Getting your data into BIDS format is often the major stumbling block

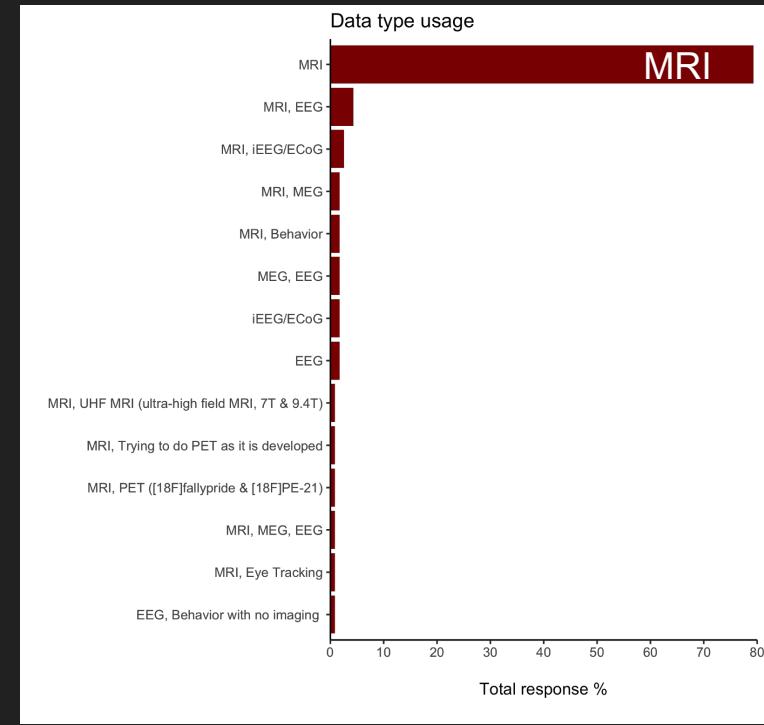
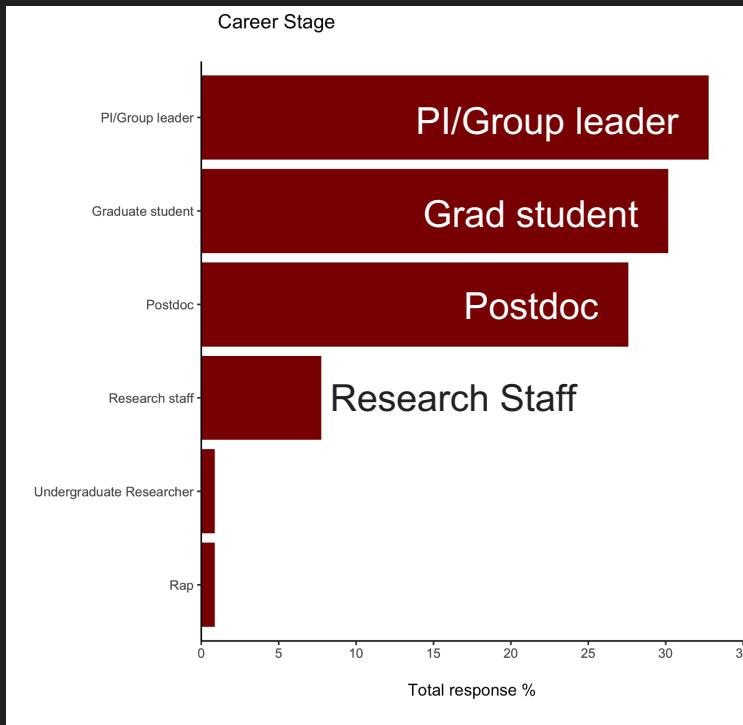
*Our data often look
more like this:*

```
data/
└── dicom
    ├── D118A_020916
    │   └── Anonymized - 3355AC90
    │       └── O'connor Sequences
    │           ├── EPI_2000_92_220 - 5
    │           ├── EPI_2000_92_220 - 7
    │           └── Localizers - 1
    ├── D118B_021616
    │   └── Anonymized - 2E78E7A7
    │       └── Oconnor Sequences
    │           ├── EPI_2000_92_220 - 6
    │           ├── EPI_2000_92_220 - 8
    │           └── Localizers - 1
    └── MoCoSeries
        ├── MoCoSeries - 8
        ├── MoCoSeries - 10
        ├── MoCoSeries - 11
        └── PhoenixZIPReport - 99
    └── RestingState - 9
    └── T1-MPRAGE - 11
    └── D118A_020916
```

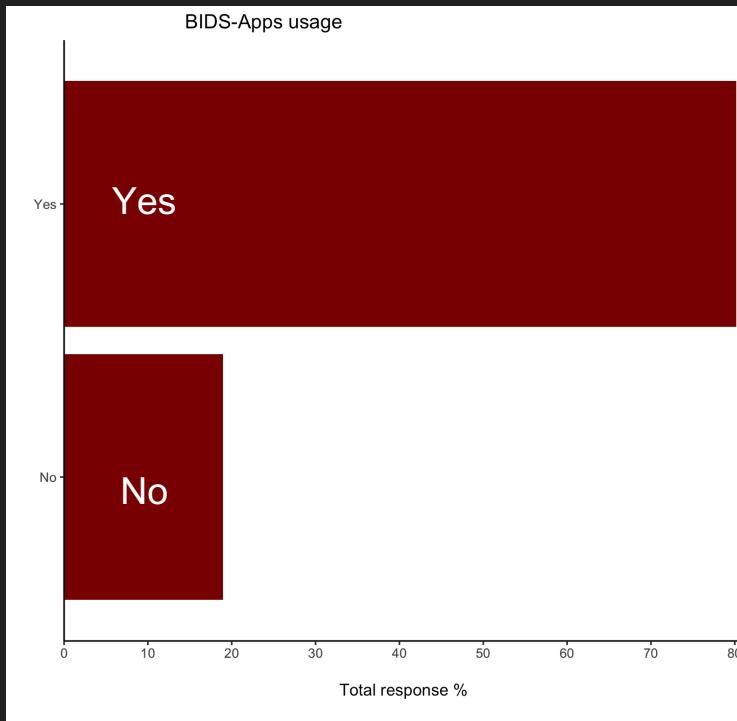
Getting Started with BIDS

- Goal of BIDS Starter Kit is to simplify the process
- Tutorials, wiki, templates, and utilities at:
<https://github.com/bids-standard/bids-starter-kit>
- Dianne's tutorials here (esp. for HeuDiConv):
<https://neuroimaging-core-docs.readthedocs.io/en/latest/pages/bids.html#creating-bids-datasets>
- My tutorial (mostly dcm2niix):
<https://sarenseley.github.io/BIDS-fmriprep-MRIQC.html>

2019 BIDS Usage Survey ($n = 116$)



2019 BIDS Usage Survey ($n = 116$)



- 81% are using BIDS Apps
 - Programs or libraries that automatically query BIDS datasets
 - <https://bids-apps.neuroimaging.io>

Major advantage of having your data in
BIDS format: BIDS apps!

BIDS Apps

Available BIDS Apps

BIDS-Apps/example	version 0.0.7	open bug issues 0	build passing	open bug pull requests 0	docker pulls 10k	439.5MB 23 layers
BIDS-Apps/freesurfer	version 1.0.0	open bug issues 0	Build Failing	open bug pull requests 0	docker pulls 10k	2.6GB 52 layers
BIDS-Apps/ndmg	version v0.1.0	open bug issues 0	build passing	open bug pull requests 0	docker pulls 7.9k	920.9MB 31 layers
BIDS-Apps/BROCOLI	version v1.0.1	open bug issues 1	build passing	open bug pull requests 0	docker pulls 505	3GB 21 layers
BIDS-Apps/FibreDensityAndCrosssection	version v0.0.1	open bug issues 0	build passing	open bug pull requests 0	docker pulls 237	576.8MB 31 layers
BIDS-Apps/SPM	version v0.0.15	open bug issues 0	build passing	open bug pull requests 0	docker pulls 1.4k	2GB 24 layers
poldracklab/mriqc	version 0.1.0	open bug issues 1	build passing	open bug pull requests 0	docker pulls 20k	3GB 41 layers
BIDS-Apps/GAP	Image not found	open bug issues 0	build passing	open bug pull requests 0	docker pulls 7	Image not found
BIDS-Apps/CPAC	version v1.01a_22	open bug issues 0	build passing	open bug pull requests 0	docker pulls 3.2k	1.6GB 45 layers
BIDS-Apps/hyperalignment	Image not found	open bug issues 0	build passing	open bug pull requests 0	docker pulls 165	Image not found
BIDS-Apps/mindboggle	version 0.0.4-1	open bug issues 2	build passing	open bug pull requests 0	docker pulls 647	1.9GB 81 layers
BIDS-Apps/MRtrix3_connectome	version latest	open bug issues 0	build passing	open bug pull requests 0	docker pulls 1.2k	7.3GB 54 layers
BIDS-Apps/rs_signal_extract	version 0.1	open bug issues 0	build passing	open bug pull requests 0	docker pulls 22k	240MB 17 layers
BIDS-Apps/aa	version v0.2.0	open bug issues 0	build passing	open bug pull requests 0	docker pulls 259	6.5GB 30 layers
BIDS-Apps/nlak	version latest	open bug issues 1	build passing	open bug pull requests 0	docker pulls 263	2.7GB 103 layers
BIDS-Apps/oppri	version 0.0.2-1	open bug issues 0	build passing	open bug pull requests 0	docker pulls 300	2.3GB 41 layers
poldracklab/fmriprep	version 1.5.0	open bug issues 0	Build Failing	open bug pull requests 0	docker pulls 3319	5GB 49 layers
BIDS-Apps/brainiak-srm	version latest	open bug issues 0	Build Failing	open bug pull requests 0	docker pulls 168	559.3MB 13 layers
BIDS-Apps/nipypipelines	version 0.3.0	open bug issues 0	build passing	open bug pull requests 0	docker pulls 291	478.1MB 20 layers
BIDS-Apps/HCPPipelines	version v3.17.0-18	open bug issues 0	build passing	open bug pull requests 0	docker pulls 1.7k	4GB 31 layers
BIDS-Apps/MAGeTbrain	Image not found	open bug issues 0	build passing	open bug pull requests 0	docker pulls 444	Image not found
BIDS-Apps/tracula	version v6.0.0-1	open bug issues 0	build passing	open bug pull requests 0	docker pulls 649	3.4GB 57 layers
BIDS-Apps/baracus	Image not found	open bug issues 0	build passing	open bug pull requests 0	docker pulls 1.1k	Image not found
BIDS-Apps/antsCorticalThickness	version v2.2.0-1	open bug issues 0	build passing	open bug pull requests 0	docker pulls 116	351.9MB 21 layers
BIDS-Apps/DPARSF	version v4.3.12	open bug issues 0	build passing	open bug pull requests 0	docker pulls 29k	1.4GB 28 layers
BIDS-Apps/afni_proc	Image not found	open bug issues 0	build passing	open bug pull requests 0	docker pulls 186	Image not found
BIDS-Apps/rshRF	version 1.0.1	open bug issues 0	build no builds	open bug pull requests 0	docker pulls 443	69MB 5 layers
BIDS-Apps/PyMVPA	version v2.0.2	open bug issues 0	build passing	open bug pull requests 0	docker pulls 144	0B 26 layers
PennBBL/qsiprep	Image not found	open bug issues 0	build passing	open bug pull requests 0	docker pulls 1.4k	Image not found

“Portable neuroimaging pipelines that understand BIDS datasets”

- Container image capturing a neuroimaging pipeline
- Same core set of command-line arguments
- Accessible via Docker Hub repo
 - Versioned & historical versions available
 - By reporting the BIDS App name and version in a manuscript, authors can provide others with the ability to exactly replicate their analysis workflow.

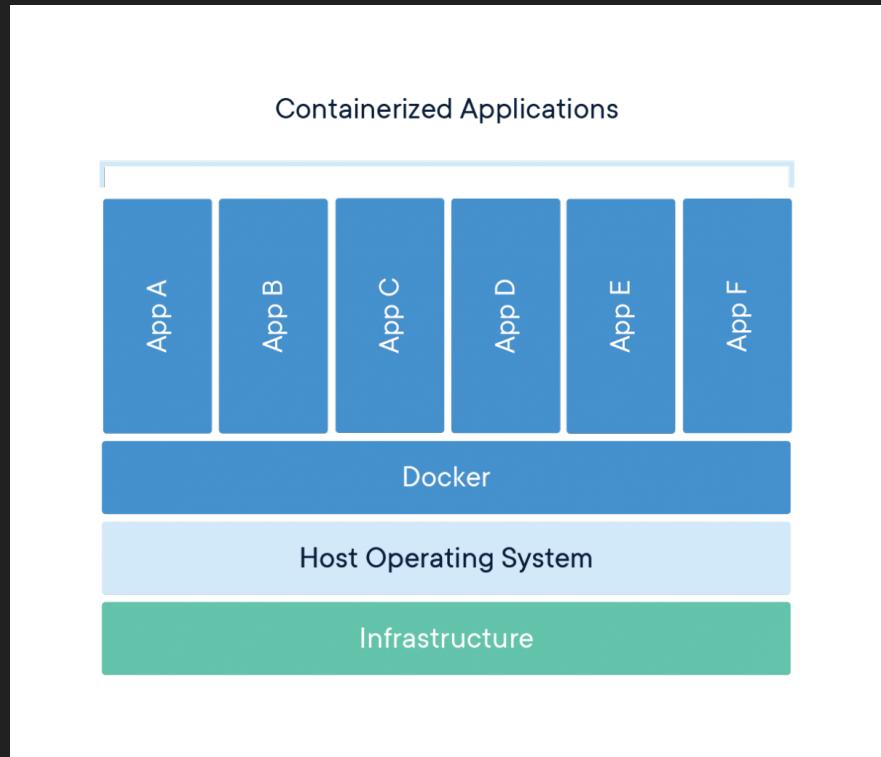
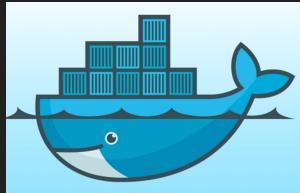
App name	Description	Applicable modalities	References
example	Example App that also serves as a template for new apps. Calculates intracranial volume.	T1w	n/a
Freesurfer	Surface extraction, longitudinal pipeline and study specific template calculation using FreeSurfer.	T1w	[19,20]
ndmg	One-click reliable and reproducible pipeline for T1w + DWI weighted MRI connectome estimation.	T1w, DWI	[21,22]
BROCCOLI	Fast fMRI analysis on many-core CPUs and GPUs.	T1w, fMRI	[23]
FibreDensityAndCrosssection	Pixel-Based Analysis (FBA) of Fibre Density and Fibre Cross-section.	DWI	[24,25]
SPM	Statistical Parametric Mapping.	T1w, fMRI	[26]
MRIQC	Quality Assessment of structural and functional MRI.	T1w, fMRI	[52]
FMRIprep	A generic fMRI preprocessing pipeline providing results robust to the input data quality as well as informative reports.	T1w, fMRI	In preparation
Quality Assessment Protocol	Quality Assessment of structural and functional MRI.	T1w, fMRI	[27]
Configurable Pipeline for the Analysis of Connectomes	Pipeline for high throughput processing and analysis of structural and functional MRI data.	T1w, fMRI	[28]
Hyperalignment	Computes hyperalignment transformations for functional alignment.	fMRI	[29]
mindboggle	Pipeline to improve the accuracy, precision, and consistency of automated labeling and shape analysis of human brain image data.	T1w	[30]
MRtrix3 connectome	Robust generation and statistical analysis of structural connectomes estimated from diffusion tractography.	T1w, DWI	[31]
nilearn	Extraction of time-series and connectomes for population analysis.	fMRI	[32]
nipypipelines	Preprocessing of functional time series for resting or task analysis	T1w, fMRI	[16]
automatic analysis (aa)	Neuroimaging pipeline system written in Matlab.	T1w, T2w, fMRI, DWI	[33]
Niak Preprocessing	Noise reduction, segmentation, coregistration, motion estimation, resampling.	fMRI	[34]
HPCPipelines	Anatomical and functional preprocessing pipelines used in the Human Connectome Project.	T1w, T2w, fMRI	[35,36]
BrainIAK-SRM	Functional alignment using Shared Response Model implementation from the Brain Imaging Analysis Kit.	fMRI	[37]
OPPNI	Optimization of Preprocessing Pipelines for NeuroImaging, for analysis of fMRI data.	fMRI	[38–41]
MAGeTbrain	Multiple Automatically Generated Templates brain segmentation algorithm	T1w	[42–45]
tracula	Automatic reconstruction of a set of major white-matter pathways from diffusion-weighted MR images	T1w, DWI	[46]

Two options for running BIDS apps:

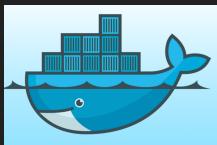
- Docker (*local*)
- Singularity (*HPC*)

Docker

- Packages all necessary code & dependencies → runs reliably across computing environments
- Runs locally
(requires root access)

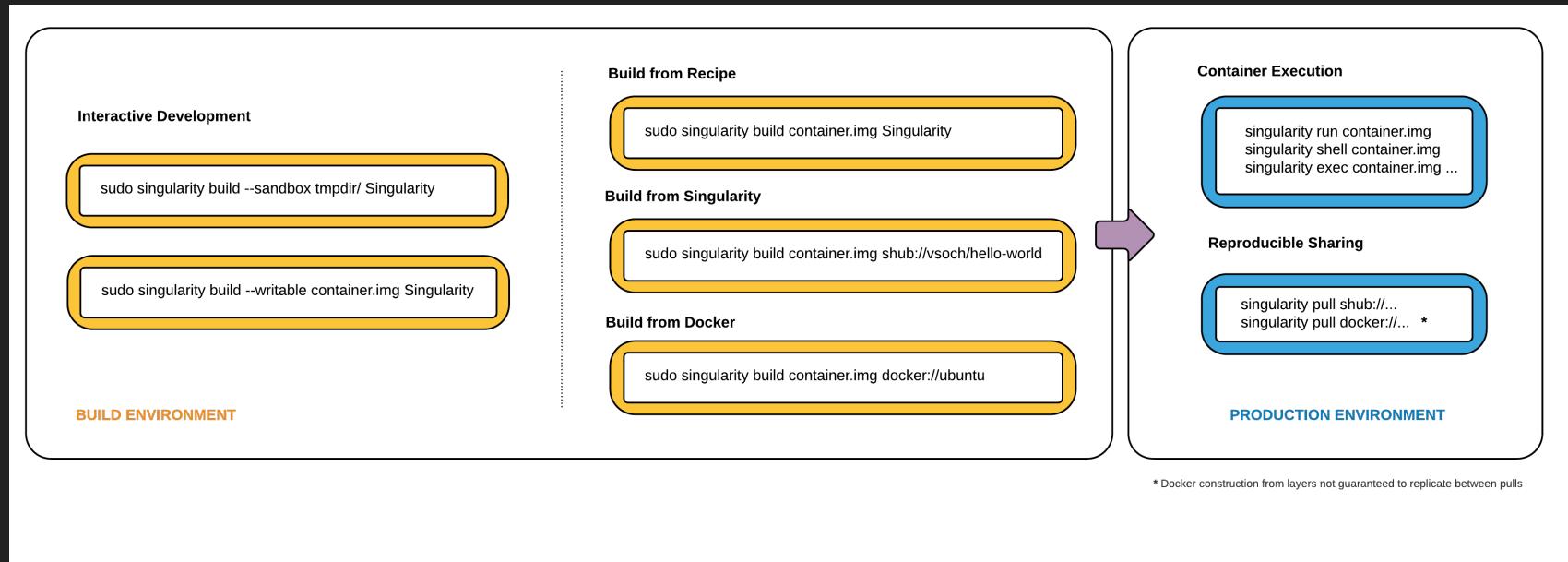


Running a BIDS app: Docker



```
docker run -ti --rm \ -v  
/Users/srycajal/data/ds005:/bids_dataset:ro \ -v  
/Users/srycajal/outputs:/outputs \ bids/example:0.0.4 \  
/bids_dataset /outputs participant --participant_label 01
```

Singularity



Running a BIDS app: Singularity

1. Save to Singularity-compatible image file:

```
docker run --privileged -ti --rm \ -v  
/var/run/docker.sock:/var/run/docker.sock \ -v  
/home/srycajal/singularity_images:/output \  
singularityware/docker2singularity \ bids/example:0.0.4
```

2. Transfer Singularity image (.img file) to cluster

3. Run .img file:

```
./bids_example-0.0.4.img /bids_dataset /outputs  
participant --participant_label 01
```

BIDS validator

Can be run online (<http://bids-standard.github.io/bids-validator/>)

- Chrome or Firefox only
- “*Selecting a dataset only performs validation. Files are never uploaded.*”
- Select your BIDS dataset parent directory and wait for it to finish validation.
- View errors and warnings.
 - Click the link at the bottom of the page to download the error log.
 - Some of the errors I ran into (and their solutions) are listed in my BIDS-conversion notebook on [my website](#).

fMRIPrep

<http://fmriprep.readthedocs.io>

fmriprep

- An open-source Nipype-based pipeline for transparent and reproducible preprocessing workflows.
- Robust to variation across datasets; intended to be “analysis-agnostic”
- Performs “minimal preprocessing” (skull stripping, motion correction, segmentation, coregistration, normalization etc.)
- Integration of Freesurfer for surface based processing (optional)

fmriprep allows researchers to...

- Take fMRI data from raw to fully preprocessed form.
- Implement tools from different software packages; achieve optimal data processing quality by using the best tools available.
- Generate preprocessing quality reports, with which the user can easily identify outliers.
- Receive verbose output concerning the stage of preprocessing for each subject, including meaningful errors.
- Automate and parallelize processing steps, which provides a significant speed-up from typical linear, manual processing.

fmriprep

Built around three principles:

- Robustness
- Ease of use
- “Glass box” philosophy

fmriprep

Built around three principles:

- Robustness
- Ease of use
- “Glass box” philosophy
- Adapts the preprocessing steps depending on the input dataset.
- Should provide results as good as possible independently of scanner make, scanning parameters or presence of additional correction scans (such as fieldmaps).

fmriprep

Built around three principles:

- Robustness
- Ease of use
- “Glass box” philosophy
- Runs automatically: minimal manual parameter input required due to dependence on BIDS standard.

fmriprep

Built around three principles:

- Robustness
 - Ease of use
 - “Glass box” philosophy
-
- Automation should not mean that one should not visually inspect the results or understand the methods.
 - Generates visual reports for each subject detailing the accuracy of the most important steps.
 - Reports and documentation help researchers to understand the process and decide which subjects should be kept for group level analysis.

Before you start

- You need a Freesurfer license: [link to register & download \(free\)](#)
- You also need fMRIPrep to be able to *find* your Freesurfer license...
 - You might need to add the path to your bash profile, if it can't locate the file.

fMRIPrep Docker wrapper

Python script that operates the Docker Engine

- Makes command-line arguments slightly more intuitive.
- <https://fmriprep.readthedocs.io/en/stable/installation.html#the-fmriprep-docker-wrapper>

When run, fmriprep-docker will generate a Docker command line for you, print it out for reporting purposes, and then run the command.

```
$ pip install --user --upgrade fmriprep-docker  
$ fmriprep-docker /path/to/data/dir /path/to/output/dir participant
```

You can also invoke Docker directly:

```
$ docker run -ti --rm \  
  -v filepath/to/data/dir:/data:ro \  
  -v filepath/to/output/dir:/out \  
  poldracklab/fmriprep:latest \  
  /data /out/out \  
  participant
```

Running fMRIPrep via Docker

```
fmriprep-docker      ← Using the fMRIPrep Docker wrapper
/Users/sarensseeley/Desktop/data/          ← Input BIDS dataset
/Users/sarensseeley/Desktop/data/derivatives   ← Output goes here (under /fmriprep)
--low-mem
--skip_bids_validation
--stop-on-first-crash
--longitudinal
--use-syn-sdc
--use-aroma
--participant_label sub-130
--task-id gAAT
-w /Users/sarensseeley/Desktop/data/derivatives/scratch ← Local scratch directory
```

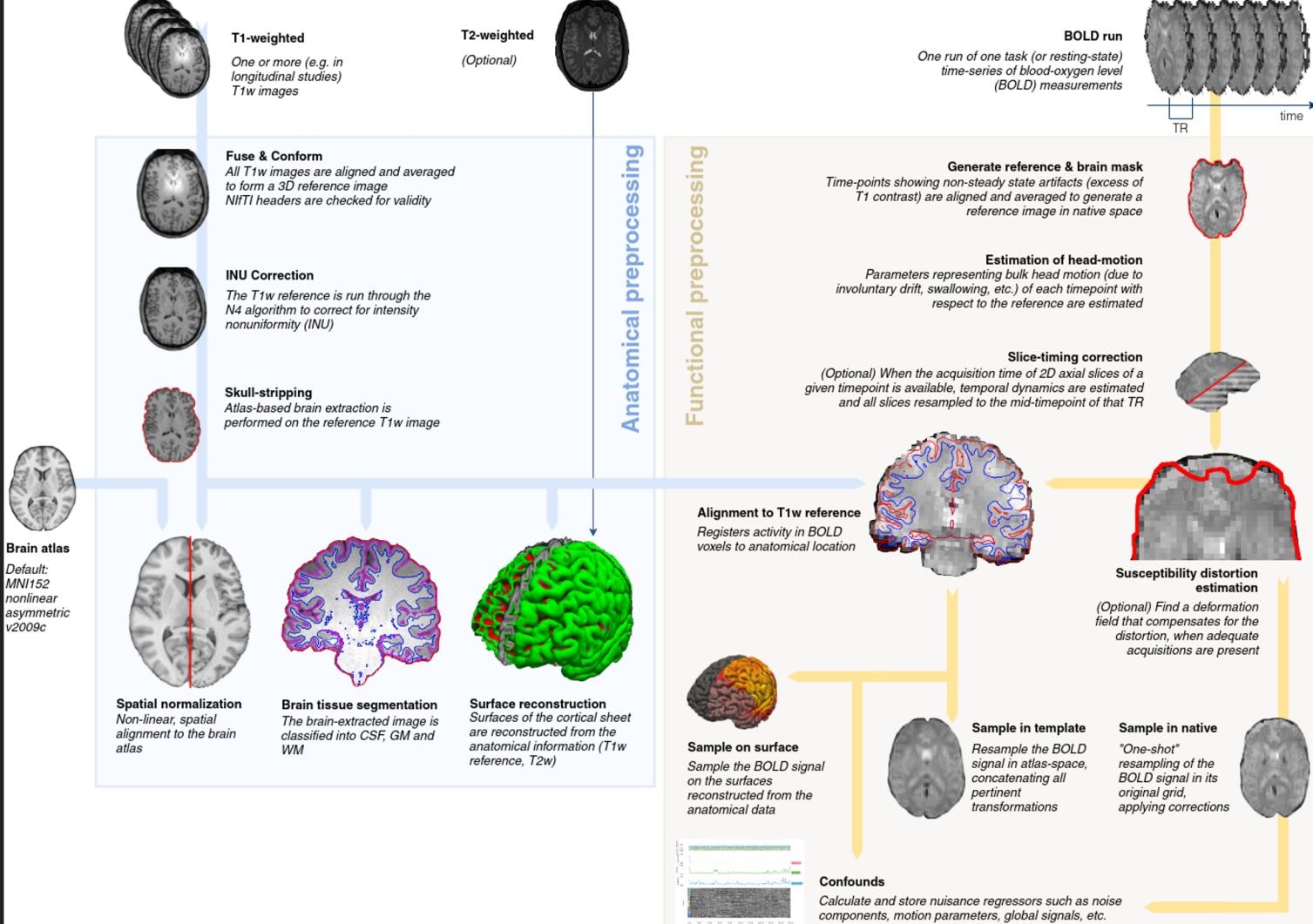


Command-line arguments; see
<https://fmriprep.readthedocs.io/en/stable/usage.html#command-line-arguments>

Tips for fMRIPrep via Singularity

- <https://fmriprep.readthedocs.io/en/stable/singularity.html>
- Multiple places where fMRIPrep may run into problems...
 - Defaults and configuration in your system
 - Environment and access to fMRIPrep resources (access to input/output data folders, BIDS validity, access and permissions to \$HOME, *TemplateFlow* operation)
- Data need to be de-identified (e.g., defacing via PyDeface or SPM)

Workflow



T1 preprocessing

N4 bias field correction (ANTs)

Skull stripping (ANTs)

3 class tissue segmentation (FSL FAST)

Robust MNI coregistration (ANTs)

EPI preprocessing

Motion correction (FSL MCFLIRT)

Skull stripping (nilearn)

Coregistration to T1 (FSL FLIRT with BBR / FreeSurfer bbregister if Freesurfer run)

Confounds estimation (nipype)

- Framewise displacement
- Global signal
- Mean tissue signal
- Temporal & anatomical CompCor

EPI transformations

Combination of:

- Motion correction affines
- EPI → T1 affine
- T1 → MNI affine
- T1 → MNI warp field

Single interpolation step

No upsampling (keep original voxel size)

Outputs

T1w

- Bias-corrected volume
- Brain mask
- Tissue segmentation (+ probability maps)
- Affine and warp to MNI (both ways)

Outputs

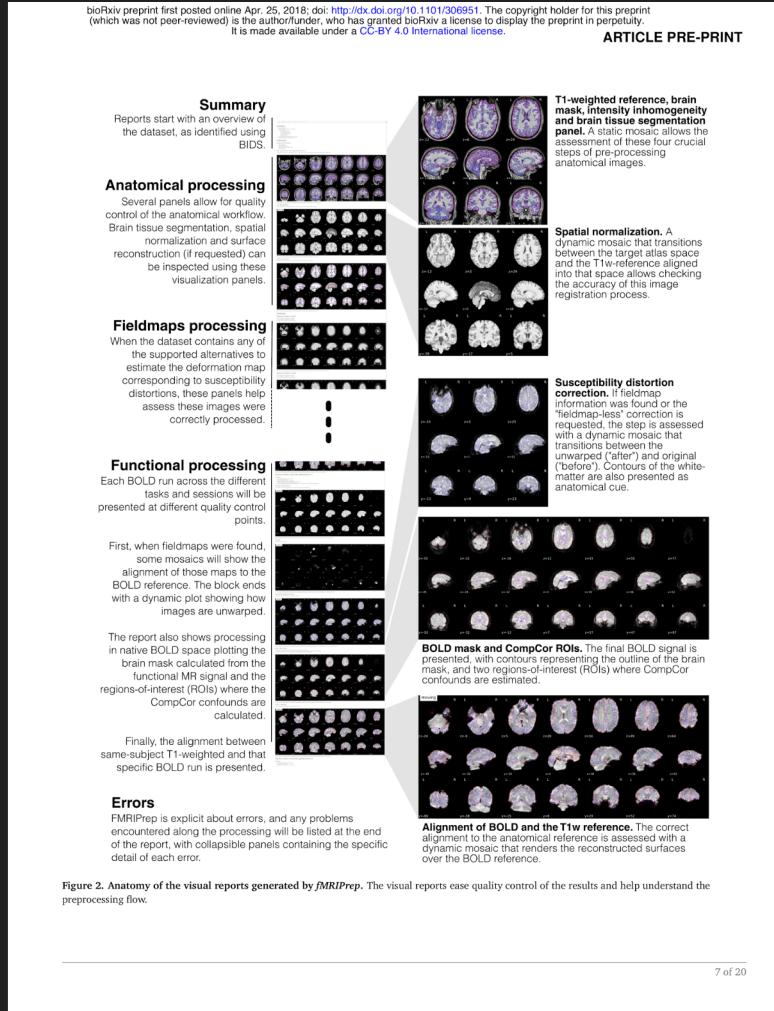
EPI

- Motion-corrected images
- Brain mask
- Affine T1w
- TSV file with all noise confounds

All volumes in MNI and native (EPI) space

fmriprep outputs

1. ***Visual QA (quality assessment) reports***: One HTML per subject, that allows the user a thorough visual assessment of the quality of processing and ensures the transparency of fMRIPrep operation.
2. ***Pre-processed imaging data*** which are derivatives of the original anatomical and functional images after various preparation procedures have been applied.
3. ***Additional data for subsequent analysis***, e.g. the transformations between different spaces or the estimated confounds.
 - a. Mean global signal, mean tissue class signal, tCompCor, aCompCor, Frame-wise Displacement, 6 motion parameters, DVARS
 - b. If the --use-aroma flag is enabled, the noise components identified by ICA-AROMA



Let's look at some reports...

Questions?

Q: Should I run quality control of my images before running *fMRIPrep*?

A: Yes.

- Pre-specify exclusion criteria for your data/experiment.
- Run QC (e.g., via [MRIQC](#)).
- Decide which subjects/sessions to keep vs. exclude from preprocessing.
- Run fMRIPrep on subset of subjects/sessions that passed QC.

Q: What am I looking for in these reports and how do I use this info? (TMI !!)

- Working on it...
<https://docs.google.com/document/d/1TE6ZWzNg8cDpvL4Vu0VGOZQLXkQ88Fa59AORzN01Avk/edit?usp=sharing> (*draft in progress, input welcome!*)
- **New** improved documentation on confounds and regressors:
<https://fmriprep.readthedocs.io/en/stable/outputs.html#confounds>

"I recall the GIFTI output from fMRIPrep is not quite the same as from the HCP project. I also recall there was some work to address that issue. Would you happen to have an update?"

CM: CIFTI, I presume? HCP doesn't use a lot of GIFTI. The upcoming release is intended to have fully-compatible CIFTI outputs. If they are worried about GIFTI I would simply say that the GIFTIs produced by fMRIPrep are consistent with each other. The functional series (func.gii) are sampled to the surfaces (surf.gii), and both viewable in FreeView and connectome_wb.

SG: i think fmriprep should not equate HCP-flavored CIFTI to CIFTI (the current CIFTI output is as valid a CIFTI file as the future HCP-flavored CIFTI output).

OE: Let's say "grayordinates CIFTI" here instead of just CIFTI. Would [@satra](#) find that more appropriate?

SG: yes grayordinates makes it more specific to HCP. i would just use HCP- instead of CIFTI, just like we use MNI instead of the underlying data format.