**fmriprep Manual**

*Author: Daniel Elbich*

*Contact: delbich10@gmail.com*

*Getting Started*

fmriprep is an preprocessing pipeline for fMRI data developed by the Poldrack lab at Standford. It is an amalgamation of multiple data processing programs, bringing together the best of each to form a single unified pipeline. The aim to develop a pipeline which is robust for all functional data & scanner types, requires minimal user input, and be as transparent as possible about the choices made for preprocessing data. This package is undergoing continuous upgrading as newer/better software becomes available, so be sure to check their website ([https://fmriprep.readthedocs.io/en/stable/#](https://fmriprep.readthedocs.io/en/stable/)) for the most current updates. Any necessary upgrades to the program should be sent as a software request to ACI support.

*Quick Need to Know*

There are 2 important items to note before actually executing fmriprep. The first is that fmriprep was designed to use the BIDS style formatting for MR data (<https://bids.neuroimaging.io/>). Therefore, I recommend that any data processed by the package should be in the BIDS formatting instead of being skipped.

Second, the package is very particular about how information is formatted to be read into the program. For example, the program only uses NIfTI files as data inputs. In addition, each set of data must have an accompanying .json file which details important aspects of data collection, such as TR or slice timing. This is less needed for the structural image but is absolutely essential for any functional data. If this file is missing the package will not run. If there are unexpected values the program will crash. Also, if your installation is on a cluster the package could have difficulty accessing licensed software that is installed in root directories (e.g. Freesurfer). It is recommended you setup a single subject first and run that data through the pipeline to debug for errors.

*Executing the package*

To execute the package, first you will need to load the module. Because this is not installed on the stack yet (as of this writing), the module will need to be loaded separately. In terminal, navigate to the main directory (parent directory containing the Software folder). Here, you want to reset where the ‘module’ command is looking for modules and load in fmriprep:

‘module use Software/fmriprep/modules’

‘module load fmriprep/latest’

Once the package is loaded, you can begin to preprocess data. The 3 absolute requirements are the path to the data, the path to save the output, and the participant, as in:

‘fmriprep /path/to/data/dir /path/to/output/dir {participant}’

Again this will have errors trying to run the if the data are not organized in BIDS style formatting to ensure this is correct.

The script 'createPBSScriptfmriprep.sh' will create PBS scripts to run the default pipeline. Simply change the file paths to correspond with the data directories location of the fmriprep module. Include '--help' to see additional options for that script.

*Reading the Output*

fmriprep will take around 8 hours to complete the complete pipeline. This time will also depend on the processing resources allocated to the job. Upon completion, navigate to the output directory you specified in the ‘fmriprep’ command. There will be 2 new folders, an fmriprep folder and a freesurfer folder. The freesurfer folder will contain all files relating to freesurfer’s processing of the anatomical image. The fmriprep folder will contain all files (and output) related to functional data processing and registration to the T1 structure and standard space (if specified). There will also be a report of the processing steps and results in the form of an .html file, which can be opened in any browser. This will be the first round of QA to check for registration, signal dropout, and data quality metrics (e.g. motion). Within each folder (anat, func) you will find the newly processed data to use in your processing program of choice.

*Optional Flags*

fmriprep does include optional arguments that can change the preprocessing choices. Below are just a few of the more pertinent choices (in my opinion), however using the flag ‘--help’ will give you the full list. Please also see the fmriprep website ([https://fmriprep.readthedocs.io/en/stable/#](https://fmriprep.readthedocs.io/en/stable/)) for more details.

Options for filtering BIDS queries:

--skip\_bids\_validation/--skip-bids-validation #assume the input dataset is BIDS compliant and skip the validation

--participant\_label/--participant-label PARTICIPANT\_LABEL [PARTICIPANT\_LABEL ...] #a space delimited list of participant identifiers or a single identifier (the sub- prefix can be removed)

-t/--task-id TASK\_ID #select a specific task to be processed

--echo-idx ECHO\_IDX #select a specific echo to be processed in a multiecho series

Options to handle performance:

--nthreads/--n\_cpus/-n-cpus #NTHREADS maximum number of threads across all processes

--omp-nthreads OMP\_NTHREADS #maximum number of threads per-process

--mem\_mb/--mem-mb MEM\_MB #upper bound memory limit for FMRIPREP processes

--low-mem #attempt to reduce memory usage (will increase disk usage in working directory)

--use-plugin USE\_PLUGIN #nipype plugin configuration file

--anat-only #run anatomical workflows only

--boilerplate #generate boilerplate only

--error-on-aroma-warnings #Raise an error if ICA\_AROMA does not produce sensible output (e.g., if all the components are classified as signal or noise)

-v, --verbose #increases log verbosity for each occurence, debug level is -vvv

Workflow configuration:

--ignore {fieldmaps,slicetiming,sbref} [{fieldmaps,slicetiming,sbref} ...] #ignore selected aspects of the input dataset to disable corresponding parts of the workflow (a space delimited list)

--longitudinal #treat dataset as longitudinal - may increase runtime

--t2s-coreg #If provided with multi-echo BOLD dataset, create T2\*-map and perform T2\*-driven coregistration. When multi-echo data is provided and this option is not enabled, standard EPI-T1 coregistration is performed using the middle echo.

--output-spaces OUTPUT\_SPACES [OUTPUT\_SPACES ...] #Standard and non-standard spaces to resample anatomical and functional images to. Standard spaces may be specified by the form ``<TEMPLATE>[:res-<resolution>][:cohort-<label>][...]``, where ``<TEMPLATE>`` is a keyword (valid keywords: "MNI152Lin", "MNI152NLin2009cAsym", "MNI152NLin6Asym", "MNI152NLin6Sym", "NKI", "OASIS30ANTs", "PNC", "fsLR", "fsaverage") or path pointing to a user-supplied template, and may be followed by optional, colon-separated parameters. Non-standard spaces (valid keywords: anat, T1w, run, func, sbref, fsnative) imply specific orientations and sampling grids

--bold2t1w-dof {6,9,12} #Degrees of freedom when registering BOLD to T1w images. 6 degrees (rotation and translation) are used by default.

--force-bbr #Always use boundary-based registration (no goodness-of-fit checks)

--force-no-bbr #Do not use boundary-based registration (no goodness-of-fit checks)

--medial-surface-nan #Replace medial wall values with NaNs on functional GIFTI files. Only performed for GIFTI files mapped to a freesurfer subject (fsaverage or fsnative).

--dummy-scans DUMMY\_SCANS #Number of non steady state volumes.

Specific options for running ICA\_AROMA:

--use-aroma #add ICA\_AROMA to your preprocessing stream

--aroma-melodic-dimensionality AROMA\_MELODIC\_DIMENSIONALITY #Exact or maximum number of MELODIC components to estimate (positive = exact, negative = maximum)

Specific options for estimating confounds:

--return-all-components #Include all components estimated in CompCor decomposition in the confounds file instead of only the components sufficient to explain 50 percent of BOLD variance in each CompCor mask

--fd-spike-threshold FD\_SPIKE\_THRESHOLD #Threshold for flagging a frame as an outlier on the basis of framewise displacement

--dvars-spike-threshold DVARS\_SPIKE\_THRESHOLD #Threshold for flagging a frame as an outlier on the basis of standardised DVARS

Specific options for ANTs registrations:

--skull-strip-template {OASIS30ANTs,NKI,MNI152NLin2009cAsym} #select ANTs skull-stripping template (default: OASIS30ANTs))

--skull-strip-fixed-seed #do not use a random seed for skull-stripping - will ensure run-to-run replicability when used with --omp-nthreads 1

Specific options for handling fieldmaps:

--fmap-bspline #fit a B-Spline field using least-squares (experimental)

--fmap-no-demean #do not remove median (within mask) from fieldmap

Specific options for SyN distortion correction:

--use-syn-sdc #EXPERIMENTAL: Use fieldmap-free distortion correction

--force-syn #EXPERIMENTAL/TEMPORARY: Use SyN correction in addition to fieldmap correction, if available

Specific options for FreeSurfer preprocessing:

--fs-license-file PATH #Path to FreeSurfer license key file. Get it (for free) by registering at https://surfer.nmr.mgh.harvard.edu/registration.html

Surface preprocessing options:

--no-submm-recon #disable sub-millimeter (hires) reconstruction

--cifti-output #output BOLD files as CIFTI dtseries

--fs-no-reconall #disable FreeSurfer surface preprocessing

Other options:

-w WORK\_DIR/--work-dir WORK\_DIR #path where intermediate results should be stored

--resource-monitor #enable Nipype's resource monitoring to keep track of memory and CPU usage

--reports-only #only generate reports, don't run workflows. This will only rerun report aggregation, not reportlet generation for specific nodes.

--run-uuid RUN\_UUID #Specify UUID of previous run, to include error logs in report. No effect without --reports-only.

--write-graph #Write workflow graph.

--stop-on-first-crash #Force stopping on first crash, even if a work directory was specified.

--notrack #Opt-out of sending tracking information of this run to the FMRIPREP developers. This information helps to improve FMRIPREP and provides an indicator of real world usage crucial for obtaining funding.

--sloppy #Use low-quality tools for speed - TESTING ONLY