

fMRIPrep tutorial: A practical guide

Qing Wang
Vincent

vincent.w.qing@gmail.com
<https://github.com/Vincent-wq>
<https://github.com/Tutorial-series>



McGill



neuro

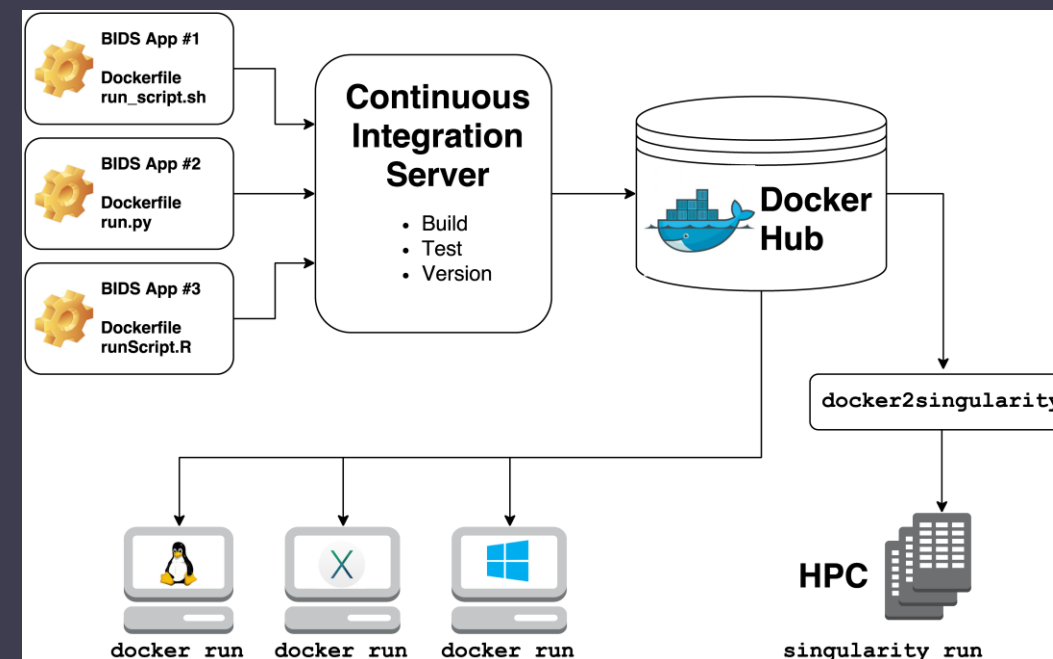
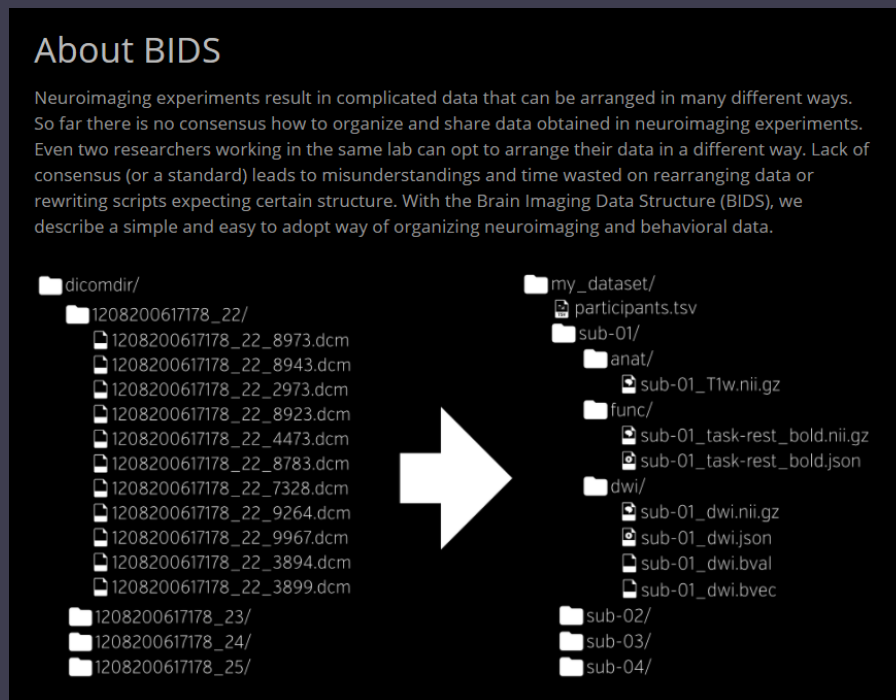
Institut-Hôpital
neurologique de Montréal

Contents

- 1 What is **fMRIPrep**, and what does it do
- 2 The inputs and outputs of **fMRIPrep**
- 3 Running **fMRIPrep** in a container (local)
- 4 Running **fMRIPrep** on HPC (Compute Canada)
- 5 Tips for running **fMRIPrep** on HPC
- 6 More Resources

NI Processing: BIDS and BIDS App

BIDS is way of organizing NI data with community consensus.



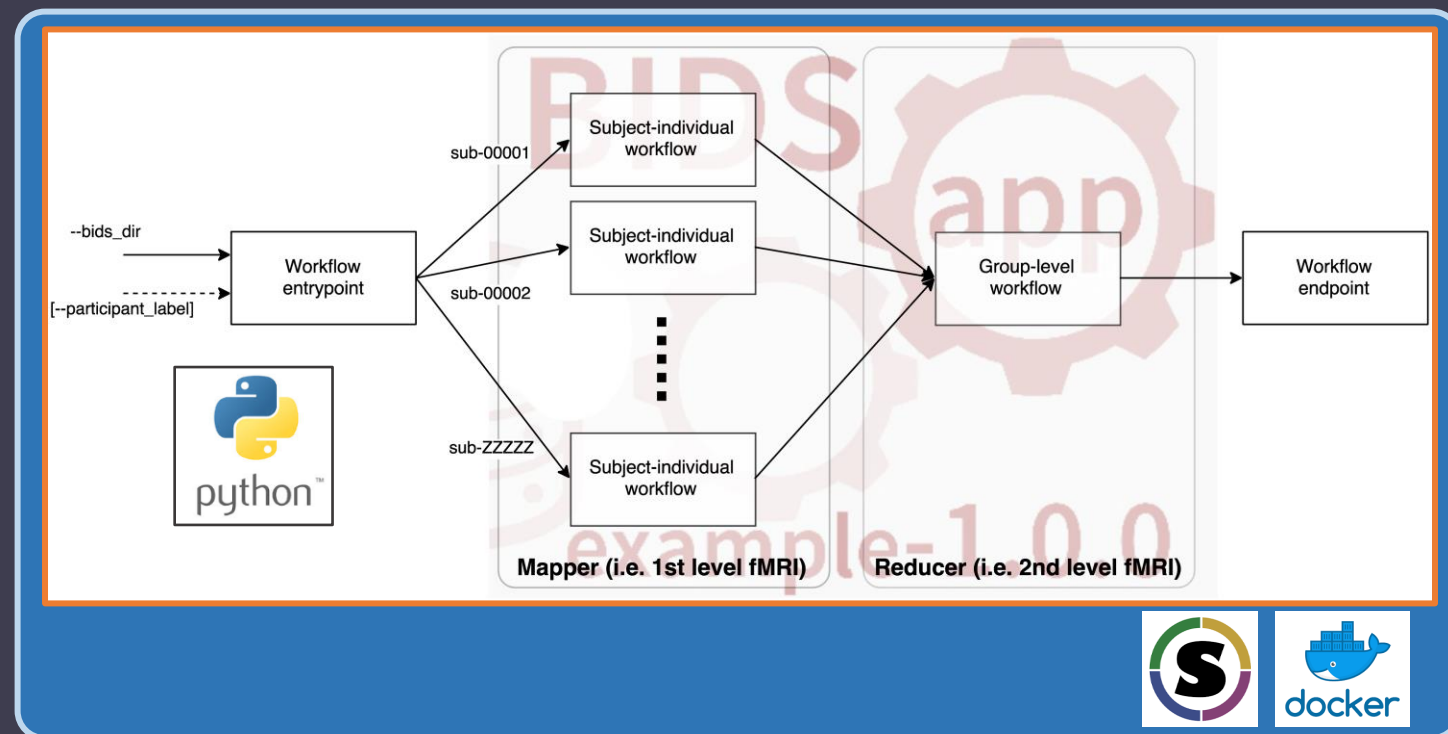
https://docs.google.com/document/d/1Wwc4A6Mow4ZPPszDIWfCUCRNstn7d_zzaWPcfCHmgI4/edit#heading=h.9146wuepc1kt
Gorgolewski KJ, et al. (2017). PLoS Comput Biol 13(3): e1005209.
<https://neuroimaging-core-docs.readthedocs.io/en/latest/pages/choropleths.html>

BIDS App: The basics

A **BIDS App** is a container image capturing a neuroimaging pipeline (most of the time, nipype wrapped) that takes a BIDS-formatted dataset as input.

Containers

- **Docker**: building, hosting and running containers locally (Windows, Mac OS X or Linux)
- **Singularity**: running containers on HPCs



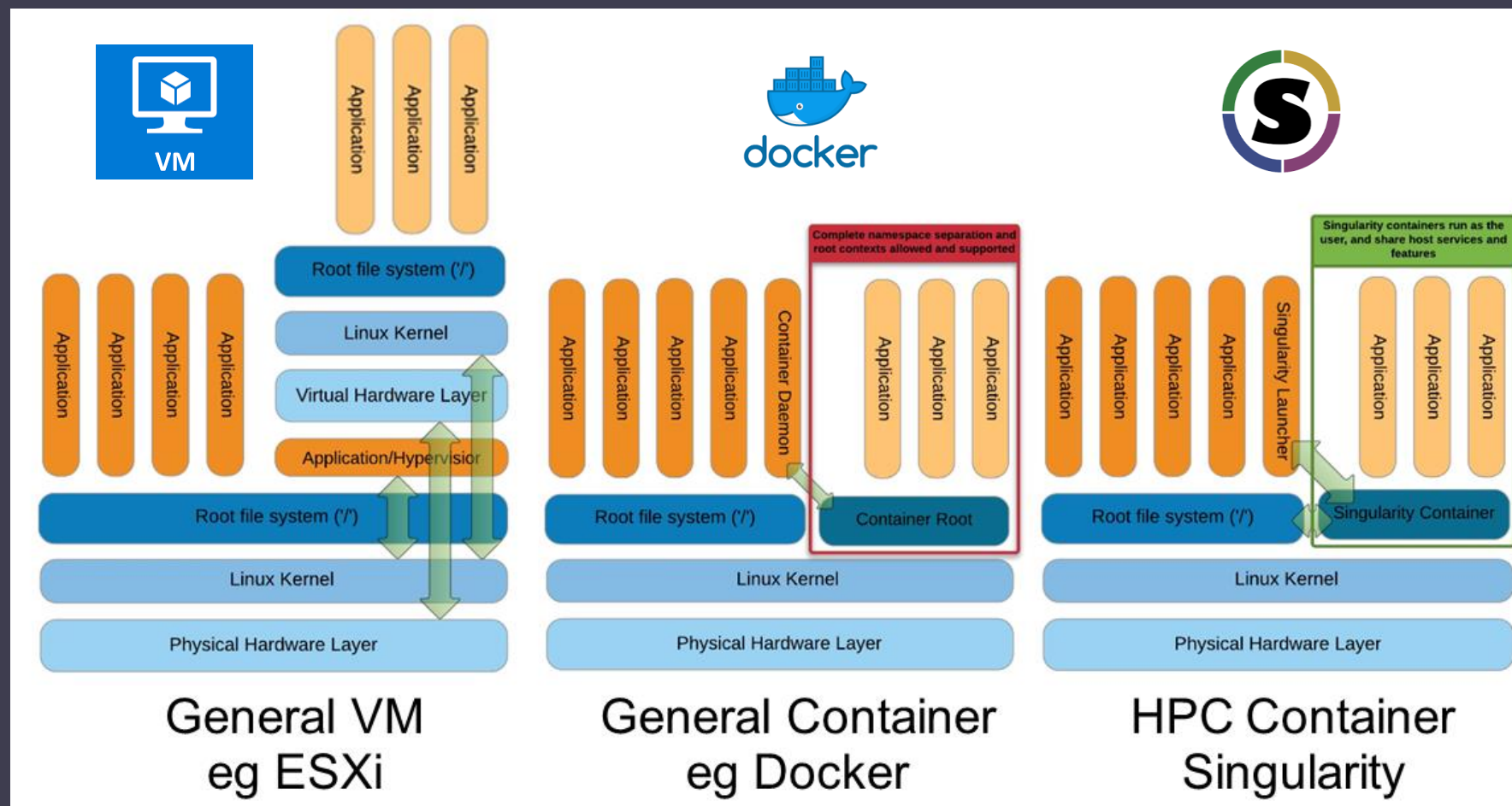
Running a BIDS App

```
BIDS_App(container image) bids_dir(input) derivative_dir(output) \  
group/participant(--participant_label xxx) (analysis_level) \  
confs(configuration for the BIDS_APP)
```

<https://bids.neuroimaging.io/>
<https://github.com/BIDS-Apps>

Gorgolewski KJ, et al. (2017). *PLoS Comput Biol* 13(3): e1005209.

Containers: A “vm” like wrapped-env for pipelines



Warning!

Be aware of the resources limit (2G memory or space for containers) in Mac OS and windows.

https://tin6150.github.io/psg/blogger_container_hpc.html

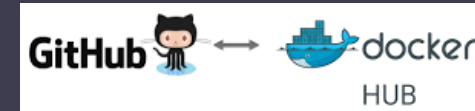


McGill

Containers: Instead of installation and dependencies

Build containers (root needed)

- `sudo docker pull poldracklab/fmriprep:latest`
- `sudo singularity build fmriprep_latest.simg docker://poldracklab/fmriprep:latest`
- `sudo docker build --no-cache -t your_container_name -f your_dockerfile .`



RUN a docker container

```
sudo docker run -it -p host_port:cont_port \  
-v host_bids_dir:cont_bids_dir:ro \  
-v host_derivates_dir:cont_derivates_dir \  
... \  
poldracklab/fmriprep:latest(docker_container:version) \  
\ cont_bids_dir cont_derivates_dir --participant_label \  
\ xxx (group) -w work_dir(conf)
```

RUN a singularity container

```
singularity run --cleanenv \  
-B host_bids_dir:cont_bids_dir:ro \  
-B host_derivates_dir:cont_derivates_dir \  
... \  
fmriprep_latest.simg (singularity_container) \  
cont_bids_dir cont_derivates_dir \  
--participant_label xxx (group) -w work_dir(conf)
```

Solve the mounting point err

```
docker run -v /var/run/docker.sock:/var/run/docker.sock \  
-v ~{HOME}/container_dir:/output \  
--privileged -t --rm singularityware/docker2singularity -m \  
"container_dir_you_need_to_add" container_name
```

<https://www.docker.com/>

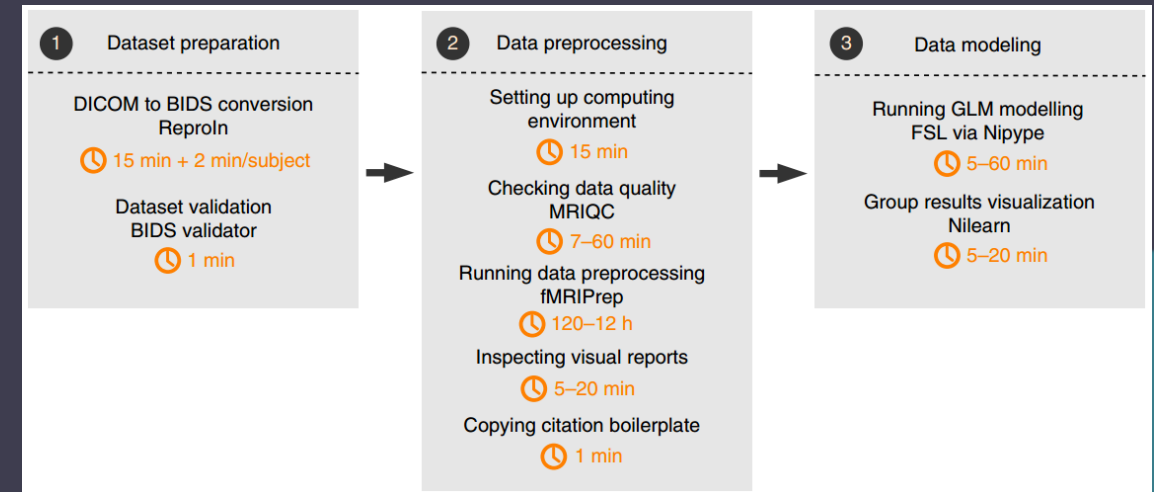
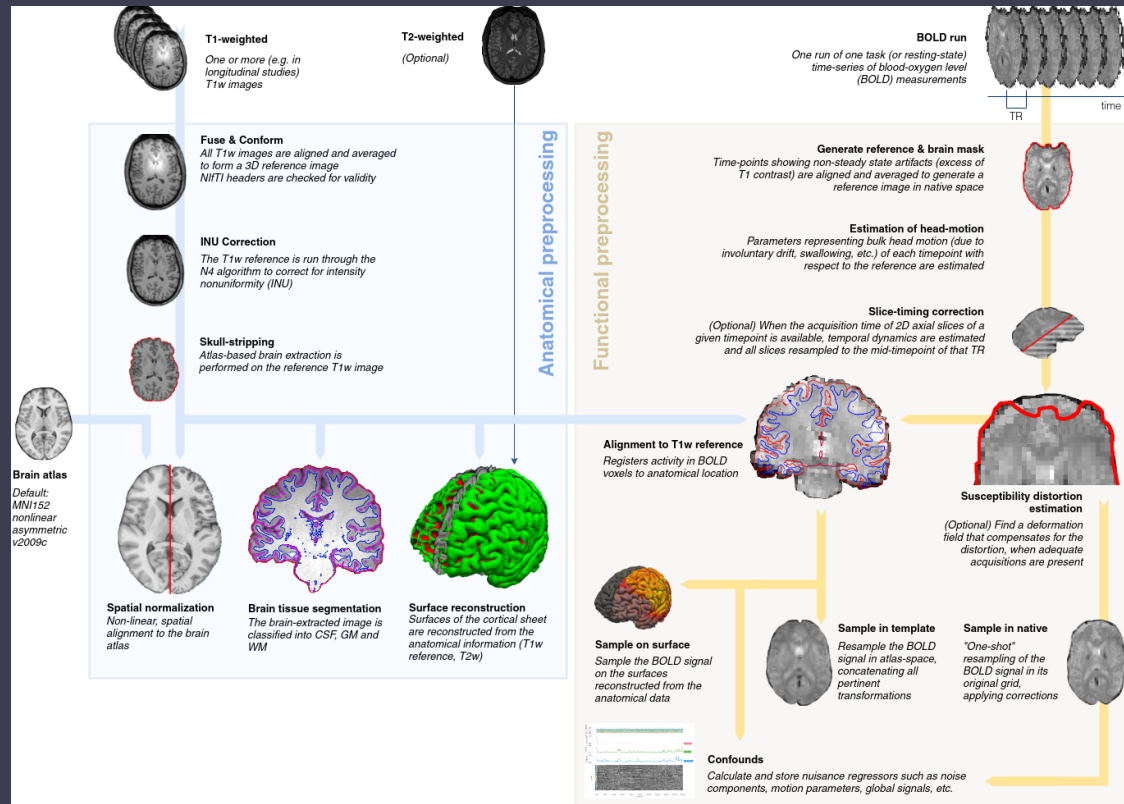
<https://sylabs.io/singularity/>

<https://hub.docker.com/>

<https://github.com/singularityhub/docker2singularity>

fMRIPrep: A BIDS App for robust anat & func MRI preprocessing

fMRIPrep is a containerized anat & func MRI preprocessing pipeline with the best tools for each step



- [FSL](#) (version 5.0.9)
- [ANTs](#) (version 2.2.0 - NeuroDocker build)
- [AFNI](#) (version Debian-16.2.07)
- [C3D](#) (version 1.0.0)
- [FreeSurfer](#) (version 6.0.1)
- [ICA-AROMA](#) (commit e8d7a58, post v0.4.4-beta)
- [bids-validator](#) (version 1.4.0)
- [connectome-workbench](#) (version Debian-1.3.2)

Warning!

Very narrow FoV images and very special study groups (infants or non-human brains) are challenging for any standard pipeline including fMRIPrep.

Esteban, et al. (2019) *Nature Methods*, 1(16): 111–16. <https://doi.org/10.1038/s41592-018-0235-4>.
Esteban, et al. (2020) *Nature Protocols*, 7(15): 2186–2202. <https://doi.org/10.1038/s41596-020-0327-3>.

<https://fmripred.org/en/stable/index.html>
<https://github.com/poldracklab/fmripred>

fMRIPrep: Basic inputs

Standard BIDS_App inputs:

```
input(bids_dir)
output(derivative_dir)
analysis_level(participant)
```

Filtering BIDS

```
--participant_label 001
--bids-filter-file ses-2.json
--skip_bids_validation
-t, --task-id rest
--anat-derivatives
```

```
ses-2.json
{"t1w":
  {"datatype": "anat",
   "session": "2",
   "suffix": "T1w"},
 "bold":
  {"datatype": "func",
   "session": "2",
   "suffix": "bold"} }
```

Specific options for sub-modules

```
--anat-only --output-spaces <SPACE>[:cohort-
<label>][:res-<resolution>][...]
--fs-license-file --fs-subjects-dir --cifti-
output
--force-bbr --use-aroma --return-all-components
--skull-strip-template --skull-strip-t1w --use-
syn-sdc
```

Workflow and env confs

```
-w, --work-dir
--resource-monitor
-v/-vvv increase log and debug level
--nprocs, --omp-nthreads, --mem, --low-mem
```

Preprocessing task	Included with fMRIPrep
Anatomical T1-weighted brain extraction	antsBrainExtraction.sh (ANTs)
Anatomical surface reconstruction	recon-all (FreeSurfer)
Head-motion estimation (and correction)	MCFLIRT (FSL)
Susceptibility-derived distortion estimation (and unwarping)	3dqwarp (AFNI)
Slice-timing correction	3dTshift (AFNI)
Intrasubject registration	bbregister (FreeSurfer), FLIRT (FSL)
Spatial normalization (intersubject co-registration)	antsRegistration (ANTs)
Surface sampling	mri_vol2surf (FreeSurfer)
Subspace projection denoising (e.g., independent or principal component analysis)	MELODIC (FSL), ICA-AROMA
Confounds	In-house implementation
Detection of non-steady states	In-house implementation

<https://fmriprep.org/en/stable/usage.html>

Esteban, et al. (2019) *Nature Methods*, 1(16): 111–16. <https://doi.org/10.1038/s41592-018-0235-4>.
Esteban, et al. (2020) *Nature Protocols*, 7(15): 2186–2202. <https://doi.org/10.1038/s41596-020-0327-3>.



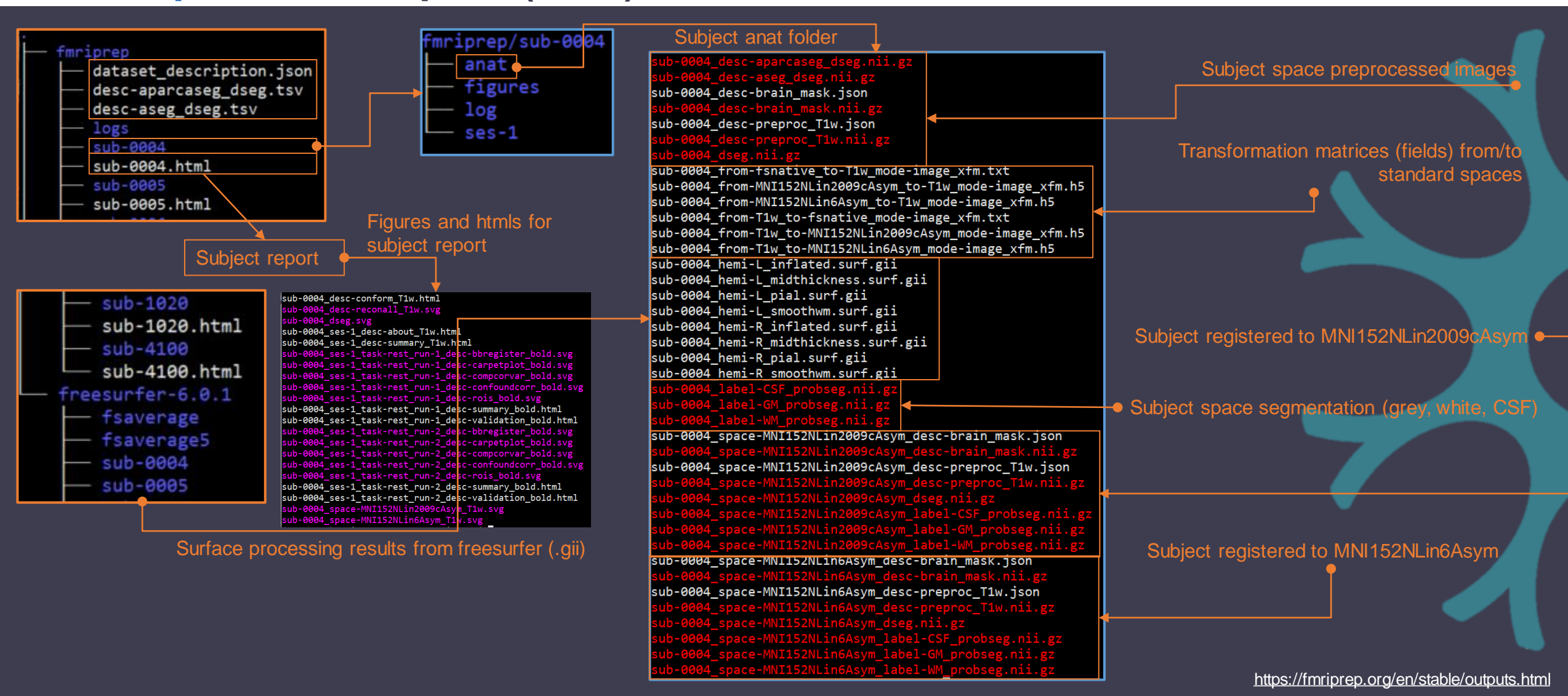
McGill



neuro

Institut-Hôpital
neurologique de Montréal

fMRIprep: Basic outputs (anat)



<https://fmriprep.org/en/stable/outputs.html>



McGill



neuro

Institut-Hôpital
neurologique de Montréal

fMRIprep: Basic outputs (func)

fmriprep/sub-0004

anat
figures
log
ses-1

```
sub-0004_ses-1_task-rest_run-1_desc-confounds_regressors.json
sub-0004_ses-1_task-rest_run-1_desc-confounds_regressors.tsv
sub-0004_ses-1_task-rest_run-1_space-fsaverage5_hemi-L_bold.func.gii
sub-0004_ses-1_task-rest_run-1_space-fsaverage5_hemi-R_bold.func.gii
sub-0004_ses-1_task-rest_run-1_space-fsLR_den-91k_bold.dtseries.json
sub-0004_ses-1_task-rest_run-1_space-fsLR_den-91k_bold.dtseries.nii
sub-0004_ses-1_task-rest_run-1_space-fsnative_hemi-L_bold.func.gii
sub-0004_ses-1_task-rest_run-1_space-fsnative_hemi-R_bold.func.gii
sub-0004_ses-1_task-rest_run-1_space-MNI152NLin2009cAsym_res-2_boldref.nii.gz
sub-0004_ses-1_task-rest_run-1_space-MNI152NLin2009cAsym_res-2_desc-aparcaseg_dseg.nii.gz
sub-0004_ses-1_task-rest_run-1_space-MNI152NLin2009cAsym_res-2_desc-aseg_dseg.nii.gz
sub-0004_ses-1_task-rest_run-1_space-MNI152NLin2009cAsym_res-2_desc-brain_mask.json
sub-0004_ses-1_task-rest_run-1_space-MNI152NLin2009cAsym_res-2_desc-brain_mask.nii.gz
sub-0004_ses-1_task-rest_run-1_space-MNI152NLin2009cAsym_res-2_desc-preproc_bold.json
sub-0004_ses-1_task-rest_run-1_space-MNI152NLin2009cAsym_res-2_desc-preproc_bold.nii.gz
sub-0004_ses-1_task-rest_run-1_space-T1w_boldref.nii.gz
sub-0004_ses-1_task-rest_run-1_space-T1w_desc-aparcaseg_dseg.nii.gz
sub-0004_ses-1_task-rest_run-1_space-T1w_desc-aseg_dseg.nii.gz
sub-0004_ses-1_task-rest_run-1_space-T1w_desc-brain_mask.json
sub-0004_ses-1_task-rest_run-1_space-T1w_desc-brain_mask.nii.gz
sub-0004_ses-1_task-rest_run-1_space-T1w_desc-preproc_bold.json
sub-0004_ses-1_task-rest_run-1_space-T1w_desc-preproc_bold.nii.gz
sub-0004_ses-1_task-rest_run-2_desc-confounds_regressors.json
sub-0004_ses-1_task-rest_run-2_desc-confounds_regressors.tsv
sub-0004_ses-1_task-rest_run-2_space-fsaverage5_hemi-L_bold.func.gii
sub-0004_ses-1_task-rest_run-2_space-fsaverage5_hemi-R_bold.func.gii
sub-0004_ses-1_task-rest_run-2_space-fsLR_den-91k_bold.dtseries.json
sub-0004_ses-1_task-rest_run-2_space-fsLR_den-91k_bold.dtseries.nii
sub-0004_ses-1_task-rest_run-2_space-fsnative_hemi-L_bold.func.gii
sub-0004_ses-1_task-rest_run-2_space-fsnative_hemi-R_bold.func.gii
```

Confounds for latter regression

Subject bold registered to fsaverage5 space

Subject bold registered to MNI152NLin2009cAsym

Subject space with segmentation (grey, white, CSF)

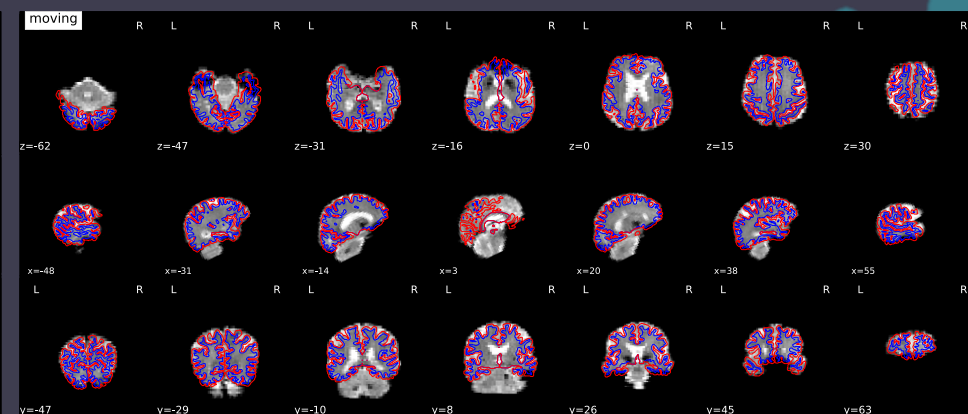
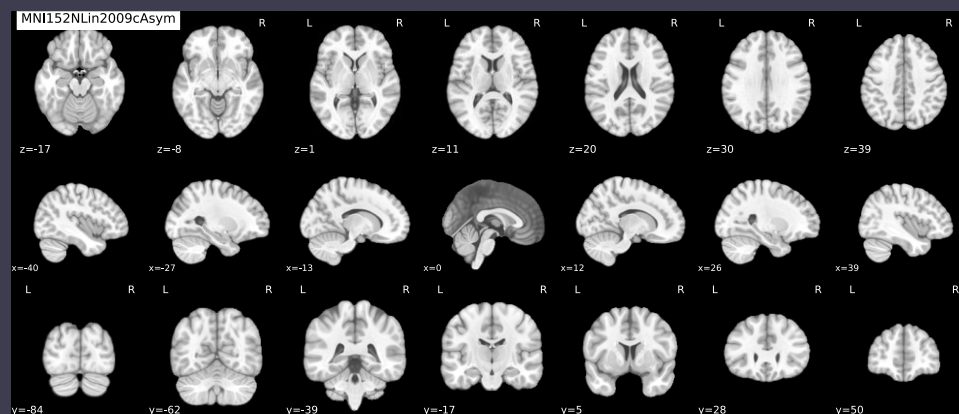
Bold preprocessing for another run

Confounds (or nuisance regressors) are variables representing fluctuations with a potential non-neuronal origin.

<https://fmriprep.org/en/stable/outputs.html>

fMRIprep: Basic outputs (report)

```
fmriprep
dataset_description.json
desc-aparcaseg_dseg.tsv
desc-aseg_dseg.tsv
logs
sub-0004
sub-0004.html
sub-0005
sub-0005.html
```



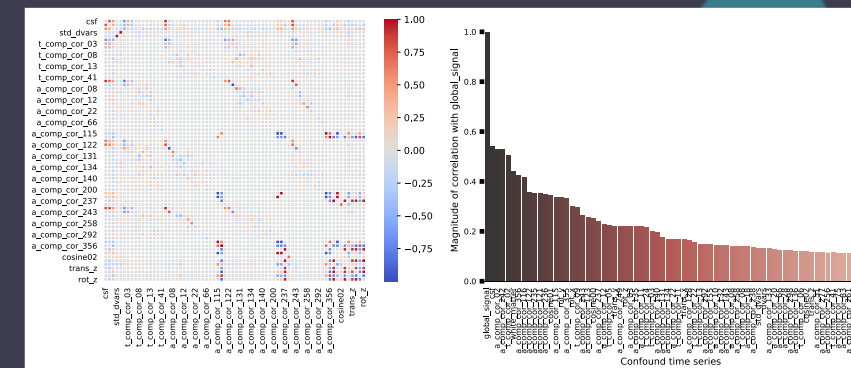
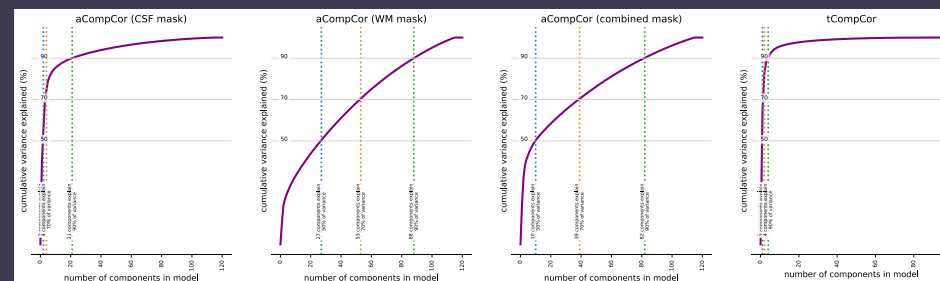
Summary

- Subject ID: 0004
- Structural images: 1 T1-weighted (+ 1 T2-weighted)
- Functional series: 2
 - Task: rest (2 runs)
- Standard output spaces: MNI152NLin2009cAsym, fsaverage, MNI152NLin6Asym
- Non-standard output spaces: anat, fsnative
- FreeSurfer reconstruction: Run by fMRIprep

Anatomical

Anatomical Conformation

- Input T1w images: 1
- Output orientation: RAS
- Output dimensions: 192x256x256
- Output voxel size: 1mm x 1mm x 1mm
- Discarded images: 0



<https://fmriprep.org/en/stable/outputs.html>

fMRIPrep: working with local container

Warning!

If you have only limited number of subjects, running fMRIPrep locally might be a good idea, or if you would like to investigate the container.

fmriprep.sh

Create out and temporal dir

```
#!/bin/bash
mkdir -p data_fmriprep_20.1.1
mkdir -p data_fmriprep_work
echo "" > data_fmriprep.log
echo "Start fmriprep QC..."
unset PYTHONPATH
singularity run -B $HOME:/home/fmriprep --home /home/mriqc --cleanenv \
-B ${HOME}/project/data_BIDS:/data:ro \
-B ${HOME}/project/data_fmriprep_20.1.1:/out \
-B ${HOME}/project/data_fmriprep_work:/work \
-B ${CODE_DIR}:/codes \
-B ${HOME}/project/templateflow:/templateflow \
-B ${LOCAL_FREESURFER_DIR}:/fsdir \
${HOME}/container_images/fmriprep_v20.1.1.simg /data /out \
participant --participant-label sub-0002 -w /work --output-spaces \
MNI152NLin2009cAsym:res-2 anat fsnative fsaverage5 --bids-filter-file \
/codes/ses-1.json --fs-subjects-dir /fsdir \
--fs-license-file /home/fmriprep/.freesurfer/license.txt \
--cifti-out 91k --return-all-components --write-graph \
--skip_bids_validation --notrack --resource-monitor
```

To be safe with container path

mriqc.sh

```
#!/bin/bash
mkdir -p data_mriqc_0.15.2
mkdir -p data_mriqc_work
echo "" > PD_mriqc.log
echo "Start group N participants QC..."
unset PYTHONPATH
singularity run -B $HOME:/home/mriqc --home /home/mriqc --cleanenv \
-B ${HOME}/project/data_BIDS:/data:ro \
-B ${HOME}/project/data_mriqc_0.15.2:/out \
-B ${HOME}/project/data_mriqc_work:/mriqc_work \
-B ${HOME}/project/templateflow:/templateflow \
${HOME}/container_images/mriqc_v0.15.2.simg /data /out participant \
--participant-label sub-0002 ... sub-1020 -w /mriqc_work \
--session-id 1 --ica --no-sub --verbose-repo --profile -vv
echo "Start group QC..."
singularity run -B $HOME:/home/mriqc --home /home/mriqc --cleanenv \
-B ${HOME}/project/data_BIDS:/data:ro \
-B ${HOME}/project/data_mriqc_0.15.2:/out \
-B ${HOME}/project/data_mriqc_work:/mriqc_work \
-B ${HOME}/project/templateflow:/templateflow \
${HOME}/container_images/mriqc_v0.15.2.simg /data /out group \
-w /mriqc_work --verbose-reports
```



McGill



neuro

Institut-Hôpital
neurologique de Montréal

fMRIPrep: HPC Basics (Compute Canada as an example)

Things you need to get ready before running codes (containers) on HPC

1. An account for HPC;
2. The documents for HPC:
3. Cluster management and job scheduling system docs (SLURM for Compute Canada), <https://slurm.schedmd.com/documentation.html>,
4. Get ssh client ready for remote terminal (ssh for linux/macOS, Bitvise SSH Client for Win...);
5. Get Globus ready for data transportation, <https://globus.computeCanada.ca/>, sftp tool is ok for small files;
6. Get your data, codes and containers ready uploaded on HPC.

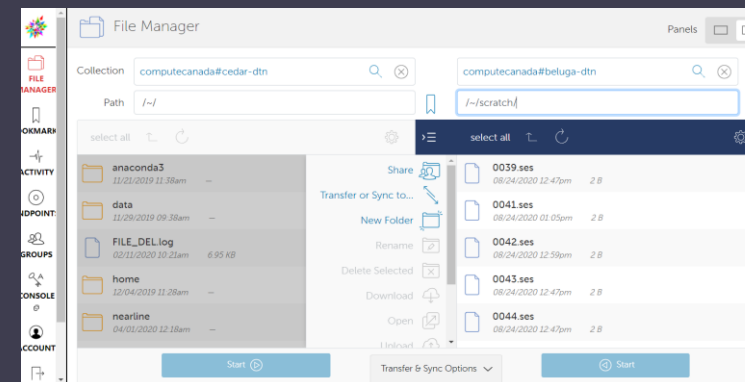
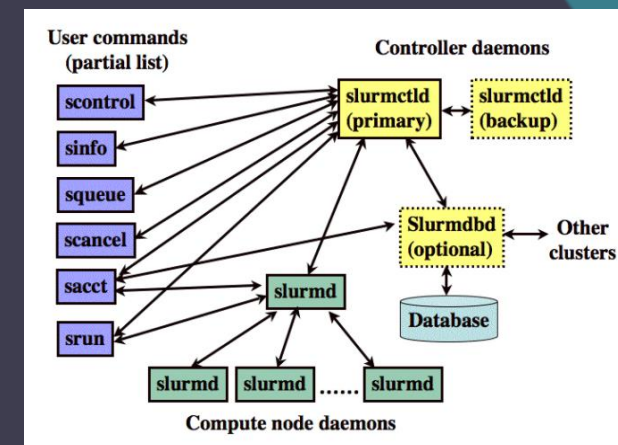
Tip1: Use **quota** to make sure you have enough space/N_files for your computation.

Tip2: **scratch** is the place where you should have all your computation.

Tip3: **sq/scancel** to check the job status and cancel job.

```
(neuroEnv) [vincentq@cedar5 scratch]$ quota
              Description      Space      # of files
/home (user [redacted])      44G/50G      156k/500k
/scratch (user vincentq)    492G/20T     749k/1000k
/project (group vincentq)    0/2048k      0/5000k
/project (group [redacted])  328T/400T    13M/33M
/project (group [redacted])  206G/1000G   19k/500k
/project (group [redacted])  250G/1000G   46k/500k
/nearline (project [redacted]) 296k/300T    74/5000
/nearline (project [redacted]) 52k/300T     13/5000
/nearline (project [redacted]) 264k/300T    66/5000
```

Warning!
Always use Globus for data transportation, especially inter-cluster data transportation, it is faster and safer.



[Cheat sheet](#)

fMRIPrep: running on HPC (Compute Canada), preparation

fmriprep.sh

```
#!/bin/bash
DATA_NAME=${@:1:1})
echo ${DATA_NAME}

WD_DIR=${HOME}/scratch
DATA_DIR=${WD_DIR}/${DATA_NAME}
BIDS_DIR=${DATA_DIR}_BIDS
CODE_DIR=${WD_DIR}/src
CODE_SLURM=${CODE_DIR}/fmriprep.slurm
CODE_COLLECT=${CODE_DIR}/fmriprep.format

SUB_LIST=${CODE_DIR}/${DATA_NAME}_fmriprep_preopt.list
CON_IMG_DIR=${WD_DIR}/container_images/fmriprep_v20.1.1.simg

OUT_DIR=${DATA_DIR}_fmriprep_anat_20.1.1
LOG_DIR=${DATA_DIR}_fmriprep_anat.log
SLURM_LOG_DIR=${DATA_DIR}_fmriprep_anat_slurm_log
WORK_DIR=${DATA_DIR}_fmriprep_anat_work

FREESURFER_LICENSE="${WD_DIR}/container_images"
TEMPLATEFLOW_HOST_HOME=$HOME/scratch/templateflow

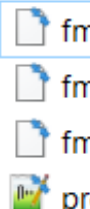
RUN_ID=$(tail -c 9 ${LOG_DIR})
if [ -z $RUN_ID ];then
    echo 'no previous run found...'
else
    echo "previous run $RUN_ID found, deleting logs..."
    rm fmriprep_vince-${RUN_ID}*.out
    rm fmriprep_vince-${RUN_ID}*.err
fi

rm -rf ${OUT_DIR}
rm -rf res/${OUT_DIR}.tar.gz
rm -rf res/${OUT_DIR}_freesurfer.tar.gz
rm -rf ${WORK_DIR}
rm -rf ${SLURM_LOG_DIR}
rm -rf res/${SLURM_LOG_DIR}.tar.gz
rm -rf ${SUB_LIST}
```

```
chmod +x ${CODE_SLURM}
chmod +x ${CODE_COLLECT}
awk -F"\t" '{print $1}' ${BIDS_DIR}/participants.tsv >> ${SUB_LIST}
sed -i '1d' ${SUB_LIST}
echo "Step1: subjects list created!"
mkdir -p ${OUT_DIR}
mkdir -p ${WORK_DIR}
mkdir -p ${SLURM_LOG_DIR}

if [ -d ${TEMPLATEFLOW_HOST_HOME} ];then
    echo "Templateflow dir already exists!"
Else
    mkdir -p ${TEMPLATEFLOW_HOST_HOME}
    python -c "from templateflow import api; api.get('MNI152NLin2009cAsym')"
    python -c "from templateflow import api; api.get('OASIS30ANTS')"
fi
echo "Step2: starting fmriprep!"
sbatch ${CODE_SLURM} ${DATA_NAME} ${CON_IMG_DIR} >> ${LOG_DIR}
```

Submit to cluster



fmriprep_anat.format
fmriprep_anat.sh
fmriprep_anat.slurm
pre_opt.json

fmriprep.format

ses-2.json

```
{
  "t1w": {
    "datatype": "anat",
    "session": "2",
    "suffix": "T1w"},
  "bold": {
    "datatype": "func",
    "session": "2",
    "suffix": "bold"}
}
```

```
#!/bin/bash
DATA_NAME=${@:1:1})
echo ${DATA_NAME}
WD_DIR=${HOME}/scratch
DATA_DIR=${WD_DIR}/${DATA_NAME}
BIDS_DIR=${DATA_DIR}_BIDS
FMRIPREP_VER=20.1.1
DERIVS_DIR_NAME=${DATA_NAME}_fmriprep_anat_${FMRIPREP_VER}
LOG_DIR_NAME=${DATA_NAME}_fmriprep_anat_slurm_log
RUN_ID=$(tail -c 9 ${DATA_NAME}_fmriprep_anat.log)
mv fmriprep_vince-${RUN_ID}*.out ${LOG_DIR_NAME}
mv fmriprep_vince-${RUN_ID}*.err ${LOG_DIR_NAME}
#2 collect output
tar -czvf res/${DERIVS_DIR_NAME}.tar.gz ${DERIVS_DIR_NAME}/fmriprep
tar -czvf res/${DERIVS_DIR_NAME}_freesurfer.tar.gz
${DERIVS_DIR_NAME}/freesurfer-6.0.1
tar -czvf res/${LOG_DIR_NAME}.tar.gz ${LOG_DIR_NAME}
echo "Step5: fmriprep for ${DATA_NAME} Finished"
```



McGill



neuro

Institut-Hôpital
neurologique de Montréal

fMRIPrep: running on HPC (Compute Canada), the core script

fmriprep.slurm

```
#!/bin/bash
#SBATCH -J fmriprep_vince
#SBATCH --time=24:00:00
#SBATCH --account=account
#SBATCH --cpus-per-task=8
#SBATCH --mem-per-cpu=8G
#SBATCH --array=1-37
# Outputs -----
#SBATCH -o %x-%A-%a_%j.out
#SBATCH -e %x-%A-%a_%j.err
#SBATCH --mail-user=youremail@gmail.com
#SBATCH --mail-type=ALL
# -----
```

Cluster setup

```
DATA_NAME=${@:1:1}
CON_IMG_DIR=${@:2:1}
WD_DIR=${HOME}/scratch
DATA_DIR=${WD_DIR}/${DATA_NAME}
BIDS_DIR=${DATA_DIR}_BIDS
DERIVS_DIR=${DATA_DIR}_fmriprep_anat_20.1.1
CODE_DIR=${WD_DIR}/src
LOG_DIR=${DATA_DIR}_fmriprep_anat_slurm_log
WORK_DIR=${DATA_DIR}_fmriprep_anat_work
SUB_LIST=${CODE_DIR}/${DATA_NAME}_fmriprep_preopt.list
```

Basic working dir

```
# subject ID and the corresponds BIDS filter
SUB_STR=$(sed -n "${SLURM_ARRAY_TASK_ID}p" ${SUB_LIST})
SUB_ID=$(cut -d'-' -f2 <<${SUB_STR})
FM RIPREP_HOME=fmriprep_home_${SUB_ID}
echo "Processing: sub-${SUB_ID} with home dir:
${FM RIPREP_HOME}"
mkdir -p ${FM RIPREP_HOME}
```

home dir for each sub

```
LOCAL_FREESURFER_DIR=${DERIVS_DIR}/freesurfer-6.0.1
mkdir -p ${LOCAL_FREESURFER_DIR}
# Prepare some writeable bind-mount points.
TEMPLATEFLOW_HOST_HOME=${HOME}/scratch/templateflow
FM RIPREP_HOST_CACHE=${FM RIPREP_HOME}/.cache/fmriprep
mkdir -p ${TEMPLATEFLOW_HOST_HOME}
mkdir -p ${FM RIPREP_HOST_CACHE}
```

Freesurfer/templateflow

```
# Make sure FS_LICENSE is defined in the container.
mkdir -p ${FM RIPREP_HOME}/.freesurfer
export SINGULARITYENV_FS_LICENSE=${FM RIPREP_HOME}/.freesurfer/license.txt
cp container_images/license.txt ${SINGULARITYENV_FS_LICENSE}
# Designate a templateflow bind-mount point
export SINGULARITYENV_TEMPLATEFLOW_HOME="/templateflow"
SINGULARITY_CMD="singularity run -B ${FM RIPREP_HOME}:/home/fmriprep -home \
/home/fmriprep --cleanenv \
-B ${BIDS_DIR}:/data:ro \
-B ${DERIVS_DIR}:/output \
-B ${CODE_DIR}:/codes \
-B ${TEMPLATEFLOW_HOST_HOME}:${SINGULARITYENV_TEMPLATEFLOW_HOME} \
-B ${WORK_DIR}:/work \
-B ${LOCAL_FREESURFER_DIR}:/fsdir ${CON_IMG_DIR}"
```

Freesurfer license

Basic singularity setup/mount

```
# Remove IsRunning files from FreeSurfer
find ${LOCAL_FREESURFER_DIR}/sub-${SUB_ID}/ -name "*IsRunning*" -type f -delete
```

```
# Compose the command line
cmd="${SINGULARITY_CMD} /data/output participant --participant-label $SUB_ID \
-w /work --output-spaces MNI152NLin2009cAsym:res-2 anat fsnative fsaverage5 \
--bids-filter-file /codes/pre_opt.json --fs-subjects-dir /fsdir \
--fs-license-file /home/fmriprep/.freesurfer/license.txt \
--cifti-out 91k --return-all-components \
--write-graph --skip_bids_validation --notrack --resource-monitor"
# --bids-filter-file ${BIDS_FILTER} --anat-only\
# Setup done, run the command
echo Running task ${SLURM_ARRAY_TASK_ID}
echo Commandline: $cmd
unset PYTHONPATH
eval $cmd
exitcode=$?
```

fMRIPrep setup

Run container

```
# Output results to a table
echo "sub-${SUB_ID}    ${SLURM_ARRAY_TASK_ID}    $exitcode" \
>> ${LOG_DIR}/${SLURM_JOB_NAME}_${SLURM_ARRAY_JOB_ID}.tsv
echo Finished tasks ${SLURM_ARRAY_TASK_ID} with exit code $exitcode
rm -rf ${FM RIPREP_HOME}
exit $exitcode
```

sv log and clear env



McGill



neuro

Institut-Hôpital
neurologique de Montréal

fMRIPrep: running on HPC (Compute Canada), debug

Tip1: Check the `%x-%A-%a_%j.out` and `%x-%A-%a_%j.err` to locate the err subject.

Tip2: Go to the `DERIVS_DIR/fmriprep/sub-xxxx/log/xxx/fmriprep.toml`, and check the log.

Tip3: Go to the `fmriprep_work(-w)/fmriprep_wf/single_subject_xxxx_wf` and check the intermediate results with `nipypecli crash *.pklz` (use the same container and `singularity shell`).

Tip4: Do a google search and [NeuroStars](#) (QA) search.

Tip5: If the problem is still there, just creating a new topic with details on [NeuroStars](#) or GitHub (@the developer☺).

Tip6: Do not forget our #tech-support slack channel ☺

fmriprep.toml

```
[environment] cpu_count = 32
exec_env = "singularity"...
[execution] bids_dir = "/data"...
[workflow] anat_only = false...
[nipype] crashfile_format = "txt"
[seeds] master = 38259...
[execution.bids_filters.t1w]
datatype = "anat"session = "1"
suffix = "T1w"
[execution.bids_filters.bold]
datatype = "func"session = "1"
suffix = "bold"
```

```
about
├── _0xe00825ec954f000118ce7296b101b1f.json
├── _inputs.pklz
├── _node.pklz
├── _report
├── report.html
└── result_about.pklz
anat_preproc_wf
├── anat_derivatives_wf
├── anat_norm_wf
├── anat_reports_wf
├── anat_template_wf
├── anat_validate
├── applyrefined
├── brain_extraction_wf
├── lut_t1w_dseg
├── split_seg
└── surface_recon_wf
bids_info
├── _0xb0bf71f52b03348f41e67cbbf949c0d6a.json
├── _inputs.pklz
├── _node.pklz
├── _report
└── result_bids_info.pklz
bidssrc
├── _0xf83ac98772fabd74cc5ef03c346ecb62.json
├── _inputs.pklz
├── _node.pklz
├── _report
└── result_bidssrc.pklz
ds_report_about
├── _0xab0de7181360e73d41dda7c2614bc0e1.json
├── _inputs.pklz
├── _node.pklz
├── _report
└── result_ds_report_about.pklz
ds_report_summary
├── _0xa28236e158d9b1a781e7869a00d87e3a.json
├── _inputs.pklz
├── _node.pklz
├── _report
└── result_ds_report_summary.pklz
summary
├── _0xa4a7cc12363b7769849f036835460b64.json
├── _inputs.pklz
├── _node.pklz
├── _report
├── report.html
└── result_summary.pklz
```

23 directories, 26 files

fMRIPrep: tips for fMRIPrep usage

Tip1: Pre-download the template with [templateflow](#).

```
python -c "from templateflow import api; api.get('MNI152NLin2009cAsym')"
```

Tip2: Running quality control ([MRIQC](#)) before running fMRIPrep.

Tip3: Skip part of the preprocessing if needed.

```
--skip_bids_validation --force-no-bbr --skull-strip-t1w skip --fs-no-reconall
```

Tip4: Resources allocation: use batch conf instead of fmriprep conf when running on HPC.
[recommendation for resources](#)

Tip5: Be careful when you have dramatic anatomical changes between sessions from subjects.

fMRIPrep always assumes that the anatomy won't change between sessions and will estimate a robust anatomical template for a subject from different sessions. This assumption does not hold for pre-and-post operation sessions.

More resources

- OGs
 - [fMRIPrep home page](#)
 - [fMRIPrep GitHub](#)
 - [BIDS](#), [BIDS App GitHub](#), [BIDS Derivatives](#)
 - [NeuroStars](#) (QA)
 - [Compute Canada document](#);
- Tutorials
 - [Stanford tutorial](#)
 - [Andrew \(Jahn\)'s Brain Book](#) (with [YouTube video](#))
 - [Using fMRIPrep for fMRI data preprocessing](#) by Gelana Tostaeva
 - [DartBrains course chapter](#) by Sasha Brietzke & Luke Chang
 - [Tutorial: BIDS, fMRIPrep, MRIQC](#) by Saren Seeley
 - [Brainlife tutorials](#)
 - [Neuroimaging Core Documentation](#) by Dianne Patterson



McGill



neuro

Institut-Hôpital
neurologique de Montréal

Thanks



McGill



neuro

Institut-Hôpital
neurologique de Montréal