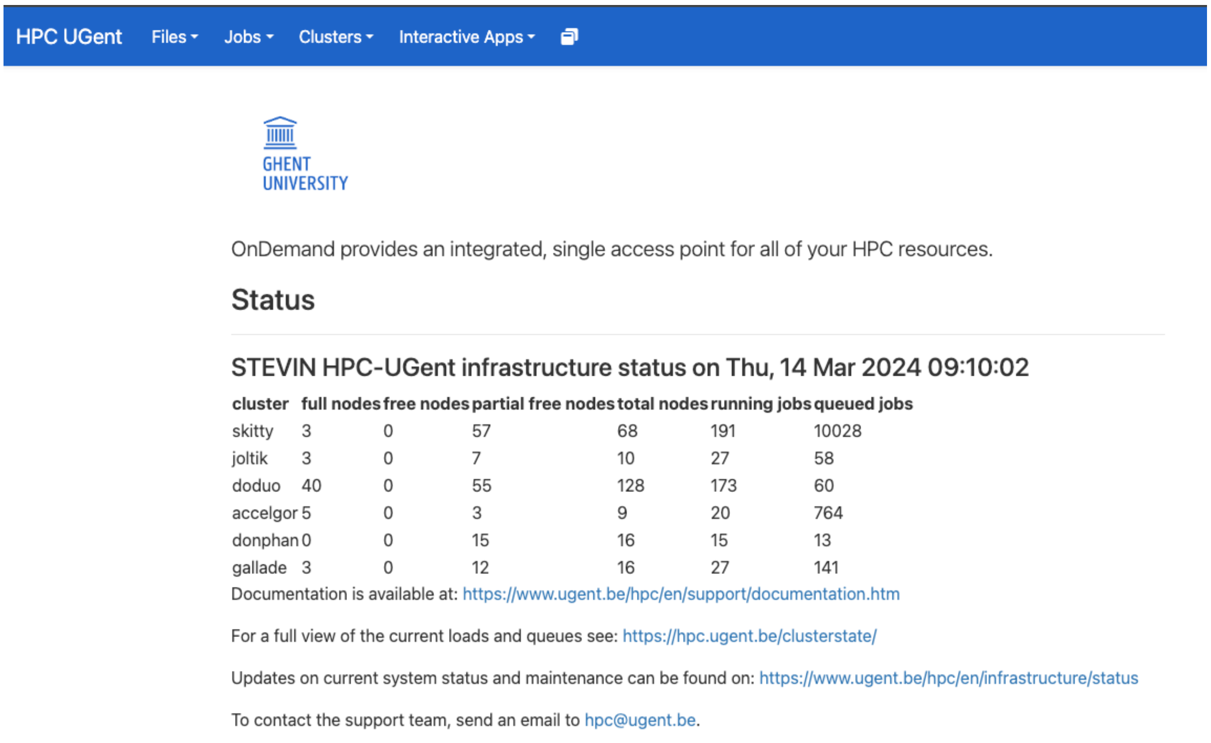
# HOW TO RUN fMRIprep on the HPC – Ugent

1. 1. If you have never used the HPC before, you need to request an account. You can find the procedure and detailed information here: https://docs.hpc.ugent.be/macOS/account/
   *If you have access to a VO (== virtual organization), that will give you much more storage space, so ask for it!
2. Access your HPC space → https://login.hpc.ugent.be/ and after the login you should arrive to this page



OnDemand provides an integrated, single access point for all of your HPC resources.

Status

STEVIN HPC-UGent infrastructure status on Thu, 14 Mar 2024 09:10:02

| cluster | full nodes | free nodes | partial free nodes | total nodes | running jobs | queued jobs |
|---|---|---|---|---|---|---|
| skitty | 3 | 0 | 57 | 68 | 191 | 10028 |
| joltik | 3 | 0 | 7 | 10 | 27 | 58 |
| doduo | 40 | 0 | 55 | 128 | 173 | 60 |
| accelgor | 5 | 0 | 3 | 9 | 20 | 764 |
| donphan | 0 | 0 | 15 | 16 | 15 | 13 |
| gallade | 3 | 0 | 12 | 16 | 27 | 141 |

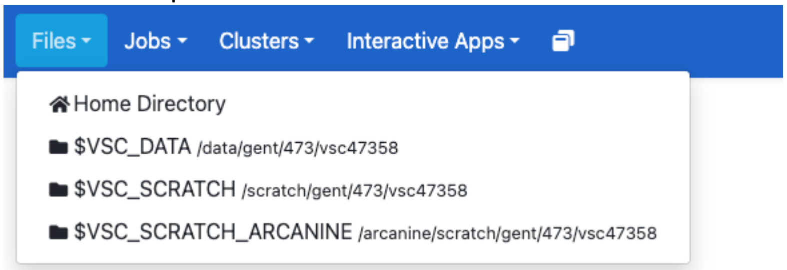
Documentation is available at: https://www.ugent.be/hpc/en/support/documentation.htm

For a full view of the current loads and queues see: https://hpc.ugent.be/clusterstate/

Updates on current system status and maintenance can be found on: https://www.ugent.be/hpc/en/infrastructure/status

To contact the support team, send an email to hpc@ugent.be.

If you click on "Files" you will see all the possible folders and space where you will put your data and scripts.


Files
Jobs
Clusters
Interactive Apps
Home Directory
$VSC_DATA /data/gent/473/vsc47358
$VSC_SCRATCH /scratch/gent/473/vsc47358
$VSC_SCRATCH_ARCANINE /arcanine/scratch/gent/473/vsc47358

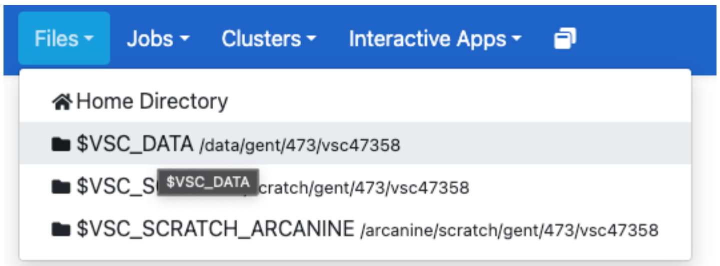
In my case, I did not join a VO yet but if you did you should have also folders with VO at the end: e.g. $VSC_DATA_VO. For now I will not use those, but if you have them I suggest you to use them (bigger storage).
As a general info:

- the Home directory and the $DATA are long-term storage slow filesystem → in the home directory you can store some scripts and configuration files, in the DATA folder (bigger space) you can store your data/results.

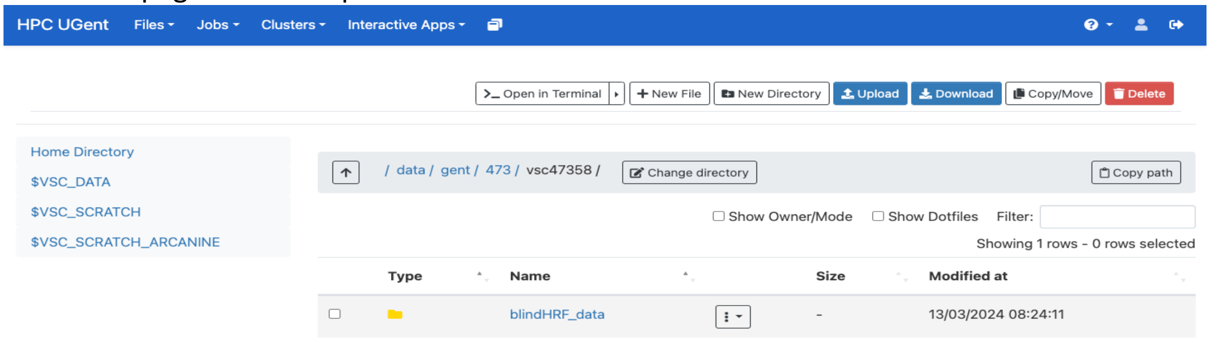
- $VSC_SCRATCH is a fast temporary storage, e.g. for transient data like the temporary files created by fmriprep during preprocessing. This is where we'll run our preprocessing analysis.

Here you can find a full description of all possible folders in the HPC: https://docs.hpc.ugent.be/macOS/running_jobs_with_input_output_data/?h=%24vsc+data

3. **Upload your data.** You need them to be in BIDS format in order to run fmriprep. We will upload them on the folder $VSC_DATA (or better $VSC_DATA_VO if you have it). Click on $VSC_DATA

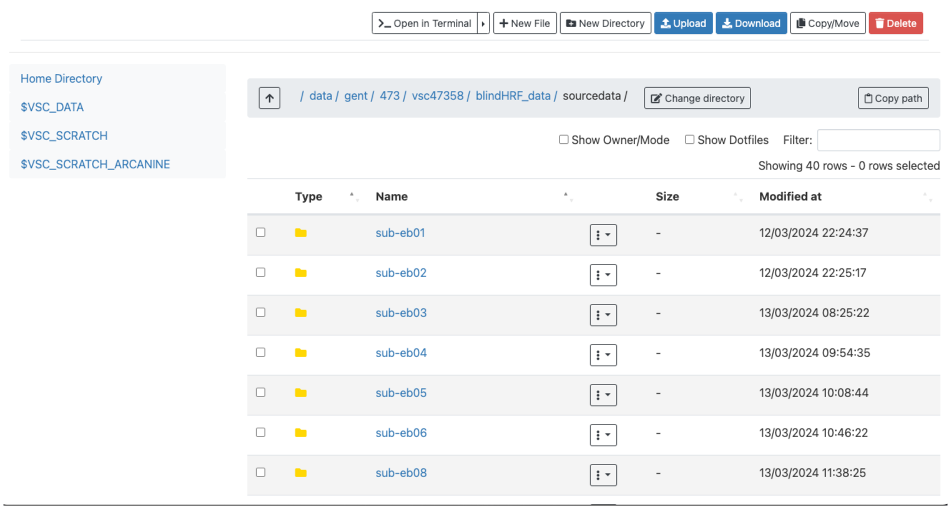

Files
Jobs
Clusters
Interactive Apps
Home Directory
$VSC_DATA /data/gent/473/vsc47358
$VSC_S $VSC_DATA cratch/gent/473/vsc47358
$VSC_SCRATCH_ARCANINE /arcanine/scratch/gent/473/vsc47358

This is the page that will open:



| Type | Name | Size | Modified at |
|---|---|---|---|
| 🟨 | blindHRF_data | - | 13/03/2024 08:24:11 |

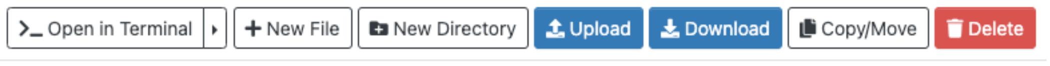
I already have my data uploaded in the folder blindHRF_data/sourcedata/*[all my subjects' folders here].*



| Type | Name | Size | Modified at |
|---|---|---|---|
| 🟨 | sub-eb01 | - | 12/03/2024 22:24:37 |
| 🟨 | sub-eb02 | - | 12/03/2024 22:25:17 |
| 🟨 | sub-eb03 | - | 13/03/2024 08:25:22 |
| 🟨 | sub-eb04 | - | 13/03/2024 09:54:35 |
| 🟨 | sub-eb05 | - | 13/03/2024 10:08:44 |
| 🟨 | sub-eb06 | - | 13/03/2024 10:46:22 |
| 🟨 | sub-eb08 | - | 13/03/2024 11:38:25 |

You have 2 options for uploading the data:

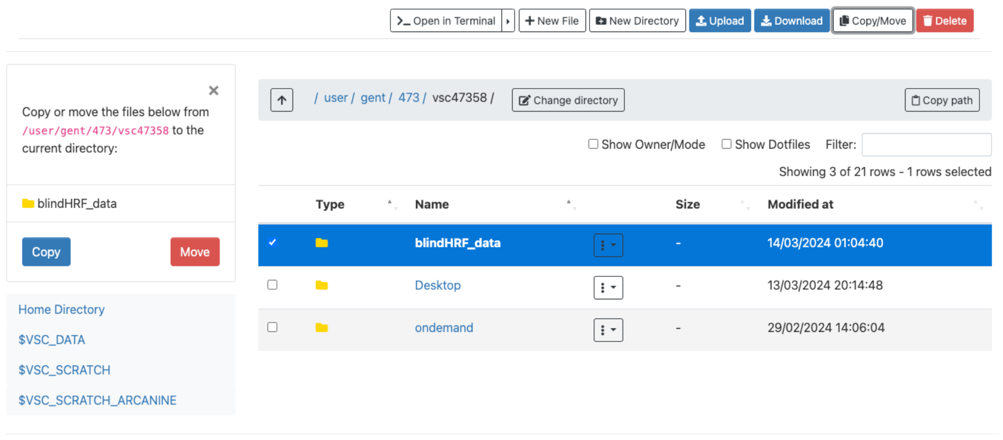
a. Use the Upload button on top of the page



This will be slower and sometimes less smooth due to internet connection

b. Use the terminal to upload the data. To use this option you need to set the SSH key access when setting your account (if you didn't do that, you can go back and add it now!)

- To use the terminal, open it in your local pc and type this line**: rsync -rzv [path_to_data_on_your_pc_goes_here] [home_directory_goes_here]**
- In my specific case this looks like: **rsync -rzv Documents/data/BlindHRF/sourcedata vsc47358@login.hpc.ugent.be:**
- You will be ask to enter your pass_phrase and then the copy procedure will start.
- Once it ends, you can open your Home Directory on HPC and you will find your data folder. In my case the 'blindHRF_data' folder is there.
- Now I just need to move it to the $VSC_DATA: click on the folder and then click on the "Copy/Move" button on the top menu. A window on the left side (like the one below) will appear.



Copy or move the files below from `/user/gent/473/vsc47358` to the current directory:

blindHRF_data

/ user / gent / 473 / vsc47358 /

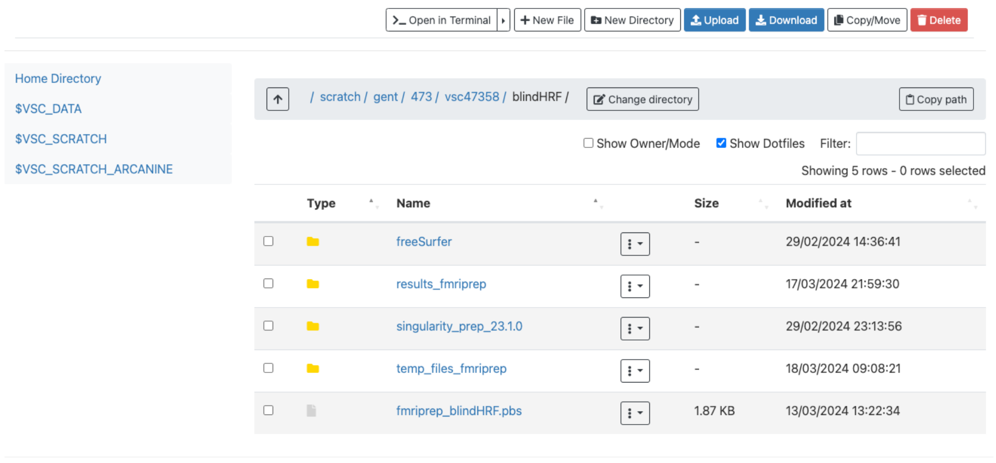
Showing 3 of 21 rows - 1 rows selected

| | Type | Name | | Size | Modified at |
|---|---|---|---|---|---|
| ✓ | 📁 | blindHRF_data | ⋮ | - | 14/03/2024 01:04:40 |
| ☐ | 📁 | Desktop | ⋮ | - | 13/03/2024 20:14:48 |
| ☐ | 📁 | ondemand | ⋮ | - | 29/02/2024 14:06:04 |

Now move to the folder were you want to copy your data: $VSC_DATA. Once you are there, you can click on the red button "Move" in the left window and the data will be moved here.

4. Now let's prepare the rest of the environment in the folder $VSC_SCRATCH.

Here I created a folder using the top button 'New Directory' and I named it 'blindHRF'. Inside this folder I have 4 folder and 1 .pbs script (see screenshot below)



/ scratch / gent / 473 / vsc47358 / blindHRF /

Showing 5 rows - 0 rows selected

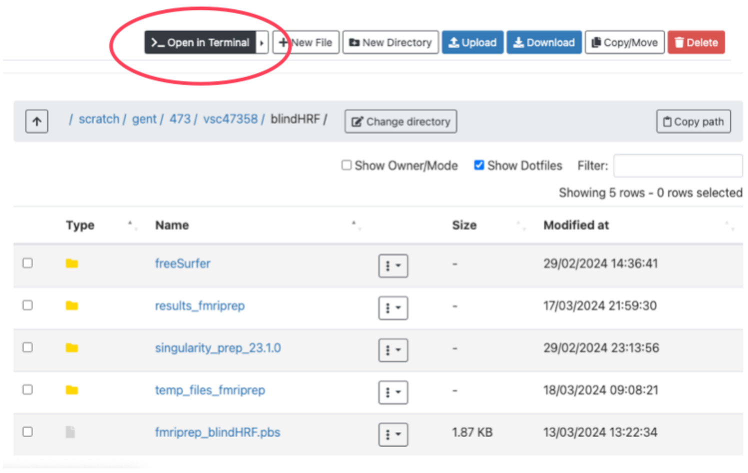
| Type | Name | Size | Modified at |
|---|---|---|---|
| 📁 | freeSurfer | - | 29/02/2024 14:36:41 |
| 📁 | results_fmriprep | - | 17/03/2024 21:59:30 |
| 📁 | singularity_prep_23.1.0 | - | 29/02/2024 23:13:56 |
| 📁 | temp_files_fmriprep | - | 18/03/2024 09:08:21 |
| 🗎 | fmriprep_blindHRF.pbs | 1.87 KB | 13/03/2024 13:22:34 |

Let's see one by one all the folder and the files inside them.

a. You need a **freesurfer license** to run some steps of fmriprep pipeline. You can download the license here (free): https://surfer.nmr.mgh.harvard.edu/registration.html. Once you get the license.txt file, put it in the folder 'freeSurfer'

b. There are different ways of using **fmriprep** (i.e. on the free cloud service OpenNeuro.org, in a Docker Container, in a Singularity Container, or in a Manually Prepared Environment). You can find a description of each of them here. For security reasons, many HPCs do not allow Docker containers, but do allow Singularity containers. The version of Singularity on ugent HPC is modern enough to create **Singularity image** directly on the HCP. Here how:

   - Click the button: >_Open in Terminal



/ scratch / gent / 473 / vsc47358 / blindHRF /

Showing 5 rows - 0 rows selected

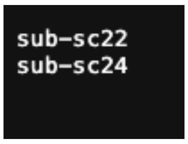
| Type | Name | Size | Modified at |
|---|---|---|---|
| 📁 | freeSurfer | - | 29/02/2024 14:36:41 |
| 📁 | results_fmriprep | - | 17/03/2024 21:59:30 |
| 📁 | singularity_prep_23.1.0 | - | 29/02/2024 23:13:56 |
| 📁 | temp_files_fmriprep | - | 18/03/2024 09:08:21 |
| 🗎 | fmriprep_blindHRF.pbs | 1.87 KB | 13/03/2024 13:22:34 |

   - Once the terminal windows opens type the following line there:

```
$ singularity build /my_images/fmriprep-<version>.simg
docker://poldracklab/fmriprep:<version>
```

     Where <version> should be replaced with the desired version of fMRIPrep that you want to download (IMPORTANT: this method only works for versions >= 2.5). For previous version see the link above.

        - In my case this looked like:
          `singularity build /my_images/fmriprep-23.1.0.simg docker://nipreps/fmriprep:23.1.0`
        - Once you get the file 'fmriprep-23.2.0.simg', put it in the folder: 'singularity_prep_23.1.0'.
    c. Create a folder named 'temp_files_fmriprep'. This is where all the (many) **temporary files** will be stored while running fmriprep. You will need to empty this folder quite often (at the end of couple of jobs) to get free space for next subjects.
    d. Create a folder 'results_fmriprep/preproc' where the final **output files** will be stored. Inside the results_fmriprep I also store a .txt file named 'subjects_to_run_blindHRF', which will contain the **sub labels** for the subjects that you want to run, for instance:



```
sub-sc22
sub-sc24
```

    e. Finally, you will need the 'fmriprep_blindHRF.pbs' script to run your analysis. At the end of this document you find a line by line description of this script.

5. Now, everything is ready and you can run fmri prep on your 2 first subjects.
    - Go in the blindHRF folder;
    - Click on '>_Open in Terminal
    - Type: `$ qsub -t 1-2 fmriprep_blindHRF.pbs`
      (Here I always keep -t 1-2, but I change each time the subject labels in the .txt file)
      → if you get an error you can read it in a file created in the blindHRF folder. You will receive an email when the job starts, fails and/or is completed.

6. After the job is completed:
    - empty the temporary files folder
    - **How to download (copy from HPC to local machine)**. Move your data in a folder named 'preproc' in the home directory. Then go in the terminal in your local machine and type:
      rsync -rzv vsc47358@login.hpc.ugent.be:preproc
      Documents/data/BlindHRF/derivatives/fmriprep_preproc/

      This command in the terminal will copy the specified folder in the HPC to the specified folder in the local pc (you need to change the highlighted path with an existing path in your computer where you want to store your data).

7. Repeat steps 5 and 6 for all your subjects!

**Line by line description of the .pbs script**

In the PBS script, the lines beginning with “#PBS” are PBS directives that specify the resource requirements and various other attributes of the job. Note that the directives must come first in the script as any directives after the first executable statement are ignored.

Let's break down each line of the PBS (Portable Batch System) script:

**First section:**
(Overall, this PBS script sets up parameters for a job named "FMRIPREP" that will run on a single node with 8 CPU cores, require 30 gigabytes of memory, have a maximum runtime of 24 hours, and send email notifications for job status updates).

1. `#!/bin/bash`: This line is called a shebang. It indicates that the script should be run using the Bash shell interpreter (`/bin/bash`). It's necessary for the system to understand which interpreter to use when executing the script.

2. `#PBS -N FMRIPREP`: This line is a PBS directive (`#PBS`). It sets the name of the job to "FMRIPREP". When the job runs, this name will be used to identify it in the job queue and in job-related notifications.

3. `#PBS -l nodes=1:ppn=8`: This line is another PBS directive. It specifies the resources needed for the job. Here, it requests one node (`nodes=1`) with 8 processors per node (`ppn=8`). This means the job will run on a single node with 8 CPU cores.

4. `#PBS -l mem=30gb`: This line is another PBS directive that specifies the amount of memory required for the job. It requests 30 gigabytes of memory (`mem=30gb`) for the job.

5. `#PBS -l walltime=24:00:00`: This line is another PBS directive that sets the maximum wall clock time for the job. It specifies a duration of 24 hours (`24:00:00`), meaning the job should complete within this time limit.

6. `#PBS -m abe`: This line is another PBS directive that specifies email notification preferences. It tells PBS to send email notifications about the job's status using the following codes:
   - `a`: Send email when the job is aborted (i.e., terminated prematurely).
   - `b`: Send email when the job begins execution.
   - `e`: Send email when the job ends.

**Second section:** Overall, this section of the script sets up the environment to process participant data in parallel by extracting participant names from a text file and storing them in an environment variable.

1. `#== READ IN PARTICIPANT NAMES==#`: This is a comment indicating that this section of the script deals with reading participant names.

2. `# This extracts the participant names from the .txt file "subjects_to_run_syn.txt"`: This is another comment providing additional context. It explains that the purpose of this section is to extract participant names from a text file named "subjects_to_run_syn.txt".

3. `# to allow participants to processed in parallel. This text file must contain`: This comment further explains the purpose of extracting participant names – to enable parallel processing of participants. It also mentions that the text file must contain the exact names of participant subdirectories in the BIDS directory.

4. `dos2unix $VSC_SCRATCH/signVWFA/results_fmriprep/subjects_to_run_syn.txt`: This command is used to convert the line endings in the text file "subjects_to_run_syn.txt" to Unix format. It ensures compatibility with Unix-based systems by standardizing the line endings.

5. `export SUBJ=`sed -n "${PBS_ARRAYID}p" $VSC_SCRATCH/signVWFA/results_fmriprep/subjects_to_run_syn.txt``: This line reads the contents of the "subjects_to_run_syn.txt" file and assigns the participant name corresponding to the current job array ID (`PBS_ARRAYID`) to the `SUBJ` environment variable. The `sed` command is used to extract the line corresponding to the current job array ID from the text file. The extracted participant name will be stored in the `SUBJ` variable for further use in the script.

**Third section:** This script segment executes FMRIPrep with specific configurations and settings for preprocessing neuroimaging data.

1. `singularity run --cleanenv $VSC_SCRATCH/signVWFA/singularity_prep_23.1.0/fmriprep-23.1.0.simg`: This command executes the FMRIPrep preprocessing tool within a Singularity container. `--cleanenv` ensures that the container environment is isolated from the host environment. The path to the Singularity image file (`fmriprep-23.1.0.simg`) specifies the container used for running FMRIPrep.

2. `$VSC_SCRATCH/signVWFA/sourcedata/`: This specifies the directory containing the input BIDS dataset. FMRIPrep will process the data located in this directory.

3. `$VSC_SCRATCH/signVWFA/results_fmriprep/preproc_sdc-syn participant`: This specifies the output directory where the preprocessed data will be stored. `preproc_sdc-syn` is the output directory, and `participant` indicates that FMRIPrep should perform participant-level processing.

4. `--fs-license-file $VSC_SCRATCH/freeSurfer/license.txt`: This specifies the path to the FreeSurfer license file needed by FMRIPrep for some processing steps.

5. `-w $VSC_SCRATCH/signVWFA/temp_files_fmriprep`: This specifies the working directory where FMRIPrep will store temporary files during processing.

6. `--ignore fieldmaps`: This flag tells FMRIPrep to ignore fieldmap data, which are used for distortion correction, during preprocessing.

7. `--use-syn-sdc`: This flag tells FMRIPrep to use synthetic field map-based distortion correction.

8. `--output-space anat MNI152NLin2009cAsym`: This specifies the output space for normalization. In this case, it is set to anatomical (anat) space in the MNI152NLin2009cAsym template.

9. `--skip-bids-validation`: This flag tells FMRIPrep to skip BIDS (Brain Imaging Data Structure) validation, allowing non-compliant BIDS datasets to be processed.

10. `--mem-mb 25000`: This specifies the maximum amount of memory (in megabytes) that FMRIPrep can use during processing.

11. `--omp-nthreads 8`: This specifies the number of OpenMP threads FMRIPrep can use for multi-threaded processing.

12. `--nthreads 8`: This specifies the total number of threads FMRIPrep can use for processing.

13. `--no-submm-recon`: This flag disables submillimeter (submm) reconstruction in FreeSurfer.

14. `--fs-no-reconall`: This flag tells FMRIPrep not to run the FreeSurfer recon-all pipeline.

15. `--participant-label=$SUBJ`: This specifies the participant label to process. The value of the `SUBJ` variable, which was set earlier in the script, is used here.