HCP D-MRI Data Analysis

Seungyong Hwang

2020 2 25

This document describes how to download, preprocess and analyze diffusion MRI (dMRI) data from the Human Connectome Project (HCP) database.

1 HCP dMRI data analysis pipeline

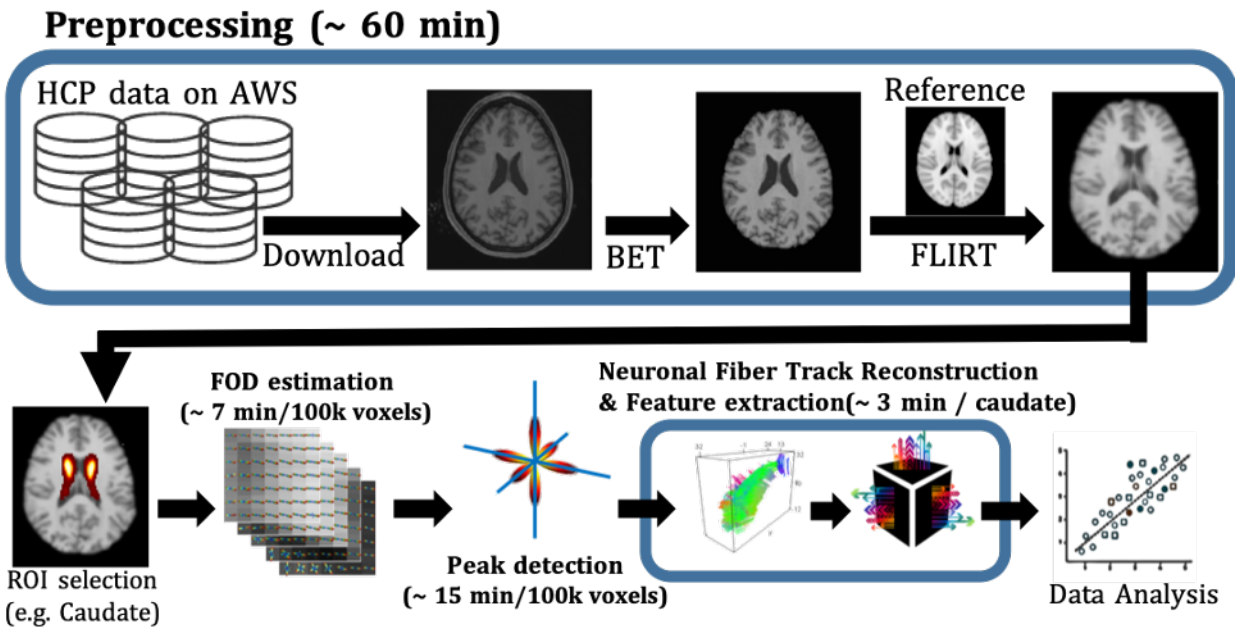


Figure 1: Data analysis pipeline of dMRI data from the Human Connectome Project (HCP) database

2 HCP dMRI data preprocessing

This section describes how to download and process diffussion MRI (dMRI) data from the Human Connectome Project (HCP) database using various *neuroconductor* R packages (https://neuroconductor.org/)

2.1 HCP dMRI data download

Step 1: Obtain AWS access credential. Your AWS access credential (access key id and secret key) is available on https://db.humanconnectome.org/app/template/Login.vm. You need tolog in your ConnectomeDB account and then click the **Amazon S3 Access Enabled** box. If you do not have an account, you need to first create one.

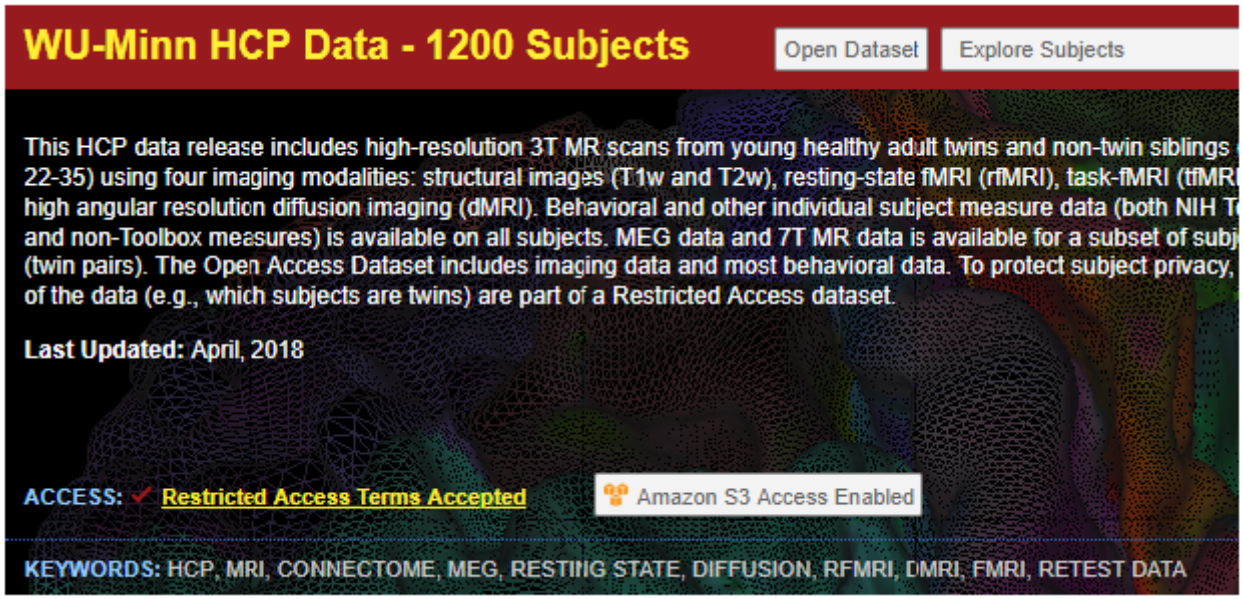


Figure 2: Wu-Minn HCP Data webpage.

Step 2: You need to install the *neurohcp* R package (https://neuroconductor.org/package/neurohcp) for batch downloading from HCP database.

Step 3: You can find ID of available subjects through the **hcp_1200_scanning_info** function in the *neurohcp* package. Two functions are usually used to download data from ConnectomeDB:

- **download_hcp_dir**: To download all files in a certain directory.
- **download_hcp_file**: To download a certain file

```
library(neurohcp)
set_aws_api_key(access_key = "your_access_key", secret_key = "your_secret_key")
hcp_info=hcp_1200_scanning_info # Information of the database
hcp_id=hcp_info$id #ID of available subjects
```

For each subject's dMRI directory, we can download five files (approximately, 1.3GB) as follows:

- data.nii.gz: dMRI data
- bvecs: gradient directions
- bvals: b-values
- nodif_brain_mask.nii.gz: a mask for the brain.
- grad_dev.nii.gz: effects of gradient nonlinearities on the bvals and bvecs for each voxel

```
# "100307" is the subject ID; an output directory should also be specified: default: tempfile()
download_hcp_dir("HCP/100307/T1w/Diffusion",verbose=FALSE, outdir="user_path")

# Downlaod the T1w image of a subject:
#  - structural volume sampled at the same resolution as the diffusion data
# "100307" is the subject ID;
#  - an output file path/name should also be specified: default: NULL
download_hcp_file("HCP/100307/T1w/T1w_acpc_dc_restore_1.25.nii.gz", verbose = FALSE,
                  destfile="user_path/out_file_name")
```

Notes: Sometimes, the corresponding files of an ID found through **hcp_1200_scanning_info** could not

be downloaded through either **download_hcp_dir** or **download_hcp_file**. This problem can occur due to two reasons.

- The first is that the dMRI data of this subject have not been registered on ConnectomeDB.
- The second is that the dMRI data have not been registered with proper directory path on AWS, even though the data are available on ConnectomeDB.

You can check whether the dMRI data are available on AWS by using the **hcp_list_dirs** function: If the result from **hcp_list_dirs** does not have any value in *parsed_resultContentsKey[[1]]*, it means you need to check the availability of file, manually.

```
dir_info=hcp_list_dirs(paste0("HCP/100307/T1w/Diffusion"))

#Check whether dMRI can be downloaded thorugh the "download_hcp_dir" function.
is.null(dir_info$parsed_result$ListBucketResult$Contents$Key[[1]])
```

2.2 HCP dMRI data preprocessing: brain extraction and registration

The *fslr* R package from *neuroconductor* (https://neuroconductor.org/package/fslr) is needed here. Also, you need to install **FSL** on your computer (https://fsl.fmrib.ox.ac.uk/fsl/fslwiki/FslInstallation). The *fslr* package is a wrapper of **FSL**.

```
library(fslr)
```

Step 1 – BET extraction (https://fsl.fmrib.ox.ac.uk/fsl/fslwiki/BET): This tool deletes non-brain tissue from an image of the whole head. The resulting image is used as input for registration (Step 2).

```
##BET step
# input: data.nii.gz -- the original dMRI data (from "download" step)
# output: data_brain.nii.gz -- extracted brain dMRI
fsl_bet(infile="/user_path/data.nii.gz", outfile="/user_path/data_brain.nii.gz")

# input: T1w_acpc_dc_restore_1.25.nii.gz -- the original T1 image (from "download" step)
# output: T1w_brain.nii.gz -- extracted T1 image.
fsl_bet(infile="/user_path/T1w_acpc_dc_restore_1.25.nii.gz",
        outfile="/user_path/T1w_brain.nii.gz")
```

Step 2 – FLIRT registration (https://fsl.fmrib.ox.ac.uk/fsl/fslwiki/FLIRT): This is a fully automated tool for linear (affine) intra- and inter-modal brain image registration. The brain image is registered to a template image. Both linearly and non-linearly generated MNI152 template images (whole head, extracted brain, brain mask and skull) are included in the folder /usr/local/fsl/data/standard/, courtesy of the MNI.

```
## Registration step
# (a) obtain transformation matrix based on T1 image registration
# input:  T1w_brain.nii.gz -- extracted T1 image (from "BET" step),
#         MNI152_T1_2mm_brain.nii.gz -- template image (from FSL)
# output: T1w_brain_flirt.nii.gz -- registered T1 image,
#         flirt.mat -- transformation matrix
# parameter: dof=12 means affine transformation
flirt(infile="/user_path/T1w_brain.nii.gz",
        outfile="/user_path/T1w_brain_flirt.nii.gz",
        omat="/user_path/flirt.mat",
        dof=12, reffile="/usr/local/fsl/data/standard/MNI152_T1_2mm_brain.nii.gz")


# (b) register dMRI  using the registration matrix from (a)
# input: data_brain.nii.gz -- extracted dMRI (from "BET" step)
```

```
#        flirt.mat -- registration transformation matrix (from (a)),
#        MNI152_T1_2mm_brain.nii.gz -- template image (from FSL)
# output: data_brain_flirt.nii.gz -- registered dMRI
flirt_apply(infile="/User/data_brain.nii.gz",
              outfile="/User/data_brain_flirt.nii.gz",
              reffile="/usr/local/fsl/data/standard/MNI152_T1_2mm_brain.nii.gz",
              initmat="/User/flirt.mat")
```

Example * example_HCP_dMRI_download_preprocess.R: Download DWI from AWS and preprocess in **FSL** (BET, FLIRT)

3 ROI Selection

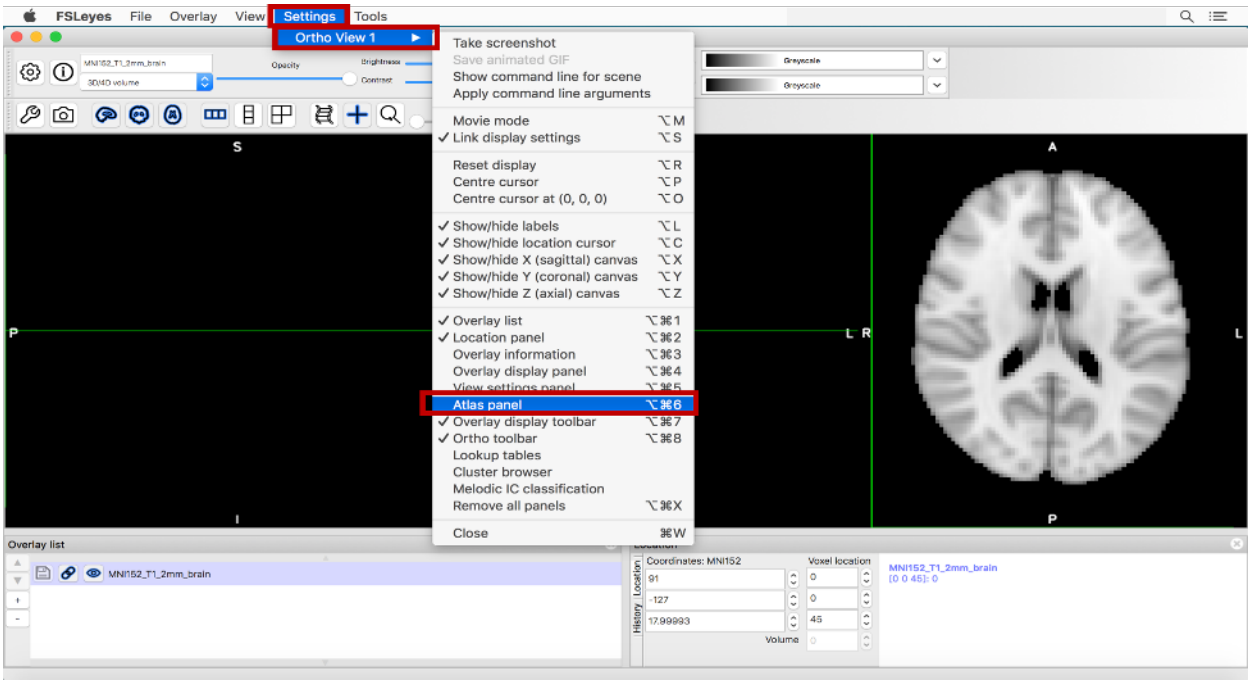
This section describes how to select a region of interest on the template brain in the **FSL** image viewer: **FSLeyes**. If **FSL** is already installed on your computer in the previous step, **FSLeyes** can be used without further installation. Basic configuration of **FSLeyes** is available on (https://fsl.fmrib.ox.ac.uk/fslcourse/lectures/practicals/intro1/index.html)

Step 1 : You have to load the template image which you have used in FLIRT registration step. In our registration step, we used "MNI152_T1_2mm_brain.nii.gz" as a template. To load this template image, in **FSLeyes**,

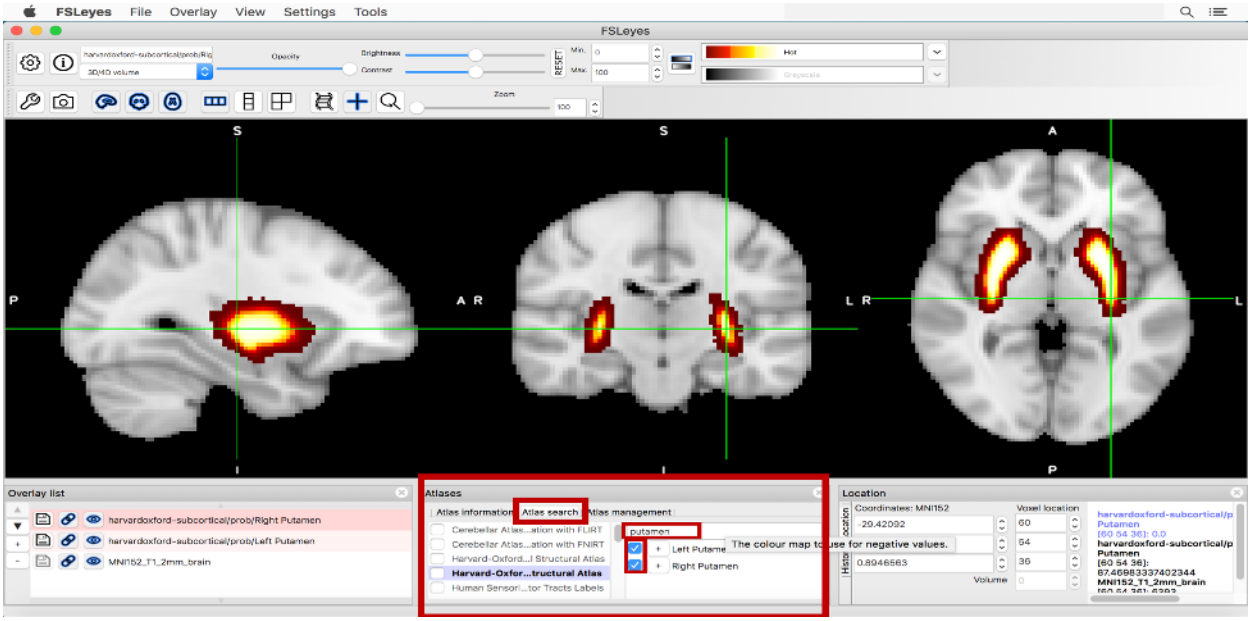
File -> Add standard -> Choose MNI152_T1_2mm_brain.nii.gz.

Step 2 : Based on the available Atalases in **FSLeyes**, we can explore each part of brain.

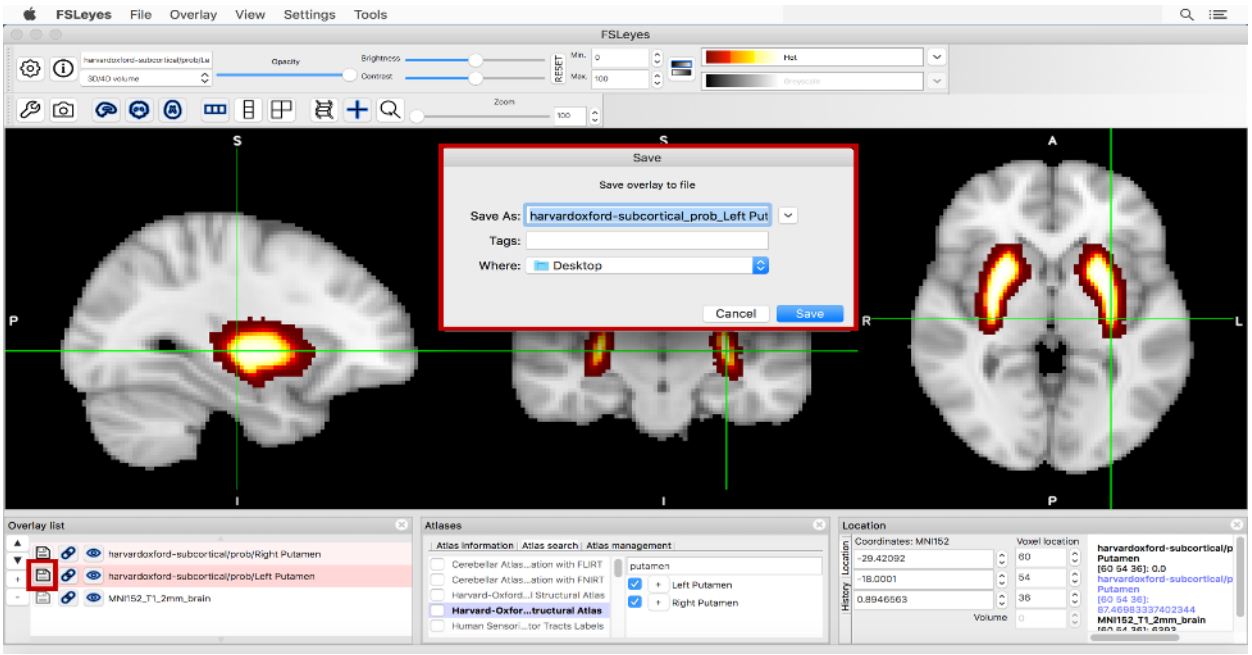
Settings -> Ortho View 1 -> Atlas Panel



Step 3: In "Atlases" Panel (at the bottom of the window), choose an atlas and a prespecified brain region: Click on the atlas search bar and search the part of brain (e.g., "putamen"). Then, the selected part of brain ("putamen") is projected onto the template image.



Step 4: Click the button (highlighted by red) to save the mask of the selected ROI as an .nii.gz file. This file will be used in the subsequent data analysis (see example_HCP_analysis.py for details).



4 FOD Estimation and Peak Detection

This part is done in *python* and *matlab*.

Data: in */data* folder:

- For HCP application:
  - bvals: b-values
  - bvecs: gradient directions
  - data_brain_flirt.nii.gz: whole brain dMRI (This is not given in the repository since registration on connectome DB is required to access these data)
  - Caudate(R).nii.gz: mask of ROI (Caudate in the right hemisphere)
  - Caudate(R)_noise.nii.gz: synthetic image for Caudate in the right hemisphere (original signals with artificially added small noises)
- FOD plotting example:
  - est_result.mat: The mean and sd of the estimated FODs from 100 numerical simulation.

Example scripts: in */example_scripts* folder:

- example_simulation_fib2.py: simulation example where the true FOD has two peaks
- example_simulation_fib3.py: simulation example where the true FOD has three peaks
- example_HCP_analysis.py: script to run HCP data application
  - Input: bvals, bvecs, Caudate(R).nii.gz, Caudate(R)_noise.nii.gz
  - Output: HCP_peaks.mat
- example_plot_fod.m: script to plot the estimated FOD in matlab
  - Input: est_result.mat

Python codes: in */python* folder:

- dwi_simulation.py: For simulating dMRI signals and evaluation of simulation results on FOD estimation.
- fod_estimation.py: Functions for the three FOD estimation methods BJS, SHridge and superCSD
- fod_HCP_application.py: Functions for HCP data processing including 1–Estimation of response function parameters, 2–ROI information organization, 3–Gradient direction extraction according to b-value groups
- FOD_peak.py: Functions for the peak detection algorithm.
- sphere_harmonic.py: Functions to evaluate the spherical harmonic basis in spherical grid.
- sphere_mesh.py: Functions for sampling schemes on the sphere.

Matlab codes: in */matlab* folder:

- fod_plotting: Functions for plotting estimated FOD

5 Tractography and Feature Extraction

This part is done in *R*.

Data: in */data* folder:

- HCP_peaks.mat: extracted peak of estimated FODs. These are used as inputs for the tracking algorithm.

Example script: in */example_scripts* folder:

- example_HCP_tractography.R: this is the script to run the tracking algorithm on the HCP application
  - Input: HCP_peaks.mat

R package: in */dmri.tracking-r*

- dmri.tracking_0.1.0.tar.gz: R package for tracking algorithm and tractography.

6 Supporting software requirements:

- R(version 3.6.2)
  - required R packages (rgl, R.matlab, dmri.tracking, neurohcp, fslr)
- python3(3.7.6)
  - required python3 packages (numpy, scipy, tqdm, nibabel, warnings)
- matlab(R2017a)
- FSL(version 6.0.3)
- Xquartz (version 2.7.11)
- Github: https://github.com/vic-dragon/BJS