DONSURF

Diffusion ON SURFace

This document is a short step-by-step guide about how to use the DONSURF code.

DATA PROCESSING:

Software requirements:

· Tested on python2.7 and python3.6

·python packages: numpy, scipy, argparse, nibabel, logging

· freesurfer (v6 is REQUIRED for the partial volume step)

·fsl (Strongly recommend to have the FSLDIR inside $PATH,

i.e export PATH=${FSLDIR}/bin:${PATH})

·diffusion_pipeline_mgh.py

·koo_orig_pvc.py

Folder Structure:

· In order to run the pipeline, the script NEEDS to have an apriori folder structure. It's very simple: for each subject, a folder with the DWI.nii.gz (4D nifti volume), bvec and bval files inside that folder. It is also NECESSARY that the files inside the diffusion folder are named WITH EXACTLY the same name as the folder. Here is an example:

-> Main folder

--> subjectA

-->subjectA.nii.gz

-->subjectA.bvec

-->subjectA.bval

--> subjectB

-->subjectB.nii.gz

-->subjectB.bvec

-->subjectB.bval

Since the code take advantage of Freesurfer's segmentation, it is very important to first, check visually that all the segmentations are OK. Moreover, all the Freesurfers MUST be together in the same folder, similar to $SUBJECTS_DIR. This can be easly done using symlinks.

Run the code:

·Example of running the code:

> python diffusion_pipeline.py –subj-list=/path/to/txt/subjlist.txt \

–dti-path=/path/to/dti/folder \

–fs-path=/path/to/fs/subjects \

–reg=bbr –dof=6 \

–tr=13677 –te=61 –bval=1000

·Inputs

> *subj-list* : It's a simple .txt file with the subjects name to process. The name MUST be the same as the subject's folders name. e.g: subjectA \n subjectB \n subjectC

> *dti-path* : Path where the folders containing the DWI data are placed.

> *fs-path* : Path with all the Freesurfers-processed subjects.

> *reg* : Type of registration algorithm. Default is bbregister

> *dof* : Degrees of Freedom for the registration algorithms

> *tr* : Repetition time. Needed for the Partial Volume Correction

> *te* : Echo time. Needed for the Partial Volume Correction

> *bval* : Bval. Needed for the Partial Volume Correction

> *debug* : If True, all the temporal files will be saved in a folder names "tmp"

Step by step pipeline's explanation:

0) Freesurfer's subject name MUST contain the DWI. E.g DWI→ subjectA; Freesurfer→tp1_subjectA001_test_svPPA

1) Motion correction. During this step, using FSL's eddy_current, the 4D DWI.nii.gz will be rotated in order to compensate for subject's movements during the acq. This rotation is also applied to the bvec matrix using the FSL's rotate_bvecs2 script. You should copy the rotate_bvec2 script to you $FSLDIR folder. (Currently working on a new version using *eddy* function)

2) BET the DWI. Skullstrip the b0 DWI image. Useful to maximize DWI-2-T1 registration.

3) Tensor fit. Using FSL's dtifit, we fit our data to a tensor. NOTE: there are other implementations of the tensor fitting such as Mrtrix one. Planning to incorporate in a near future.

4) DWI-2-T1 registration. There are several algorithms to perform the registration. The default (and recommended) it's the bbregister by Freesurfer. Sometimes, it might be useful to use a regular linear registration using the mri_coreg command. The user can also input the number of the degrees of freedom (--dof), which can be useful if the registration fails with default parameters. NOTE: I am planning to incorporate ANTS nonlinear SyN registration and also RegSeg (to maximize cortical matching)

5) Partial volume correction. We gonna use the tissue percentage obtained using FreeSurfer

5.1) If the subject doesn't have a "fine segmentation", run Freesurfer's gtmseg command to compute it. Requires FSv6.

5.2) Run the command mri_gtmpvc. It will take advantage of the cortical segmentation and the previously computed registration. It will generate a map of tissue probabilities (+ other outputs that we won't need).

5.3) Run the Koo et al 2009 implementation. Afterwards, the resulting PVC volume will be threshold for MD >0. Due to small GM percentage, some voxels might have neg values.

6) Surface projection. We will move to the native surface 3 different metrics. First, the cortical MD sampling at the middle cortical ribbon. Second, the PVC corrected MD sampled at the middle cortical ribbon. Finally, the Superficial White Matter diffusivity (SWM) which computes the mean between the white surface and -3mm.

6.1) Once the volumes are sampled to native surface, we normalize the resulting volumes to the fsaverage space.

6.2) Afterwards, we binarize the resulting volume to generate a mask/label of "where do we have signal?". This is important, since some DWI volumes can be cutted due to misplaced field of view during the MRI acquisition. We NEED to generate a mask where we have signal, in order to smooth the data just inside that mask. Otherwise, we will introduce noise to our data due to the smoothing process (vertex with zero value will result to a [very] small value, and vertex close to zero-value regions will become much lower signal).

6.3) Intersect between the ?h.cortex label and the resulting label obtained in the previous step.

7) Smooth the cortical volumes INSIDE the computed label.

8) Copy the resulting diffusivity files inside each individual Freesurfers folder. This will make the statistical analysis much easier.

9) There is a log file inside each subjects folder, in order to have a tracking of the exact commands used and to easily detect where the code fails (if fails).

***

At this step, I suggest to visualize/inspect the registrations one by one. A mismatch or bad registration might results to spurious statistical results, since CSF inclusion would affect considerably the further analysis. I have a wrapper for a fast visualization if you are interested.

***

STATISTICS:

Some scripts have been written to make things easier when performing statistics. It also provides the option to harmonize data for multicenter analysis using Fortin et al 2017, NeuroImage [combat].

Software requirements:

·**Tested on** python2.7 and Python3.6+

·nibabel

Freesurfer6 (needed for the mri_binarize option)

·combat folder

·Matlab

·prepare_GLM.py

·glm2X.m

·run_GLM_dti.sh

·multiple_comp.sh

Folder Preparation:

This is mandatory to run before the statistical analysis. The script *prepare_GLM.py* will generate the folder structure, prepare files, and harmonize the input data if required for further statistical analyses (mainly correlations and group comparisons).

·Example of running the code:

```
> python prepare_GLM.py –fdir=/path/where/stats/computed/afterwards   \
–glmtxt=/path/to/txt/with/glminfo \
–subj-dir=/path/to/Freesurfers/folder/with/subjects/with/MDdata \
–name=MD-GM \
--harmon
```

·Inputs

> *fdir* : Path where all the necessary files for further statistical analysis will be placed. The input file ?h_y.mgh, the mask, the X.mat and the folders for the results.

> *glmtxt* : This is the info that will be used to generate the X.mat files used by mri_glmfit. You can find an example of this file in the Appendix. For multicenter studies and the harmonization step, the last covariate MUST be the center.

> *subj-dir* : Path with all the Freesurfers processed subjects. These subjects must contain the volume that will be used for the statistics. E.g $subj-dir/subjectA/surf/lh.GM-MD.fwhm15.fsaverage.mgh

> *name* : Name of the metric to consider. For diffusion it will be GM-MD, GM-MD-koo, SWM-MD. This is useful if you have other metrics sampled to surface such as PET data.

> *harmon* : If included, the script will compute the harmonization volumes. It will use the last column of the glmtxt as the center variable (the one used to compensate center artifacts)

Notes:

1) The script will also generate the files ?h.framemask.mgh. As we have seen before, there is the chance that some subjects do not have data for certain vertex (due to field of view, aggressive skullstrip, registration errors, etc). This mask file will tell mri_glmfit which subjects should be included to perform the statistics. i.e for the vertex #5, it will check how many subjects DO HAVE data, and perform the stats only using those subjects. This will result in GLMs with different subjects for each vertex. Even though this might sound critical, it is not. The number of subjects that are not included is usually small, and it is preferable to skip few subjects rather than include values of zero in your statistics.

Run Statistics

Once you have the folder and the necessary files generated, running the statistical analysis (group comparison and correlation with GLM) it's very straightforward.

1) Generate a txt file named "contrasts" inside the $statsfolder/GLMDIR folder. Write the specific contrasts for that analysis (i.e 0 1 0 0 0 if your variables are dx, age, sex, education). Note that the first column, its always dedicated to the intercept. If you have further interest, you can open the X.mat file in matlab and you will check that its equal to your glmtxt + first column of 1s

2) Run the statistics script run_GLM_dti.sh. This script will compute statistics using GLM. It will also compute multiple comparisons using the already computed Monte Carlo simulations. This will be done for each hemisphere independently. The user have to indicate if the data has been harmonized or not (files prefix stuff).

·Example of running the code:

```
> ./run_GLM_dti.sh -f ${MGHFOLDER}/stats -z 0
```

APPENDIX:

glmtxt example. No header is included in the glmfile.txt. For multicenter analysis, the last covariate MUST BE the center/site. In this example the header would be (subjname; dx_group; age; sex; center). The prepare_GLM.py script will use the first column to prepare the files, and the other ones to generate the X.mat file. Importantly, the script includes an extra first column with all ones, as the intercept.

subjA	1	54.32	0	1
subjB	1	76.54	1	2
subjC	0	87.65	0	2
subjD	0	43.21	1	1