hClustering tree-building quick-start guide

1. Prior requirements

a. dmri preprocessing prerrequesites:

Install if necessary the following packages:
FSL
ANTS
FREESURFER
MRTRIX
PYTHON
NIPYPE (for python)
A **CONDOR** SYSTEM FOR CLUSTER COMPUTING (for tractography scripts)

Clone and compile the pipeline code from github:
https://github.com/dmordom/dmri_prepro_nipype.git.

To run the pipeline the following prior data is required:

- A T1 image, named "t1.nii(.gz)".
- A dMRI 4D dataset (with intercalated b0 images), named "data.nii(.gz)".
- A set of two ASCII files with the *b*-values and the *b*-vector directions related to the dMRI acquisition named "bvals" and "bvecs" respectively.

b. hClustering commands prerrequesites:

Install if necessary the following library packages:

Boost [libboost-all]:	Tested with version 1.48, higher versions should also work.
via [libviaio]:	Provided as part of the repository. For vista files handling.
nifti-1 [libniftiio]:	For nifti files handling.
lznz [libznz]:	Required by nifti library.

clone and compile the hClustering code from github:
https://github.com/dmordom/hClustering.git

2. Running the preprocessing pipeline:

Some variables can and must be tuned in p0_main.py:

- tract_number (integer): the number of streamlines that are wished to be generated per seed voxel
- tract_step (float): the tracking step size
- freesurfer_dir (path): the directory where the freesurfer recon_all outputs will be written (or are located)
- data_dir (path): the input data directory
- register_to_mni (boolean): If true the t1 data will be registered to an mni template before generating the surfaces (but then dmri data projection will not work unless dmri data is also registered)
- use_condor (boolean): use cluster computing
- use_sample (boolean): generate only a small sample of tracts from the gm/wm boundary
- clean (boolean): eliminate workflow intermediate outputs after a sucessful run
- pipe_start (string): pipe script point where to start processing the first subject
- pipe_stop (integer): pipe script point where to stop processing each subject
- pipe_restart (string): pipe script point where to start processing the next subject
- subject_list (string list): a list with the names of all subjects to be processed as per the input folders
- workflow_dir (path): directory for the sample option workflow
- output_dir (path): directory for the sample option outputs
- chunk_nr (integer): number of chunks into which to divide the tractography computing for the sample option
- workflow_dir (path): directory for the workflow intermediate outputs
- output_dir (path): directory for the outputs
- chunk_nr (integer): number of chunks into which to divide the tractography computing.

Generate the output directory and workflow directories before running.
Copy the track_script_header.sh and track_script_body.sh files from the mri_prepro_nipype clone directory to the output directory before running.

Once the previous steps have been completed open a console and console run:
"FSL --version 5.0"
"export PATH=/your_ants_path/ANTS/antsbin/bin:$PATH"
"MRTRIX"
"Freesurfer"
"SUBJECT_DIR=your_freesurfer_subjects_dir"
" python PATH/dmri_prepro_nipype/p0_main.py"

The pipeline will then start running and perform all the steps automatically until all the preprocessing steps are completed.

3. Building the Hierarchical tree:

Two outputs from the pipeline are required to compute the hierarchical tree:

- /OUTPUT_DIR/SUBJECT/roi/SUBJECT_hclust_roi_left(right).txt: the file with the dataset size, number of streamlines per voxel, tractogram seed coordinates and tractogram IDs per coordinate. Referred to as "roi.txt" below.
- /OUTPUT_DIR/SUBJECT/compact_tracts/lh(rh): the folder with the seed compact tractograms generated. Referred to as "compact_tracts/" below.

An output directory (referred to as "tree_output/" below) and a temporary files directory (like "/tmp") will also be needed.
The recommended neighbourhood value is 26, the relative threshold 0.001 (tract voxels visited by less than one-per-thousand of the streamlines will be set to 0 prior to computing distances) and the maximum neighboring distance to 0.1. Number of meta-leaves should be selected considering the number of seeds in the set and the desired maximum granularity level. For information on the rest of the available options refer to the hClustering commands manual or type buildctree -h at console prompt.

Considering what has been explained above, a possible(recommended) command to compute the tree would be:

```
buildctree -r roi.txt -I compact_tracts/ -O tree_output/ -T /tmp -t 0.001
   -d 0.1 -c 26 -N 2000 –v -m 4 –k
```

An alternative command using vista formats would require correspondingly formatted inputs (roi coordinates should vi in vista and tracts should be .v) and adding the --vista option:

```
buildctree -r roi_vista.txt -I compact_tracts_vista/ -O tree_output/ -T
   /tmp -t 0.001 -d 0.1 -c 26 -N 2000 –v -m 4 –k --vista
```

4. Visualize tree outputs with openwalnut:

www.openwalnut.org.