Diffusion MRI, Distortion Correction, & DTI

Contents:

HCP Diffusion MRI Data

Due to the large size of the HCP diffusion MRI datasets, many of the commands that we instruct you to "run" in this practical are not actually practical to run on the HCP course machines. Many of the commands are therefore shown FYI rather than intended to be run. The output from each command has been "pre-baked" for you.

Study each command carefully, so you understand what it does, then look at the pre-baked output from each of the stages. If you are uncertain about a command/output please ask one of the instructors to explain it to you.

Viewing Diffusion-Weighted data

- `cd /home/hcpcourse/day3-wednesday/Diffusion_practicals/100307/Diffusion`
- Type `$ ls raw/`

This lists the directory so you can see a list of files that are obtained from a typical HCP diffusion MRI acquisition prior to any pre-processing.

This includes a number of *.nii.gz image files, as well as *.bval and *.bvec that contain the information on the diffusion-sensitising magnetic field gradients. There are six of these groups of files, each corresponding to a 10-minute scan session. The six data files will be merged to one large file after distortion correction.

The *.bval files contain a scalar value for each applied gradient, corresponding to the respective b-value. The *.bvec files contain a 3x1 vector for each gradient, indicating the gradient direction. The entries in *.bval and *.bvec are as many as the number of volumes in the corresponding *.nii.gz file.

- You can quickly see the contents of these two files, by typing cat raw/"name".bval and cat raw/"name".bvec respectively (and where you replace "name" by the actual name of the file you want to list, e.g. `cat raw/100307_3T_DWI_dir95_LR.bval`).

The ith volume in the data corresponds to a measurement obtained after applying a diffusion-sensitising gradient with a b-value given by the ith entry in *.bval and a gradient direction given by the ith vector in *.bvec. We will use FSL's viewer (fslview) to quickly have a look at the data. Bring up FSLView open one of the .nii.gz files, for instance:

```
fslview raw/100307_3T_DWI_dir95_LR.nii.gz &
```

This is the diffusion data before correction for distortions. Look in particular at the basal slices and you will see very evident left-right distortions causing the brain to appear very asymmetric.

- It is a 4D image so you can view successive volumes using the “+” button next to the Volume option in the lower left.
- Turn on movie mode (Click the ‘movie’ button at the top, 5th button from the right; use hover-over tool-tips if needed.)
- In the Intensity control (right side of toolbar), keep Min at 0, set Max to 2000.
- Set the movie loop frame rate to 200ms (in the Misc tab after opening the Options dialog, 2nd button from right).

Look at the axial and coronal views and note the movement between volumes caused by eddy current-induced distortions. The reason you see this as a "movement" is that each volume is associated with a different diffusion gradient, and hence different distortions. You will also notice that the intensity changes quite drastically between frames, which is because there are three different b-values in this file (go back to the corresponding .bval file if you want to remind yourself what they are). These intensity changes can make it hard to appreciate the "movement" caused by the distortions.

If you want to make this clearer, you can use select_dwi_vols to obtain a file with just one of the b-values/shells. You do this with the command:

```
select_dwi_vols "image file" "bval file" "output name" "bval"
```

where you choose the file you want to split and the shell you are interested in. Specifically if you wanted to see the b=2000 shell from the LR file with 95 volumes your command would be:

```
./select_dwi_vols raw/100307_3T_DWI_dir95_LR \
raw/100307_3T_DWI_dir95_LR.bval shell_2000 2000
```

This has already been run for you, so that you can now load the result into fslview:

- *Select* **File: Add ../shell_2000** (i.e., one directory up).
- *Turn on* the movie (and adjust the intensity range if you like). The distortion induced "movement" should now be much clearer.

This particular subject happened to lie reasonably still, but in general you would often see additional "movement" due to actual subject head movement.

- *Turn off* movie mode and close the shell_2000 file (**File: Close**).

- *Re-open* **raw/100307_3T_DWI_dir95_LR** (**File: Open** - if it is no longer loaded in flsview) and have a look through the different volumes (different "timepoints" in the 4D data). See if you can work out which volumes had no diffusion gradient applied (e.g. "0" and "16", can you find more?).

Now, let's have a look at all the data associated with a voxel.

- *Choose* a CSF voxel (e.g. enter [68,101,50] for x,y,z in the lower left)

- *Choose* from the menu **Tools->TimeSeries** to observe the data associated with this voxel.

A new window will appear with a plot of the signal intensity at the chosen location for the different diffusion-weighted volumes. Notice the few high intensity values (>8000) and the low intensities (<2000) for most of the datapoints. The former correspond to the b=0 images, the latter to diffusion-weighted images, for which maximal attenuation of the CSF signal has occurred.

Now, add to the plot the timeseries for a white matter voxel. To do that:

- use the (+) button on the plot and choose a nearby voxel in the corpus callosum (e.g. [71, 109, 50]). Can you tell and explain the differences in the data?
- Use (+) again and now choose a grey matter cortical voxel (e.g. [84, 141, 50]).

Can you explain the differences? Why is the signal attenuation lower than in the CSF?

For volumes with diffusion gradients applied, see if you can work out the direction of the gradient. Remember that diffusion data appear darker in places where there is more diffusion along the gradient direction. You should be able to work out the approximate gradient direction by looking at a diffusion weighted image, and comparing darker areas with your knowledge of white matter orientation.

- *Look*, for example, at the Corpus Callosum in volume numbers 1 and 3 in the FSL viewer.

To check if you are right - e.g. for diffusion encoding direction for volume 3 (the fourth volume), type in your terminal window:

```
cat raw/100307_3T_DWI_dir95_LR.bvec | awk '{print $4}'
```

Note fslview starts counting from 0, and awk starts from 1! The awk command is extracting the 4th vector from the bvec file, which contains the normalized vectors giving the gradient directions in x, y and z.

- *Close* the timeseries window in fslview (press '**X**' in upper right corner).

TOPUP - Correcting for Susceptibility-induced Distortions and Identifying susceptibility-induced distortions

From the raw/100307_3T_DWI_dir95_LR.nii.gz file choose a volume without diffusion weighting (e.g. the first volume). You can extract this as a standalone 3D image, by entering in the terminal window:

```
fslroi raw/100307_3T_DWI_dir95_LR my_nodif_LR 0 1
```

which will generate a new image called **my_nodif_LR**.

- In fslview, **File: Add ../my_nodif_LR** and have a look at different slices.

Notice that as you go to more inferior slices, the frontal and temporal part of the brain starts to appear distorted (e.g. "squashed", "elongated" or "pushed to the side"). These distortions are always present in EPI images and are caused by differences in the magnetic susceptibilities of the regions being imaged. They are present along the phase-encoding (PE) direction used in the acquisition, and their sign changes with the sign of the PE direction that has been used. The raw/100307_3T_DWI_dir95_LR has been acquired with a Left→Right PE direction.

- *Extract* the first volume from the raw/100307_3T_DWI_dir95_RL and call it **my_nodif_RL**:
  ```
  fslroi raw/100307_3T_DWI_dir95_RL my_nodif_RL 0 1
  ```

- *Open* **my_nodif_RL** in fslview and superimpose it on the my_nodif_LR that was already there.

This image, from the same subject, has been acquired with the opposite PE direction (Right→Left).

- *Double-click* the file names at the bottom of fslviewer to switch on and off the visibility of this image to see how the distortions change sign between my_nodif_LR and my_nodif_RL.

Regions that are squashed in the first volume appear elongated in the second volume and vice versa. Unsurprisingly, the regions that were very obviously distorted when viewed in the my_nodif_LR image

change a lot as you switch back and forth between my_nodif_LR and my_nodif_RL.

Perhaps more disturbing is that some regions that weren't so obviously distorted change a lot as well, implying that they are also distorted, only not quite so much.

- For example, *look* at the Cerebellum in a coronal view at ~y=45 as you switch back and forth.

In these images, the distortions are reasonably clear even in the "non-obvious" regions thanks to the resulting left-right asymmetry. For data acquired with phase-encoding along the AP direction (more typical) it can be very hard to see.

We will correct these distortions by combining the two b=0 images using the TOPUP tool. We will then pass the results on to the EDDY tool where they will be applied to the correction of all diffusion data.

- *Close* fslview.

Running topup

topup uses the information from pairs (or sets) of images acquired with different phase-encode directions/polarities to calculate what the (off-resonance) field that the images were acquired in must have been. First you need to merge the LR and RL images into a single volume using:

```
fslmerge -t my_b0_LR_RL my_nodif_LR my_nodif_RL
```

Next one needs to create a text file acqparams.txt that contains information about the PE direction, the sign of the LR and RL volumes and some timing information obtained by the acquisition. For the file you just created (my_b0_LR_RL) the file would look like
-1 0 0 0.11154 1 0 0 0.11154
and can be created using:

```
echo "-1 0 0 0.11154" >acqparams.txt ; echo "1 0 0 0.11154" >>acqparams.txt
```

The first three elements represent a vector that specifies the direction of phase encoding. The non-zero number in the first column means that it is along the *x*-direction. A -1 means that *k*-space was traversed Left→Right and a 1 that it was traversed Right→Left. The final column specifies the "total readout time", which is the time (in seconds) between the collection of the centre of the first echo and the centre of the last echo. In the FAQ section of the online help for topup, there are instructions for how to find this information for Siemens scanners. There are also example EPI images to help you get the sign of the phase-encoding right (even the author of topup frequently resorts to those images).

The file should contain as many entries as there are volumes in the image file that is passed to topup. You can name this file anything you like. We tend to name it acqparams.txt to make it easy to remember

what it is.

You are now (almost) ready to run topup. But first we need to remove one slice from the data since the config file that we want to use with topup assumes an even number of slices and all HCP diffusion data has an odd number. You can do that with the command:

```
fslroi my_b0_LR_RL my_b0_LR_RL 0 -1 0 -1 0 110
```

If you were to actually run all the commands in this practical, all the files would have to be stripped of one slice like this. It would be easiest done directly on the file in ./raw but DON'T do it. It has already been done for you in all the prebaked results. Now, finally, we can run topup with the command:

```
topup --imain=my_b0_LR_RL --datain=acqparams.txt --config=b02b0.cnf \
--out=my_topup_b0_LR_RL --iout=my_iout --fout=my_fout
```

This will take a little while to run, so unless you are planning a coffee break at this time we recommend you:

- *Control-C* out of topup and instead look at the "pre-baked" results.

The pre-baked results have been calculated in a slightly different way. Instead of relying on just one pair of images we have used three pairs. This was achieved by extracting the first volume (all of them b0) from each of the files raw/100307_3T_DWI_dir9?_LR.nii.gz and raw/100307_3T_DWI_dir9?_RL.nii.gz. These have been collected into a single file topup/b0_LR_RL.nii.gz. Have a look at this file in fslview to make sure you understand it. Since there are now six volumes in the file that we want to run topup on, it must be reflected in the acqparams.txt file. Have a look at it with the command:

```
cat topup/acqparams.txt
```

We ran topup on these files with the command:

```
topup --imain=b0_LR_RL --datain=acqparams.txt --config=b02b0.cnf \
--out=topup --iout=iout --fout=fout
```

You can have a look at the results by cd'ing into the topup directory. The two main result files are called topup_fieldcoef.nii.gz, which contains information about the off-resonance field, and topup_movpar.txt, which specifies any movement between the first and the remaining volumes in b0_LR_RL. The names of these result files are derived from the value of the --out parameter. Had we used --out=rubbestad the output files would have been named rubbestad_fieldcoef.nii.gz and rubbestad_movpar.txt instead.

The topup_fieldcoef.nii.gz file can be viewed with fslview (but if a file of different dimensions is currently loaded, first use File -> Close). It looks like a low resolution fieldmap, and it contains the spline coefficients for the field that topup has estimated. If one wants to inspect the actual field, one needs to look at the output associated with the --fout parameter, in this case fout.nii.gz. You will see that it looks

very much like a "regular" fieldmap and provided that the readout time was specified correctly in the acqparams.txt file it will be quantitative in Hz. Finally the output associated with the --iout output (in this case iout.nii.gz) shows you all the volumes in b0_LR_RL.nii.gz after correction for distortions.

Data acquisition requirements for topup

If you want to use topup on your own data, a minimum requirement is that two spin-echo (SE) EPI images with different PE-directions have been acquired. The best is to acquire them with opposing PE-directions (i.e. A→P AND P→A or L→R AND R→L). An SE-EPI image is the same as a b=0 image acquired as a part of a diffusion protocol. Just as for fieldmaps, this pair must be acquired at the same occasion as the full diffusion protocol, the subject must not leave the scanner in between, and no re-shimming can be done. Ideally the subject should also be in the same position, so it is a good idea to acquire them reasonably close in time.

Parenthetically, we note that for the HCP, topup is also used to compute the field maps for fMRI data, by acquiring two spin echo images along with the gradient echo fMRI.

EDDY - Correcting for Eddy Currents

Preparing to run eddy

There are a few things that eddy needs in order to run. Firstly, it will need a general idea of what is brain and what isn't. To obtain that we generate a brain mask using BET (Brain Extraction Tool) on the corrected b0 data. Remember that topup/iout.nii.gz contains the b0-volumes corrected for susceptibility distortions.

The next two commands you can try if you want (they are fast), but are optional.

In order to create a single, average, correct volume one can use the command:

```
fslmaths topup/iout -Tmean hifi_b0
```

Turning that into a mask can be done using:

```
bet hifi_b0 hifi_b0_brain -m -f 0.3
```

The first command creates a file named hifi_b0.nii.gz that you can check with fslview; the second command creates a file named hifi_b0_brain_mask.nii.gz that you should also check.

The next thing one needs to do is to create a single file that contains all the data, b0 and diffusion weighted alike. This has already been done for you (so **don't** try it yourselves) with a command like:

```
fslmerge -t eddy/LR_RL `echo raw/100307_3T_DWI_dir9?_LR.nii.gz \
raw/100307_3T_DWI_dir9?_RL.nii.gz`
```

resulting in a file named eddy/LR_RL.nii.gz. We also need to create corresponding files with b-values and b-vectors corresponding to the entire data set. Again this has been done for you with commands like:

```
paste `echo raw/100307_3T_DWI_dir9?_LR.bval \
raw/100307_3T_DWI_dir9?_RL.bval` > eddy/LR_RL.bvals
```

and

```
paste `echo raw/100307_3T_DWI_dir9?_LR.bvec \
raw/100307_3T_DWI_dir9?_RL.bvec` > eddy/LR_RL.bvecs
```

The next file we need to create is a text file with one entry for each volume in LR_RL.nii.gz and where that entry is an index into both the acqparams.txt file (telling how that volume was acquired) and into the ./topup/topup_movparams.txt file giving an (approximate) subject "location" as assessed by topup for each volume. These "locations" are used by eddy as a starting guess when estimating subject movement, and can hence be helpful when there is a lot of subject movement. We started with six files and extracted a b0 volume from each of these. For each of these b0 files topup has estimated movement parameters (a "location") and our assumption will be that all volume in the file was in the same location as the first b0 volume. Hence our index file should start with 95 "1"s, followed by 96 "2"s etc. The following commands can generate a file like that:

```
indx=""
ii=1
for p in 1 2
do
    for n in 95 96 97
    do
      for ((i=1; i<=$n; i+=1))
      do
          indx="${indx} $ii"
      done
      ii=$((ii+1))
    done
done
echo $indx > $./eddy/index.txt
```

Running eddy

eddy has already been run in the eddy directory with the command:

```
eddy --imain=LR_RL --index=index.txt --mask=hifi_b0_brain_mask \
--acqp=acqparams.txt --bvals=LR_RL.bvals --bvecs=LR_RL.bvecs \
--topup=../topup/topup --flm=quadratic --niter=5 --rms --nvoxhp=1000 \
--sep_offs_move --peas --out=eddy_corrected --ff=10
```

You will recognise many of the files that we just created, but some parts of this command might need some explaining. topup was the name given as the --out parameter when we ran topup; the parameter -

-topup=../topup/topup will cause eddy to look for the files ../topup/topup_fieldcoef.nii.gz and ../topup/topup_movpar.txt.

The parameter --flm=quadratic specifies that we assume a quadratic model for the EC-fields. That is our current recommendation, and you are unlikely to ever have to use any other model. The --rms flag specifies that we want a text file with the root-mean-square voxel displacement caused by movement for each volume. The --peas flag specifies that we want to do a "post-eddy" check that the different shells are all aligned to each other and to align them if they are not. The --niter=5 parameter specifies that eddy should run for five full iterations. This is typically sufficient, but for data with **a lot** of movement one might want to run, for example, ten iterations instead. Finally the parameters --nvoxhp=1000, --sep_offs_move and --ff=10 are all "technicalities" that we recommend.

As eddy performs a simultaneous registration of all volumes in the data set, it is quite memory and CPU hungry (especially on HCP data); you should **not** try to run the command above. Instead, we suggest you take a look at the "pre-baked" results.

Understanding eddy output

- In the terminal:

```
cd eddy
ls
```

You will find that eddy has produced a number of output files. The two "main" ones are **eddy_corrected.nii.gz** and **eddy_corrected.eddy_parameters**. The former contains the data in LR_RL corrected for susceptibility, eddy currents and subject movements.

As before, it can be quite difficult to assess how well/badly eddy has done because of the vastly different intensities in the different shells. We therefore suggest that you again use the select_dwi_vols command to extract a shell of your choice from LR_RL.nii.gz and eddy_corrected. We have already extracted for you the b=1000 shell, before distortion correction (file LR_RL_shell_1000.nii.gz) and after correction (file eddy_corrected_shell_1000.nii.gz).

- In the terminal:

```
fslview &
```

- In fslview, **File: Open: LR_RL_shell_1000.nii.gz** and **File: Add: eddy_corrected_shell_1000.nii.gz**
- *Press* the **Movie** button.
- *Toggle* the visibility of the "top" file on and off to see how well the distortions/movements have been corrected.

The other "main" result is **eddy_corrected.eddy_parameters** that contains the parameters for movement and the chosen eddy-current model (quadratic).

In addition to these results, eddy will also have generated the following ASCII files:
eddy_corrected.eddy_movement_rms
eddy_corrected.eddy_post_eddy_shell_alignment_parameters
eddy_corrected.eddy_outlier_map
eddy_corrected.eddy_outlier_n_stdev_map
eddy_corrected.eddy_outlier_report

The first of these is the file with root-mean-squared voxel displacements incurred by subject movement. The first column contains values relative to the first volume and the second column "delta" values, i.e. values relative to the previous volume. This file can be used both to find bad data sets and bad volumes, though as always it is difficult to give any absolute recommendations as to what constitutes "too much" movement.

The second file contains estimates of "residual" movement between shells (and relative to the b0 "shell"). If the --peas was specified in the eddy command these estimates will have been applied to the data (i.e. they are no longer "residual").

The remaining files are all associated with outlier detection. When a subject's movements coincide in time with the diffusion weighting part of the sequence, it can lead to complete or partial signal loss in a slice (or group of slices in the case of multi-band data). These files are always generated as part of quality control that eddy attempts to provide, but no action is taken unless one has specified the --repol flag. All three files are text-files. The first two are "matrices" with one row per volume and one column per slice.

The .eddy_outlier_map file is binary and each element is a one (indicating an outlier) or a zero. By default an outlier is a slice that is more than four standard deviations away from its expected value. The .eddy_outlier_n_stdev_map is organised in the same way but instead of ones and zeros each value specifies how many outliers away a given slice in a given volume is. The .eddy_outlier_report file finally is a "plain text" file that lists all the outlier slice-volumes and how many standard deviations away they are.

All of these "outlier" files can be used for quality control, where many outliers imply a lot of movement and possibly a problematic data set. For instance, display the contents of eddy_corrected.eddy_outlier_report to get a quick summary of which slice in which volume is an outlier:

```
cat eddy_corrected.eddy_outlier_report
```

Data acquisition requirements for eddy

This section is for those who might want to use eddy on their own data. eddy works by comparing observed scans with one set of distortions with predicted scans with different distortions. It is therefore crucial that the predictions really have different distortions from the observations.

There are two ways of ensuring that. One is to acquire each diffusion direction twice with opposing phase-encode directions. This has the disadvantage of prolonging the total acquisition, but if one is acquiring more than one repetition (for SNR purposes) anyway, it is a good idea to acquire the two repetitions with different phase-encodes. This is the strategy used by the HCP.
The other, not mutually exclusive, option is to make sure that the diffusion is sampled on the whole sphere. This is as opposed to on the half sphere, for which many existing diffusion schemes are optimised. From a diffusion perspective, it doesn't matter if diffusion is measured along **g** or along -**g**, but from a distortion perspective the two will be each other's polar opposites. This means that if $\mathbf{g}_1$ and $\mathbf{g}_2$ are two diffusion vectors with a small angle between them (i.e. they have similar diffusion information) it is better to instead use $\mathbf{g}_1$ and $-\mathbf{g}_2$. There are examples of diffusion schemes on the whole and half sphere in the online help for eddy.

In general: the data requirements for eddy do not need to increase the acquisition time, and they are completely compatible with the "diffusion requirements". Hence, it doesn't cost anything to do it. However, data that has not been acquired according to those requirements are unlikely to be perfectly corrected by eddy so it is better to think/ask before rather than after the acquisition.

One way of ensuring a direction set that is optimal both for diffusion and eddy is to use the FSL app gps. If you, for example, want to have a 64 direction diffusion protocol that eddy will work well on, you can type:

```
gps --ndir=64 --optws --out=my.bvecs
```

Then my.bvecs will be a text-file with optimal diffusion directions. You will need to complement it with any *b*=0 volumes you want to intersperse.

A closer look at outlier-detection with eddy

You have already seen how eddy reports on what it thinks are outliers. In addition it can attempt to "repair" a data set if one specifies the --repol (REPlace OutLiers) flag. It will then replace the "missing" slice with its own estimate of what that slice "ought to have been". This can be particularly valuable when scanning "difficult" subjects such as children or confused elderly subjects where movement may otherwise lead to invalid examinations.

We have deliberately introduced two partial outliers in the data set you have just looked at. (The "outlier contaminated" data are in the ../with_outliers directory.) Are you tempted to look through the 110x576=63360 slices manually to look for outliers? A more attractive option is to run eddy in “with_outliers” mode using the command:

```
eddy --imain=LR_RL --index=index.txt --mask=hifi_b0_brain_mask \
--acqp=acqparams.txt --bvals=LR_RL.bvals --bvecs=LR_RL.bvecs \
--topup=../topup/topup --flm=quadratic --niter=5 --rms --nvoxhp=1000 \
--sep_offs_move --peas --out=eddy_corrected --ff=10 --repol
```

Eddy will not only find outliers for you, it will also replace them with what it thinks they should have been. Though you cannot, of course, actually run the command right now, you can look at the results we have already generated for you. In the with_outliers directory you will find an additional file compared to when eddy was run without the --repol flag. It is called **eddy_corrected.eddy_outlier_free_data.nii.gz**. If you look at it with fslview you will find that it looks exactly like LR_RL.nii.gz **except** that all slices that eddy thought were outliers have been replaced. If you look at eddy_corrected.nii.gz (which was generated also without the --repol flag) you will see that it is corrected for movements and distortions, but that additionally the outlier slices have also been replaced.

The outliers that were deliberately introduced were slice 46 in volume 3 and slice 53 in volume 4.

Have a look at these in both: **eddy_corrected.eddy_outlier_free_data.nii.gz** and **eddy_corrected.nii.gz**. You can also compare them to the corresponding slices in **eddy/LR_RL.nii.gz** (uncorrected, no outliers) and **eddy/eddy_corrected.nii.gz** (corrected, no outliers).

Fitting Diffusion Tensors

Registering corrected data to T1-space

After distortion correction, the HCP diffusion MRI data (i.e. eddy_corrected) contain a set of directions and bvalues acquired with LR phase-encoding (the first half of the volumes in eddy_corrected.nii.gz) and the same set of directions and bvalues acquired with RL phase-encoding (the second half of the volumes). To reduce file sizes we average the two phase-encodings, so that in the final dataset every volume is the average of the corresponding LR and RL acquisition. Furthermore, we transform the data to the T1-space, so that data from all structural modalities are aligned into the same space. To do that, the data need to be transformed and the bvecs need to be rotated according to the transformation.

Registration of diffusion space to T1 space within the same subject is a rigid-body transformation problem (i.e. only translations and rotations are used). For the HCP, we use boundary-based registration (BBR), which aligns the WM/GM boundary surface between the input and target volumes using a two-pass approach. First we use the FSL script epi_reg to perform a BBR registration between the mean b0 of the eddy_corrected data to the T1w image of the same subject (../T1w/T1w_acpc_dc_restore_brain.nii.gz). We can extract a mean b0 using: (**don't** run the following two commands):

```
select_dwi_vols eddy/eddy_corrected eddy/LR_RL.bvals reg/meanB0 0 -m
```

and register that to T1w as:

```
epi_reg --epi=reg/meanB0 --t1=../T1w/T1w_acpc_dc_restore \
--t1brain=../T1w/T1w_acpc_dc_restore_brain --out=reg/diff2str
```

In the HCP pipelines, a second pass of BBR-registration is performed using Freesurfer's bbregister, which

refines the epi_reg output. The two passes have already been run for you, and the final transform is reg/diff2str.mat. The data then can be then transformed to T1 space using flirt and the applyxfm option that applies the reg/diff2str.mat to the data.

> [Parenthetical note: A similar two pass approach to BBR is used for registering fMRI data to the T1w image in the HCP pipelines.]

We also need to rotate the bvecs accordingly to be consistent with any rotations induced by the transformation. We use the script Rotate_bvecs.sh to perform such a rotation. The end result after distortion correction, averaging of LR and RL data and transformation to T1 space is in the folder **../T1w/Diffusion**. This contains the data.nii.gz file and the corresponding bvals, bvecs files.

Fitting the DTI model and visualising using Workbench

```
cd ../../T1w/Diffusion
```

We can fit the DTI model using dtifit from the command line. This will fit the diffusion tensor model at each voxel indicated by the nodif_brain_mask using the measurements contained in the data file for that voxel. It will then generate maps of features extracted from the tensor, including the eigenvalues and eigenvectors of the tensor, the fractional anisotropy (FA) and the mean diffusivity (MD). Notice that the DTI model is good for low b value data. As the HCP data contain low as well as high b value acquisitions, we first need to extract the lower b value volumes (e.g. the b=1000 s/mm2 directions). The new data, bvals and bvecs files are named as data_b1000, bvals_b1000, bvecs_b1000. To fit the DTI model, we need to run the following command, which has already run for you:

```
dtifit -k data_b1000 -m nodif_brain_mask -r bvecs_b1000 -b bvals_b1000 \
--gradnonlin=grad_dev -o dti
```

The switch --gradnonlin=grad_dev takes into account gradient nonlinearities (spatial nonuniformities in the magnetic field gradients produced by the scanner, which are particularly pronounced in the customized HCP scanner because the head position is a bit off-center). This produces a "corrected" bval and bvec in each voxel that represents better the true diffusion-sensitising gradient at this location than the nominal values. The output files of dtifit are named as dti_*.

We will now switch visualisation to wb_view, as it allows better representations of fibre orientations. Load the fractional anisotropy and mean diffusivity maps into workbench by launching wb_view in your terminal window (which should be **/home/hcpcourse/day3-wednesday/Diffusion_practicals/100307/T1w/Diffusion**):

```
wb_view dti_FA.nii.gz dti_MD.nii.gz &
```

- *Look* at the differences in contrast between the two maps. FA highlights white matter regions, while

MD appears pretty homogeneous between white and gray matter, showing hyperintensities in CSF voxels.

- *Load* the principal DTI eigenvalue **dti_L1** (**File->Open File->Open Volume File...**). Larger values exist in white matter regions compared to gray matter.
- *Load* the secondary eigenvalue **dti_L2**. Observe that L2 is lower in the white matter.
- Also *load* **dti_L3**. The contrast between white and gray matter is even larger, with much lower values being observed in the former.

L1, L2 and L3 do not vary that much in the gray matter (or in the CSF). These differences between L1, L2 and L3 give rise to diffusion anisotropy and lead to high FA values in the white matter and low FA in the gray matter. Keep in mind the diffusion tensor ellipsoid we saw in the lecture for a pictorial representation of the DTI eigenvalues.

We will now visualize the principal DTI eigenvector map onto of the FA. To do this, we first need to convert the FSL output to a format readable by workbench. This can be done using the provided DTIV1_to_Workbench.sh script. What this does effectively is reading the dtifit output dti_V1 that describes vectors using Cartesian coordinates and transforms them to representations using Spherical coordinates. In the terminal window run:

```
./DTIV1_to_Workbench.sh dti_FA.nii.gz dti_V1.nii.gz \
../Whole_Brain_Trajectory_1.25.nii.gz ./
```

This will produce a file called **dtiV1.fiberTEMP.nii**.

- *Open* this file in workbench (**File->Open File->Open Connectivity – Fiber Orientations TEMPORARY Files...**).
- *Open* the **Features Toolbox** (from the menu **View->Features Toolbox->Attach to Right**) and *tick* **Display Fibers**.

What you see in every voxel is the principal eigenvector orientation from every estimated tensor.

- To make sure that you display eigenvectors from a single slice, *set* the **Slice Above/Below Limit** entries to less than the voxel size (e.g. 0.5 and -0.5).
- *Zoom-in* and *change* the view to coronal/sagittal by selecting appropriately from the **Slice Plane** in the Toolbar. You will use additional visualisation options available in the Features Toolbox during the next practical.

End of Diffusion Practical

Authors:

The authors of this practical were Matteo Bastiani, Stamatios Sotiropoulos, and Jesper Andersson, with contributions from Jennifer Elam, Matt Glasser, and David Van Essen.