**GOGNESTIC 2022: Diffusion Tractography and Connectivity Analysis with MRTrix**

This document describes an example of tractography analysis using MRTrix commands. It includes information taken from the MRTrix documentation as well as Andrew Jahn’s Brain Book MRTrix tutorials.

The full MRTrix tutorials and documentation can be found here:

<https://mrtrix.readthedocs.io/en/latest/>

And the more detailed tutorials by Andrew Jahn can be found here:

<https://andysbrainbook.readthedocs.io/en/latest/MRtrix/MRtrix_Introduction.html>

**Diffusion Tractography and Connectivity Analysis**

In this session we will cover preprocessing of diffusion MRI data using the MRTrix interface, how to generate fibre orientation densities (FODs) and a whole brain tractogram. This will then be used to generate a connectivity matrix, or connectome, which can be used to perform group-level connectivity analyses.

**Download the data and set up**

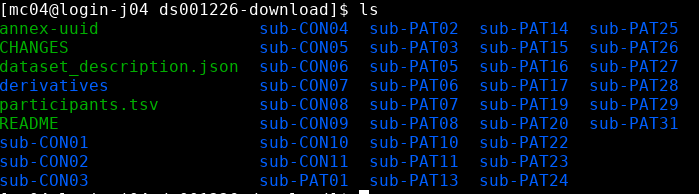
First of all, you need to install MRTrix if you haven’t done so already. For instructions please see:

<https://www.mrtrix.org/download/>

For this tutorial we will use the Brain Tumor Connectomics Data which is available from OpenNeuro:

<https://openneuro.org/datasets/ds001226/versions/00001>

Once you have downloaded the data, the directory including all of the individual subject datasets will be your working directory. In my case this is called ‘ds0011226\_download’ and this is what it contains:



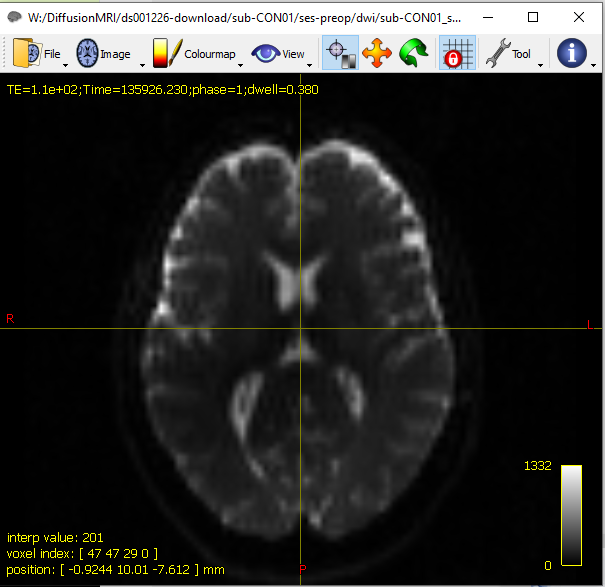
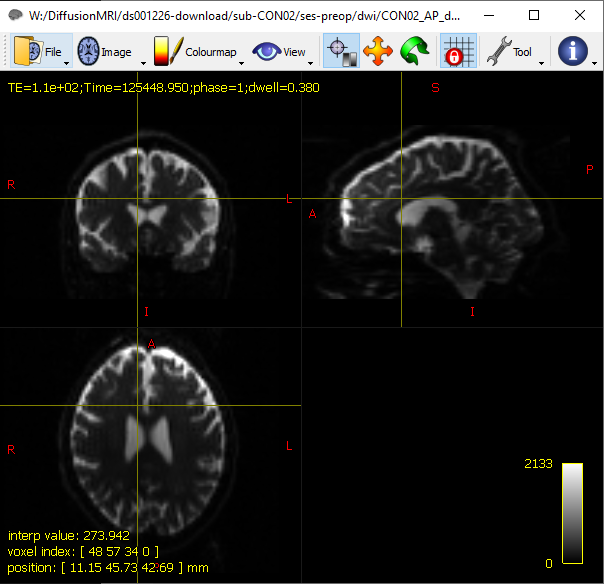
All of the steps described below have been implemented in the scripts **MRtrix\_dMRI\_preprocessing.sh**, **MRtrix\_dMRI\_CSD\_tractography.sh** and **MRtrix\_dMRI\_connectome.sh** which you can download from <https://imaging.mrc-cbu.cam.ac.uk/methods/COGNESTIC2022> .

**Visualising the data**

mrview is the MRTrix image viewer. We can load images directly from the command line, for example:

mrview image1.mif image2.mif

Or we can launch the viewer by simply typing mrview in the command line and then load images, overlays, etc, using the graphical user interface (File -> Open…).

# DWI denoising

MRtrix includes the command dwidenoise, which implements dMRI noise level estimation and denoising based on random matrix theory. The method exploits data redundancy in the patch-level PCA domain ([[Veraart2016a]](https://mrtrix.readthedocs.io/en/latest/reference/references.html#veraart2016a), [[Veraart2016b]](https://mrtrix.readthedocs.io/en/latest/reference/references.html#veraart2016b) and [[CorderoGrande2019]](https://mrtrix.readthedocs.io/en/latest/reference/references.html#corderogrande2019)). The method uses the prior knowledge that the eigenspectrum of random covariance matrices is described by the universal Marchenko-Pastur (MP) distribution.

## Recommended use

Image denoising must be performed as the first step of the image-processing pipeline. Interpolation or smoothing in other processing steps, such as motion and distortion correction, may alter the noise characteristics and thus violate the assumptions upon which MP-PCA is based.

Typical use will be:

dwidenoise dwi.mif out.mif -noise noise.mif

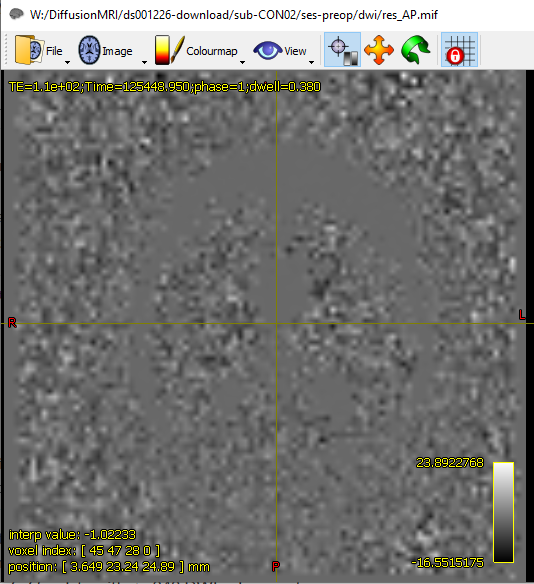
where dwi.mif contains the raw input DWI image, out.mif is the denoised DWI output, and noise.mif is the estimated spatially-varying noise level.

We always recommend eyeballing the residuals, i.e. out - in, as part of the quality control. The lack of anatomy in the residual maps is a marker of accuracy and signal-preservation during denoising. The residuals can be easily obtained with

mrcalc dwi.mif out.mif -subtract res.mif

mrview res.mif



**Removal of Gibbs artefacts (optional)**

MRtrix includes the command mrdegibbs, which attempts to remove Gibbs ringing artefacts from MRI images using the method of local subvoxel-shifts proposed by Kellner et al.

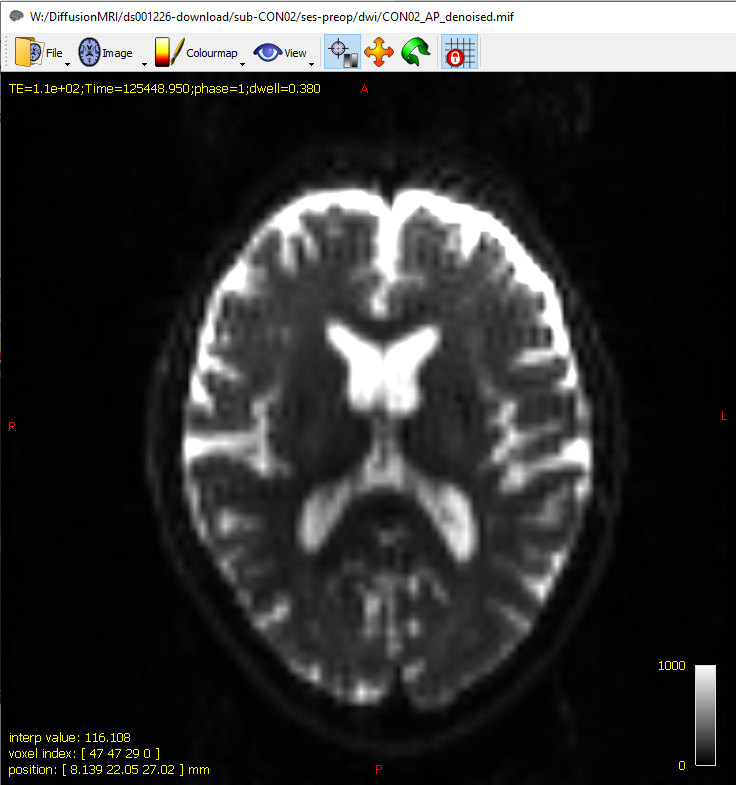
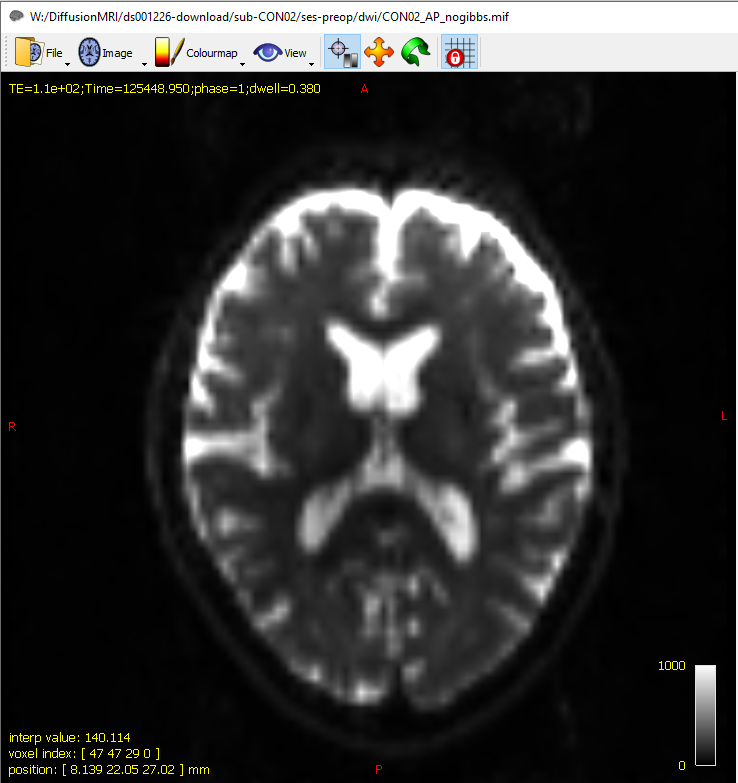
This command is designed to run on data directly after it has been reconstructed by the scanner, before any interpolation of any kind has taken place. You should not run this command after any form of motion correction (e.g. not after dwifslpreproc). However, if you intend running dwidenoise, you should run denoising before this command to not alter the noise structure, which would impact on dwidenoise’s performance.

Note that this method is designed to work on images acquired with full k-space coverage. Running this method on partial Fourier (‘half-scan’) data may lead to suboptimal and/or biased results, as noted in the original reference. There is currently no means of dealing with this; users should exercise caution when using this method on partial Fourier data, and inspect its output for any obvious artefacts.

Typical use will be:

mrdegibbs out.mif out\_nogibbs.mif

where out.mif is the denoised DWI output, and out\_nogibbs.mif is the corrected image.



DWI distortion correction

The dwipreproc script is responsible for performing general pre-processing of DWI series. This script is intended to provide convenience of use of the FSL software tools topup and eddy for performing DWI pre-processing, by encapsulating some of the surrounding image data and metadata processing steps. It is intended to simplify these processing steps for most commonly-used DWI acquisition strategies, whilst also providing support for some more exotic acquisitions. More information on use of the dwipreproc command can be found at the following link: <https://mrtrix.readthedocs.io/en/3.0_rc2/reference/scripts/dwipreproc.html>

The “-topup\_options” and “-eddy\_options” command-line options allow the user to pass desired command-line options directly to the FSL commands topup and eddy. The available options for those commands may vary between versions of FSL; users can interrogate such by querying the help pages of the installed software, and/or the FSL online documentation: <https://fsl.fmrib.ox.ac.uk/fsl/fslwiki/topup/TopupUsersGuide> ; <https://fsl.fmrib.ox.ac.uk/fsl/fslwiki/eddy/UsersGuide>

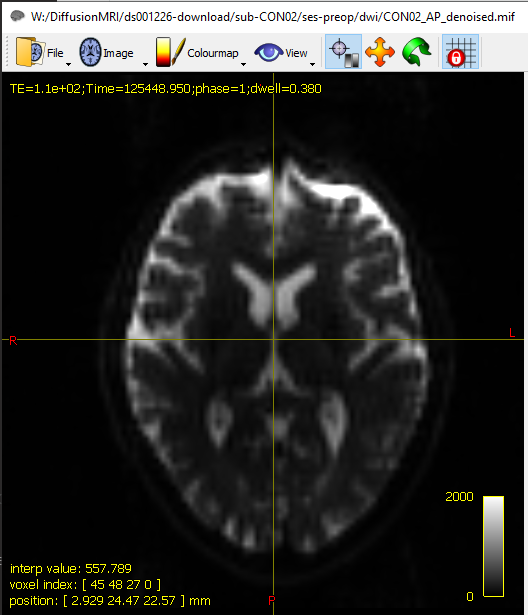
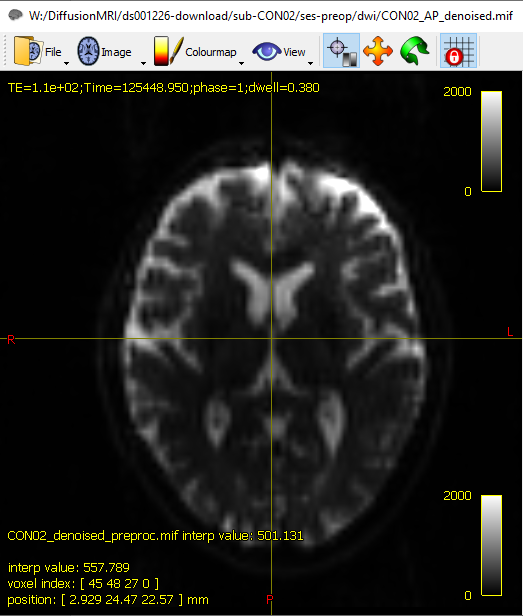
## Example usage

* For DWIs all acquired with a single fixed phase encoding; but additionally a pair of b=0 images with reversed phase encoding to estimate the inhomogeneity field:

mrcat b0\_ap.mif b0\_pa.mif b0\_pair.mif -axis 3;

dwipreproc DWI\_in.mif DWI\_preproc.mif –pe\_dir AP -rpe\_pair -se\_epi b0\_pair.mif -readout\_time 0.72 -align\_seepi –eddy\_options “ –slm=linear”

Here the two individual b=0 volumes are concatenated into a single 4D image series, and this is provided to the script via the -se\_epi option. Note that with the -rpe\_pair option used here, which indicates that the SE-EPI image series contains one or more pairs of b=0 images with reversed phase encoding, the FIRST HALF of the volumes in the SE-EPI series must possess the same phase encoding as the input DWI series, while the second half are assumed to contain the opposite phase encoding direction but identical total readout time. Use of the -align\_seepi option is advocated as long as its use is valid (more information in <https://mrtrix.readthedocs.io/en/3.0_rc2/reference/scripts/dwipreproc.html>).

**Create a brain mask for subsequent steps**

We will use the dwi2mask command for this step, but first it can be useful to apply bias correction to the pre-processed dwi data. Both steps can be achieved using the two commands below:

dwibiascorrect DWI\_preproc.mif DWI\_preproc\_unbiased.mif -bias bias.mif

dwi2mask DWI\_preproc\_unbiased.mif mask.mif

Note that the -ants option will only work if you have ANTs installed on your system. If not, you can use -fsl instead, but this option is discouraged due to its strong dependence on initial brain masking and its inability to correct voxels outside of this mask. To check the output type:

mrview mask.mif



**Constrained Spherical Deconvolution (CSD)**

# Response function estimation

A prerequisite for spherical deconvolution is obtaining the response function(s), which is/are used as the kernel(s) by the deconvolution algorithm. For the white matter, the response function models the signal expected for a voxel containing a single, coherently oriented bundle of axons [[Tournier2004]](https://mrtrix.readthedocs.io/en/latest/reference/references.html#tournier2004) [[Tournier2007]](https://mrtrix.readthedocs.io/en/latest/reference/references.html#tournier2007). In case of multi-tissue variants of spherical deconvolution, response functions for other tissue types are introduced as well; typically to represent grey matter(-like) and/or CSF(-like) signals [[Jeurissen2014]](https://mrtrix.readthedocs.io/en/latest/reference/references.html#jeurissen2014) [[Dhollander2016a]](https://mrtrix.readthedocs.io/en/latest/reference/references.html#dhollander2016a).

In MRtrix3, the [dwi2response](https://mrtrix.readthedocs.io/en/latest/reference/commands/dwi2response.html#dwi2response) script offers a range of algorithms to estimate these response function(s) directly from your dataset itself. This process of estimating response function(s) from the data is non-trivial. No single algorithm works for any possible scenario, although some have proven to be more widely applicable than others.

## General recommendations

### Choice of algorithm

While many algorithms exist, the following appear to perform well in a wide range of scenarios, based on experience and testing from both developers and the [MRtrix3 community](http://community.mrtrix.org):

**Single-tissue CSD:** If you intend to perform (single-tissue) [Constrained spherical deconvolution](https://mrtrix.readthedocs.io/en/latest/constrained_spherical_deconvolution/constrained_spherical_deconvolution.html#constrained-spherical-deconvolution) (e.g. via dwi2fod csd), the [tournier](https://mrtrix.readthedocs.io/en/latest/constrained_spherical_deconvolution/response_function_estimation.html#tournier) algorithm is a convenient and reliable way to estimate the single-fibre white matter response function:

dwi2response tournier dwi.mif wm\_response.txt

Other options include the [fa](https://mrtrix.readthedocs.io/en/latest/constrained_spherical_deconvolution/response_function_estimation.html#fa) or [tax](https://mrtrix.readthedocs.io/en/latest/constrained_spherical_deconvolution/response_function_estimation.html#tax) algorithms.

**Multi-tissue CSD or global tractography:** If you intend to perform a multi-tissue analysis, such as [Multi-shell multi-tissue constrained spherical deconvolution](https://mrtrix.readthedocs.io/en/latest/constrained_spherical_deconvolution/multi_shell_multi_tissue_csd.html#msmt-csd) (e.g. via dwi2fod msmt\_csd) or [Global tractography](https://mrtrix.readthedocs.io/en/latest/quantitative_structural_connectivity/global_tractography.html#global-tractography) (e.g. via tckglobal), the [dhollander](https://mrtrix.readthedocs.io/en/latest/constrained_spherical_deconvolution/response_function_estimation.html#dhollander) algorithm is a convenient and reliable way to estimate the single-fibre white matter response function as well as the grey matter and CSF response functions:

dwi2response dhollander dwi.mif wm\_response.txt gm\_response.txt csf\_response.txt

Other options include the [msmt\_5tt](https://mrtrix.readthedocs.io/en/latest/constrained_spherical_deconvolution/response_function_estimation.html#msmt-5tt) algorithm.

The dataset we are using for COGNESTIC is a multi-shell dataset and therefore we will use the dhollander algorithm for response function estimation:

dwi2response dhollander DWI\_preproc\_unbiased.mif wm\_response.txt gm\_response.txt csf\_response.txt –voxels voxels.mif

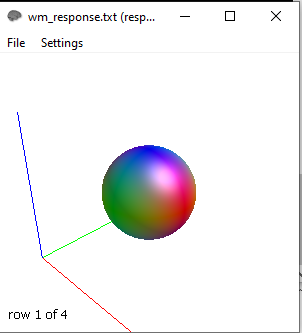
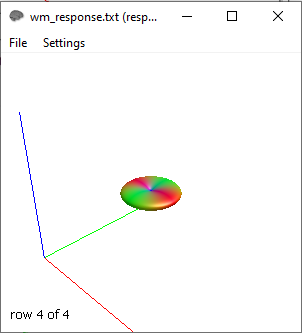
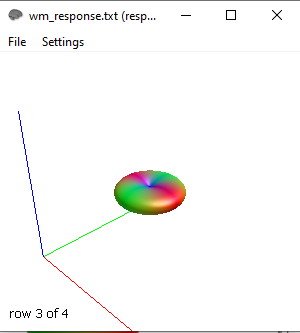
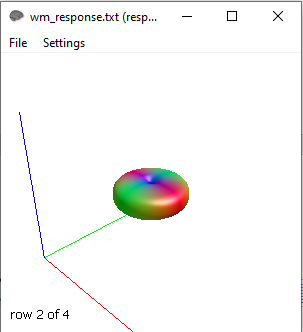
It may also be helpful to check which voxels were selected by the algorithm to estimate the response function(s) from. For any [dwi2response](https://mrtrix.readthedocs.io/en/latest/reference/commands/dwi2response.html#dwi2response) algorithm, this can be done by adding the -voxels option, which outputs an image of these voxels.

### Checking the results

In general, it’s always worthwhile checking your response function(s):

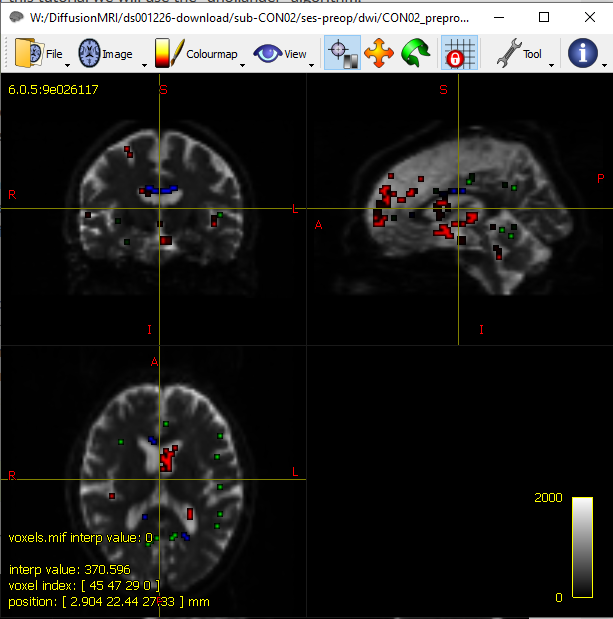
shview wm\_response.txt

Use the left and right arrow (keyboard) keys in this viewer to switch between the different b-values (‘shells’) of the response function, if it has more than one b-value (this would for example be the case for the outputs of the [dhollander](https://mrtrix.readthedocs.io/en/latest/constrained_spherical_deconvolution/response_function_estimation.html#dhollander) algorithm).

The voxels.mif image can be overlaid on the dwi.mif dataset using the [mrview](https://mrtrix.readthedocs.io/en/latest/reference/commands/mrview.html#mrview) image viewer for further inspection.

mrview DWI\_preproc\_unbiased.mif -overlay.load voxels.mif



# Multi-shell multi-tissue constrained spherical deconvolution

## Introduction

Multi-Shell Multi-Tissue Constrained Spherical Deconvolution (MSMT-CSD) exploits the unique b-value dependencies of the different macroscopic tissue types (WM/GM/CSF) to estimate a multi-tissue orientation distribution function (ODF) as explained in [[Jeurissen2014]](https://mrtrix.readthedocs.io/en/latest/reference/references.html#jeurissen2014) As it includes separate compartments for each tissue type, it can produce a map of the WM/GM/CSF signal contributions directly from the DW data. In addition, the more complete modelling of the DW signal results in more accurate apparent fiber density (AFD) measures and more precise fibre orientation estimates at the tissue interfaces.

## User guide

### Prerequisites

MSMT-CSD relies on multi-shell high angular resolution diffusion imaging (HARDI) data, containing multiple b-values. The number of tissue types that can be resolved is limited by the number of b-values in the data (including b=0). To resolve the three primary tissue types in the brain (WM, GM & CSF), the acquisition should contain at least 2 shells along with the b=0 volumes (i.e. 3 unique b-values).

In addition, this command expects that suitable multi-shell multi-tissue response functions have already been computed.

### Invocation

Multi-shell multi-tissue CSD can be performed as:

dwi2fod msmt\_csd DWI\_preproc\_unbiased.mif wm\_response.txt wmfod.mif gm\_response.txt gmfod.mif csf\_response.txt csffod.mif

where:

* DWI\_preproc\_unbiased.mif is the dwi data set (input)
* <tissue>\_response.txt is the tissue-specific response function (input)
* <tissue>fod.mif is the tissue-specific ODF (output), typically full FODs for WM and single scalars for GM and CSF

Note that input response functions and their corresponding output ODFs need to be specified in pairs.

Typically, you will also want to use the -mask option to avoid unnecessary computations in non-brain voxels:

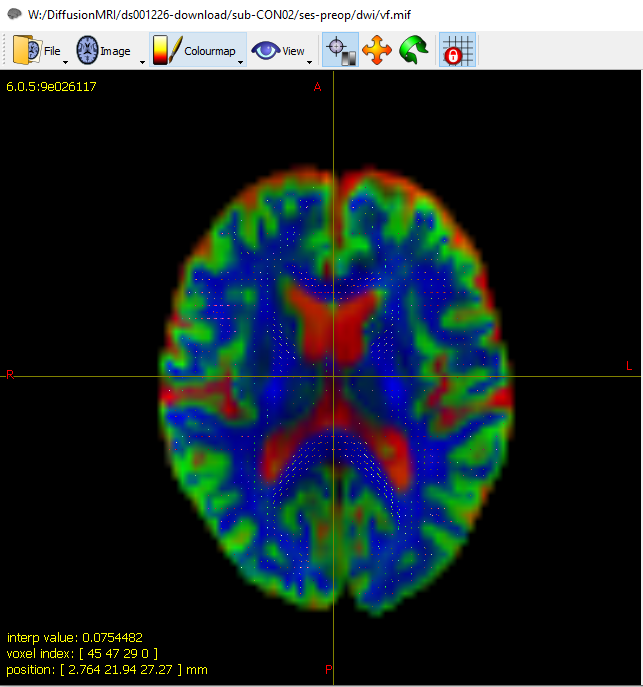
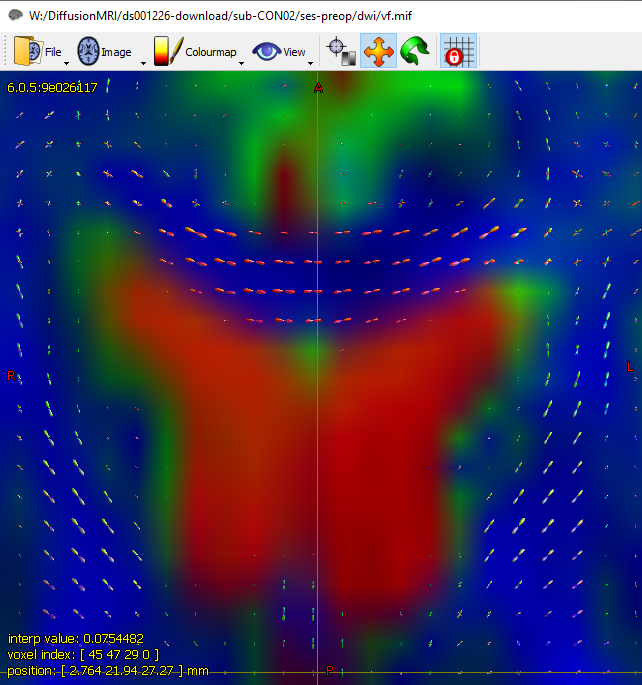
dwi2fod msmt\_csd -mask mask.mif DWI\_preproc.unbiased.mif wm\_response.txt wmfod.mif gm\_response.txt gmfod.mif csf\_response.txt csffod.mif

RGB tissue signal contribution maps can be obtained as follows:

mrconvert -coord 3 0 wmfod.mif - | mrcat csffod.mif gmfod.mif - vf.mif

The resulting WM FODs can be displayed together with the tissue signal contribution map as:

mrview vf.mif -odf.load\_sh wmfod.mif

## Normalization

For group-level analyses, mtnormalisewe will need to **normalize** the FODs. This is to ensure that any differences in intensity in the images will not bias the group effects. To do this, we can use the mtnormalise command:

mtnormalise wmfod.mif wmfod\_norm.mif gmfod.mif gmfod\_norm.mif csffod.mif csffod\_norm.mif -mask mask.mif

# Anatomically-Constrained Tractography (ACT)

### Tissue segmentation

Before performing tractography we need to identify tissue boundaries for placement of ‘seeds’. The first step is tissue segmentation, and FSL’s FAST algorithm is recommended. The 5ttgen script using the fsl algorithm interfaces with FSL to generate the necessary image data from the raw T1 image, using BET, FAST and FIRST. Note that this script also crops the resulting image so that it contains no more than the extracted brain (as this reduces the file size and therefore improves memory access performance during tractography); if you want the output image to possess precisely the same dimensions as the input T1 image, you can use the -nocrop option.

First, we need to convert the anatomical T1-weighted image:

mrconvert T1w.nii.gz T1.mif

Next we will use the command 5ttgen to segment the T1-weighted image into five tissue tipes:

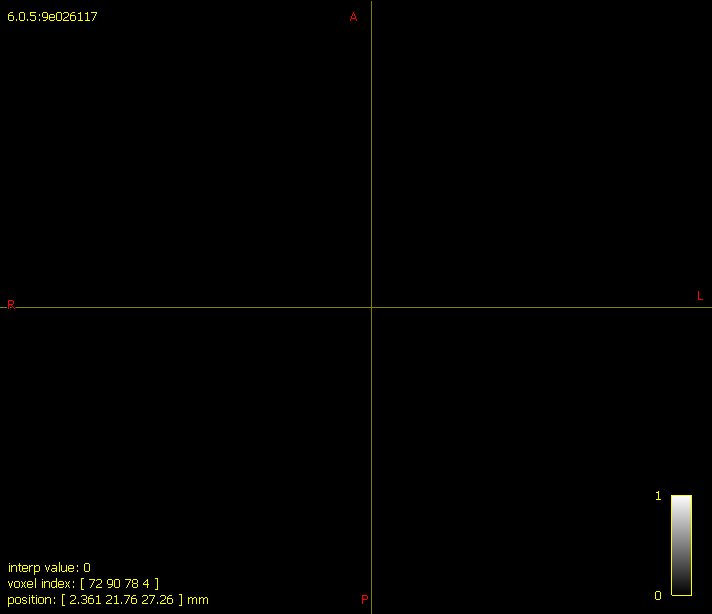
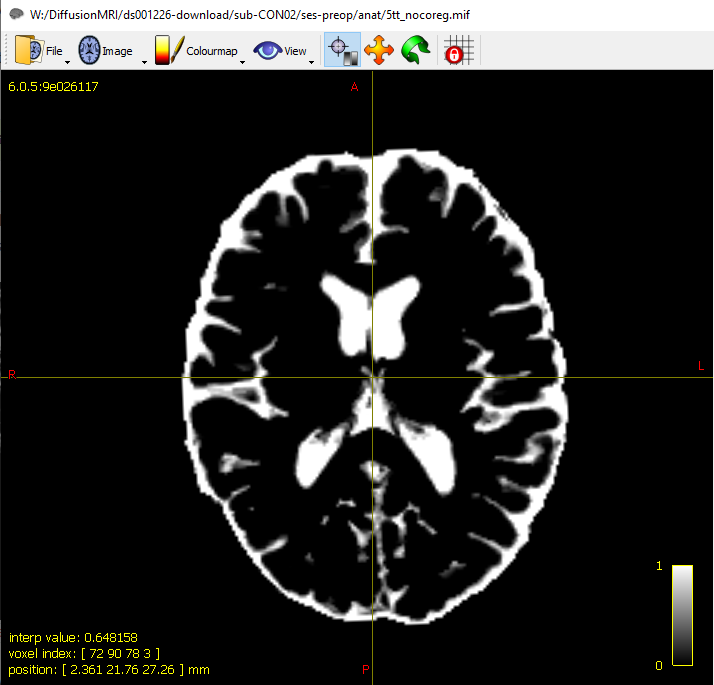
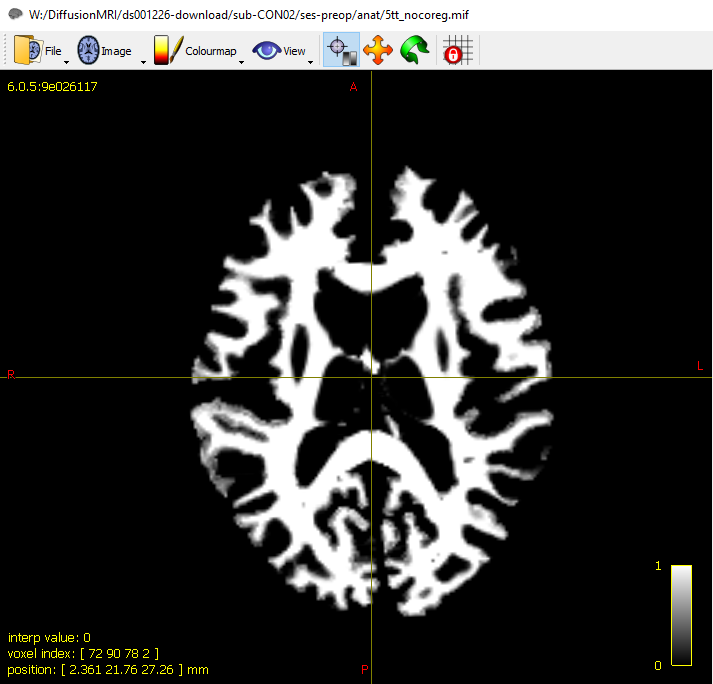
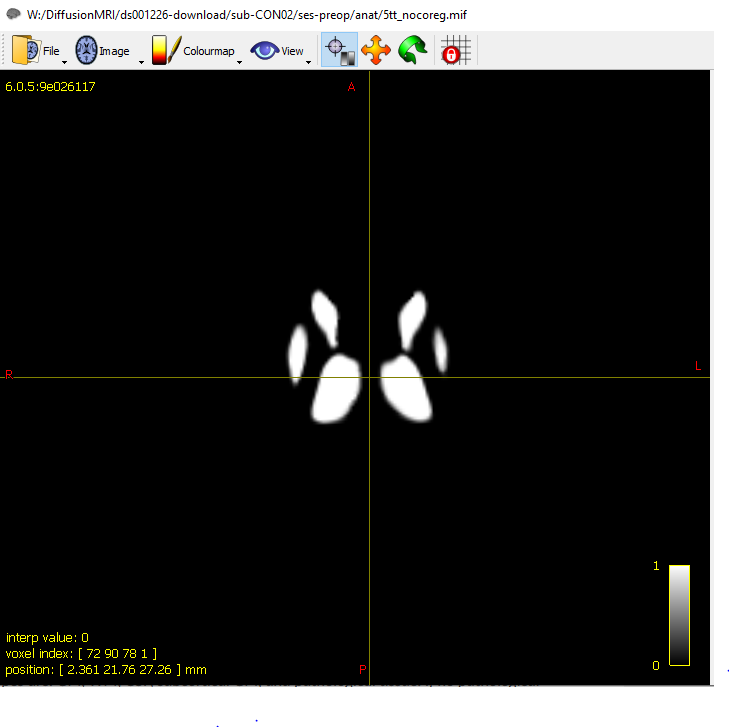
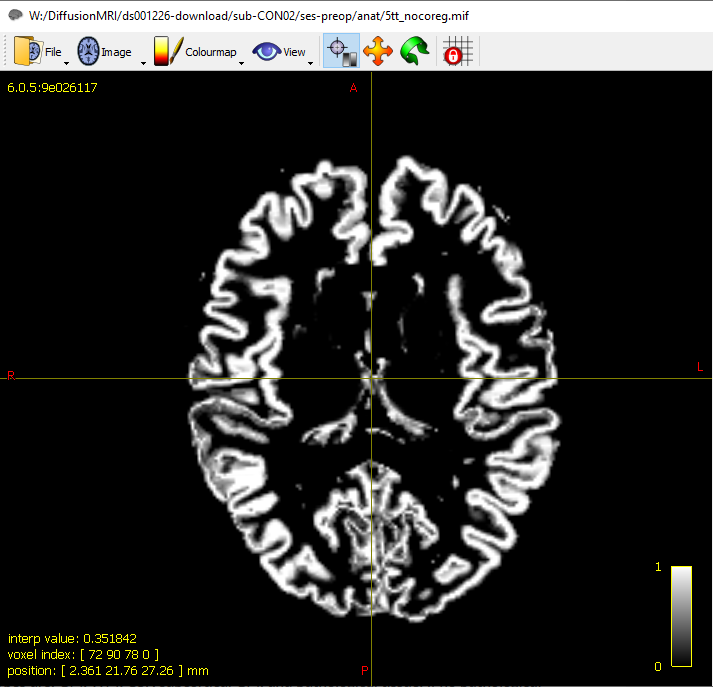
1. Cortical grey matter
2. Sub-cortical grey matter
3. White matter
4. CSF
5. Pathological tissue

5ttgen fsl T1.mif 5tt\_nocoreg.mif

To look at the results:

mrview 5tt\_nocoreg.mif

If no pathological tissue is detected, the final volume will be blank.



### Image registration

MRtrix developers recommend registering the T1-contrast anatomical image to the diffusion image series before any further processing of the T1 image is performed. By registering the T1 image to the diffusion series, rather than the other way around, reorientation of the diffusion gradient table is not necessary; and by doing this registration before subsequent T1 processing, any subsequent images derived from the T1 are inherently aligned with the diffusion image series. This registration should be rigid-body only; if the DWI distortion correction is effective, a higher-order registration is likely to only introduce errors. MRtrix does not have any inbuilt image registration tools, so we will use FSL’s FLIRT to achieve this.

The first step is to extract the b=0 images from the pre-processed DWI data and create an average:

dwiextract DWI\_preproc\_unbiased.mif - -bzero | mrmath – mean mean\_b0.mif -axis 3

Next, we need to convert mean\_b0.mif back into nifti format, required for FSL tools:

mrconvert b0\_mean.mif b0\_mean.nii.gz

We now use FSL’s FLIRT for the registration step:

flirt –ref mean\_b0.nii.gz –in T1w.nii.gz –dof 6 –omat anat2diff\_flirt.mat

Next we use the transformconvert command to transfer the transformation matric back to MRtrix:

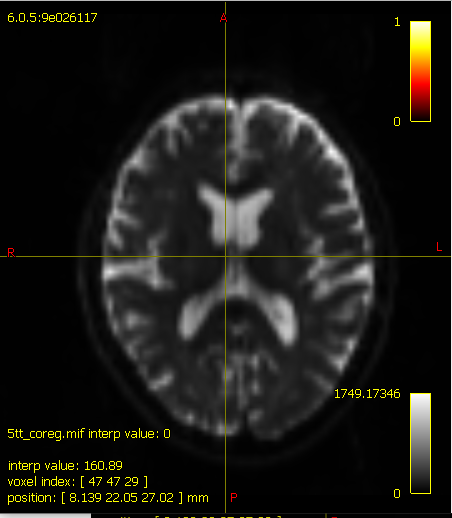
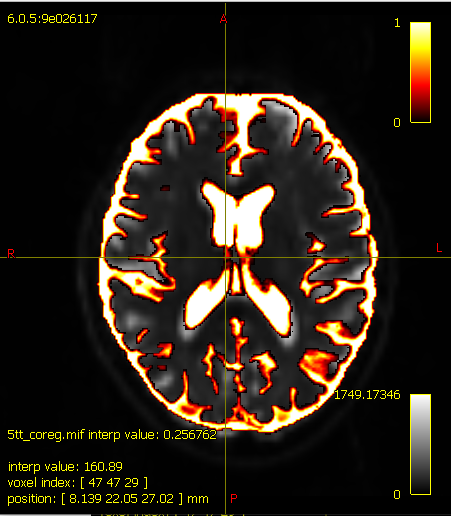
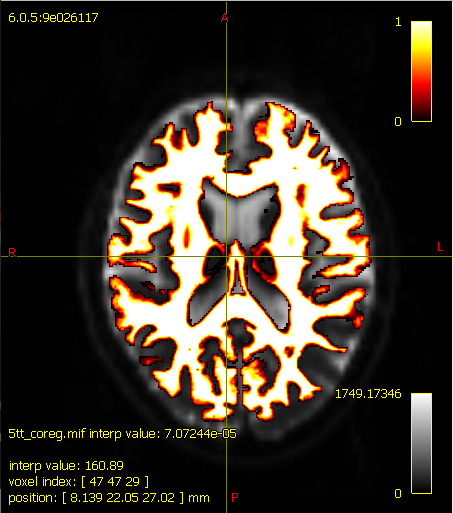
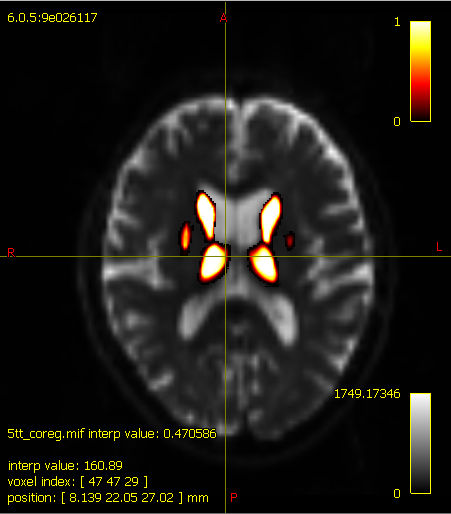
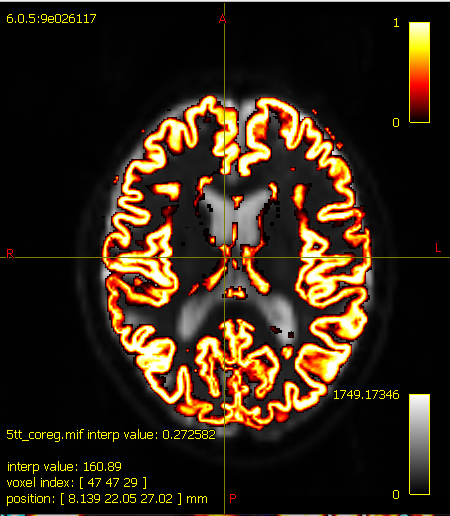
transformconvert anat2diff\_flirt.mat T1w.nii.gz mean\_b0.nii.gz flirt\_import anat2diff\_mrtrix.txt

and finally we apply the transformation to 5tt\_nocoreg.mif:

mrtransform 5tt\_nocoreg.mif –linear diff2anat\_mrtrix.txt 5tt\_coreg.mif

To check the results we can use mrview:

mrview DWI\_preproc\_unbiased.mif -overlay.load 5tt\_coreg.mif



The following binaries are provided for working with the 5TT format:

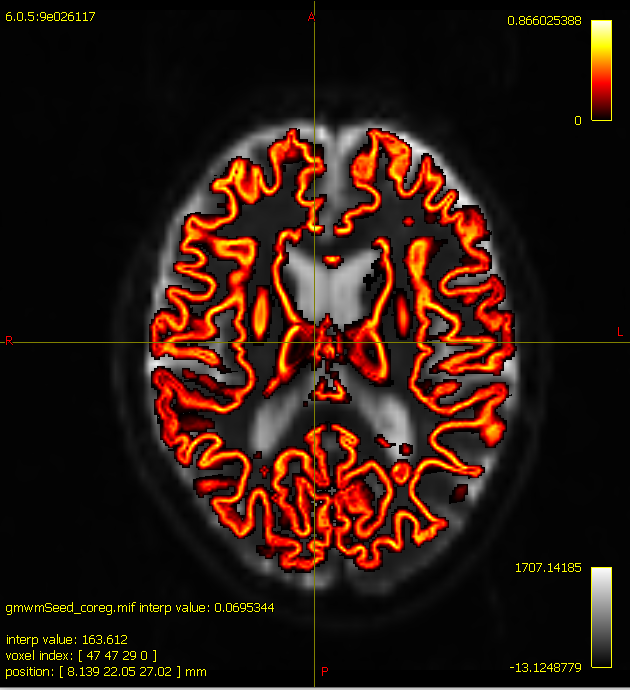
* 5tt2gmwmi: Produces a mask image suitable for seeding streamlines from the grey matter - white matter interface (GMWMI).
* 5tt2vis: Produces a 3D greyscale image suitable for visualisation purposes.
* 5ttcheck: Check that one or more input images conform to the 5TT format.
* 5ttedit: Allows the user to edit the tissue segmentations. Useful for manually correcting tissue segmentations that are known to be erroneous (e.g. dark blobs in the white matter being labelled as grey matter); see the command’s help page for more details.

For the next step, we will use 5tt2gmwmi to generate the GM/WM boundary mask which will be used to seed the streamlines:

5tt2gmwmi 5tt\_coreg.mif gmwmSeed\_coreg.mif

And we can check the result with mrview:

mrview DWI\_preproc\_unbiased.mif -overlay.load gmwmSeed\_coreg.mif



## Using ACT

Now that the necessary pre-processing steps are completed, using ACT is simple: just provide the tissue-segmented image to the tckgen command using the -act option. For example:

tckgen -act 5tt\_coreg.mif -backtrack -seed\_gmwmi gmwmSeed\_coreg.mif -nthreads 8 -maxlength 250 -cutoff 0.06 -select 10000000 wmfod\_norm.mif tracks\_10M.tck

This command will generate 10 million streamlines, using an algorithm known as iFOD2 (see below), which will use a probabilistic streamline approach. This is the default for MRTrix, but other algorithms can be found at [this site](https://mrtrix.readthedocs.io/en/latest/reference/commands/tckgen.html).

**iFOD2 (default):** Second-order Integration over Fiber Orientation Distributions. A probabilistic algorithm that takes as input a Fiber Orientation Distribution (FOD) image represented in the Spherical Harmonic (SH) basis. Candidate streamline paths (based on short curved “arcs”) are drawn, and the underlying (trilinear-interpolated) FOD amplitudes along those arcs are sampled. A streamline is more probable to follow a path where the FOD amplitudes along that path are large; but it may also rarely traverse orientations where the FOD amplitudes are small, as long as the amplitude remains above the FOD amplitude threshold along the entire path.

Below you can find more information about the options used with the tckgen command:

**-act image** use the Anatomically-Constrained Tractography framework during tracking; provided image must be in the 5TT (five-tissue-type) format.

**-backtrack** allow tracks to be truncated and re-tracked if a poor structural termination is encountered.

**-seed\_gmwmi image** seed from the grey matter - white matter interface (only valid if using ACT framework). Input image should be a 3D seeding volume; seeds drawn within this image will be optimised to the interface using the 5TT image provided using the -act option.

**-nthreads number** use this number of threads in multi-threaded applications (set to 0 to disable multi-threading).

**-maxlength value** set the maximum length of any track in mm (default: 100 x voxelsize).

**-cutoff value** set the FOD amplitude cutoff for terminating tracks (default: 0.1 for FOD-based algorithms; threshold multiplied by 0.5 when using ACT).

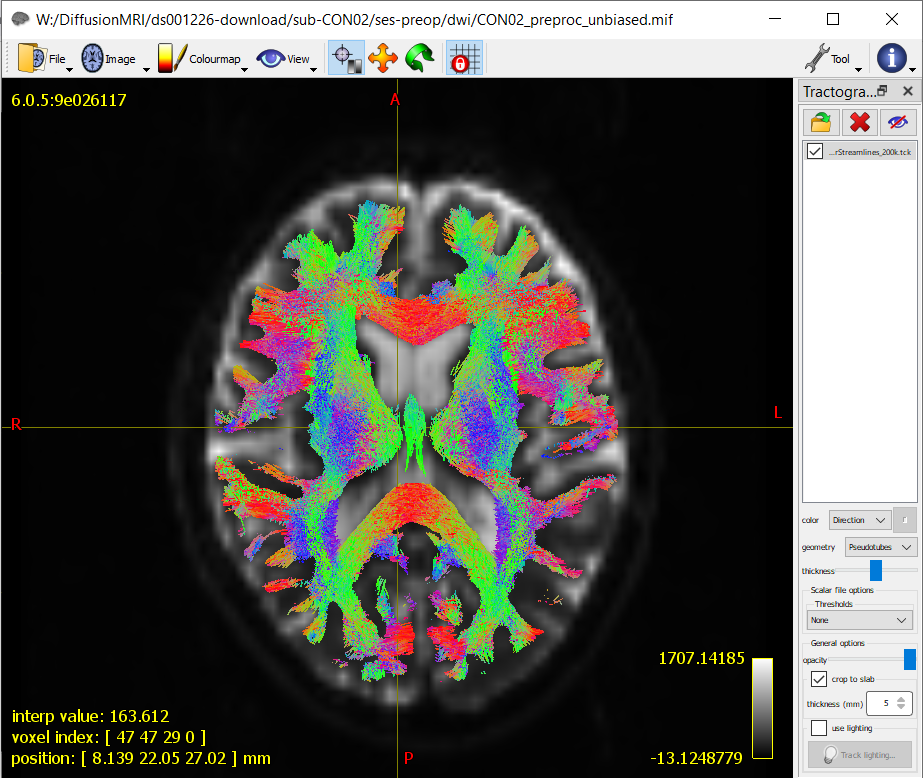
**-select number** set the desired number of streamlines to be selected by tckgen, after all selection criteria have been applied (i.e. inclusion/exclusion ROIs, min/max length, etc). tckgen will keep seeding streamlines until this number of streamlines have been selected, or the maximum allowed number of seeds has been exceeded (see -seeds option). By default, 5000 streamlines are to be selected. Set to zero to disable, which will result in streamlines being seeded until the number specified by -seeds has been reached.

If you want to visualize the output, first you should extract a subset of streamlines by using tckedit:

tckedit tracks\_10M.tck -number 200k smallerStreamlines\_200k.tck

This can then be loaded into mrview:

mrview DWI\_preproc\_unbiased.mif -tractography.load smallerStreamlines\_200k.tck



Finally, we will need to calculate weights to counterbalance the overfitting of some WM tracts. This is because some tracts will be threaded with more streamlines than others, due to differences in orientation densities. This results in certain tracts being over-represented by the amount of streamlines that pass through them not necessarily because they contain more fibers, but because the fibers tend to all be orientated in the same direction.

To counter-balance this overfitting, the command tcksift2 will create a text file containing weights for each voxel in the brain:

tcksift2 -act 5tt\_coreg.mif -out\_mu sift\_mu.txt -out\_coeffs sift\_coeffs.txt -nthreads 8 tracks\_10M.tck wmfod\_norm.mif sift\_10M.txt

**Creating the Connectome**

The next step is to generate a connectivity matrix, or **connectome**, showing the number of streamlines connecting different areas of the brain. First, we need to parcellate the brain into regions of interest (ROIs), and the most common method is to use a brain atlas. A common approach for this is to use one of the atlases included in **Freesurfer**. As discussed in the structural MRI tutorial, the first step is to run recon-all (please remember to set the environment variable SUBJECTS\_DIR to your desired output directory):

recon-all –i T1w.nii.gz –s sub01\_recon - all

To create the connectome we will use the aparc+aseg.mgz generated by freesurfer recon-all, which includes the parcellated cortical ribbon at the same time as the segmented subcortical structures uses the Desikan-Killiany atlas.



When recon-all has finished, we will need to convert the labels of the FreeSurfer parcellation to a format that MRtrix understands. The command labelconvert will use the parcellation and segmentation output of FreeSurfer to create a new parcellated file in .mif format:

labelconvert sub01\_recon/mri/aparc+aseg.mgz $FREESURFER\_HOME/FreeSurferColorLUT.txt /usr/local/mrtrix3/share/mrtrix3/labelconvert/fs\_default.txt sub01\_parcels.mif

We then need to create a whole-brain connectome, representing the streamlines between each parcellation pair in the atlas (in this case, 84x84). The “symmetric” option will make the lower diagonal the same as the upper diagonal, and the “scale\_invnodevol” option will scale the connectome by the inverse of the size of the node:

tck2connectome -symmetric -zero\_diagonal -scale\_invnodevol -tck\_weights\_in sift\_10M.txt tracks\_10M.tck sub01\_parcels.mif sub01\_parcels.csv -out\_assignment assignments\_sub01\_parcels.csv

**Viewing the Connectome**

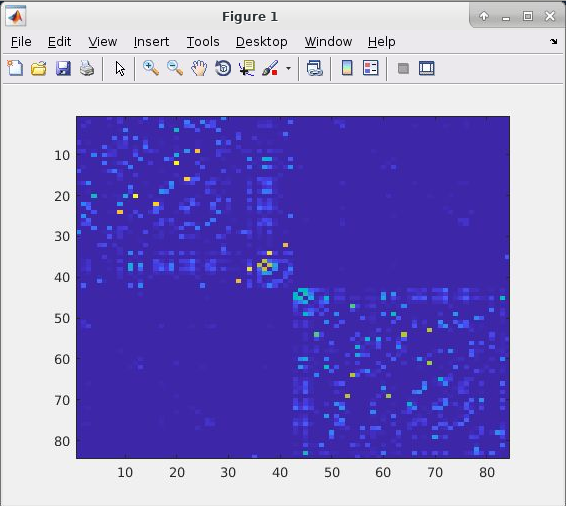
Once you have created the parcels.csv file, you can view it as a matrix in Matlab. First, you will need to import it:

connectome = importdata('sub01\_parcels.csv');

And then you will need to view it as a scaled image, so that higher structural connectivity pairs are brighter:

imagesc(connectome)

You should see an image like this:



The most noticeable feature is a division of the figure into two distinct “boxes”, representing increased structural connectivity within each hemisphere. You will also observe a relatively brighter line traced along the diagonal, representing higher structural connectivity between nearby nodes. The brighter areas in the opposing bottom-left and upper-right corners of the two hemisphere “boxes” represent increased structural connectivity between the subcortical regions.

To make these associations more obvious, you can change the scaling of the colour map:

imagesc(connectome, [0 1])

