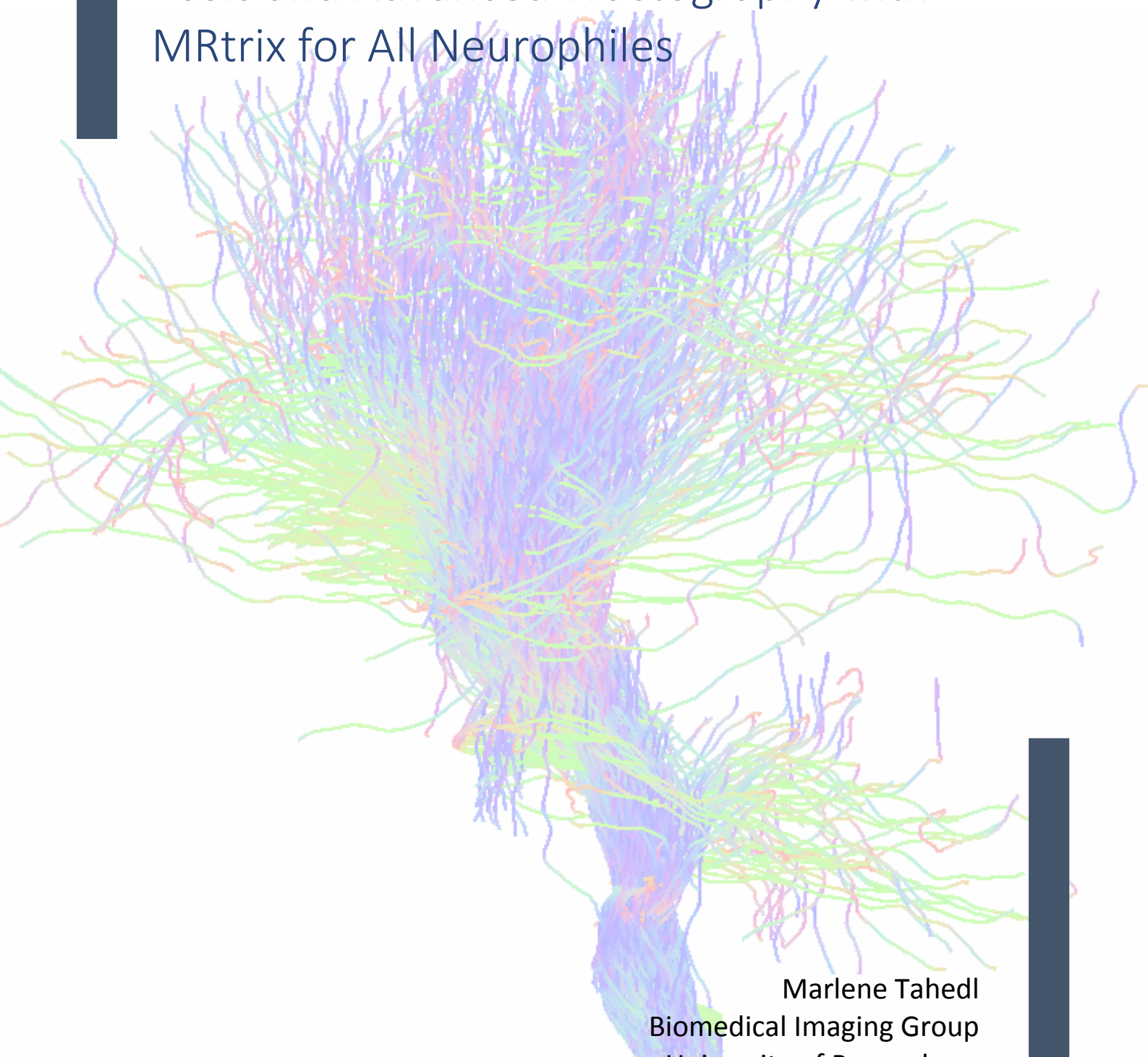Beyond the tensor model

BATMAN

Basic and Advanced Tractography with MRtrix for All Neurophiles

Marlene Tahedl
Biomedical Imaging Group
University of Regensburg

1   General tutorial info

1.1   About the tutorial

Welcome to *BATMAN* – Basic and Advanced Tractography with MRtrix for All Neurophiles! If you want to learn about the latest approaches in diffusion weighted imaging (DWI), this is where you want to be ☺. Most students and researchers that have been working with DWI and/or tractography will likely have worked with diffusion tensor imaging (DTI) (Basser et al., 1994) to estimate how nerve fibers are oriented within the brain. However, the diffusion tensor model, on which DTI is based on, does not perform well in brain regions containing crossing or kissing (i.e. tangentially touching) fibers. The reason for this is that the tensor model approaches fiber orientation with an ellipsoid shape. In a region where fibers cross, the orientation estimation of the tensor model will approach a sphere and thus cannot capture the orientation of two separate fibers. This is an especially severe problem when considering that up to 90% of all brain voxels contain crossing fibers (Jeurissen et al., 2013). Several strategies have been suggested to deal with this problem, and most recently, the development of Constrained Spherical Deconvolution (CSD) (Tournier et al., 2004, 2007) has raised the attention of the field, since it evidently outperforms DTI or other alternatives in regions of crossing-fibers (Farquharson et al., 2013; Tournier et al., 2008).

Based on the improvements CSD has introduced to tractography, the main drivers of this method developed an easy-to-use and freely-available software to perform CSD-based tractography: MRtrix. Following the success of CSD, the MRtrix developers have put forth more algorithms to improving the biological plausibility of fiber tracking: These algorithms include Anatomically Constrained Tractography (ACT) (Smith et al., 2012), which rejects such streamlines that end in biologically implausible tissue (e.g. the cerebrospinal fluid, CSF); spherical-deconvolution informed filtering of tractograms (SIFT) (Smith et al., 2013), which corrects for the fact that longer streamlines tend to be overestimated in tractography; and finally multi-shell multi-tissue CSD (MSMT) (Jeurissen et al., 2014), which – among others – improves tractography in voxels containing partial volumes by exploiting the differences in b-value sensitivity of different tissue types. In this tutorial, we aim to make the user familiar with how to implement all those algorithms into their own tractography processing pipeline. For that, we (hopefully!) provide easily understandable instructions for each step, with ready-to-run bash commands. We also provide a tutorial dataset, using state-of-the-art scanning parameters (Jeurissen et al., 2014; Tournier et al., 2013), which we are happy to share for the purpose of this tutorial. More information on the tutorial data can be found in the upcoming section. For the reader who's interested to learn more about the theory of diffusion-weighted imaging (DWI) and methods to improve tractography, we provide a list with suggested readings on the most important topics discussed in this tutorial in the appendix. Although we are providing very brief descriptions of each of those topics in the tutorial, by far we cannot cover all important technical details. Even though you should be able to complete the tutorial with only minimum knowledge on DWI, you will benefit more if you know more about the theory. So check out some of the suggested readings, or find different ones!

Another important thing is that throughout most of the tutorial, we will work in a bash terminal. Although we will provide all necessary commands, if the word terminal sounds unfamiliar to you, you might want to check out this webpage to learn about the very most basics on bash.

We designed the tutorial in such a way that no prior knowledge of MRtrix is necessary. As we walk you through the tutorial, you will hopefully get more and more familiar with MRtrix concepts, its command line usage as well as its graphical user interface (GUI). For further

information on the software and its usage, we provide a list of useful online resources to learn more. Other software requirements are discussed in an upcoming section.

OK – we're almost ready to get started! One more note: Please keep in mind to cite the original work if you are going to use any of the features discussed in this tutorial in your research. All software is freely-available, so keep in mind to honor the intellectual property of the scientists who enable that. For that, we once more refer you to the relevant sections of the tutorial, the list of further readings and the complete references list.

All right, you should now be familiar with the idea of this tutorial. Next, we'll take a look at the tutorial data and then we can hit the keys!

1.2 About the tutorial data

1.2.1 Tutorial data structure

The tutorial data comes in a directory called BATMAN/. In it, you will find three subfolders; two containing MR-data (DWI images and T1-high-resolution images), and one containing supplementary files. Within the DWI directory, you will find four more subdirectories, each containing dicom files from different DWI acquisition parameters. Lastly, in the Supplementary_Files directory, we provide any intermediate files that the user should create throughout this tutorial, such that he/she can compare his/her own results to the sample solution. Additionally, this serves as a backup in case one of the files can't be created for whatever reason, or if the computation would take too long. If necessary, just copy the respective files from the Supplementary_Files/ into the DWI/ directory, which is also where all analyses files will be created (if not otherwise specified). Furthermore, the Supplementary_Files/ directory contains a copy of this tutorial text, as well as a useful MATLAB function which helps to create and implement b-vector files suitable for SIEMENS scanners (check out the help page of that function to learn more). To summarize, you will find the following directory structure:

```
BATMAN
├── DWI
│   ├── b0_PA
│   ├── b1000_AP
│   ├── b2000_AP
│   └── b3000_AP
├── Supplementary_Files
└── T1
```

Note: If you have trouble to access the associated tutorial data, please don't hesitate to contact the author:
marlene.tahedl@stud.uni-regensburg.de

1.2.2 Data acquisition

All DWI images were acquired on a Siemens Prisma 3T MRI system equipped with a 64-channel receiver head coil and using a multi-shell acquisition scheme based on previous recommendations (Jeurissen et al., 2014; Tournier et al., 2013). Diffusion weighting of b=1000, 2000, and 3000 s/mm² were applied in 17, 31, and 50 directions, respectively. For each b-value, five images without diffusion-sensitizing gradients (i.e. "b0 images") were acquired. The phase-encoding direction of those data was anterior-posterior (AP). Additionally, three b = 0 s/mm² images were acquired with opposite phase encoding (i.e. PA) for the purpose of EPI distortion correction (Andersson et al., 2003; Holland et al., 2010; but we will also discuss this further in the tutorial during data preprocessing). To ensure uniform coverage across the shells, the gradient-encoding directions were generated based on a previously published repulsion model which takes into account multiple shells (Caruyer et al., 2013). Other DW

2   Getting started: Preprocessing

2.1   Preparatory steps

We recommend you define a project directory in which all analyses of this tutorial are carried out. This can be done by opening a terminal and typing, e.g.

```
tutorialDir=/Users/Marlene/MRI/projects/BATMAN
```

→ *Remember to adjust the path to wherever in your system you want to do this tutorial ;-)*

Now change into the `DWI/` directory of the tutorial data, e.g. by typing

```
cd $tutorialDir/DWI
```

and concatenate all b-images into MRtrix's-specific data format, the so-called `.mif` image ("**MR**trix **i**mage **f**ormat"; you can find more details on that data format [here](#)):

```
mrcat b1000_AP/ b2000_AP/ b3000_AP/ dwi_raw.mif
```

→ *A warning about mosaic slice timing will probably occur, but you can ignore this; the data is fine!*

2.2   Denoising

- ♦ Purpose: Estimate the spatially varying noise map
- ♦ Main reference(s): Veraart et al., 2016b, 2016a

```
dwidenoise dwi_raw.mif dwi_den.mif −noise noise.mif
```

→ *This step will take around 5–10 minutes to complete, depending on your hardware*

Inspect the results: Calculate the difference between the raw and the denoised images and mrview them. Additionally, also mrview the noise map (Figure 1):

```
mrcalc dwi_raw.mif dwi_den.mif −subtract residual.mif
mrview noise.mif residual.mif
```

→ *In the MRview-GUI, you can change between the displayed images with PageUp/Down*
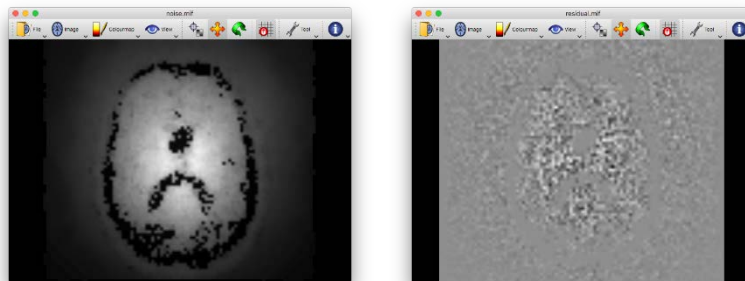
Figure 1. `noise.mif` (left) and `residual.mif` (right)

2.3   Unringing

- ♦ Purpose: remove Gibb's ringing artefacts
- ♦ Main reference(s): Kellner et al., 2016
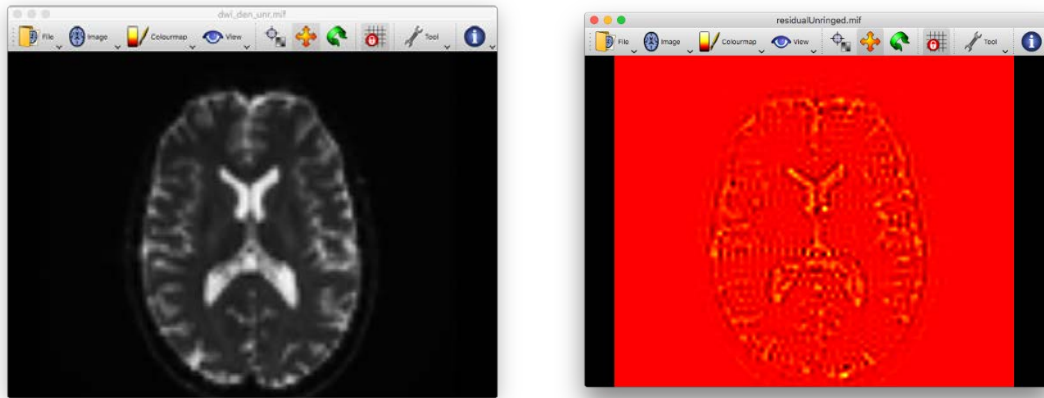
Figure 2. dwi_den_unr.mif (left) and the residual image between the denoised and the denoised plus unringed image (right). The colormap of the right image was changed to heat in the MRview GUI for better visibility

```
mrdegibbs dwi_den.mif dwi_den_unr.mif —axes 0,1
```

→ *The "axes" option must be adjusted to your dataset: With this option, you inform the algorithm of the plane in which you acquired your data: —axes 0,1 means you acquired axial slices; -axes 0,2 refers to coronal slices and —axes 1,2 to sagittal slices!*

Inspect the results. Calculate the difference between the denoised image and the unringed image. Mrview the resulting image. It's also always a good idea to inspect the DW-image, just to make sure everything is fine (Figure 2):

```
mrcalc dwi_den.mif dwi_den_unr.mif —subtract residualUnringed.mif
mrview dwi_den_unr.mif residualUnringed.mif
```

→ *As above, you can change between the displayed images with PageUp/Down!*

2.4   Motion and distortion correction

♦ Purpose: The purpose of this step should be pretty self-explanatory ;)
♦ Main reference(s):
  o EPI-distortion correction: Holland et al., 2010 (suggest using a pair of b0s in in phase encoding (PE) and reversed PE correction)
  o B0-field inhomogeneity correction: Andersson et al., 2003; Smith et al., 2004 (FSL's *topup* tool is called by MRtrix's preprocessing tool *dwipreproc*)
  o Eddy-current and movement distortion correction: Andersson and Sotiropoulos, 2016 (FSL's *eddy* tool is called by MRtrix's preprocessing tool *dwipreproc*)

For EPI distortion correction, we will use a pair of b0 images (Holland et al., 2010): One which was acquired in phase encoded (PE) direction (i.e. in the same direction as the non-b0 images) and one in the reversed PE direction. In the tutorial data, we acquired several b0-images in both PE directions. The purpose of this is to get a cleaner b0 for either direction by taking the mean. Therefore, we need to calculate the mean image for both directions. For the PE-direction (in our case anterior-posterior, AP), we will first extract all

b0-images from the `dwi_den_unr.mif` and then calculate the mean. In MRtrix, this can be done with one command:

```
dwiextract dwi_den_unr.mif - -bzero | mrmath – mean mean_b0_AP.mif
–axis 3
```

→ "-" denotes that the output will be piped as an input into the next command, which follows after "|". Like this, no additional output files are created!

→ "`-axis 3`" denotes that the mean image will be calculated along the third axis

Now calculate the mean image of the b0s in the reversed phase-encoded direction (in our case PA):

```
mrconvert b0_PA/ - | mrmath – mean mean_b0_PA.mif –axis 3
```

Further preprocessing requires to concatenate the two mean b0-images into one file. Note that here **order matters**: MRtrix expects the first image to be the b0 in PE direction, and the last to be the b0-image in reversed PE direction. We therefore type:

```
mrcat mean_b0_AP.mif mean_b0_PA.mif –axis 3 b0_pair.mif
```

To better understand the effects the b0-paired EPI-distortion correction, overlay the two images in mrview

```
mrview mean_b0_AP.mif –overlay.load mean_b0_PA.mif
```

→ In the GUI, change the colourmap of the main image via the colourmap menu from the top ribbon (see Figure 3). To change the colormap of the overlay, choose the tool menu from the top ribbon. An additional menu will appear where you can also adjust the opacity of the overlay image. It also helps to change the view to orthogonal (View menu → Ortho view). This allows you to better spot those locations where the two images do not overlap, and therefore those regions which will especially benefit from this type of EPI-distortion correction, improving later tractography.

Now we are ready to run motion and distortion correction. In MRtrix, both steps are carried out in one go, using the `dwipreproc` script. We select the `dwi_den_unr.mif` as input and will name the output `dwi_den_unr_preproc.mif`. We specifiy the PE direction via the –pe_dir option (in our case AP). For EPI inhomogeneity correction, we selected the b0-paired option (`-rpe_pair` option; for other options see the `dwipreproc` help page). We pass the b0_pair-file to the script (which will call FSL's *topup* tool) via the –se_epi option. Via the option –eddy_options, we can further adjust option on how MRtrix should call FSL's *eddy* tool. In our case, we select the option `–-slm=linear`, which will correct for the fact that in the two shells b=1000 and b=2000, sampling is moderately asymmetric due to few directions:
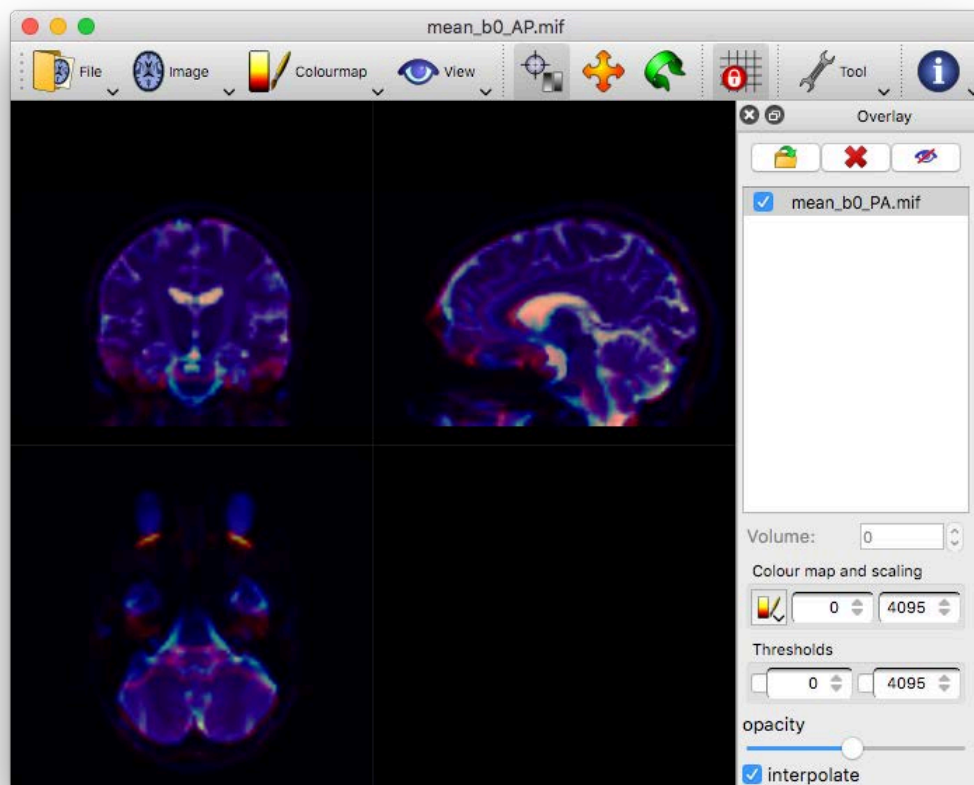
Figure 3. Overlay of the two mean b0 images: AP (cool) and PA (heat). Note that the overlay is especially poor in frontal, occipital, as well as deep cerebellar regions. In those regions, the b0-paired EPI distortion correction will especially improve tracking results.

```
dwipreproc dwi_den_unr.mif dwi_den_unr_preproc.mif –pe_dir AP –rpe_pair
–se_epi b0_pair.mif –eddy_options " --slm=linear"
```

→ *Note that the* eddy_options *are passed within double quotes. After the opening quote, an empty space must follow. Importantly, the options within the* eddy_options *are called with a double hyphen rather than with a single hyphen!*

→ *This command will take several hours to compute. You will best run this overnight. If you don't want to wait so long for the purpose of this tutorial, you can find the output of this command in the tutorial data, in the subfolder* Supplementary_Files/. *There, you'll find the output of the above command,* dwi_den_unr_preproc.mif. *Just copy this file into your* DWI/ *folder and continue with the next step.*

2.5   Bias field correction

♦   Purpose: Improve brain mask estimation

♦   Main reference: Tustison et al., 2010

This step is meant to improve brain mask estimation, which will be performed hereafter. However, if no strong bias fields are present in the data, running this script might actually *deteriorate* brain mask estimation and result in inferior brain mask estimation (for an example, see Figure 4A, left). In the case of the tutorial data, we *will* apply the bias field

correction, because it does not result in inferior brain mask estimation (Figure 4A, right), and should therefore improve brain mask estimation. In any case, you should always check your brain masks (we will estimate the brain mask in the next step). If you want to learn more about this script, check the MRtrix help of `dwibiascorrect`. Note that we recommend to use this script with the –ants option (Tustison et al., 2010) and for that ANTS must be installed on your system. Another useful option is to output the estimated bias field using the `-bias` option (Figure 4B). You can best display this with a heat colormap (gray command).

```
dwibiascorrect –ants dwi_den_unr_preproc.mif
dwi_den_unr_preproc_unbiased.mif –bias bias.mif
```

```
mrview bias.mif –colourmap 2
```
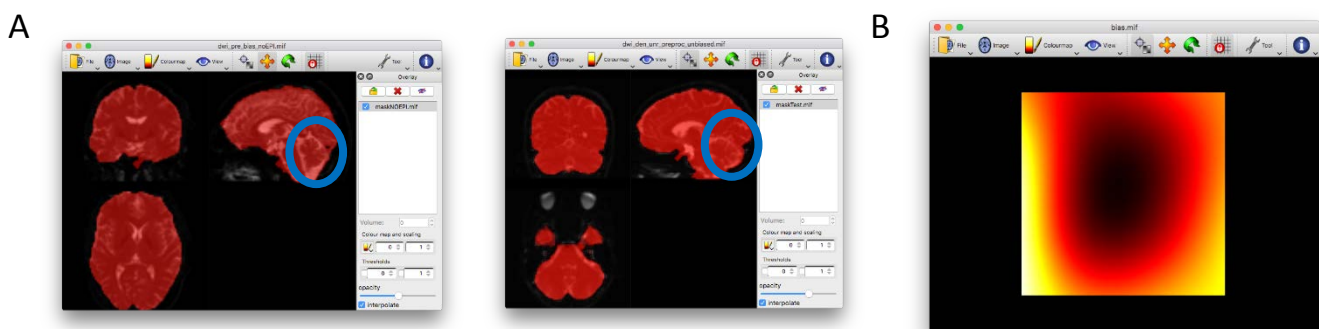

Figure 4. (A) Example of how running `dwibiascorrect` can potentially deteriorate brain mask estimation. In both images, a binary brain mask is overlaid in red on a biascorrected DW-image. In each case, the brain mask was estimated from the DW-biascorrected image. In the left image, the blue circle marks an area beneath the cerebellum which is clearly not part of the brain and should not be included in the brain mask. However, in case of that data set, running `dwibiascorrect` evoked this error (without running `dwibiascorrect`, this error did not occur; not shown). However, not in all cases `dwibiascorrect` results in flawed brainmask estimation (right). Here, the brain mask perfectly aligns at the borders of the cerebellum (blue circle). The take-home-message of this figure is that if you decide to include `dwibiascorrect` into your pipeline, you must always check subsequent brain mask estimation for any such potential errors. The right image shows the tutorial data. Since brain mask estimation is OK even when applying `dwibiascorrect`, we decided to include this step in our pipeline. (B) Estimated biasfield from the tutorial data, displayed with a heat colormap.

2.6   Brain mask estimation

♦   Purpose: Create a binary mask of the brain. Downstream analyses will be performed within that mask to improve biological plausibility of streamlines and reduce computation time.

```
dwi2mask dwi_den_unr_preproc_unbiased.mif
mask_den_unr_preproc_unb.mif
```

→   *Remember to check the brain mask in MRview. You can do this as outlined above (biasfield correction)*

3   Fiber orientation distribution

3.1   Response function estimation

♦ Purpose: Estimate different response functions for the three different tissue types: white matter (WM), gray matter (GM), and cerebrospinal fluid (CSF).
♦ Main reference: Dhollander et al., 2016

Figure 5. Voxels used for response function estimation of the different tissue types (RGB), displayed on the preprocessed DW-image. Red shows voxels used for CSF-response function estimation, blue GM and green WM.

Now we are finished with data preprocessing and can start to create streamlines. To get there, we first need to estimate the orientation of the fiber(s) in each voxel. As briefly outlined in the introduction, we will do this with constrained spherical deconvolution (CSD; Tournier et al., 2004, 2007) instead with the tensor model, which was shown to outperform the performance of DTI in regions of crossing/kissing fibers (Farquharson et al., 2013). To perform CSD, a so-called "response function" (RF) is necessary which is used as a kernel for deconvolution. For example, the RF in white matter models the signal which is expected if there was only a fiber bundle with one coherent orientation present in that voxel. However, there are many voxels that voxels with so-called "partial volumes", i.e. voxels containing both white and gray matter, or such with white matter and CSF. CSD will be flawed in such voxels. But we can improve our results in such regions if we estimate different RFs for different tissue types. This is best done with DW data with different b-values, since different tissue types are best sensitive for different b-values. This idea is at the core of so-called multi-shell multi-tissue CSD (MSMT; Jeurissen et al., 2014). We acquired the tutorial data with different b-values (0, 1000, 2000, and 3000) so that we can do MSMT. We will now continue straight with response function estimation, but check out the original article on MSMT (Jeurissen et al., 2014) to learn more!

```
dwi2response dhollander dwi_den_unr_preproc_unbiased.mif wm.txt
gm.txt csf.txt –voxels voxels.mif
```

→ *Note that the outputs of this command are order-dependent: The first output will always be the response function of white matter (`wm.txt`), the second of the gray matter (`gm.txt`) and the third of CSF (`csf.txt`)!*
→ *With the –voxels-option, you can output an image of those voxels which are selected for the response function estimation of each tissue type. We recommend to take a look at that image first (Figure 5). Type*

```
mrview dwi_den_unr_preproc_unbiased.mif -overlay.load voxels.mif
```

Next, we'll take a look at the response functions. MRtrix provides a special viewer for that, called with *shview*. To initialize it, type

```
shview wm.txt
```

to display the upper left corner in Figure 6. You're looking at the response function of the white matter in the first "shell", i.e. the angular resolution of white matter estimated from only b-values = 0 s/mm^2 (the "b0" images). The shape of this response function is a sphere, which denotes isotropic diffusion. However, in white matter, we would expect *an*isotropic
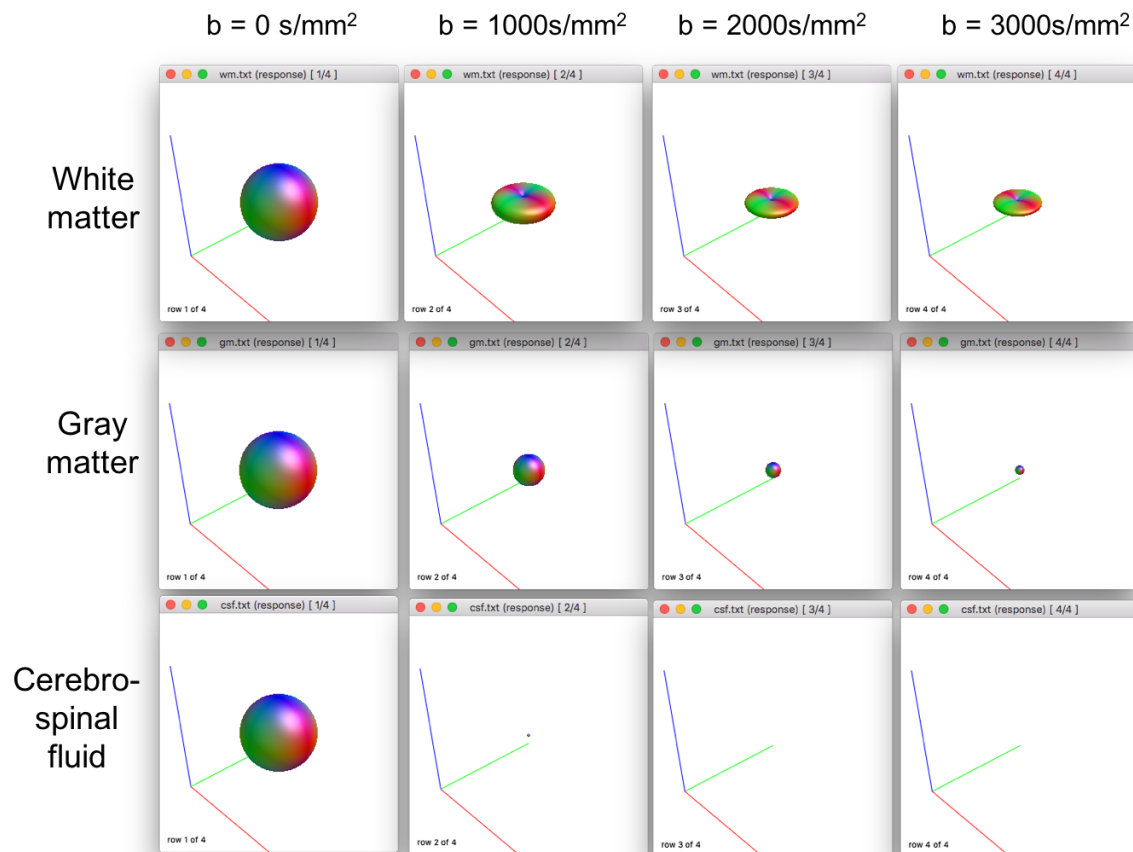
Figure 6. Tissue-dependent response functions estimated separately for different tissue types. For each tissue type, the response function was estimated from an average of the data of the voxels shown in Figure 5.

diffusion! So why do we see a spherical shape here? The answer is simple: Remember that we are only looking at the RF estimated from the *b0*-images, i.e. from those volumes which were acquired *without* gradient directions! Therefore, with b0 data, we *cannot* estimate any angular resolution from white matter. However, we *can* estimate such angular resolution from the data *with* the gradients switched on. If you still have the *shview*-GUI open, press the right arrow key. Now you're looking at the white matter RF *estimated from the next shell*, i.e. the b1000 data (Figure 6, first row, second column). This RF has a donut-like shape. This basically means that with b1000 data, we can already estimate angular resolution and therefore anisotropy of white matter. The angular resolution increases even further for the last two shells (i.e. b2000 and b3000), which you can inspect by pressing the right arrow key further (Figure 6, first row, third and fourth columns). Do you see how higher b-values flatten the RF further? The flatter the RF, the higher the angular resolution and ultimately, the smaller crossing-fiber-angles can be resolved. However, also note that the drawback of higher b-values is a lower signal-to-noise ratio (Dietrich et al., 2008). If you're interested in optimizing b-values for CSD, we recommend to take a look at this paper: Tournier et al., 2013.
Now close the *shview*-GUI and type

```
shview gm.txt
```

to display the shell-dependent response functions for gray matter. Again, push the right arrow key to inspect the four different response functions (Figure 6, second row). For gray matter, the shape of the RF is always spherical. This is because gray matter tissue is by nature isotropic

(i.e. water can diffuse freely and is not restricted). However, the amplitude varies: For larger b-values, you get smaller amplitudes. This means that the isotropy of gray matter can be most sensitively estimated with b0, but even with gradient directions (i.e. b1000/2000/3000), the algorithm still estimates GM to be isotropic, although at diminishing amplitudes.
Now also close that GUI and type

```
shview csf.txt   .
```

Again, you get four different RFs (Figure 6, third row). Like in gray matter, the shape of the RF is spherical. This is because just like gray matter, CSF is isotropic by nature. However, unlike gray matter, CSF RFs can almost only be estimated from the b0 data (first row). When the direction gradients are switched on (rows 2–4), the amplitude of the RF gets so small that it is almost invisible. This means that CSF isotropy is only estimated with b0. With that information, it is possible to differentiate gray matter from CSF: Although both are isotropic, they show different amplitudes for different b-values. With this information, the downstream fiber orientation distribution and tracking are enhanced for those voxels with partial volumes. This is another core idea of MSMT (Jeurissen et al., 2014).

3.2 Estimation of Fiber Orientation Distributions (FOD)
♦ Purpose: In every voxel, estimate the orientation of all fibers crossing that voxel.
♦ Main reference(s): Tournier et al., 2004, 2007

Having the different response functions for the different tissue-types at hand, it is possible to differentiate between those tissue types. We can now continue with the next step and can finally estimate the orientation distribution of the fibers in each voxel (FOD)! Run the following command to carry out FOD:

```
dwi2fod msmt_csd dwi_den_unr_preproc_unbiased.mif –mask
mask_den_unr_preproc_unb.mif wm.txt wmfod.mif gm.txt gmfod.mif csf.txt
csffod.mif
```

To check the results of FOD, we need two pieces of information: a) we need to know if FOD could accurately resolve crossing fibers, e.g. by inspecting FOD estimation in locations known to contain crossing fibers by anatomy and b) whether the estimation of white matter FODs (which is what we will ultimately need for downstream fiber tracking) was only performed within the borders of the white matter. Since – again – anatomically, fibers are only expected to occur in white matter, an accurate algorithm should not estimate white matter fibers in regions containing gray matter or CSF. One of the aims of MSMT is to differentiate between tissue types using information from different b-values, which was performed during response function estimation. Therefore, what we need is to display the white matter FOD on a map which shows the estimated volume fraction of each tissue type. We can do this with the following set of commands:

```
mrconvert –coord 3 0 wmfod.mif - | mrcat csffod.mif gmfod.mif – vf.mif
mrview vf.mif –odf.load_sh wmfod.mif
```

Figure 7 shows some results of this process. In that figure, we marked some locations which are suited for a quality check (see the highlighted areas of Figure 7).
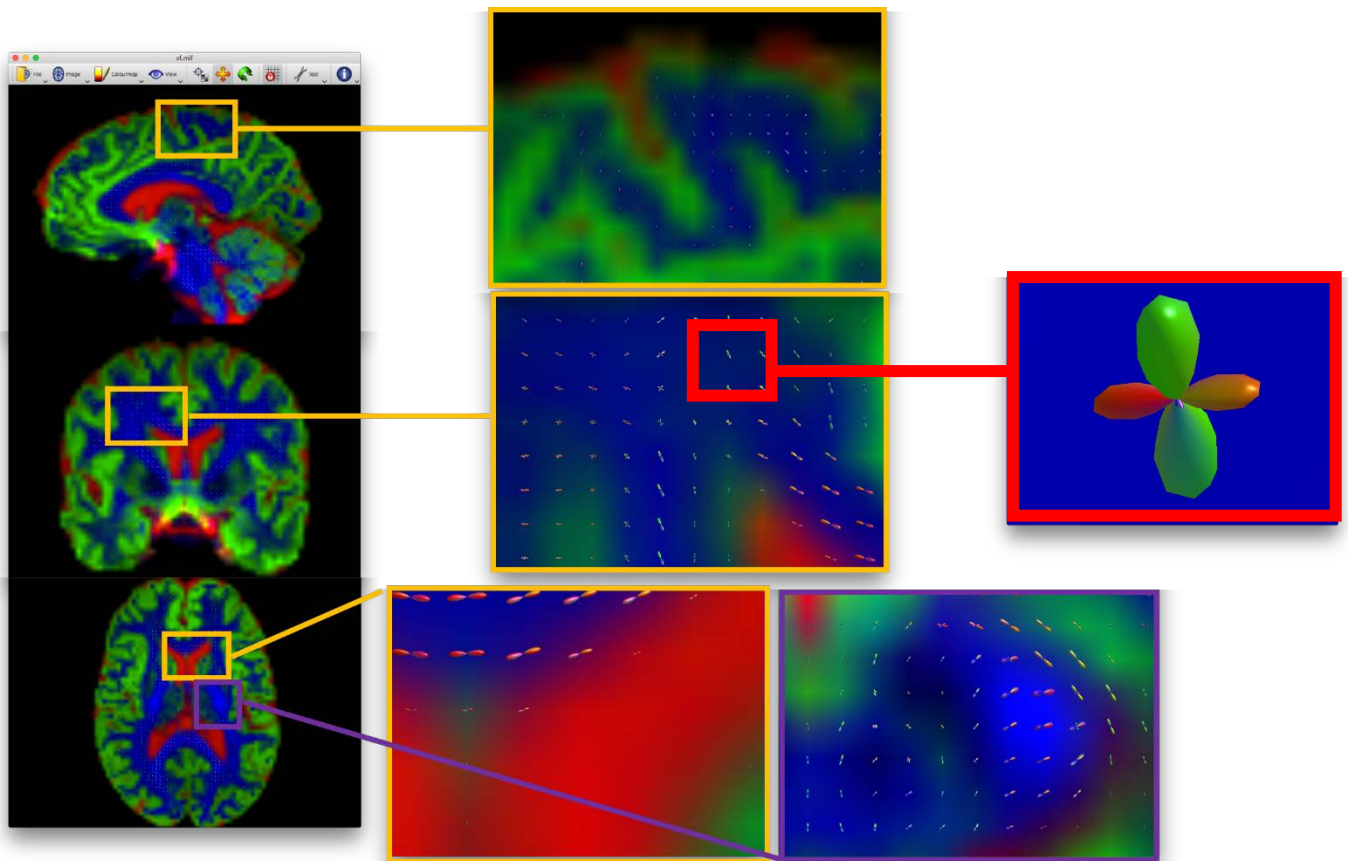
Figure 7. Quality check of Fiber Orientation Distribution (FOD) estimation. Red insert shows the FOD of a single voxel. Orange boxes show regions which contain crossing fibers by anatomy, and are therefore especially fit for quality check of FOD. Also, note that FOD is almost only performed within the white matter boundaries (blue areas). CSF (red areas) and gray matter (green areas) are free from FODs, which is consistent with anatomy! This is a result of multi-shell multi-tissue constrained spherical deconvolution!

3.3   Intensity Normalization

♦ Purpose: Correct for global intensity differences (especially important when performing group studies!)

When you have multiple subjects, this step helps to make the FODs comparable between your subjects, by performing a global intensity normalization. Although we only have a single dataset in the case of the tutorial, we will nevertheless carry it out so that you are prepared for your own group study!

```
mtnormalise wmfod.mif wmfod_norm.mif gmfod.mif gmfod_norm.mif csffod.mif
csffod_norm.mif —mask mask_den_unr_preproc_unb.mif
```

→ As a little exercise, you can try to check these results yourself: Proceed analogous to what we did in the previous step. First, concatenate the images to obtain a normalized volume fraction map (e.g. `vf_norm.mif`).   Then display the white matter normalized FODs (`wmfod_norm.mif`) onto that image. Is the quality of the normalized FODs maintained? To judge that, find the locations shown in Figure 7 and compare.

4   Creating a whole-brain tractogram

4.1   Preparing Anatomically Constrained Tractography (ACT)

♦ Purpose: Increase the biological plausibility of downstream streamline creation.
♦ Main reference(s): Smith et al., 2012

4.1.1   Preparing a mask for streamline termination

Having the FOD at hand, we could now go straight to creating streamlines. However, if we decide to include Anatomically Constrained Tractography (ACT) as an additional step into our pipeline, we can further increase the biological plausibility of streamline creation: When we will perform fiber tracking later on, we will use a probabilistic algorithm which is going to identify streamlines that end, for example, in CSF. From anatomy, however, we know that *such streamlines do not exist*! ACT rejects those (and other) streamlines. In order to use ACT, we need a high-resolution T1-weighted image. In the tutorial data set, you will find such data, as completely unpreprocessed dicom files, here: BATMAN/T1.

Now, we need to make the dicom data fit for ACT. This requires two steps: a) Preprocess the data and make them suitable for ACT. In MRtrix, this means to create a so-called "5tt-image", a 4-dimensional preprocessed T1-weighted image with **5** different **t**issue **t**ypes (cortical gray matter, subcortical gray matter, white matter, CSF and pathological tissue, which is usually just an empty volume – at least for the sake of this tutorial). B) Co-register that image to the DWI. While the first step is very easy in MRtrix, the second step is a little more tricky and requires the use of external software. In our example, we will use FSL to co-register that image. But let's start with the first step. Type:

```
mrconvert ../T1/ T1_raw.mif
5ttgen fsl T1_raw.mif 5tt_nocoreg.mif
```

→ *Note again that FSL must be installed on your system in order for this command to work.*
→ *This step can take up to half an hour to complete. If you don't want to wait that long, check the Supplementary_Files/ directory of your tutorial data and find the resultant file there!*

Check the output of that command (Figure 8):

```
mrview 5tt_nocoreg.mif
```

Now the T1 data is preprocessed and segmented into five tissue types as described above. However, since we plan to use this information for improving fiber tracking, i.e. on DW data, these two datasets must be co-registered. We strongly suggest to register the T1 to the DW data, and not the other way around, since DW data suffers from such transformations more than the T1 data does. Like this, our source image is the T1-data and the target image is the DW data. However, we cannot use the entire 4-D DW data as a target but we need a 3-D image. For that, the best-suited image is the b0, since it is not gradient-weighted. Since we acquired several b0 images, we will simply take the mean. Furthermore, we only need to perform registration with six degrees-of-freedom (DOF), since our target image (DW data) is already corrected for eddy currents, movements, etc. More DOFs will not be beneficial and might even introduce errors. The following set of commands take care of the described steps:
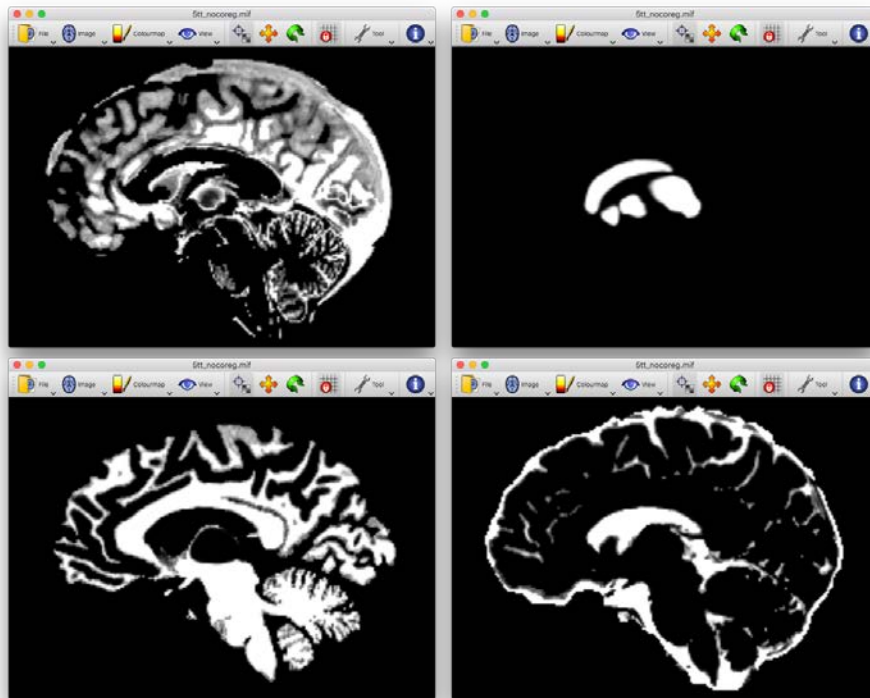
Figure 8. T1-highresolution anatomical image after running 5ttgen. The resulting image is a four-dimensional, fully preprocessed T1-weighted image. The fourth dimension has five elements, each corresponding to the segmentation of a different tissue type. Four of those tissue types are shown here: cortical gray matter (upper left), subcortical gray matter (upper right), white matter (lower left) and CSF (lower right). Note that the fifth element corresponds to pathological tissue and will be simply an empty volume for most analyses. For the sake of this tutorial, that fifth element can be ignored.

```
dwiextract dwi_den_unr_preproc_unbiased.mif - -bzero | mrmath – mean
mean_b0_preprocessed.mif –axis 3

mrconvert mean_b0_preprocessed.mif mean_b0_preprocessed.nii.gz
mrconvert 5tt_nocoreg.mif 5tt_nocoreg.nii.gz

flirt –in mean_b0_preprocessed.nii.gz –ref 5tt_nocoreg.nii.gz –interp
nearestneighbour –dof 6 –omat diff2struct_fsl.mat

transformconvert diff2struct_fsl.mat mean_b0_preprocessed.nii.gz
5tt_nocoreg.nii.gz flirt_import diff2struct_mrtrix.txt

mrtransform 5tt_nocoreg.mif –linear diff2struct_mrtrix.txt –inverse
5tt_coreg.mif
```

Check the results to see that the two datasets are now really aligned. Also, convince yourself that they were not realigned before the preprocessing! For that, load the DW image with both the co-registered and the native (i.e. before co-registration) 5tt-image (Figure 9):

```
mrview dwi_den_unr_preproc_unbiased.mif –overlay.load
5tt_nocoreg.mif –overlay.colourmap 2  -overlay.load 5tt_coreg.mif –
overlay.colourmap 1
```

→ *This command loads the preprocessed DW-image as a main image with two different overlays: The native 5tt will appear as an overlay in cool colors (colormap 2), the co-registered 5tt will have hot colors (colormap 1). This will give you an image like the one in Figure 9A.*

16
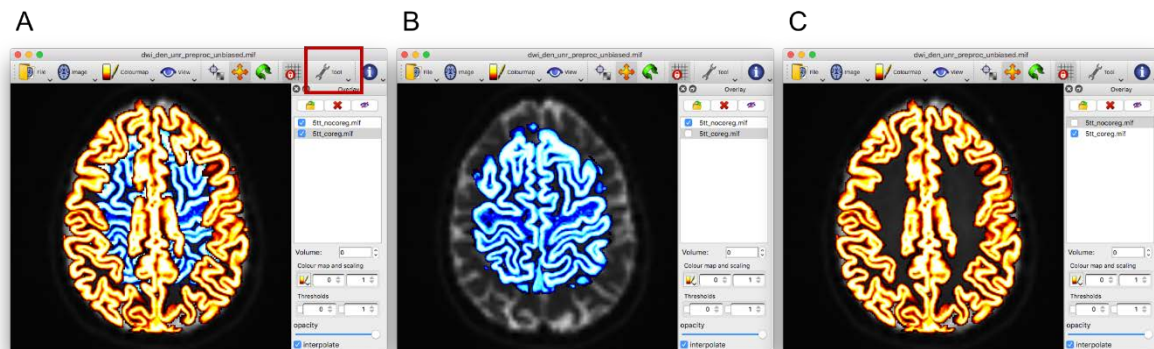
Figure 9. Checking the effects of co-registration. (B) After the 5-tissue-type segmentation is complete, the `5tt` (cool colors) and the DW image (main gray image) are not aligned. (C) After co-registration (hot colors), they match well. (A) shows both the co-registered and the native `5tt` simultaneously on the DWI. To watch them separately, choose "Overlay" from the Tool menu (red rectangle in A) to see the menu on the right. There, you can mark which overlays you want to display.

4.1.2   Preparing a mask of streamline seeding

We can improve the biological plausibility of our streamline creation even further. In the previous step, we have defined where our streamlines must not *end*. Now we will define where they should *start*! Again, from anatomy, we know that the gray-matter/white-matter-boundary should be a reasonable starting point for that. In MRtrix, it is very easy to create an mask of the gray-matter/white-matter-boundary, which we can later use as a mask for streamline seeding. Type:

```
5tt2gmwmi 5tt_coreg.mif gmwmSeed_coreg.mif      .
```

> → *One drawback of starting the streamlines at the gray-matter/white-matter-boundary is that streamlines between subcortical regions are hardly reconstructed. Therefore, if you are interested between such connections, you should create an additional streamlines file where you seed from a (binary) subcortical mask only, and combine both results. Alternatively, you can start the streamlines randomly within a whole-brain (binary) mas,. You will find details on streamline initialization on the* `tckgen` *help-page, and we will also discuss that command after this step.*
> → *Note that since we are creating the seed mask from the co-registered* `5tt`-*image, the resultant seed mask will be co-registered as well. You can check this by viewing the seed mask on the DW, e.g. by typing:*

```
mrview dwi_den_unr_preproc_unbiased.mif –overlay.load
gmwmSeed_coreg.mif
```

If everything went fine, you should now see something similar to Figure 10!

4.2   Creating streamlines

Ready for the big moment? Now we have everything set for an (almost) perfect streamline creation! The relevant command for that in MRtrix is `tckgen`. We will perform a "basic" streamline creation with mostly default parameters and few extra options. It's important to note that we will do *probabilistic* rather than deterministic tractography. However, the command has several options, and once you are familiar with the basic procedure you should definitely play around with those, too (e.g. you can specify which algorithm to use – if you do not specify an algorithm, the default is a probabilistic one)! Check the help page of that command for a complete list of `tckgen`'s options.
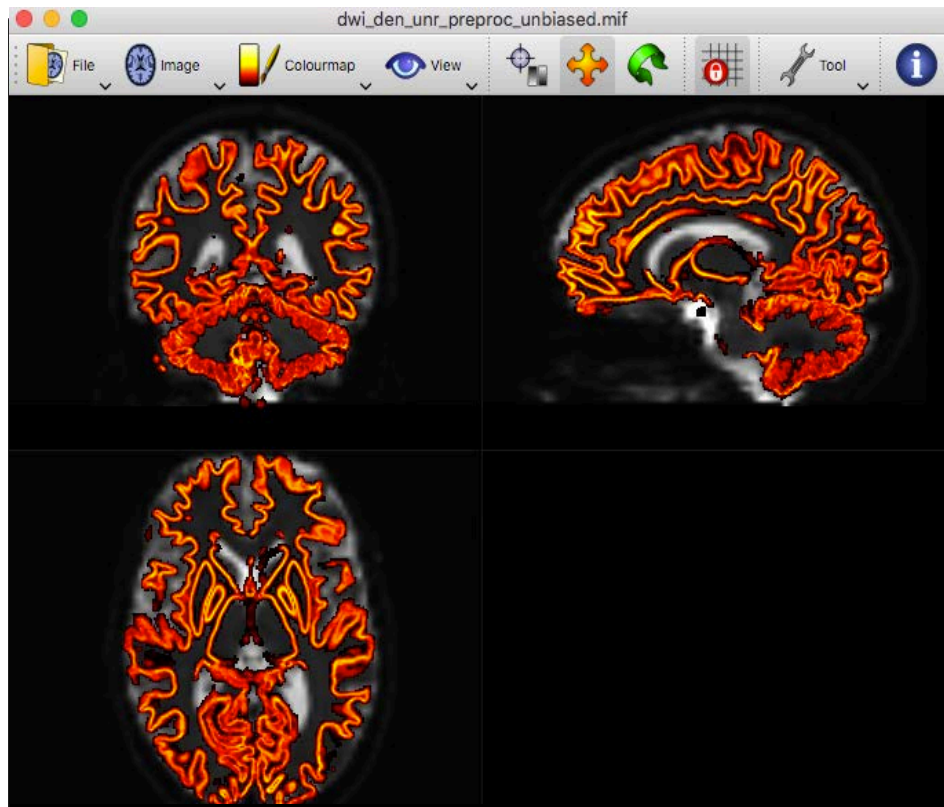
Figure 10. Checking the results of seed mask creation. If everything went fine, the seed mask (hot-colored overlay) should mark the area at the transition of gray and white matter on the DW image (main gray image).

Those options which we will make use of include the number of streamlines to be created, as well as the option which allows the use of ACT. Additionally, we will call "`-backtrack`" – an option specific to ACT and probabilistic fiber tracking, which basically resamples rejected streamlines from a defined number of steps back. You can find more about this in Smith et al. (2012).

Considering the number of streamlines, we chose to go big and create 10 million streamlines. This takes a while – but not as long as one might expect: a few hours (2–4 hours with an i5 processor) should be enough to finish this command. So now type

```
tckgen –act 5tt_coreg.mif –backtrack –seed_gmwmi gmwmSeed_coreg.mif
–select 10000000 wmfod_norm.mif tracks_10mio.tck
```

> → *The streamline tracking is based on the normalized fiber orientations of the white matter (`wmfod_norm.mif`)*
> → *Note that in MRtrix, files with streamlines have the ending `.tck` (for "tracks")*

Once the product is finished (or after you copied the data file from the `Supplementary_Files/` directory), you should check the results in MRview. For that, use the option –`tractography.load`. You *could* (but please hold in for a moment!) load the entire file on the DW data, e.g. by typing

```
mrview dwi_den_unr_preproc_unbiased.mif –tractography.load
tracks_10mio.tck
```

However, we strongly suggest **not** to do that! The `.tck` file is really big (> 4.1 GB!), and this will definitely challenge your RAM. We recommend to randomly choose a subset of the 10 million tracks. This can be done easily with `tckedit`:

```
tckedit tracks_10mio.tck –number 200k smallerTracks_200k.tck
```

→ *This command creates a random subsample of two hundred thousand tracks from the original 10-million-tracks image*
→ *Note that MRtrix understands "k" to be 1000*

Viewing that image should be no problem:

```
mrview dwi_den_unr_preproc_unbiased.mif –tractography.load
smallerTracks_200k.tck
```

What you see should look like Figure 11. Make sure to scroll through the volume and check the fit of your streamlines: Do they all end at the gray-matter/white-matter-boundary? If everything went well, they should, and this is a result of using ACT. Especially the sagittal slice of Figure 11 illustrates this: The streamlines wind perfectly around the (gray matter) gyri. This is exactly what we expect from anatomy: Streamlines should end as they enter the cortex.

Figure 11. Results of streamline creation, displayed as an overlay on the DW image. The tracks are color-coded (i.e. red indicates streamlines from right to left, green from anterior to posterior, blue from superior to inferior. Note that the streamlines terminate reliably at the gray-matter/white-matter-boundary. This is a result of ACT.

4.3   Reducing the number of streamlines

 ♦ Purpose: Filtering the tractograms to reduce CSD-based bias in overestimation of longer tracks compared to shorter tracks; reducing the number of streamlines
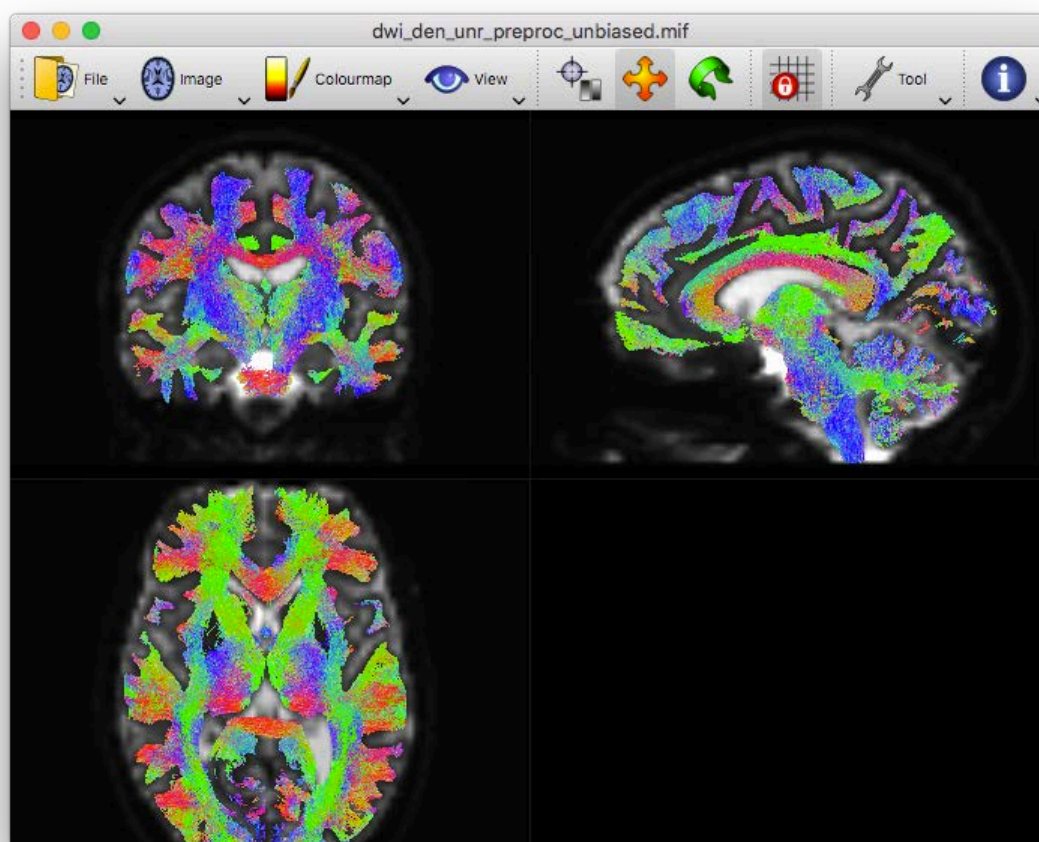 ♦ Main reference: Smith et al., 2013

Our track-file has one more problem: One of the pitfalls of CSD is that the density of long tracks is overestimated compared to short tracks. For example, imagine two tracks – a short and a long one – but *equally thick*: The long track will get more "hits" during probabilistic streamline creation, just because it is longer and the probability of a "hit" is more likely – as a result, the thickness of the tracks will be overestimated. Similarly, in regions of crossing fibers, most streamlines will follow the straightest path, thereby also generating a bias towards streamline density of that straight track. The first problem can be taken care of by seeding only from the gray-matter/white-matter-boundary, which is what we did during ACT. However, the second problem remains. Smith et al. (2013) suggest an algorithm that takes care of that bias, referred to as *spherical-deconvolution informed filtering of tracks* (SIFT). The name of that algorithm indicates that it filters tracks based on information drawn from CSD. To understand this, take another look at the red insert in Figure 7. There, you see the fiber orientation distribution in a single voxel. The volume of each of those lobes gives a relative estimation of the volume of the tract passing through each lobe. SIFT uses those relative estimations and filters the tractograms to match those proportions. Additionally, it reduces the size of the track-file significantly. However, one problem of SIFT is that there is no straightforward recommendation on how long to filter. While SIFT is running, every streamline removal reduces a cost function, which indicates how well the streamline density of the track-file fits the volume estimated from the fiber orientation distribution. The more the cost function is reduced, the better the fit – but at the same time, fewer streamlines mean less differential information. You might have to play around with different degrees of filtering to see what fits you best. Also, check the SIFT help page to get more information on the different filtering options. In this tutorial, we chose to reduce our 10-million-track-file by a factor of 10. Again, we can use the ACT option to improve biological plausibility. Type

```
tcksift —act 5tt_coreg.mif —term_number 1000000 tracks_10mio.tck
wmfod_norm.mif sift_1mio.tck
```

→ *Note that we must specify a FOD-file which SIFT can use to estimate the density of each tract. In our case, we use the normalized white-matter fiber orientation distribution (`wmfod_norm.mif`).*
→ *Possibly, a warning may pop up while SIFT is running, informing you that SIFT has reached a "quantisation error". This indicates that MRtrix thinks you filtered too heavily and that you might have to rethink your filtering options. Anyway, for the sake of this tutorial, you can ignore that warning.*
→ *Again, this command takes several hours to complete. Run it overnight, or copy the resultant file (`sift_1mio.tck`) from the `Supplementary_Files/` directory of the tutorial data!*

As always, remember to check the data. Again, do not display all the tracks at once but a random subset to spare RAM. Type

```
tckedit sift_1mio.tck —number 200k smallerSIFT_200k.tck

mrview dwi_den_unr_preproc_unbiased.mif —tractography.load
smallerSIFT_200k.tck
```

What you seed will look much like Figure 11, so we won't show it again here. The reason why both track-files look the same is partly due to that we displayed a subset of 200,000 tracks for both track-images. Although relatively small, it is still too much for our eyes to make out any substantial differences. Therefore, to understand and see what SIFT does, we need to choose an even smaller subset of both track-images and compare them. Try a subset of 10,000 tracks for both images and display them next to each other:

```
tckedit tracks_10mio.tck –number 10k superSmall_10k.tck

tckedit sift_1mio.tck –number 10k superSmall_sift_10k.tck

mrview dwi_den_unr_preproc_unbiased.mif –tractography.load
superSmall_10k.tck &

mrview dwi_den_unr_preproc_unbiased.mif –tractography.load
superSmall_sift_10k.tck
```

Figure 12 shows what you should see. Inspect the corpus callosum: Without SIFT (left), it appears really dense. After SIFT, it is much less dense. This tells us that without SIFT, that track is highly overestimated, since SIFT uses anatomically-based information (the FOD) to correct for any biases considering track density. Therefore, it is definitely a good idea to include SIFT for tractogram creation. This was the last step of tractogram construction. Now we'll delve into how we can do region-of interest-based analyses on that tractogram.

4.4   Region-of-interest filtering of tractograms

Although the whole-brain tractogram is nice to look at, for many analyses you will not want that much information but only a specific subset. For example, you might be interested in properties of the corticospinal tract (CST). Therefore, you need to create a track-file which exclusively contains streamlines of the CST. You can again use tckedit to do that. Tckedit has an option called –include, which filters a track-file such that only those streamlines crossing the specified region remain. You can either provide a binary mask as a region of
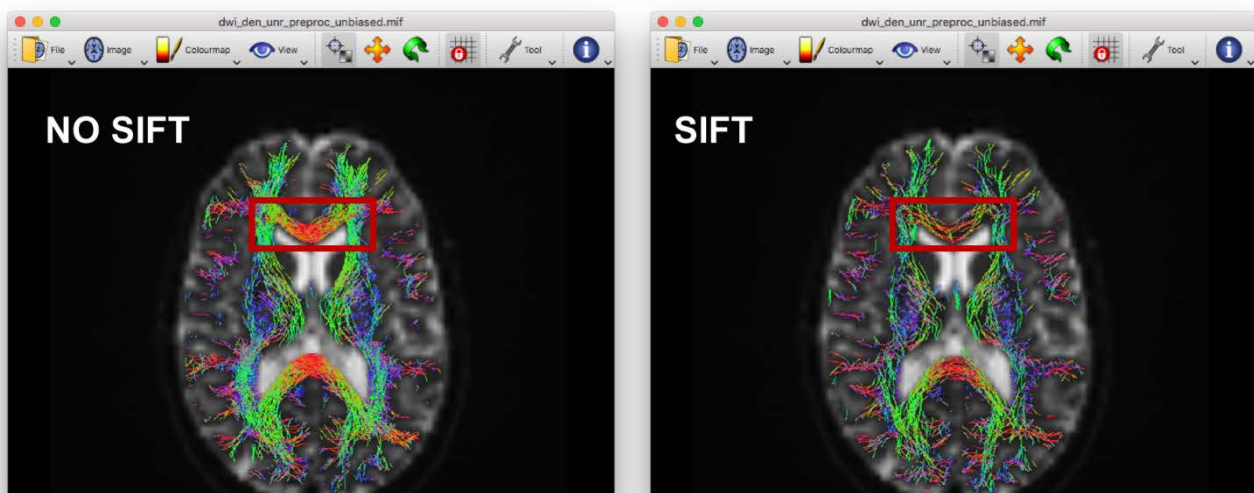

Figure 12. Visualizing the effects of SIFT. Both track images are a subset of ten thousand tracks randomly chosen from the original track file without SIFT (left) and with SIFT (right). Without SIFT, the corpus callosum (red box) is overestimated. SIFT corrects for that bias.

interest (ROI), or your ROI can be a coordinate (x, y, z) with a specific radius. Here, we will demonstrate the second option. At first, you need to identify the coordinates of a region within the CST. Load the filtered track-file on the DW image and find a suited spot (Figure 13A):

```
mrview dwi_den_unr_preproc_unbiased.mif –tractography.load
smallerSIFT_200k.tck &
```

→ *To select a ROI, it's good to see all 3 planes of the volume. Therefore, choose the "Ortho view" option from the "View" toolbar.*

At the bottom left corner, you can see both the voxel- and the real-world position (i.e. the position in millimeters) of the cursor. Note that the position information refers to the main image, i.e. in our case the DW image. However, for the ROI analysis we need to provide the position of the track-file, i.e. the overlaid image. Since the DW and the track-file are in the same space, also the real-world positions are the same. Tckedit therefore always expects coordinate positions in **real-word space**, not in voxel space! As can be seen in Figure 13A, the position [–0.6, –16.5, –16.0] is a good spot to filter CST tracks: This is right before where the CST divides into a right and a left part, so that we should get both sides of the CST equally well. An easy way to get your cursor in that position is by selecting "View Options" from the Tool menu and then entering the coordinates in the "Focus" section. The GUI on the right of Figure 13A will pop up. Just enter the coordinates into that GUI where it says "Positions" and there you go. Now that we have identified an adequate position, we can do the ROI-filtering. Do not close MRview yet, we will need it again in a moment. In the terminal, type

```
tckedit –include -0.6,-16.5,-16.0,3 sift_1mio.tck cst.tck
```

→ *The coordinates (x,y,z) have to be provided as a comma-separated list without spaces with four values: the fourth value indicates the radius of the sphere your ROI should have (in millimeters).*
→ *Of course, we filter on the entire SIFT-track-file (i.e. the one with 1 million tracks) rather than on the one we are displaying (200,000 tracks). The 200k-track-file is only there for making visualization more RAM-sparing!*

This will only take a few seconds to complete. Switch back to MRview and choose "Tractography" from the "Tool" menu. A GUI pops up which shows all track files that are currently loaded. At the moment, there is only one file, the smallerSIFT_200k.tck. Disable this by unselecting the tick. Now load the newly created CST-track file: Click on the folder symbol and choose the cst.tck file. Scroll around the volume (you can best do this by hitting the following icon from the main tool bar:) and try to get a good view of the track, e.g. like the one show in Figure 13B.

As you are scrolling through the volume, you will soon notice that it is difficult to judge the three-dimensional shape of the created CST-file on the two-dimensional planes. We will therefore switch to the volume render mode. Select the "Volume render" option from the "View" menu. Now you see the main image (i.e. the DW image) in its 3D shape. If you select the green horseshoe icon from the menu, you can rotate that volume around. However, right now you do not see the CST-track file any more since it is covered by the main image. We therefore need to "cut" that volume open. To do that, select "View options" from the "Tool" menu. A menu pops up, and in the lower half of that menu there is an option called "Clip planes". You can add as many planes as you want, in any of the three orientations (i.e. sagittal,
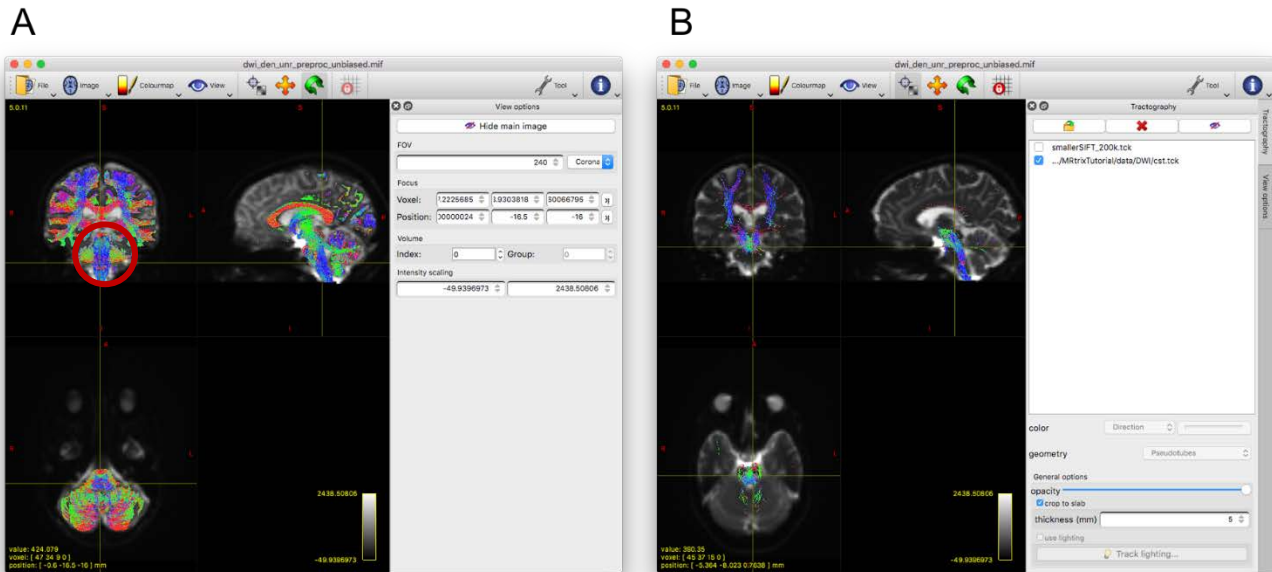
Figure 13. Region-of-interest-based (ROI) filtering of the whole-brain tractogram. (A) The first step is to find the coordinates in real-world (i.e. millimeter space) of the track-file which you want to filter on. The coordinates can be found at the lower left corner, where it says "Position". We selected a ROI right before the corticospinal tract divides into a right and a left branch (red circle). (B) After `tckedit` is completed, you can manually load the newly created track-file via the "Tractography" option from the "Tool" menu. Click on the folder-icon to load other track-files, and/or (dis)enable currently loaded track-files by (un)ticking them.

coronal and axial). Play around with this option. In Figure 14A we set three clip planes – one for each orientation. You can change the relative position of each plane by tagging that plane and then hit the arrow keys. Rotate the volume around to get an impression of what the CST-tracking looks like. Note that it is beautifully fan-shaped, and this is exactly what is expected from anatomy! Moreover, there are streamlines preserved which connect the left and the right CST – a feature which is often hard to recover because those are crossing fibers. This illustrates the strength of CSD again to resolve crossing fibers! If you want the main image to disappear, hit the "M" key (for **M**ain image), and hit it again to see it again (Figure 14B).

Before we finish the tutorial on tractogram construction, one more hint: Of course, you can also load all track-images onto the T1-image, which will give you a nicer anatomical contrast. For that, you need to get the T1-image (`T1_raw.mif`) in the same space as the track-image, i.e. in diffusion space. For that, we need a transformation matrix between diffusion and anatomical space. We already created such a matrix in a [previous step](), when we registered the `5tt`-image (which was derived from the T1 and is therefore in the same space) to diffusion space: `diff2struct_mrtrix.txt`. Try to register the `T1_raw.mif` to diffusion space now yourself to obtain a `T1_coreg.mif`. You can use the exact same `mrtransform` command as when we registered the `5tt` to diffusion space! This should work smoothly, but if you still have trouble finishing this step, you can find the `T1_coreg.mif` in the `Supplementary_Files/` directory. In any case, you should definitely view the track-files on the T1-image in volume render mode, and enjoy what you see ☺

5   Connectome construction

We have now dealt with the most important features of tractogram construction. As beautiful as such analyses are – what you will mostly need in the end to evaluate your data is quantitative information on the tracks, so-called "structural connectivity" (SC). A typical research question is, for example, which regions have reduced SC in a patient group compared
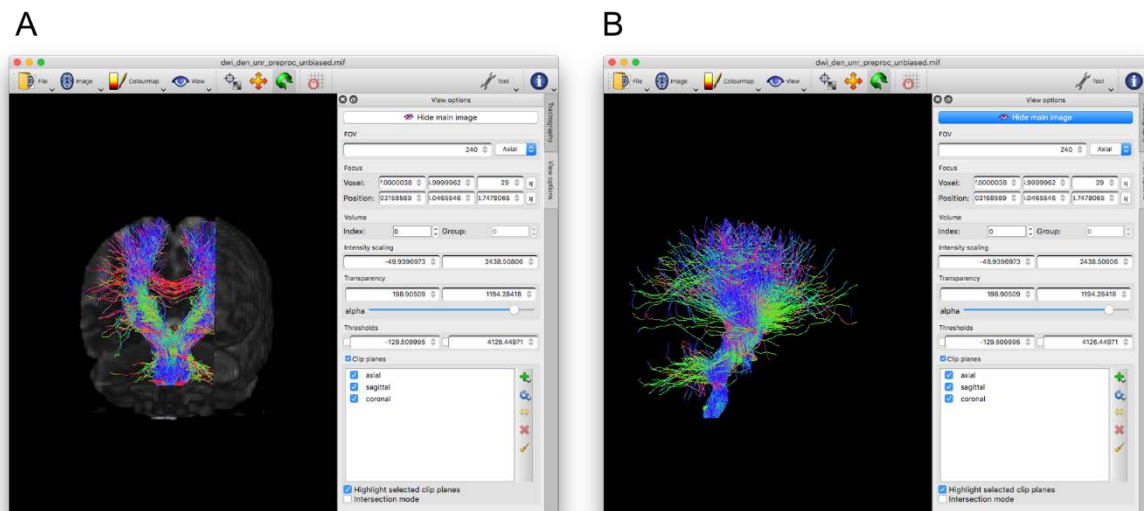
Figure 14. The volume render mode. To get there, first select the "Volume render" option from the "View" menu. Then, select the "View options" from the "Tool" menu to see the GUI which you see on the right menu in (A) and (B). In the lower half, note the option "Clip planes". Add some planes: In (A), we selected one axial, one sagittal and one coronal slice. To change the relative position of each slice, mark them on the menu and push arrows up/down/right/left, until you are happy with what you see. If you want the main image to disappear completely, hit the "M" key (B). If you hit "M" again, the main image will reappear. Note that (B) shows the same track-file as (A), but from a lateral view.

to healthy controls. A straightforward way to approach this question is to take an atlas and count how many tracks connect each atlas region to every other of the atlas regions. The result of this analysis is a SC matrix. This part of the tutorial is on how to obtain such a matrix in MRtrix and shows some nice features to visualize them.

5.1   Preparing an atlas for structural connectivity analysis

- ♦ Purpose: Obtain a volumetric atlas-based parcellation image, co-registered to diffusion space for downstream structural connectivity (SC) matrix generation
- ♦ Main reference: Glasser et al., 2016a (for the atlas used here for SC generation)

The first decision we have to make when doing SC analyses is which atlas to use. In this tutorial, we chose the recently published *Human Connectome Project Multi-Modal Parcellation 1.0* ("HCP MMP 1.0") (Glasser et al., 2016a). In this atlas, regional boundaries are drawn based on integrative information from cortical architecture (cortical thickness, myelin), function (task-fMRI), connectivity and/or topography (resting-state fMRI). If you are not familiar with that atlas, check out the original publication (see main reference 5.1) to learn more about how that atlas was created.

To construct an atlas-based SC matrix in MRtrix, you need a volumetric atlas-based parcellation image. Importantly, the integers assigned to each atlas region should start at 1 and increase by 1. Unfortunately, many atlases have more or less random integers assigned to their atlas regions. The atlas we chose is based on FreeSurfer processing, and it is not so straightforward to prepare a file which meets those requirements. In the appendix, we provide a detailed step-by-step instruction on how to get there. That instruction can also be modified to be used with other atlases. We recommend you try those steps yourself, but for the sake of this tutorial you can just copy the resulting file from the Supplementary_Files/ directory into the DWI/ directory. The relevant file is called hcpmmp1_parcels_coreg.mif.

5.2   Matrix generation

♦ Purpose: Gain quantitative information on how strongly each atlas region is connected to all others; represent it in matrix format

You can now use the atlas-based parcellation image to gain quantitative information on how strongly each of those atlas regions connects to all others. A straightforward way to determine that connection strength is the streamline count: The idea is to overlay the atlas-based parcellation onto the whole-brain tractogram and simply count how many streamlines from that region reach every other region. Since this metric is highly dependent on the number of streamlines in the track-file as well as the atlas region's volume (the bigger the volume, the more streamlines will are created from that region), it is a good idea to normalize those values. MRtrix offers different ways to scale the track count. Check out the help page for more details, and to identify the scaling option which fits your analysis best. Here, as outlined above, we encourage to scale by the atlas region's volume. Type:

```
tck2connectome –symmetric –zero_diagonal -scale_invnodevol
sift_1mio.tck hcpmmp1_parcels_coreg.mif hcpmmp1.csv –out_assignment
assignments_hcpmmp1.csv
```

→ *We like to use the two options "`-symmetric`" and "`-zero_diagonal`", which symmetrize the resulting SC matrix (if not specified, the lower triangular will be left blank) and set the diagonal to zero. We find that this makes subsequent manipulation of the matrix easier in external software, e.g. MATLAB.*
→ *We will need the output of the "`-out_assignment`" option in the next section. That option is not necessary for SC matrix generation, but for the moment just specify it and check out the next section to see what it is and what it is good for!*

This will only take a few seconds to complete. The resulting file is a `.csv` file, which contains a matrix in the size of the number of atlas regions squared. In our case, this is a 379x379 matrix: 180 atlas regions on each cortical hemisphere, based on the HCP MMP 1.0, plus 19 subcortical regions, based on the FreeSurfer segmentation (9x2 homologs plus brainstem). You can visualize this matrix now easily with your favorite software. Figure 15A shows how such a visualization can look like using MATLAB (The Mathworks, Natic USA). As you can see, the matrix looks pretty uniformly blue. That is because by default, the color-coding ranges between 0 and the maximum value of the matrix, even if the maximum value is an outlier. In structural connectivity, the word "outlier" is not quite appropriate, since the distribution of the element values typically follows the so-called power-law (Figure 15C and D): The histogram shows many small values and few extremely high values. Biologically, such a distribution makes sense for structural connectivity, because we know from anatomy that many cortical regions are not structurally connected at all, but there are a few regions which are extremely well connected. These are the reasons we can't make out much when looking at the SC matrix color-coded up to the maximum element value. If we reduce the limit of the matrix, the properties of the matrix are more evident (Figure 15B): First of all, we can see that intra-hemispherically, SC is relatively high (in Figure 15B, the upper-left red box marks intrahemispheric connections within the left hemisphere and the lower-right red box SC within the right hemisphere). Moreover, the bright main diagonal hits the eye (marked purple in Figure 15B), which denotes high SC between spatially close regions. Third, the last few rows/columns are relatively bright (green box): Remember that we ordered the atlas regions

in such a way that the last few regions encompass the subcortical segmentation. Therefore, this means that subcortical regions connect well to many cortical regions. Fourth, the minor diagonal (blue box in Figure 15B) stands out, although to a lesser degree than the other three properties. The minor diagonal shows the connection between the so-called "homolog" regions, i.e. corresponding regions on the right and left hemispheres, e.g. right and left premotor cortices. Also, from anatomy, we know that such regions really *are* structurally connected; many of those tracks use the corpus callosum to cross hemispheres. However, due to the crossing-fiber problem, detecting such connections has proven a problem with diffusion imaging, but as we can see in Figure 15B, our pipeline succeeds to do so. This property of the SC matrix is therefore an important quality control criterion for any SC analyses.
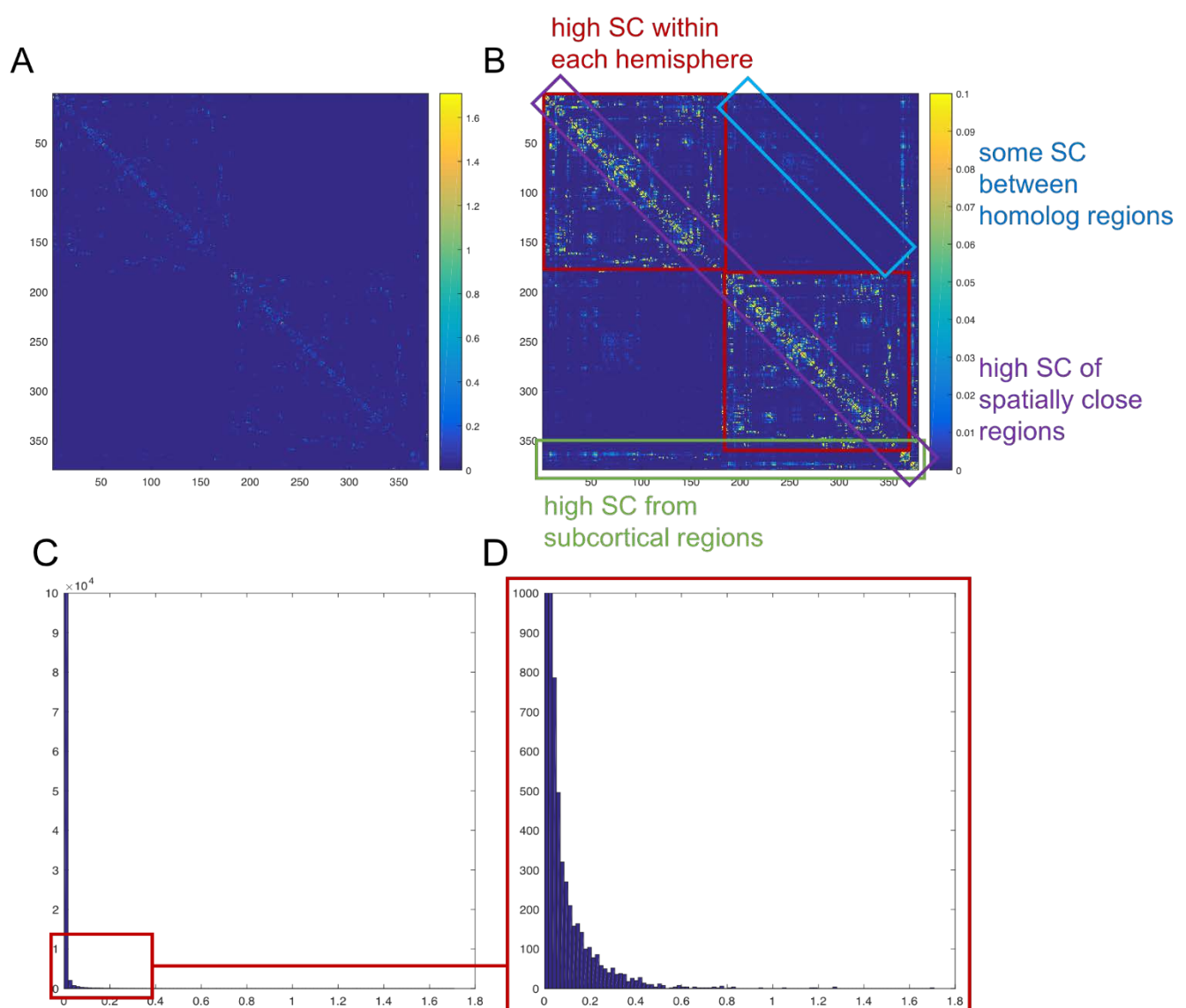
Figure 15. Structural connectivity (SC) matrix and its properties. (A) shows the SC matrix with the colorbar ranging from 0 to the highest element value. (B) shows the same matrix with a reduced colorbar. Now, more distinct features of the matrix are recognizable, such as increased intra- and interhemispheric SC, high SC of subcortical regions and high SC between spatially close regions (colored boxes). (C) and (D) show histograms of all matrix elements' distribution. The distribution follows a power law with many zero-elements (C). (D) shows the same histogram with a reduced y-axis to enhance the distribution of the non-zero elements.

5.3   Selecting connections of interest

We will now look at how to select streamlines corresponding specific brain regions. We will look at two different strategies: a) Selecting all streamlines that connect two regions and b) selecting all streamlines that emerge from a region of interest (ROI). For both strategies, the function connectome2tck can be used. Check out the help page of this command before you continue.

5.3.1   Extracting streamlines between atlas regions

In the first case, the initial question to deal with is: *Which* are the regions that you are interested in? In this example, we chose to extract a homolog connection, since such regions should connect strongly, as we already found looking at the SC matrix in Figure 15. More specifically, we are interested in the motor cortex. Therefore, we have to identify those atlas regions that correspond to the left and right motor cortices. The HCP MMP 1.0 atlas, which we used in this tutorial, subdivides the motor cortices into several subregions (see the supplementary material of the original article: Glasser et al., 2016b). Looking at that article, we find that one core region of the motor cortex is simply called "4". Since the atlas is symmetric across hemispheres, there is a region "4" on the left and the right hemisphere, called "L_4" and "R_4" respectively. We next have to identify the indices assigned to those regions in our atlas-based parcellation image (hcpmmp1_parcels_coreg.mif). To do that, open the color lookup table (the ordered version: hcpmmp1_ordered.txt). Look for regions "L_4" and "R_4". You will see that they correspond to indices 8 and 188.

To identify all tracts that connect those regions, MRtrix needs the track-file and information on every single streamline about its connecting regions. This is a text file with two columns and as many rows as you have streamlines. In our case, this means one million rows. The first column specifies the index of the starting regions and the second column the index of the ending region. This file can be easily created with the function tck2connectome. Remember that when we ran that command in the previous section, we specified an option called –out_assignments, which we said we would need later. This is exactly the option that creates such a text file. Now we have all information that we need. We can now type:

```
connectome2tck –nodes 8,188 –exclusive sift_1mio.tck
assignments_hcpmmp1.csv moto
```

→ *With the option* -exclusive*, you specify that you only want to select tracks between the two regions*
→ *The last argument specifies the prefix that the resultant file will have. We chose "moto", since we are analyzing the motor cortex. The complete file name includes by default also the node indices that you are analyzing, so that in our case we will get a file called "moto8-188.tck"*

Now look at your results. If you completed the earlier exercise and co-registered the T1-image to diffusion space, use that file as main image. If you did not do that exercise, you should either do it now or use the DW image as main image (however, the T1 will look nicer). Type

```
mrview T1_coreg.mif –tractography.load moto8-188.tck
```

and select the "Volume render" mode. Choose the "Clip planes" option as discussed above and try to make your image look like Figure 16A. Quite a nice track, isn't it? ☺ Also, in Figure 16A, we displayed the atlas regions of interest as yellow and orange overlays. This can be done with the following trick:
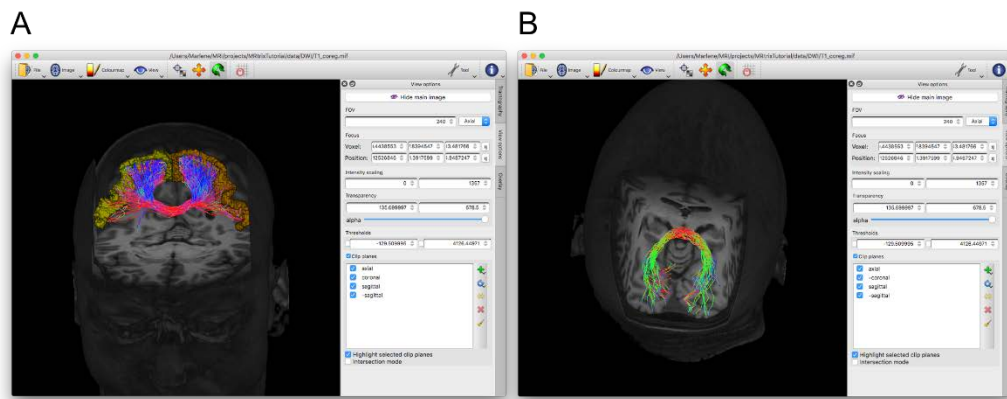
Figure 16. Selecting streamlines between two atlas regions. (A) shows all streamlines connecting homolog regions of the motor cortex. Additionally, the analyzed atlas regions are plotted as an overlay in yellow (right motor cortex) and orange (left motor cortex). (B) shows the streamlines connecting the left and right V1.

```
mrcalc hcpmmp1_parcels_coreg.mif 8 –eq L_4.mif

mrcalc hcpmmp1_parcels_coreg.mif 188 –eq R_4.mif
```

This extracts all voxels that have the index 8 (i.e. 188) from the parcellation image, which in our case corresponds to the left (i.e. the right) region "4" of the motor cortex. In MRview, the newly created files can be displayed via the "Overlay" option from the "Tool" menu.

If you want more practice, try to select all streamlines that connect the left and right V1 from the visual cortex! You can proceed analogously to our demonstration of area "4". Your result should look like what you see in Figure 16B.

One more useful hint. If you ever wish to merge two atlas regions, first extract each region individually via the `mrcalc –eq` option, as just explained. Then merge the regions, using e.g. `mrcalc roi_one.mif roi_two.mif –max merged_roi.mif`!

5.3.2   Extracting streamlines emerging from a region of interest

If you wish to visualize all streamlines that emerge *from* a ROI, the procedure is similar. First, decide which region(s) to analyze. For this example, we decided to choose the thalamus. As we discussed earlier when looking at the SC matrix, we saw that subcortical regions are highly connected to most other cortical regions, which makes the thalamus a particularly interesting region to analyze. We can analyze the left and the right thalamus at the same time. In the color lookup table, you will find that the corresponding indices are 362 (left thalamus) and 372 (right thalamus). Type

```
connectome2tck –nodes 362,372 sift_1mio.tck assignments_hcpmmp1.csv
–files per_node thalamus
```

→ *Note that unlike when selecting pathways between two regions, here we must not use the option –exclusive! Only then we will get all streamlines emerging from our ROI!*

→ *With the `–files per_node` option, we specify that we want our resulting files should be one file for each region/node that we analyze, which includes all streamlines emerging from those regions. If we do not specify this, we will get an individual file for each other region of ROI connects to, which in the case of the thalamus are hundreds of files!*
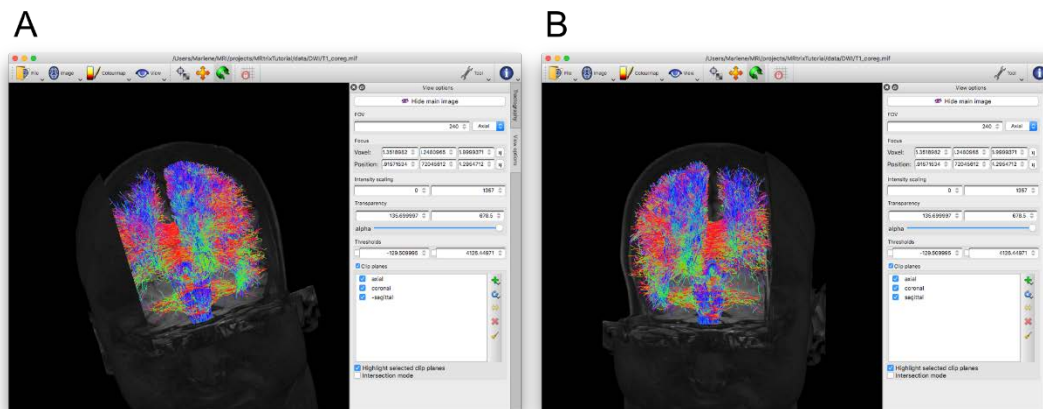
Figure 17. Selecting all streamlines that emerge from a region of interest. (A) shows all streamlines emerging from the left thalamus, (B) shows all streamlines emerging from the left thalamus. Note that although both thalami connect strongly to many cortical region, the connection density is higher on the ipsilateral hemisphere respectively!

Now look at the results. Start with the left thalamus:

```
mrview T1_coreg.mif –tractography.load thalamus362.tck
```

and switch to the "Volume render" mode and play around with the "Clip plane" option to see something like Figure 17A. Note that as expected, the thalamus connects to many cortical regions, but the connection density on the ipsilateral side is even stronger. Now, load the result from the right thalamus via the "Tractography" option from the "Tool" menu. The respective file is called `thalamus372.tck`. Remember to unselect the track-file corresponding to the left thalamus (simply untick the file `thalamus362.tck`). Figure 17B shows what you should see: Like the left thalamus, also the right thalamus connects to most other cortical regions, with higher density on the ipsilateral side – in this case the right hemisphere!

6   Connectome visualization tool

6.1   Getting familiar with the tool

Up until now, we have talked a lot about how to visualize track-files in MRview. This last part of the tutorial deals with visualization strategies of the SC matrix that we built before (Figure 15). MRtrix provides a special tool for that, called the *connectome visualization tool*. Before we introduce you to some of its basic features, it is important to clarify two terms which are fundamental for the informed use of that tool: **nodes** and **edges**.

If you are familiar with network analyses or graph theory, you are probably very confident with those terms. If not, here are some brief definitions: On a general level, a node is a point of redistribution. Two nodes are connected by an edge. For example, in the airline transportation system, nodes could be airports and edges the flight routes between the airports. In structural brain network analyses, nodes typically are cortical regions and edges are the tracks connecting those cortical regions. Often, cortical regions are derived from atlas-based parcellations. In our case, we have 379 nodes because we have 379 atlas regions. If you look at our SC matrix in Figure 15B, every node is represented by a row/column. Each of those nodes can potentially connect to 378 other nodes via tracks, or edges (all but itself). The strength of all of these connections is represented by the individual element values:

Remember that the element values are no more than the (scaled) number of streamlines between every possible pair of nodes. Therefore, the element values tell us about how strongly two nodes are connected. For example, the element value in the first column and the second row indicates how strongly atlas region 1 is connected to atlas region 2, and so on. If you want to learn more about brain network analyses, here is a good review article to start with: Rubinov and Sporns, 2010.

But for now, this information is sufficient to complete the tutorial and we can look at different strategies to visualize nodes and edges in MRtrix. To use the connectome visualization tool, you minimally need the following two files: the atlas-based parcellation image and a SC matrix file based on that parcellation. In the course of this tutorial, we have created both and can therefore type:

```
mrview hcpmmp1_parcels_coreg.mif –connectome.init
hcpmmp1_parcels_coreg.mif –connectome.load hcpmmp1.csv
```

→ *You have to specify the parcellation image twice, otherwise MRview is confused and won't load anything, or crash. As a result, the parcellation image will be loaded as a 2D file into MRview. However, the connectome visualization tool is designed for three dimensions, therefore it doesn't make much sense to display the 2D-parcellation. Just hit "M" to hide that image and continue!*

What you now see is a 3D-representation of the SC matrix (Figure 18A)! The little spheres represent the nodes. The geometric positioning of the nodes corresponds to the center of each atlas region on a 3D volume. The lines represent the edges, and their color indicates the connection strength (by default, on a hot colorbar). So basically, all information from the SC matrix is there. However, sometimes all information is too much, and in our case, the connectome looks messy. So now it's all about selecting which sets of information to represent, and making it prettier ☺! To do this, select the "Connectome" option from the "Tool" menu. A GUI appears, on which you can change the properties of both node and edge streamline visualization. Get a first feeling of the effects of changing those parameters by changing the "Colour" option in the "Node visualization" section to "random". Also, try to increase the node scaling to 2. We encourage you to play around with the other options as well! For example, you could try to change the node color to match a color lookup table.
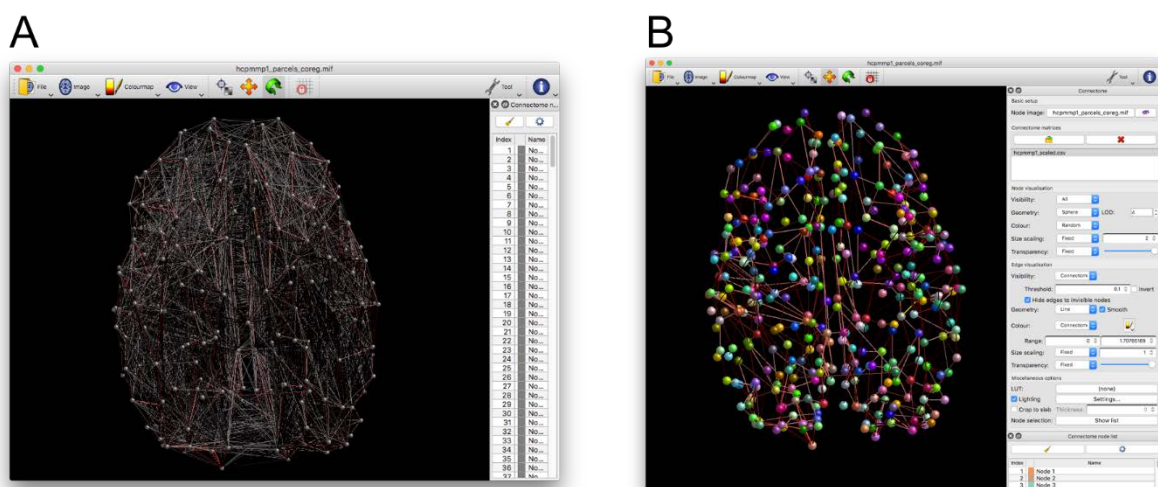

Figure 18. First steps in using the connectome visualization tool. After initializing the tool, you will see something like in (A). After increasing the threshold for edge visibility and smoothing the edges, as well as choosing a different ("random") color for the nodes and increasing their scaling factor to 2, it already looks clearer and prettier (B).

Choose that option and select the color lookup table we've used throughout this tutorial (`hcpmmp1_ordered.txt`) and see what you get! Also, note that the "Node" list at the very top of the GUI now gives you the names of each node/atlas region. Try highlighting different nodes on that list by clicking on them. Also try to highlight different nodes of that list simultaneously. What changes?

In terms of node visualization, there is another option called "Vector file". With this option, you can provide a vector file with as many entries as you have atlas regions. The data format must be logical, which means that those atlas regions which you want to hide are assigned "0" and those regions which you want to display get a "1". Note that the nth entry of that vector file corresponds to the nth atlas region of the parcellation image, defined in the (ordered) color lookup table. To test this option, you can now either create such a vector file yourself (e.g. in MATLAB) and load it in, but we also provide you with such a file in the `Supplementary_Files/`. It's called `randvec.csv`, and is a random selection of nodes, so don't be surprised that what you see may look scattered.

When you're done, focus on the GUI's section "Edge visualization". At first, we will get rid of all streamlines with low connection strength. Remember from Figure 15C and D that there are very many connections with very low connection strength. You can make such connections disappear by manipulating the "Threshold": Again, from Figure 15C and D we can see that a connection strength of 0.1 should reduce the number of visible edges substantially. Also, try out different values! Note that as you increase the threshold, you will also see nodes to which no edges connect. If you want to hide them and only see nodes with an edge, you can do that by changing the "Node visibility" to "Degree>1". Check out the "Edge smoothing" option as well, if you want a thicker representation of the edges. Does your visualization look similar to what you see in Figure 18D? Again, we encourage you to test out other available options. When you're done, move on to the next section to learn about how to visualize nodes and edges more "anatomically" than just spheres and lines!

6.2   Node geometry: Meshes

Right now, the nodes are represented as relatively boring spheres. MRtrix provides the possibility to represent nodes as a cortical meshes, corresponding to the atlas region which they represent. Take a look at Figure 20A to see what we mean by this. To get to such a visualization, all you need to do is to create a mesh file from your atlas-based parcellation image. And this is how that's done:

```
label2mesh hcpmmp1_parcels_coreg.mif hcpmmp1_mesh.obj
```

→ *Currently, MRtrix only supports the `.obj`-file format for meshes!*

Now switch back to MRview. There, change the node "Geometry" to "Mesh". Select the `hcpmmp1_mesh.obj` image and there you go. If what you see does not look like Figure 20A and you are not seeing all nodes, just change the "Node visibility" back to "All". The other options of "Node visualization" are of course not affected by the mesh-visualization and still apply as discussed in the previous section. When you're done, continue with the next section to learn how to make edges look fancy as well!

6.3   Edge geometry: Streamlines

Now that your nodes look more "anatomical", let's try to do the same with the edges. Right now, they are simply straight lines that connect its two linking nodes in the shortest possible way. However, we know from biology that two cortical regions are not always connected by the shortest possible path, but they might have to take some "turns" to bypass other "obstacles", such as other tracks. It would be nice if the edges in our connectome visualization would at least indicate the route via which two nodes are connected! For us, this is not a big problem because after all, we already created a whole brain tractogram and therefore *know* the routes that our streamlines take. With that information, we can easily reconstruct a mean representation of the "true" edge routes. In MRtrix, this is implemented in the command `connectome2tck`, which we used before when we selected all streamlines connecting two cortical regions. To derive a single file with the mean representation of the tracks that connect all possible pairs of atlas regions, we can use `connectome2tck` with the `-exemplars` option and provide the atlas-based parcellation image.

```
connectome2tck sift_1mio.tck assignments_hcpmmp1.csv exemplar –files
single –exemplars hcpmmp1_parcels_coreg.mif
```

→ *"exemplar" is the prefix name for your output file, such that the resulting file will be called "exemplar.tck"*
→ *With "-files single" you specify that your output should be merged into one file. This is necessary for the downstream visualization in MRview.*

This operation might take a few minutes to complete. Once you're done, you can use the file as follows: Switch to MRview. In the "Edge visualization" section, choose "Streamline" from the "Geometry" menu. Then, select the newly-created exemplars file (`exemplar.tck`). If you still have selected "All" for "Node visibility", you will probably see no change even after the file is loaded. If that is the case, get rid of some nodes (e.g. by switching back to the `randvec.csv` file). Figure 19 shows an example of what your visualization could look like. Once you have done this, also try changing the "Edge geometry" to "Streamtubes"!

6.4   Manipulating the visualization to match a research question

You should now be familiar with the basic features of the connectome visualization tool. However, the manipulations we have performed in the last three sections were more or less random. They looked pretty but were not meaningful. To make them meaningful, you need a research question and an analysis strategy which at the end hints you to some specific nodes or edges, and then you can use the connectome visualization tool to display them. Probably, you will often perform those analyses in MATLAB, and after your manipulations you'll get a matrix file with your results. You can provide those files directly to the tool, via the "Visibility → Matrix file" option (for both nodes and/or edges). Then, the connectome visualization will change – scaled according to your manipulated matrix. So, you're on your own from now on! However, we won't let you go without guiding you through one more example: Together, we will try to answer the question of which structural connections are the strongest in our brain (at least – in this tutorial – in my brain ☺).
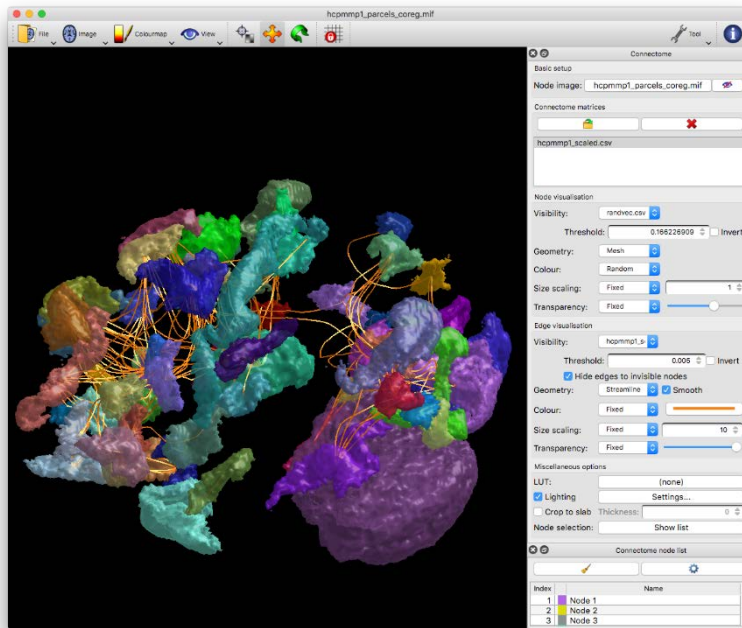
Figure 19. Making node and edge visualization more anatomical with the connectome visualization tool. Work through the tutorial to find out how to get there!

Before we start, think about this question for a moment: Which cortical regions do you think are the strongest connected ones and why? Have you made your guess? OK, then let's check it: Switch back to MRview. For prettiness, make sure to have your nodes loaded as meshes and the edges as streamlines. For the moment, we are only interested in the nodes, so you can decrease the node transparency even if that means the edges are covered.

When you are set, think about how to operationalize our research question: We want to know which connections are the strongest, which means nothing else than we want to filter out those connections which have the highest values in our SC matrix (Figures 15A and B). Therefore, all we need to do is adjust the threshold of our "Edge visibility". Additionally, we want our "Node visibility" to match the edge visibility, i.e. we only want to see those nodes to which edges connect. Therefore, we change the "Node visibility" to "Degree>=1". So now start to change your threshold. As you choose the value, orient at the distribution of matrix values as seen in Figures 15B and C. Start with a low value, e.g. 0.1. You should see what you see in Figure 20A (notice that in this is the only figure where we are showing the brain from a left lateral view rather than the right lateral view – this has a reason, as you will see in a minute!). Now slowly increase the threshold, for example by steps of 0.1. Remember that the highest matrix element value is roughly 1.7, so work your way towards that number. In Figure 20B, we show the edge visibility thresholded by 1.0. Now, very few edges and nodes are left. Which nodes are they? Obviously, many areas in the visual cortex are left, on both hemispheres, as well as some regions around the motor cortex. Additionally, a few regions of the inferior frontal cortex survive this threshold, but only on the left hemisphere. Well, which main functions are located at the inferior frontal cortex of the left hemisphere only (at least in right-handed people, like myself)? Of course, language! So, there you have the best-connected regions of the brain: regions corresponding to vision, motor function, as well as language.

Now that we know about the best-connected nodes, we want to see the corresponding edges as well. After all, we have yet to clarify whether the motor cortex is left because it connects so strongly to visual/language areas or because its subregions connect so strongly. To see that, decrease the node transparency (Figure 20C). Now you see the edges with the highest connectivity strengths, those edges to the very right of the connectivity distribution (Figure 15D). Obviously, the remaining edges are very short. More importantly, they connect nodes *within* rather than *between* brain lobes. Now we can answer our initial question completely: The (or rather: my ☺) brain's strongest connections are those between subregions of the visual, motor and language areas. What was your guess?
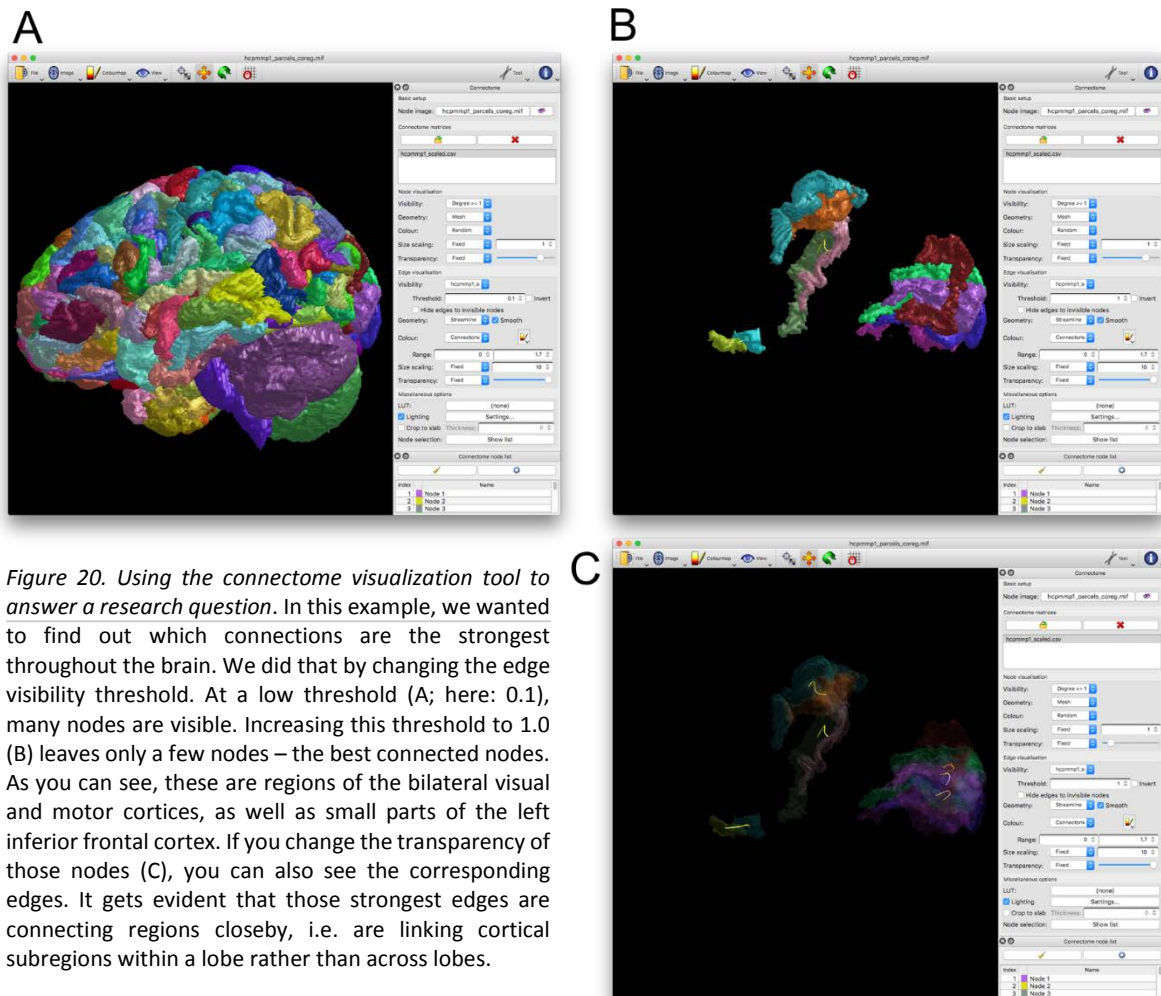
A

B

C

Figure 20. Using the connectome visualization tool to answer a research question. In this example, we wanted to find out which connections are the strongest throughout the brain. We did that by changing the edge visibility threshold. At a low threshold (A; here: 0.1), many nodes are visible. Increasing this threshold to 1.0 (B) leaves only a few nodes – the best connected nodes. As you can see, these are regions of the bilateral visual and motor cortices, as well as small parts of the left inferior frontal cortex. If you change the transparency of those nodes (C), you can also see the corresponding edges. It gets evident that those strongest edges are connecting regions closeby, i.e. are linking cortical subregions within a lobe rather than across lobes.

7   Farewell

You've come to the end of this tutorial. Congratulations for sticking with us so long! We hope you have enjoyed our little walk through the world of diffusion imaging, and we hope we could arouse your interest in this technique, or – if that was not necessary any more – shown you methods on how to improve your current pipeline. If you haven't already, we strongly encourage you to read a little deeper into the literature we provided throughout this tutorial, as well as to take a look at the following list of online resources and further readings – there is so much more to discover about DWI, tractography and MRtrix – we could only cover small parts here, and this already is a fairly long tutorial! But we hope to have you equipped well enough to continue yourself. So we leave you to your own now! We sincerely wish you a wonderful time and hope that fiber tracking improves your understanding of that unique, fascinating organ, the human brain! And don't forget to share once you've learned something new ☺

Cheers!
Snake §.

8   Appendix

8.1   Preparing a parcellation image for structural connectivity analysis

The following script demonstrates how a raw T1-image can be prepared in such a way that a structural connectivity matrix can be built for the HCP MMP 1.0 atlas (Glasser et al., 2016a). Of course, you can use analogous commands with your atlas of choice. The following set of commands require that FreeSurfer runs on your system. Remember that FreeSurfer requires that an environment variable called `$SUBJECTS_DIR` is defined, which points to a folder where all anatomical analyses take place. That folder is typically called `fs_subjects` or only `subjects` (if you don't have that environment variable already defined, you can do this by typing, for example, $SUBJECTS_DIR=*/YourPath*/fs_subjects).

Importantly, the `fs_subjects` directory must contain another subdirectory called `fsaverage`. This is a directory where all information on the FreeSurfer standard brain is saved. The `fsaverage` directory contains numerous subdirectories, called "label", "mri", "surf", etc. Importantly for us, the "label" subdirectory of `fsaverage` must contain two annotation files of the HCP MMP 1.0 atlas (one for the left hemisphere, one for the right hemisphere). This can be obtained by following the instructions on the atlas' [figshare webpage]. However, this can be a tricky process, and additionally requires that another software, the [Connectome Workbench], runs on your system. Nevertheless, we recommend you give this a shot and see how far you get! If you can't go on, however, we provide you the necessary annotation files. You find them in the `Supplementary_Files/` directory of your tutorial data, called `lh.hcpmmp1.annot` and `rh.hcpmmp1.annot`. You can just copy those files into the `fsaverage/label` directory and continue with your analysis. Before we start with the first few steps of preparing the T1 image for SC matrix construction, one more hint: In FreeSurfer commands often call a specific dataset using the subject name. In our case, we chose "snake" as a subject name. Of course, you can replace that with whatever name you want! So here we go:

```
# 1. Convert the raw T1.mif image to nifti-format, which is
necessary for subsequent analyses

mrconvert T1_raw.mif T1_raw.nii.gz

# 2. Since the HCPMMP1-atlas is a FreeSurfer-based atlas, you have
to preprocess the T1 image in FreeSurfer. This will take several
hours to complete.

recon-all –s snake –i T1_raw.nii.gz –all

# 3. Map the annotation files of the HCP MMP 1.0 atlas from
fsaverage to you subject. Remember to do that for both hemispheres:

mri_surf2surf --srcsubject fsaverage --trgsubject snake --hemi lh -
-sval-annot $SUBJECTS_DIR/fsaverage/label/lh.glasser.annot --tval
$SUBJECTS_DIR/snake/label/lh.hcpmmp1.annot

mri_surf2surf --srcsubject fsaverage --trgsubject snake --hemi rh -
-sval-annot $SUBJECTS_DIR/fsaverage/label/rh.glasser.annot --tval
$SUBJECTS_DIR/snake/label/rh.hcpmmp1.annot
```

In your $SUBJECTS_DIR/fs_subjects directory, there is now a directory called "snake" (or whatever name you chose ☺). In there, you will find the same directory structure as in fsaverage. Also like in fsaverage, the files ?h.hcpmmp1.annot lie in the /label subdirectory. We will now map those annotations on the volumetric image, additionally labeling subcortical structures. The resulting image will have the adequate atlas-based segmentation, but with more or less random integers assigned. MRtrix requires that the integers start with 1 and increase by 1. For that, we need to provide two color-lookup tables: one with the original integers and one with the ordered integers. In the latest release of MRtrix, those are available in $MRtrix3/share/MRtrix3/labelconvert. If you don't find them, we provide them in the Supplementary_Files/ directory of the tutorial data. They are called hcpmmp1_original.txt and hcpmmp1_ordered.txt. Note that the color-lookup tables contain the name of each atlas region – so if you want to know which integer belongs to which atlas region, this is the file you should consult. In our case, the name of the atlas regions are taken from the supplementary material of the original publication of that HCP MMP 1.0 atlas: Glasser et al., 2016b.

Finally, we need to co-register that parcellation image to diffusion space, using the transformation matrix which we already created in a [previous step]. Assuming you are still in the directory BATMAN/DWI, the following set of commands takes care of all those steps:

```
# 4. Map the HCP MMP 1.0 annotations onto the volumetric image and add
(FreeSurfer-specific) subcortical segmentation. Convert the resulting
file to .mif format (use datatype uint32, which is liked best by
MRtrix).

mri_aparc2aseg --old-ribbon --s snake --annot hcpmmp1 --o hcpmmp1.mgz

mrconvert —datatype uint32 hcpmmp1.mgz hcpmmp1.mif

# 5. Replace the random integers of the hcpmmp1.mif file with integers
that start at 1 and increase by 1.

labelconvert hcpmmp1.mif ../Supplementary_Files/hcpmmp1_original.txt
../Supplementary_Files/hcpmmp1_ordered.txt hcpmmp1_parcels_nocoreg.mif

# 6. Register the ordered atlas-based volumetric parcellation to
diffusion space.

mrtransform hcpmmp1_parcels_nocoreg.mif —linear diff2struct_mrtrix.txt —
inverse —datatype uint32 hcpmmp1_parcels_coreg.mif
```

8.2   Online Resources

Where?	What?
http://www.MRtrix.org/	Official MRtrix webpage
http://community.MRtrix.org/	User issues, mailing list
https://github.com/MRtrix3/MRtrix3	More user issues, bug fixes (more technical discussions)
https://www.youtube.com/watch?v=lfmAtXV0XJE https://www.youtube.com/watch?v=lQWucXuAXR8	Screening of an MRTrix3 tutorial at CIMAT 2016 by founder JD Tournier (Parts 1 and 2)

8.3   Further readings

The references listed here are by far not exhaustive. We chose to provide only very few literature, which we find helpful, for each of the different topics to not overwhelm you. However, in that literature, you will be guided to other useful references that will help you understand the topics better!

What?	Brief description	Example reference(s)
Basics on diffusion weighted imaging	The DWI signal differentiates tissue types based on the tissue-dependent diffusion of water molecules.	Mori and Tournier, 2013
	Advancements in DWI since the year 2000.	Tournier et al., 2011
Artifacts and pitfalls		
Hardware-related	MRI system needs to provide stable and strong gradient pulses, which is manipulated by b-factors, pulse widths and effective diffusion time; ways out: high-performance gradient coils.	Bihan et al., 2006
B0 field inhomogeneity	For a perfect DW scan, an absolutely homogeneous b0 field is required. This is however not possible because of the subject and different magnetization exposure of molecules. In MRtrix, `dwipreproc` can correct for this, which used FSL's `topup` is one way to correct for this.	Andersson et al., 2003; Schenck, 1996; Smith et al., 2004
Eddy currents	During DWI, strong gradient pulses are quickly switched on and off. This induces spiral-shaped currents in the body of the conductor (Lenz's law), which leads to geometric distortion of the MR images. This is best corrected for during post-processing (e.g. with FSL's `eddy` or MRtrix's `dwipreproc`, which uses `eddy`).	Andersson and Sotiropoulos, 2016; Bihan et al., 2006
Motion artifacts	DWI is especially sensitive to motion artifacts, since the phase shifts induced by the gradients happen at a microscopic scale. The easiest way to correct for this is during post-processing (e.g. with FSL's `eddy` or MRtrix's `dwipreproc`, which uses `eddy`).	Andersson and Sotiropoulos, 2016; Bihan et al., 2006
EPI artifacts	In echo-planar imaging, strong distortion occurs along the phase-encoding direction because of the relatively long time difference of the k-space-readout. By acquiring a pair of b0	Holland et al., 2010; Tournier et al., 2013

	images (i.e. an additional b0 in reversed phase-encoding direction, this can be easily corrected for, e.g. in MRtrix's `dwipreproc`).	
Tractography algorithms	Comparison of the performance of different algorithms: CSD is superior to Q-Ball imaging.	Tournier et al., 2008
Diffusion tensor imaging (DTI)	The tensor model is a 3x3 matrix which geometrically describes an ellipsoid. It can be used to describe the (an)isotropy of water diffusion. However, it is difficult to resolve crossing fibers with the tensor model since those will blur the ellipsoid, making it look more like a sphere instead of a cross.	Basser et al., 1994
Q-Ball imaging	Alternative to DTI	Tuch, 2004
Diffusion spectrum imaging (DSI)	Alternative to DTI to address crossing fiber problem: Many directions (up to >500) and extremely high b-values (>10,000s/mm^2) are needed. Results in very good resolution within "hard-to-track"-regions, e.g. cerebellar cortex. However, high SNR due to high b-values/eddy-currents, etc. This requires long scanning times, which are often not practicable. Additionally, there are no hints that DSI performs superior to CSD, which requires much less time.	Wedeen et al., 2008
Constrained spherical deconvolution (CSD)	An alternative to DTI to correct for crossing fibers. First, a response is estimated, which then deconvolves the DW data in all voxels. This outputs the so-called fiber orientation distribution, which can be represented as a geometric shape with many lobes. It was shown that this can resolve crossing fibers even of angles as sharp as 30°.	Tournier et al., 2004, 2007
DWI acquisition parameters	In CSD, there is a minimum number of directions you need to acquire in order to resolve a certain resolution. Acquiring more than that number will not further improve the data and can acquisition time can therefore be optimized. This optimization is highly confounded by the b-value. E.g., for a b-value of 3000, acquiring 45 data is sufficient.	Tournier et al., 2013
Algorithms to improve tractography		

Anatomically constrained tractography (ACT)	From anatomy, we know that nerve fibers do not end at certain brain regions (e.g. the CSF). By excluding such regions in streamline creation, the biological plausibility of fiber tracking is improved.	Smith et al., 2012
Multi-shell multi-tissue CSD (MSMT)	CSD performs optimally in pure white matter voxels. In voxels with partial volumes of CSF/gray matter, the algorithm performs imperfect. By acquiring DW images with different b-values, one can correct for this because different tissue types are optimally sensitive to different b-values.	Dhollander et al., 2018, 2016; Dhollander and Connelly, 2016; Jeurissen et al., 2014
Spherical-deconvolution informed filtering of tracks (SIFT)	Probabilistic fiber tracking overestimated the density of long tracks. This bias can be corrected for by filtering according to the properties of the fiber orientation distribution.	Smith et al., 2013

8.4   Complete list of references

Andersson, J.L.R., Skare, S., Ashburner, J., 2003. How to correct susceptibility distortions in spin-echo echo-planar images: Application to diffusion tensor imaging. Neuroimage 20, 870–888. https://doi.org/10.1016/S1053-8119(03)00336-7

Andersson, J.L.R., Sotiropoulos, S.N., 2016. An integrated approach to correction for off-resonance effects and subject movement in diffusion MR imaging. Neuroimage 125, 1063–1078. https://doi.org/10.1016/j.neuroimage.2015.10.019

Basser, P.J., Mattiello, J., LeBihan, D., 1994. MR diffusion tensor spectroscopy and imaging. Biophys. J. 66, 259–67. https://doi.org/10.1016/S0006-3495(94)80775-1

Bihan, D. Le, Poupon, C., Amadon, A., 2006. Artifacts and Pitfalls in Diffusion MRI. J. Magn. Reson. Imaging 24, 478–488. https://doi.org/10.1002/jmri.20683

Caruyer, E., Lenglet, C., Sapiro, G., Deriche, R., 2013. Design of multishell sampling schemes with uniform coverage in diffusion MRI. Magn. Reson. Med. 69, 1534–40. https://doi.org/10.1002/mrm.24736

Dhollander, T., Connelly, A., 2016. Generating a T1-like contrast using 3-tissue constrained spherical deconvolution results from single-shell (or multi-shell) diffusion MR data. Conf. ISMRM Work. Break. Barriers Diffus. MRI.

Dhollander, T., Ra, D., Connelly, A., 2018. Accuracy of response function estimation algorithms for 3-tissue spherical deconvolution of diverse quality diffusion MRI data Accuracy of response function estimation algorithms for 3-tissue spherical deconvolution of diverse quality di usion MRI data 3–6.

Dhollander, T., Raffelt, D., Connelly, A., 2016. Unsupervised 3-tissue response function estimation from single-shell or multi-shell diffusion MR data without a co-registered T1 image, in: Conference: ISMRM Workshop on Breaking the Barriers of Diffusion MRI. Lisbon, Portugal, p. 5 pp.

Dietrich, O., Raya, J.G., Reeder, S.B., Ingrisch, M., Reiser, M.F., Schoenberg, S.O., 2008. Influence of multichannel combination, parallel imaging and other reconstruction techniques on MRI noise characteristics. Magn. Reson. Imaging 26, 754–762.

https://doi.org/10.1016/j.mri.2008.02.001

Farquharson, S., Tournier, J.D., Calamante, F., Fabinyi, G., Schneider-Kolsky, M., Jackson, G.D., Connelly, A., 2013. White matter fiber tractography: why we need to move beyond DTI. J. Neurosurg. 118, 1367–1377.

Glasser, M.F., Coalson, T.S., Robinson, E.C., Hacker, C.D., Harwell, J., Yacoub, E., Ugurbil, K., Andersson, J., Beckmann, C.F., Jenkinson, M., Smith, S.M., Van Essen, D.C., 2016a. A multi-modal parcellation of human cerebral cortex. Nature 536, 171–178. https://doi.org/10.1038/nature18933

Glasser, M.F., Coalson, T.S., Robinson, E.C., Hacker, C.D., Harwell, J., Yacoub, E., Ugurbil, K., Andersson, J., Beckmann, C.F., Jenkinson, M., Smith, S.M., Van Essen, D.C., 2016b. A multi-modal parcellation of human cerebral cortex. Nature 536, 171–178. https://doi.org/10.1038/nature18933

Holland, D., Kuperman, J.M., Dale, A.M., 2010. Efficient correction of inhomogeneous static magnetic field-induced distortion in Echo Planar Imaging. Neuroimage 50, 175–183. https://doi.org/10.1016/j.neuroimage.2009.11.044

Jeurissen, B., Leemans, A., Tournier, J.D., Jones, D.K., Sijbers, J., 2013. Investigating the prevalence of complex fiber configurations in white matter tissue with diffusion magnetic resonance imaging. Hum. Brain Mapp. 34, 2747–2766. https://doi.org/10.1002/hbm.22099

Jeurissen, B., Tournier, J.D., Dhollander, T., Connelly, A., Sijbers, J., 2014. Multi-tissue constrained spherical deconvolution for improved analysis of multi-shell diffusion MRI data. Neuroimage 103, 411–426. https://doi.org/10.1016/j.neuroimage.2014.07.061

Kellner, E., Dhital, B., Kiselev, V.G., Reisert, M., 2016. Gibbs-ringing artifact removal based on local subvoxel-shifts. Magn. Reson. Med. 76, 1574–1581. https://doi.org/10.1002/mrm.26054

Mori, S., Tournier, J.D., 2013. Introduction to Diffusion Tensor Imaging, 1st edition. Academic Press, Oxford.

Rubinov, M., Sporns, O., 2010. Complex network measures of brain connectivity: uses and interpretations. Neuroimage 52, 1059–69. https://doi.org/10.1016/j.neuroimage.2009.10.003

Schenck, J.F., 1996. The role of magnetic susceptibility in magnetic resonance imaging: MRI magnetic compatibility of the first and second kinds. Med. Phys. 23, 815–50. https://doi.org/10.1118/1.597854

Smith, R.E., Tournier, J., Calamante, F., Connelly, A., 2013. NeuroImage SIFT : Spherical-deconvolution informed fi ltering of tractograms. Neuroimage 67, 298–312. https://doi.org/https://doi.org/10.1016/j.neuroimage.2012.11.049

Smith, R.E., Tournier, J.D., Calamante, F., Connelly, A., 2012. Anatomically-constrained tractography: Improved diffusion MRI streamlines tractography through effective use of anatomical information. Neuroimage 62, 1924–1938. https://doi.org/10.1016/j.neuroimage.2012.06.005

Smith, S.M., Jenkinson, M., Woolrich, M.W., Beckmann, C.F., Behrens, T.E.J., Johansen-Berg, H., Bannister, P.R., De Luca, M., Drobnjak, I., Flitney, D.E., Niazy, R.K., Saunders, J., Vickers, J., Zhang, Y., De Stefano, N., Brady, J.M., Matthews, P.M., 2004. Advances in functional and structural MR image analysis and implementation as FSL. Neuroimage 23 Suppl 1, S208-19. https://doi.org/10.1016/j.neuroimage.2004.07.051

Tournier, J.-D., Calamante, F., Gadian, D.G., Connelly, A., 2004. Direct estimation of the fiber orientation density function from diffusion-weighted MRI data using spherical deconvolution. Neuroimage 23, 1176–85.

https://doi.org/10.1016/j.neuroimage.2004.07.037

Tournier, J.-D., Mori, S., Leemans, A., 2011. Diffusion tensor imaging and beyond. Magn. Reson. Med. 65, 1532–1556. https://doi.org/10.1002/mrm.22924

Tournier, J., Yeh, C., Calamante, F., Cho, K., 2008. Resolving crossing fibres using constrained spherical deconvolution: Validation using diffusion-weighted imaging phantom data. Neuroimage 42, 617–625. https://doi.org/10.1016/j.neuroimage.2008.05.002

Tournier, J.D., Calamante, F., Connelly, A., 2013. Determination of the appropriate b value and number of gradient directions for high-angular-resolution diffusion-weighted imaging. NMR Biomed. 26, 1775–1786. https://doi.org/10.1002/nbm.3017

Tournier, J.D., Calamante, F., Connelly, A., 2007. Robust determination of the fibre orientation distribution in diffusion MRI: Non-negativity constrained super-resolved spherical deconvolution. Neuroimage 35, 1459–1472. https://doi.org/10.1016/j.neuroimage.2007.02.016

Tuch, D.S., 2004. Q-ball imaging. Magn. Reson. Med. 52, 1358–72. https://doi.org/10.1002/mrm.20279

Tustison, N.J., Avants, B.B., Cook, P.A., Zheng, Y., Egan, A., Yushkevich, P.A., Gee, J.C., 2010. N4ITK: improved N3 bias correction. IEEE Trans. Med. Imaging 29, 1310–20. https://doi.org/10.1109/TMI.2010.2046908

Veraart, J., Fieremans, E., Novikov, D.S., 2016a. Diffusion MRI noise mapping using random matrix theory. Magn. Reson. Med. 76, 1582–1593. https://doi.org/10.1002/mrm.26059

Veraart, J., Novikov, D.S., Christiaens, D., Ades-Aron, B., Sijbers, J., Fieremans, E., 2016b. Denoising of diffusion MRI using random matrix theory. Neuroimage 142, 394–406. https://doi.org/10.1016/j.neuroimage.2016.08.016

Wedeen, V.J., Wang, R.P., Schmahmann, J.D., Benner, T., Tseng, W.Y.I., Dai, G., Pandya, D.N., Hagmann, P., D'Arceuil, H., de Crespigny, A.J., 2008. Diffusion spectrum magnetic resonance imaging (DSI) tractography of crossing fibers. Neuroimage 41, 1267–1277. https://doi.org/10.1016/j.neuroimage.2008.03.036