SLT Project Report: Segmentation of neuron bundles from Diffusion MRI

Antonio Orvieto
orvietoa@student.ethz.ch

Sjoerd van Bekhoven
vsjoerd@student.ethz.ch

Moisés Torres
moisest@student.ethz.ch

Abstract

As in any problem of this kind, the key to get good results is to understand how the data is generated and how specific biological properties are translated into the numbers we see in our files. In this report we are going to explore a particular similarity measure to solve the bundle segmentation problem with superparamagnetic clustering. This only has to be considered as a starting point for some possible further work on the subject, and not as a solution of the problem. Details about our doubts and difficulties will be covered in the report.

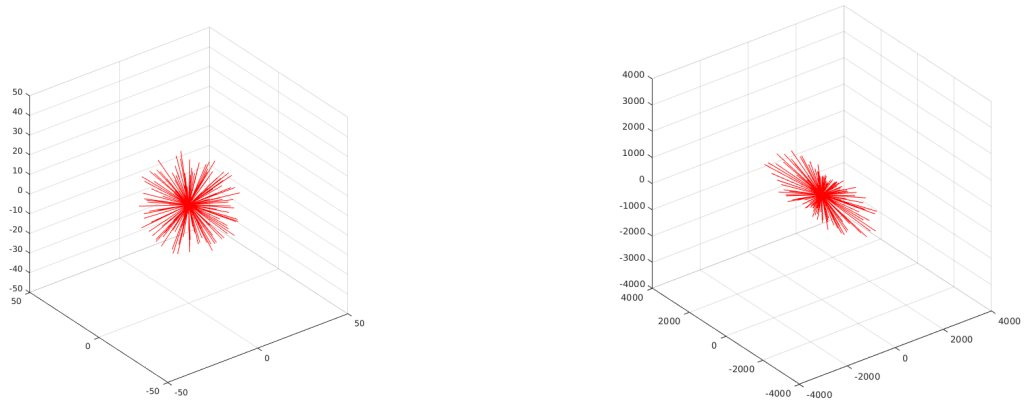
1 The Diffusion Tensor

Let us start from the beginning: for every point (x, y, z) in the brain, we are given measurements about the water diffusion at that point. The `nii` file combined with the `bvecs` gives us the diffusion [1] strength at that point (voxel), sampled at 164 directions (however 20 of them are not valid). Our task is to segment neuron bundles using this diffusion information. As it is very well described in [1], white matter tracts in the brain are characterized[2] by a strongly anisotropic diffusion in the direction of the fiber, whereas grey matter, CSF spaces and basically any other part of the brain has isotropic diffusion.

In Figure 1 one can see how different the diffusion is when the signal comes from a white matter voxel (b) or not (a).

[1] Actually we are given a signal which is inverse proportional to the diffusion. Hence the maximum diffusion direction is the minimum signal direction.

[2] This is not actually true at fiber crossings



(a) Isotropic case: p=(63,46,1)

(b) Anisotropic case: p=(63,46,26).

Figure 1: Different modes of diffusion.

Moreover, diffusion is a random transport phenomenon, furthermore the diffusion signals can be elegantly modeled by a Multivariate Normal Distribution in $\mathbb{R}^3$ [2,3]: the maximum diffusion direction corresponds to the eigenvector associated with the largest eigenvalue (all of them are positive) of the covariance matrix of this distribution. This model is reasonable because:

1. Water diffusion is related to Brownian Motion and therefore to the normal distribution.
2. The shape of our samples from this distribution (i.e. our data for every voxel) is very similar to an ellipsoid.

In order to get some intuition about the latter (hence to get the plots in Figure 1), we loaded the whole `nii` file on MATLAB and we used the `Ellipsoid fit` library [3] to get an estimate of the eigenvectors and eigenvalues of the covariance matrix at each voxel. After that (the procedure only takes a couple of hours in a regular laptop for the whole dataset) we computed for every voxel a quantity called Fractional Anisotropy [1,4]: let $\lambda_1, \lambda_2, \lambda_3$ be the eigenvalues of an ellipsoid:

$$FA = \frac{\sqrt{(\lambda_1 - \lambda_2)^2 + (\lambda_3 - \lambda_2)^2 + (\lambda_1 - \lambda_3)^2}}{\sqrt{2(\lambda_1^2 + \lambda_2^2 + \lambda_3^2)}}. \tag{1}$$

The Fractional Anisotropy is 0 if the ellipsoid is actually a sphere (completely isotropic) and is 1 in the extreme anisotropic case: this quantity is very convenient to identify the strong diffusion areas of the brain. In Figure 2 you can see a slice of the brain at $x = 60$, where the red pixels have a fractional anisotropy greater than 0.34 (this is a bit high and some white matter is of course not plotted).

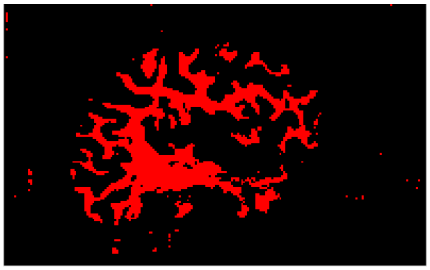


Figure 2: White matter at the section x=60, $FA > 0.34$

Now we should ask ourselves a crucial question: how do I identify a fiber? There are several methods to do that, most of them are described in [1]. However we are going to use pairwise clustering, in

[3] `http://ch.mathworks.com/matlabcentral/fileexchange/24693-ellipsoid-fit`

particular Blatt's super-paramagnetic clustering technique [5]. The only parameter we can choose in the algorithm is the J_{ij}, which is the cost of not assigning the voxel i and the voxel j to the same fiber (cluster): J_{ij} should be high when i and j are similar (so we want to keep them in the same cluster) and should be small when i and j are not similar. But how do we measure similarity between ellipsoids?
Many approaches exist right now to do this, and most of them depend on the application: a good list of such similarity measures can be found in [4]. However, we are going to use the simplest of them: for every voxel we only keep the maximum diffusion (minimum signal) direction, and then we compute J_{ij} as a simple function of the angle between the vectors: let v_i and v_j be the maximum diffusion directions of the voxels i and j respectively, we define:

$$J_{ij} = \exp\left[-k\left(\frac{\langle vi, vj\rangle}{||v_i||\,||v_j||}\right)^2\right]. \tag{2}$$

Where we set $k = 6$ (it appears to work, but maybe it needs to be tuned in properly, maybe with some local considerations like the parameter a in [5]). J_{ij} is (almost) 0 for orthogonal diffusion directions, and it is 1 for parallel diffusion directions. Moreover we will only consider in our clustering points with relatively high fractional anisotropy (not too high, as we would loose the fiber structure). This has of course both drawbacks and advantages. We start enumerating the advantages:

1. Data compression (i.e. get minimum signal direction) can be done in MATLAB[4] before the clustering procedure.
2. The computation of the J_{ij} is trivial at each step. Moreover the Fractional Anisotropy threshold eliminates a lot of points, making the computation faster.
3. Considering just the diffusion directions (and not lengths for example) is convenient as the diffusivity in the brain is not constant and depends on the individual [7].
4. The method can be easily validated, as can be seen in the last section.

However the method does have some serious drawbacks:

1. It is not clear whether our compression results in less robustness: in our opinion this is not really a big problem, as not completely anisotropicity is not due to the noise, but rather to the MRI technology.
2. This method fails when dealing with crossing fibers. Further information can be found in [8]
3. Other similarity measures could perform way better than this. In the future we would definitely love to experiment with the Pollari distance measure [9]

2 Super-paramagnetic Clustering

For our clustering algorithm, we have implemented the (nonparametric) procedure described in Blatt's 1997 paper [5]. Algorithms such as histogram clustering (which is parametric) will fail, if not modified correctly, to capture the connectedness property required for the fibers as it only considers the statistics of each point in isolation.

The main idea for the selected approach is to model our data by using a Potts model. Each data point is assigned a Potts spin and then an interaction between neighboring points is introduced. This magnetic system will exhibit three different phases: ferromagnetic, super-paramagnetic and paramagnetic. In the intermediate regime (super-paramagnetic) clusters of relatively strongly coupled spins become ordered, whereas different clusters remain uncorrelated. We will take advantage of this property to perform the clustering of the data.

[4]We do this fitting the ellipsoid with the previous library. Unfortunately such a library does not exist for Python and it is really difficult to program: the ellipsoid fitting problem for dimension 3 or more is a well known challenging problem, further reference can be found in [6]

2.1 Potts model simulation: Swendsen-Wang algorithm

In order to identify the range of temperatures corresponding to the super-paramagnetic regime, we make use of the fact that when at this phase, the magnetic system will experiment a jump in the order parameters. Specifically, a sharp transition between the ferromagnetic and the super-paramagnetic regimes is observed in the susceptibility χ:

$$\chi = \frac{N}{T}(\langle m^2 \rangle - \langle m \rangle^2) \tag{3}$$

In the above formula can be seen that the susceptibility is related to the variance of another thermal quantity, the magnetization $\langle m \rangle$:

$$\langle m \rangle = \sum_{\mathcal{S}} m(\mathcal{S}) P(\mathcal{S}) \quad \text{with} \quad P(\mathcal{S}) = \frac{1}{Z}\, exp(-\frac{\mathcal{H}(\mathcal{S})}{T}),\ m(\mathcal{S}) = \frac{qN_{max}(\mathcal{S}) - N}{(q-1)N} \tag{4}$$

To obtain the magnetization, the thermal average of the magnetization over all possible spin configurations $\mathcal{S}$ has to be calculated. Direct evaluation of this sum is intractable, since the number of configurations increases exponentially with the size of the data (q^N).

To overcome this problem, we will use the Swendsen-Wang (SW) Monte Carlo simulation method (details on how to use our code are described in the `README.md` file). Basically we will sample possible configurations out of the set $\mathcal{S}$ according to the Boltzmann probability distribution from (4). Then the magnetization calculation turns out to be simply the average over the sampled configurations:

$$\langle m \rangle \approx \frac{1}{M} \sum_{i=1}^{M} m(\mathcal{S}_i) \quad \text{with} \quad \mathcal{S}_i \in \{\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_M\} \tag{5}$$

In order to construct such configurations, the SW algorithm proposes the following steps for each one of the temperatures:

1. Initialize the value of the Potts spins at random.
2. For all neighbouring points[5], establish a bond with probability:

$$p^f_{i,j} = 1 - exp(-\frac{J_{i,j}}{T}\ \delta_{i,j}) \tag{6}$$

3. Find the connected subgraphs, the SW-clusters.
4. Generate a new Monte Carlo step by assigning a random spin value to all the nodes in the same cluster.
5. Calculate the magnetization of the new configuration.
6. Go to step 2 until the number of iterations is reached.
7. Average the magnetization obtained for each configuration.

To improve the efficiency of the algorithm, it is straightforward to perform the simulation of each temperature in parallel as they are completely independent of each other. In our implementation, we used the Python package `joblib` to achieve this. With this improvement our code was able to take advantage of the large number of cores available at the Euler cluster (which we used to run the Monte Carlo simulations), speeding up the runtime notably.

We can see that through our simulation procedure we have been able to obtain an estimate of the susceptibility χ as a function of the temperature T, which reflects the thermodynamic phases of the system. Hence, we can identify the range of temperatures in which our system is in the super-paramagnetic regime and use this information to perform the clustering of our data.

[5]Instead of letting all pairs interact, K-nearest neighbors are used due to computational reasons.

2.2 Clustering the data

As we mentioned in the beginning, in the super-paramagnetic regime clusters of relatively strongly coupled spins become ordered. Thus, for our final step we will perform another Monte Carlo simulation to estimate the behaviour of our system in this phase. We will select an adequate temperature based on the previously obtained plot of the susceptibility, and for this temperature, we will use the SW algorithm described earlier. In this case, for each Monte Carlo step we will measure the two-point connectedness C_{ij}, which is the probability that two nodes belong to the same cluster. To calculate it we just keep track of the number of times that two nodes were in the same cluster in the different Monte Carlo steps and take the average at the end. Once we have this, we perform the clustering in the following way:

1. First, we build the "core" of the clusters by bonding the nodes with $C_{ij} > 0.5$[6]. As mentioned in the Blatt paper [5], the result depends weakly on this threshold.

2. Capture nodes lying on the periphery of the clusters, by linking them to the neighbour with highest C_{ij}.

3. The resulting subgraphs conform the clustering of the given data.

3 Results

For the results of the simulation in this section, we use a subset of $26 \times 26 \times 26 = 17576$ voxels with $x \in [50, 75]$, $y \in [80, 105]$ and $z \in [55, 80]$. The number of iterations in the SW Monte Carlo model is 400 per temperature with 50 burn-in samples, ranging over 100 temperatures from $T = 0$ to $T = 1$. As mentioned before, we set a threshold on the fractional anisotropy (FA), in this case FA > 0.25. This leaves us a with a total of $N = 8942$ voxels. Each voxel interacts with its 10-neighborhood, which results in ~ 60.000 (i, j) interactions in total. The results of the Monte Carlo simulation of Potts models with the Swendsen-Wang method with the above settings can be found in Figure 3.

When looking at the plot of the susceptibility density χ^T/N, we find a strong peak at $T = 0.41$ which indicates a ferromagnetic to super-paramagnetic transition, or more informally the moment that a large cluster breaks into several still rather large clusters. This transition can also be spotted in the magnetization plot, which suddenly diminishes at $T = 0.41$.

For the transition from super-paramagnetic to paramagnetic, we look for a sudden decrease, which we find at about $T = 0.54$. From the susceptibility density plot, we therefore identify the the super-paramagnetic phase to be from $T = 0.41$ to $T = 0.54$. Note that between $T = 0.46$ and $T = 0.54$ a plateau can be spotted, which indicates large, stable clusters. Therefore, we pick a temperature in the plateau (which is in the super-paramagnetic phase) to create a clustering, in this case $T = 0.49$.

The results of the clustering at $T = 0.49$ can be found in Figure 5 and Figure 4. Figure 5 shows the distribution of the voxels over the clusters. In total, we found 1044 clusters of which 6 have a size larger than 200. These are plotted in Figure 4. We can clearly see some "stripes" that could very well denote fibers, such as for example the yellow and red cluster. To verify whether this indeed is the case, we validate our model in the next section.

[6] Although in the Blatt paper [5] the spin-to-spin correlation G_{ij} is used instead of C_{ij}, the authors also mention that is completely equivalent and that they only do it for didactic purposes.

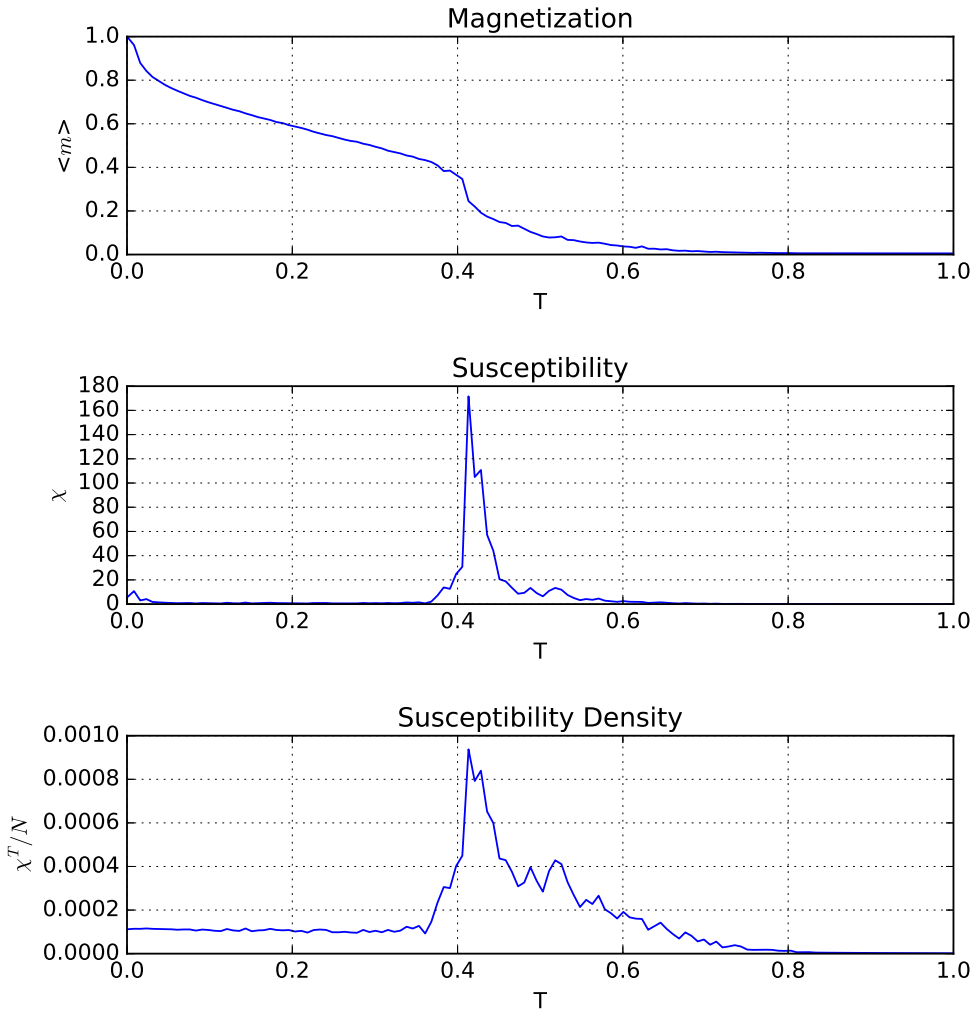


Figure 3: Results of the Monte Carlo simulation of Potts models with the Swendsen-Wang method

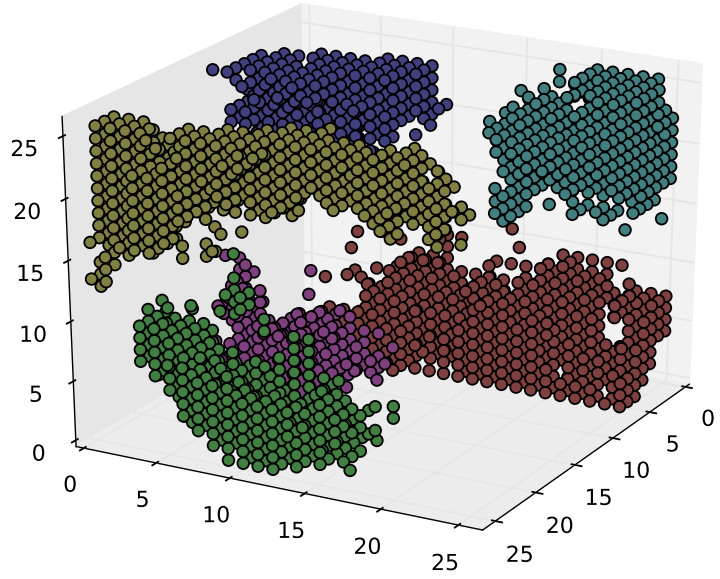


Figure 4: Clustering

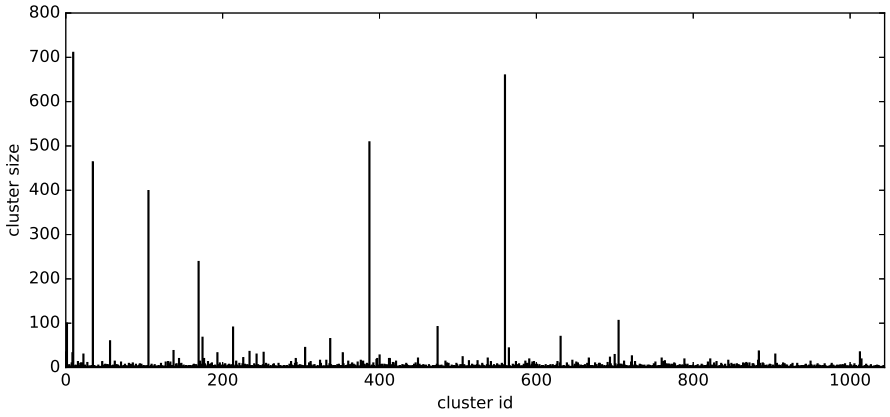


Figure 5: Cluster Distribution

4 Validation

Now that we have established a clustering, we ask our selves: do the clusters actually denote fibers? Since the answer to this is generally unknown for this dataset, we at least verify our methodology. In this final section we attempt to do so by validating the results of a clustering with respect to the cost function we have constructed.

The cost function J_{ij} used in the clustering algorithm is constructed in such a way that it prefers parallel maximum diffusion direction and is thus averse of clustering datapoints with orthogonal maximum diffusion direction. To verify that the clustering is in correspondence with this behavior, we look at a 2D slice of the dataset ($x = 67$, $y \in [20, 79]$ and $z \in [50, 90]$), so that we can easily visualize the maximum diffusion direction by means of an arrow. An example of such a visualization can be found in Figure 6, only showing the datapoints with a fractional anisotropy of FA > 0.25, just as in our original results. In this figure, one can somewhat manually classify the fibers by following the arrows.

Now, let us look at the result of our clustering algorithm on the same slice of data, which can be found in Figure 7. In the clustering, a very large part of the datapoints is clustered together, but there are several, significantly smaller clusters. If we focus on the yellow cluster at the bottom right, we can clearly see that the datapoints that are separated from the big red cluster are denoted by vectors that are orthogonal to the vectors denoting the datapoints in the big red cluster close to them. If we look at other small clusters, we see similar behaviour. On the other hand, one can also pinpoint sections in the red cluster that contain datapoints that are denoted by orthogonal vectors, which ideally should not happen.

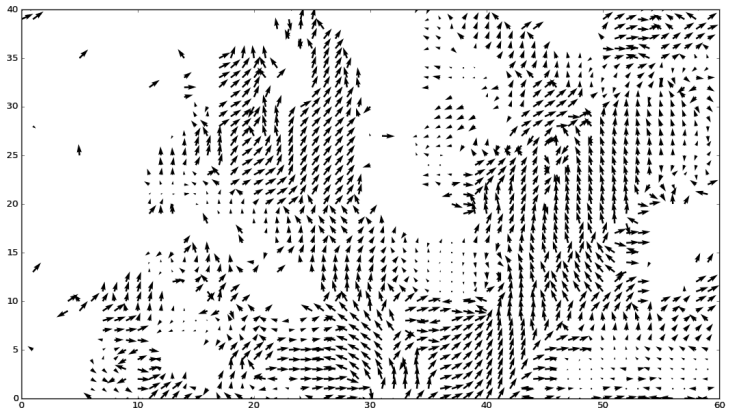


Figure 6: Maximum diffusion directions for 2D slice of the brain

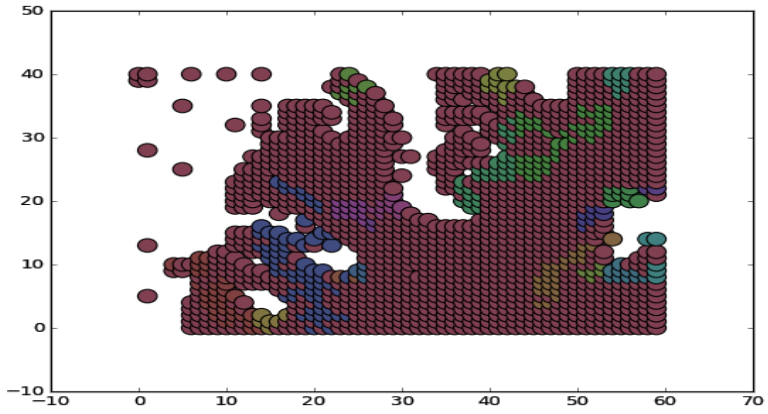


Figure 7: Clustering for the previous slice

Another, more general way of validating our results, could be verifying whether the largest clusters in our clustering contain the majority of what we consider white matter for higher thresholds of the fractional anisotropy. For example, we have used only datapoints with FA > 0.25 for the subset in our experiments. By increasing this threshold, we will capture datapoints of which we are, by

our previous assumptions, more and more sure that they are white matter. By comparing these datapoints with the datapoints in our clusters created for threshold FA > 0.25, our clustering can be considered "good" if it keeps a somewhat steady ratio of captured datapoints. Such a plot can be found in Figure 8 in which we can see that the number of datapoints captured by our 6 biggest clusters decreases way less steep than the total number of datapoints that we, by our assumptions, consider white matter.

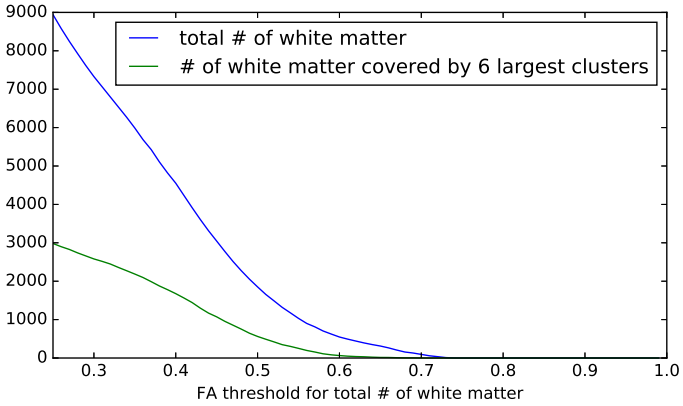


Figure 8: Validation with total number of white matter vs. the white matter covered by the 6 biggest clusters

To sum up, with these examples we just try to get some insight of whether or not the designed cost function J_{ij} succeeds in modeling both the connectedness and diffusion direction properties that characterize white matter tracts. However this only gives us an intuition of the correctness of our algorithm. More rigorous ways of validating tractography results are still a matter of research.

References

[1] P. Mukherjee, J. Berman, S. Chung, C. Hess, and R. Henry, "Diffusion tensor mri imaging and fiber tractography: Theoretic underpinnings," *American Journal of Neuroradiology*, vol. 29, no. 4, pp. 632–641, 2008.

[2] A. L. Alexander, J. E. Lee, M. Lazar, and A. S. Field, "Diffusion tensor imaging of the brain," *Neurotherapeutics*, vol. 4, pp. 316–329, Jul 2007. 17599699[pmid].

[3] P. J. Basser, J. Mattiello, and D. LeBihan, "Estimation of the Effective Self-Diffusion Tensor from the NMR Spin Echo," *J. Magn. Reson.*, vol. Series B, 103, pp. 247–254, 1994.

[4] T. H. J. M. Peeters, P. R. Rodrigues, A. Vilanova, and B. M. ter Haar Romeny, *Visualization and Processing of Tensor Fields: Advances and Perspectives*, ch. Analysis of Distance/Similarity Measures for Diffusion Tensor Imaging, pp. 113–136. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009.

[5] M. Blatt, S. Wiseman, and E. Domany, "Data clustering using a model granular magnet," *Neural Computation*, vol. 9, pp. 1805–1842, Nov 1997.

[6] D. Turner, *An Algorithm for Fitting an Ellipsoid to Data*. 1999.

[7] D. Alexander, J. Gee, and R. Bajcsy, "Similarity measures for matching diffusion tensor images," in *In Proceedings of the British Machine Vision Conference (BMVC*, pp. 93–102, 1999.

[8] J. C. Gee and D. C. Alexander, *Visualization and Processing of Tensor Fields*, ch. Diffusion-Tensor Image Registration, pp. 327–342. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006.

[9] M. Pollari, T. Neuvonen, M. Lilja, and J. Lotjonen, "Comparative evaluation of voxel similarity measures for affine registration of diffusion tensor mr images," in *2007 4th IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pp. 768–771, April 2007.