1 Parametric models

1.1 Parameter estimation and mapping

1.1.1

As seen in class, we apply the logarithm to the diffusion tensor model to obtain a linear equation and perform a linear estimation, with the pseudo-inverse of the design matrix. After having followed the method described in the question sheet, we obtain Figure 1.

The results are satisfactory, even though in the FA maps, we see that the border all around the brain raises numerical issues, and gives us an unrealistic render. This question of the border is quite important when processing the scans, because we have a lot of black pixels corresponding to non-useful data (it is just the background). Given that we use a log map, we cannot use it on the background (because log(0) is undefined), so we have to add a condition to verify if the minimum value of the voxel is greater than 0 before applying the log. We can also roughly resize the scans to reduce the number of black pixels to process.

1.1.2

We are using the MATLAB function `fminunc` which corresponds to unconstrained optimization. We can adjust this optimization by changing the tolerance on the parameter or the function output or changing the number of iterations. We can imagine that to obtain different optimization results, we will have to modify these parameters, but also find relevant starting points, so as not to fall into local minima. The fit we have obtained can be seen in Figure 2.

The fit obtained is not ideal. We can see it has the potential to catch up to the data values for $b = 0$, but the fit is far away from the data values for $b = 3000$. This explains why the RESNORM is this high compared to the expected value that should have been around `5e6` or `8e6`. One of the reasons why we obtain a high SSD value is because the parameters we have estimated are not realistic.

For example, when we had estimated the parameters with the linear estimate in 1.1.1 for one particular voxel, the values we obtained were: `[8.3550 0.0009 0.0001 -0.0002 0.0007 -0.0003 0.0008]`. The first value represents $\log S(0)$, so it gives around `4e3`, which is in the same order of magnitude as our new estimate. The values for diffusivity also match the d we have estimated. But the value of f cannot be negative, in our model $f \in [0, 1]$ (it represents a ratio between the ball part and the stick part). This might be why we obtain such a bad fit, the values exceed or are way under what is expected (especially for $b = 3000$). Let's adapt our method to take into account this limitation.

1.1.3

Still using `fminunc`, we have chosen to modify our parameters this way:

$$S(0) = x_1^2, \qquad d = x_2^2, \qquad f = \exp(-x_3^2), \qquad \theta = x_4 \quad or \quad \pi \exp(-x_3^2), \qquad \phi = x_5 \quad or \quad 2\pi \exp(-x_3^2)$$

Intuitively, we would not think about constraining θ and ϕ given that we use them in $\cos$ and $\sin$, which are periodic functions, so we would just have to take the modulo of these angles to retrieve a unique solution. Let's look at the graphs we obtain, shown in Figures 3 and 4.

It seems that the optimizer would rather we constrained θ and ϕ. The SSD value is much closer to what we expect, so are the values of S0, d and - especially - f. The main change is the estimation of $b = 3000$ values. We are now in the same error range as the $b = 0$ values, and so the fit respects the measurement noise variance, and the values agree with what we expect.

1.1.4

To assess if we have really found a global minimum or not, we will follow the method suggested in the coursework sheet. We start by adding Gaussian noise scaled to each parameter (e.g for f, given that it is comprised between 0 and 1, we should watch out not to have too much variance, otherwise it would get out of bounds). We will repeat the optimization step, beginning from the given starting point, and add noise at each iteration, and store the SSD value. We repeat this operation multiple times. The results are shown in Figure 5.

We can see that the `RESNORM` mostly takes 3 values, and we can identify the global minimum a priori: `5e6`. This is the value we have found earlier, so our routine confirms that this is the global minimum. We can also see that it is not easy to fall into the global minimum, and so that we will have to repeat the optimisation step and explore around our starting point to find the right parameters each time. According to our graph, we can be sure at 95% to find the global minimum between 5 and 10 iterations. We have repeated the experiment, but now for different voxels, chosen randomly over the slices (we add a condition that discards the background voxels). The results are shown in Figure 6.

Once again we can see that the SSD values eventually converge towards `5e6`, which confirms once again it is our global minimum. But this time, we see a new minimum value `0.3e6`. This corresponds to the $b = 0$ voxels, which are in much less proportion than the $b = 3000$ voxels. In both cases, the global minimum is reached within 10 iterations, and confirms our first results. Now that we have seen that our method is consistent, let's apply it to the whole slice.

1.1.5

We therefore apply the previous routine to the whole 72 slice. More precisely, for each voxel we run 10 times our optimizer, and we keep the parameters associated with the lowest `RESNORM`. According to the previous questions, the global minimum should be reached by one of these sets. Let's note that we run the unconstrained version of the MATLAB optimizer `fminunc` while trying to force constraints, so there might be numerical issues during the computation. We have added a try/catch block so that in case the optimization fails, the program keeps on running. Finally, to counter some other numerical issues, such as some voxels not reaching the global minimum, we process the maps by discarding the abnormally high values, and we finally normalize each map. The obtained results are presented in Figures 7-9.

The results are satisfactory. The S_0 and f maps look accurate. The mean diffusivity map is more precise than the one shown in Figure 1, here optimization achieves a better result than linear estimation, and gives us precious new information. Finally, the `RESNORM` map is homogeneous, showing that the global minimum is in general found for every voxel: which validates our approach. We can still detect some numerical issues: black pixels in the middle of the f and d maps, and the "exterior" part of the brain in f is not quite accurately estimated, probably because the value of our data in this part is lower than the "interior" part of it.

As for the fibre direction map, we can still recover the main movements of the fluids by looking at the unscaled map: isotropic directions in the parts storing brain fluids, and clear directions in the fibres. When we scale the map by f, we eliminate the "ball" part of the ball-and-stick model, and therefore only recover the fibres' directions. We have then added a scaling coefficient so that we could still see the arrows. We have zoomed in on the middle-right part of the scaled map to see if the movements estimated are correct. Indeed, the fluid goes from top to bottom in the front part of the brain. The estimates are correct.

To conclude this part, we have seen that even an unconstrained optimizer can give us a meaningful estimation. There are still numerical issues to the constrained nature of our parameters, and we need to study the convergence of the algorithm, but this is a good first step before trying to estimate more complex models. The linear DTI is also useful for more simple models, and to provide more validation data for our optimization routine. Still, before using

1.1.6

fmincon
Instead of using unconstrained optimization and forcing the constraints within the function (which may eventually lead to numerical errors as seen above), we will use constrained optimization with the `fmicon` function from MATLAB. Figure 10 shows us that the convergence towards the global minimum is similar to our unconstrained version, so we can apply the same routine on the slice 72.

Figures 11 and 12 present the results using `fmincon`. A priori, there does not seem to be much of a difference between constrained and unconstrained results. The major difference would be the d and f maps, because they were subject to a lot of dead pixels due to numerical issues with the `fminunc`. Here, the maps are more complete, and the borders appear less noisy too. The quiver map is also more complete, and we can visualize way better the fibres on the left of Figure 12. The constrained method seems to handle better the smallest values. Overall, we have won in image quality and information. But what have we lost? Here is the processing time of each method using MATLAB's `tic` and `toc`:

- for `fminunc`: `Elapsed time is 496.608760 seconds.`
- for `fmincon`: `Elapsed time is 4248.204070 seconds.`

The constrained optimization is almost ten times longer in processing time compared to unconstrained optimization. The question here is: are we ready to spend more time on the method for this gain in information? Because, although there is a gain in information, it is not hugely significant. One could stick to the unconstrained method and run it multiple times to enhance the quality, and fix the few numerical issues. Otherwise, it could be interesting to come up with a new model that would separate the estimation of S_0, f and d from ϕ and θ, so that a trade-off can be found. We could also, instead of running each optimization step for a fixed number of iterations, stop as soon as the value gets under a `RESNORM` of `6e6` for example, as we know that it will be the global minimum.

Informed starting points
We will now use the linear diffusion tensor model to obtain first estimates for S_0 and d, and then add noise from this estimate to look for the global minimum. We start by running the error plot from **1.1.4** to see how fast the convergence has become. Indeed, when comparing Figure 5 (non-informed points) and Figure 13 (informed points), we can see that the global minimum can be found in under 5 iterations. Let's apply this method to the parameter estimation of the whole slice. We have also added a condition which makes the iterations stop when the `RESNORM` drops below $6e6$ i.e the global minimum, so we can compare the convergence speed of the informed version versus the non-informed one. The results can be found in Figures 14-17.

Comparing the two results, we can see that there are fewer numerical errors in the informed version, because the global minimum is found more quickly. There is also more information on the quiver map. The estimation is a bit worse on the "exterior" part of the brain, because the voxel intensity here is much lower than on the "interior" part of the brain and therefore the DT model gives a worse estimation, making the starting point further from the global minimum. We have had to add a condition on the DT estimate so we don't have end up too far: we take the maximum between the S_0 estimate and 2500, and we take the maximum between the d estimate and 0. These values might be too far from the global minimum in the "exterior" part, but at least we can limit the number of numerical issues. The main result for us lies in the processing time:

- for the informed version: `Elapsed time is 309.100425 seconds.`
- for the non-informed version: `Elapsed time is 326.579426 seconds.`

Now, we can be sure that starting from a point in which we have more confidence, makes the search for the global minimum easier.

Using the derivative of the objective function
We have computed the gradient of our ball-and-stick model, the equations can be found in Figure 20. Unfortunately, using the argument `CheckGradients` in the function, we can see that the algorithm stops because our computed gradient differ from the finite-difference computed by MATLAB (see Figure 21). We can see that for the constrained version, this difference is quite small, but MATLAB demands a relative difference of maximum `1e-6`, therefore it does not work. There might be numerical issues due to the small parameter values, leading to this error.

We have still run the algorithm (Figures 18 and 19), and as we can see the global minimum is not found, although we could argue the chain obtained is more stable than the previous error plots we have had.

1.2 Uncertainty estimation

1.2.1

The classical bootstrap results can be found in Figure 22. We can see that for S_0, we obtain a nice Gaussian around a mean value of approximately 2400, which is a correct estimation for our model. But, for the d and f parameters, we can see that there is a consensus around a small d, and a high f. This can be verified through our estimates in the previous and next parts. On the other hand, the θ and ϕ parameters seem to fluctuate a lot, once again this is something verified in our previous and next results: this is because we might be located in a zone where the fluids can move freely, therefore there is not only one correct result. Thus, we can extract a 2-sigma range for S_0, because it takes high values and it is easier to tweak it, but we cannot really say the same for the other parameters, even though the bootstrap procedure has provided us with new information about the range of these parameters.

1.2.2

The MCMC results can be found in Figure 23. This graph gives us additional information compared to the bootstrap one. We can here see a convergence of the chain. This indicates us that the voxel admits one global minimum, and no other local minima, or local minima too far from the global one. We even tried to add a bigger noise value when the chain starts converging to make the chain vary again, but the global minimum seems too attractive for the model. This also explains why the d and f values have also converged in the bootstrap procedure. This is both good and bad news for our parameter estimation: we know that once we are in a minimum, it is likely that it is the global one. But on the flipside, if we do not let the estimation run for enough time, we might end up with a real anomaly in the parameters and `RESNORM` (could be the reason for the numerical instabilities from part **1.1**).

1.3 Model selection

1.3.1

First of all, we need to get an idea of the possible `RESNORM` values. We will use our routine seen in **1.1.4** to compute the error value, using the starting point from **1.1.6**, before trying to estimate better starting points. The standard deviation of this new data being 5000 times lower than the one from the former data, we should expect a `RESNORM` around `1e3`. As we can see on Figure X, there is actually a global minimum around `15`. We can see in Figure 24 that constrained optimization seems more stable, but unconstrained optimization obtains a lower score. At the moment there is no clear advantage.

The final graphs show that the global minimum is found more often, and that the fit is quite good (there is less variance in this data compared to the data in parts **1.1** and **1.1**), with no real difference between constrained and unconstrained methods (ϕ and θ are bigger in the unconstrained one, it might lead to less numerical issues).

1.3.2

After performing the same routine as in **1.3.1**, we have now a fit of the two new models, and the DT model (Figures 26-28). The DT model is not flexible enough, this could have been expected from part 1. The two new models seem to fit correctly the data, although the `RESNORM` and the stability of the optimization seem less ideal compared to the ball-and-stick model. We can imagine that this could differ when exploring different voxels, where the structure of the brain could benefit from a zeppelin shape rather than a ball shape.

1.3.3

Figure 29 exhibits the AIC and BIC values computed for the four different models. Only looking at the `RESNORM`, and given that all the X-and-stick models have 5 parameters, we could have guessed the ranking. In this part of the brain, the ball-and-stick model is the most capable of explaining what is happening. The DT model is not appropriate now that we have more robust tools. For the three others though, there is not too much difference between them, the two new zeppelin models could prove useful in the following. The fact that the rankings are consistent whether we use AIC or BIC also validates our results.

1.3.4

Figures 30-32 show our results for the X-and-two-sticks models. We can see that the fit is definitely better for these 2-compartments models. The ball-and-two-sticks model is getting close to the ball-and-stick model. Let's look at their AIC/BIC in next question to see if we can achieve something better using these new models.

1.3.5

Figure 33 shows the evolution of the AIC/BIC values over the 6 voxels of the challenge dataset. The two last voxels give `NaN` and `Inf` values for the eigenvalues, and therefore we cannot compute all the models, but the data from the first 4 voxels already gives us enough insight.

The new 2-compartments models outperform the 1-compartment models (except for the ball-and-stick model which is only outperformed by the ball-and-two-sticks model). Through this graph, we see the importance of trying to learn the inner structure of the brain, because brain molecules and structures are much more complex, and need to be modeled as good as possible so that we can extract information from scans.

1.3.6

Finally, Figures 34 and 35 show us how we can take advantage of these different models. Figure 34 confirms what we had said in the previous parts: that depending on the area of the brain (lakes, or fibers), the structures that pass through it differ, and that is what the zeppelin models account for, this deformation of the cells when going through fibers. We can see that the zeppelin models are especially relevant in the ”outside” part of the brain (where the ball-and-stick model gave poorer performance in the previous parts!).

In the end, as displayed in Figure 35, we need to take the best models depending on the area where we are. We can see that the `RESNORM` maps highlight different areas of performance, and we need to retain only the ones where the performance is better. The last maps do show how important model selection is. Selecting the best model gives us a low `RESNORM` overall (compared to the high `RESNORM` overall in the ball-and-stick model). The Akaike’s weights help to some extent, but I think we need to come up with new objective functions to take into account the coherence of the different models in each area, and how they can give information on one another in each voxel.