SSY186 - Diagnostic Imaging, DTI and Curve Fitting. Lab 3

Rita Laezza, Pavel Gueorguiev

May 2016

Introduction

The purpose of this lab was to introduce the concept of Diffusion Tensor Imaging (DTI) and how it can be used to investigate the white matter fibers of the brain as well as the mammary fibroglandular tissue. For this laboratory exercise the software used was MATLAB and 3DSlicer.

Part 1: Exploring a breast DTI scan

Because the 3D Slicer does not analyse DTI data stored in DICOM format we begun by converting the breast image to Nrrd format. For that purpose we selected the "Dicom to Nrrd Converter" module. After exploring the data from an experimental DTI scan of the breast using 3D Slicer we were asked to answer a few questions:

1. In this DTI sequence 25 different directions have been used plus a 0 gradient direction for a total of 26 directions. For this information we selected Modules → Volumes → Active Volume: "breast" → Diffusion Editor → Gradients.

2. In the same location of the 3D Slicer one can also see that the b-value used was 1000, which corresponds to the gradient strength.

3. By looking at the values of each of the gradient vectors it becomes evident that their magnitude is one, meaning they are unit vectors.

4. The first gradient direction is $\mathbf{g} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$ because it provides the signal without diffusion sensitizing gradient, S_0. This provides the baseline image, which is necessary to calculate the rest of the signals using the equation:

   $$S(\mathbf{g}) = S_0 \cdot exp(-b\mathbf{g}^T\mathbf{D}\mathbf{g})$$

   where, $S(\mathbf{g})$ is the DTI image, $\mathbf{g}$ corresponds to each gradient direction, b corresponds to the diffusion weighing value and $\mathbf{D}$ the diffusion tensor.

5. The 3D Slicer software uses the Nrrd format because it is the one typically used in diffusion weighted image analysis and tractography. Hence, Dicom format images first have to be converted to Nrrd format, so that they can be recognized by Slicer3 as a legitimate diffusion weighted volume. Basically, the "Dicom to Nrrd Converter" module parses the Dicom header to extract necessary information about measurement frame, diffusion weighting directions, b-values, etc, and write out a Nrrd image.[1]

Part 2: Computing the diffusion tensor and associated scalar measures

For this part of the lab we used MATLAB to compute the diffusion tensor, $\mathbf{D}$, and associated scalar measures for the DTI breast scan in part 1. Similarly to the previous exercise, we were asked to answer a few questions:

[1] http://www.slicer.org/slicerWiki/index.php/Modules:DicomToNRRD-3.6

1. The folder "breastDTIdata" that was provided contained the DICOM files necessary for exploring slices 1 and 19 (of the original 40 axial slices) of a breast DTI scan. In this folder there were a total of 52 different .dcm files which corresponded to the diffusion signals measured in each of the 26 gradient directions for both slices. For example, the files "IM-0009-0001.dcm" and "IM-0009-0019.dcm" correspond to the images in the first (probably S_0) gradient direction for slice 1 and 19, respectively. Then come the files "IM-0009-0041.dcm" and "IM-0009-0059.dcm" which correspond to the images in another gradient direction for the same slices, and so on. The indexes are separated by intervals of 40 because the original acquisition consisted of 40 images.

2. After carefully reading through the M-file we made the appropriate changes:

   (a) At line 46 of the `diffusion_tensor_and_scalars.m` file we added the following code:

```
 % INSERT YOUR CODE HERE TO SHOW THAT EACH DIRECTION IS A UNIT VECTOR
mag = zeros(25,1);
for i=1:number_of_directions
    mag(i) = sum(dirs(i,:).^2); % the magnitude of each gradient is approx. 1
end
```

   Which output: mag = [1.0000 0.9996 1.0000 0.9997 1.0000 1.0002 1.0009 1.0004 0.9999 1.0000 0.9992 1.0004 0.9993 1.0003 1.0007 1.0006 1.0000 0.9992 1.0002 0.9997 0.9992 0.9995 1.0007 1.0003 0.9995]

   (b) At line 78 (former 74) we added the following code:

```
% INSERT YOUR CODE HERE TO DISPLAY THE b=0 IMAGE FOR SLICE 19
imagesc(base_img);
colormap gray
title('b=0 image for the slice 19');
```

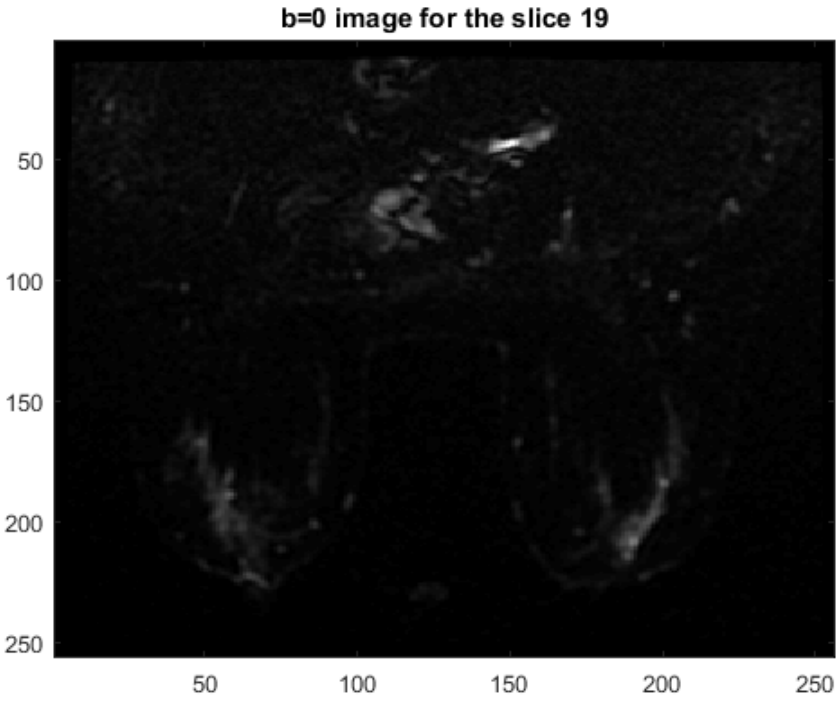

Figure 1: Plot of the baseline image for slice 19

   (c) At line 136 (former 127) we added the following code:

```
% INSERT YOUR CODE HERE TO COMPUTE:
% (1) fractional anisotropy (FA), and
% (2) volume ratio (VR).
FA(x,y) = sqrt(3/2)*sqrt(num/den);
VR(x,y) = (evals(1,1)*evals(2,2)*evals(3,3))/mean_diffusivity(x,).^3;
```

Which provided the following images for Fractional Anisotropy and Volume Ratio:

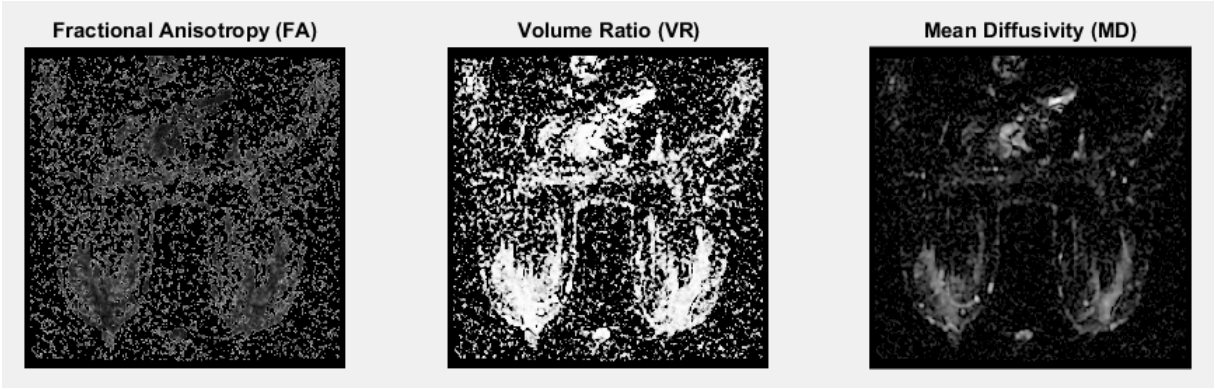

Figure 2: Plot of the different scalar measures

3. **Mean diffusivity** (MD) or trace is a scalar measure of the total diffusion within a voxel. **Fractional Anisotropy** (FA) is a scalar measure (between 0 and 1) of the degree of anisotropy of a diffusion process, where 0 corresponds to an isotropic process. **Volume ratio** (VR) is the ratio of an ellipsoid volume to the volume of a sphere of radius $\bar{\lambda}$.

4. B is the acquisition matrix as stated in slide 12/25 in the computational diffusion MRI lecture given by Mohammad Alipoor. It corresponds the G matrix also referred to in equation 5.

Part 3: Exploring white matter fibers

In this part of the lab exercise we worked with the 3D Slicer again. This time we followed a tutorial created by Sonia Pujol. While we went through the tutorial about diffusion MRI of the brain we answered the following questions:

1. Considering the first part of the tutorial about *Diffusion Data Loading and Tensor Estimation* (slide 17), we determined that 14 gradient directions (whith two of them being $\mathbf{g} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$) were used for the DTI sequence.

2. The b-value used was equal to 800 s/mm^2.

3. As previously mentioned, there were two gradient directions equal to $\mathbf{g} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$), which means that two baseline images (b=0) were acquired.

4. Considering the second part of the tutorial about *Scalar Measurements* (slide 33), we found the voxels with the highest FA values. They were located in the white matter of the brain, which contains a high amount of nerve fibers. Since FA is a measure of the Fractional Anisotropy of a diffusion process, it makes sense that the part of the brain with the largest values is located where most of the diffusion process occurs: in the aligned nerve fibers (axons) of the brain. On the other hand, the gray matter shows a low FA value, which can be explained by the fact that it is composed mostly of cell bodies of the neurons, where there is hardly any diffusion.

5. Considering the third part of the tutorial about *Region of Interest Based Tractography* (slide 33), we obtained an image of the tracts as tubes coloured with FA values, seen in figure 3.

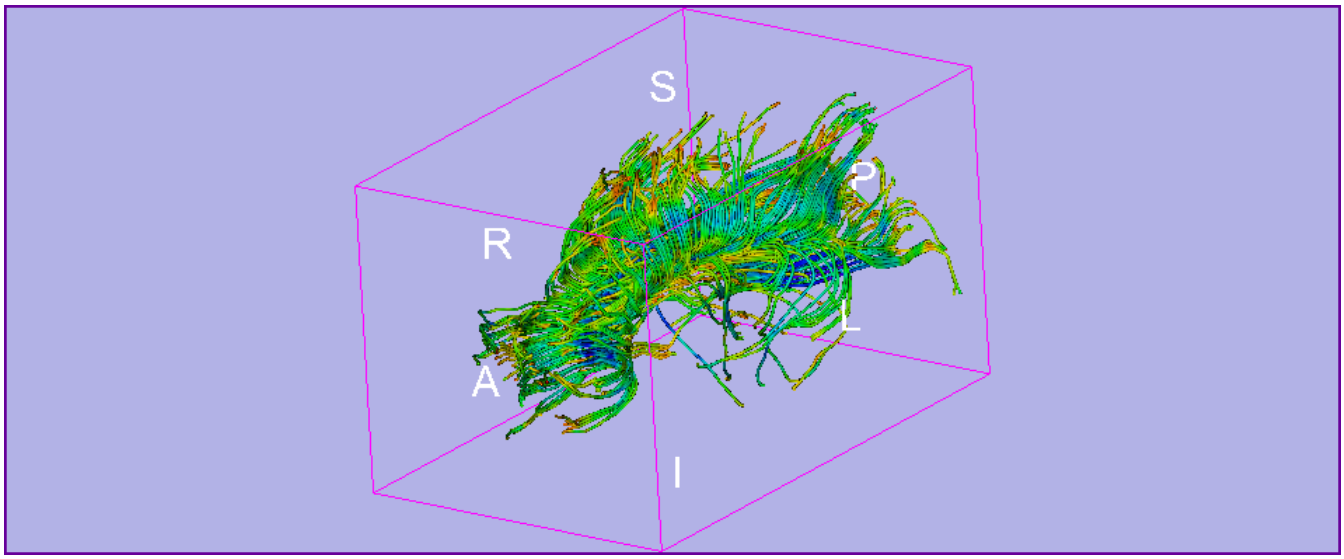

Figure 3: Tractography with colour mapping of the Fractional Anisotropy values

Part 4: Analysis of MRI Data: Diffusion Signal Curve Fitting

Outliers can be identified by graphing the raw data and removing the points that seems to be far from the general trend, before passing to the curve fitting algorithm. Below is the function used to fit the data onto the curve. It is a linear function of the format $y = mx + b$ to perform the linear fitting, onto the logarithmic data.

```
function F = linFun(x, xdata)
F = x(1)*xdata + x(2);
end
```

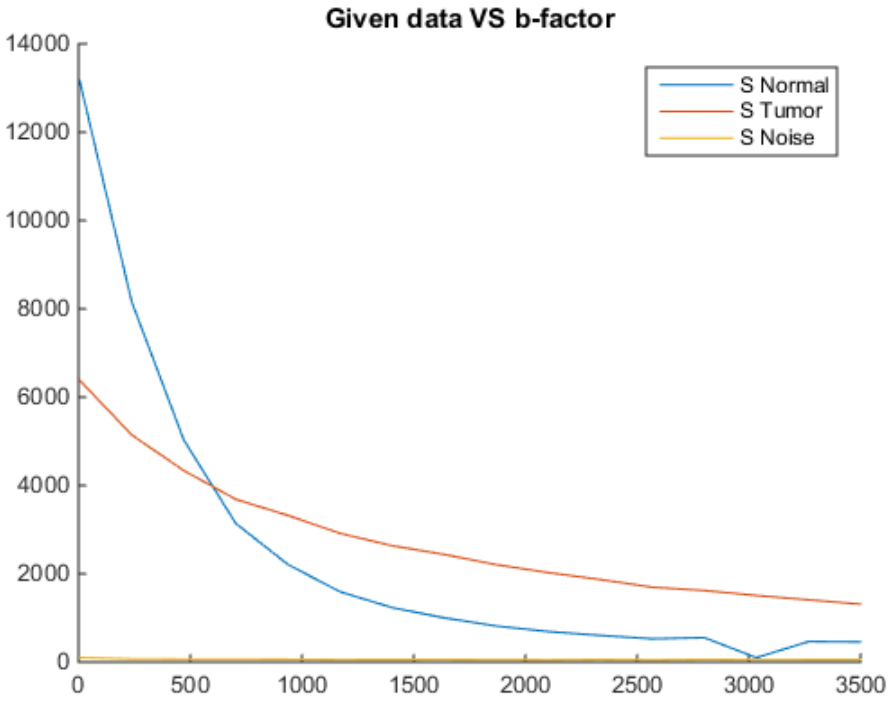

Figure 4: Raw data of Normal, Tumour and Noise levels versus the b factor

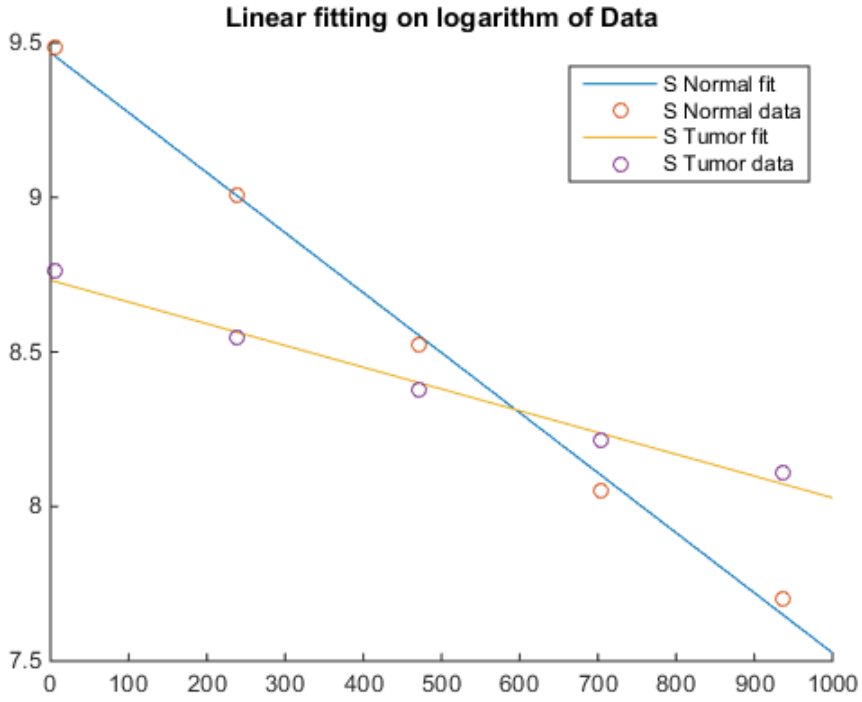

Figure 5: Linear fit onto the b-values below $1000\frac{s}{mm^2}$

For the non-linear fitting function we use a the monoexponential function given and call the Levenberg-

Marquardt method to solve the least squares minimization problem. The function follows.

```
function F = expFun(x, xdata)
F = x(1)*exp(-xdata*x(2));
end
```

And we use the lsqcurvefit function with the following parameters.

```
x0 = [0 0];
ub = [14000 0.1];
lb = [10000 0];
xdata  = data(1:5,1);
ydata1 = data(1:5,2);
y_hat1 = lsqcurvefit(@expFun,x0,xdata,ydata1,lb,ub, ...
    'Algorithm = levenberg-marquardt')
```

The resulting curve is obtained.

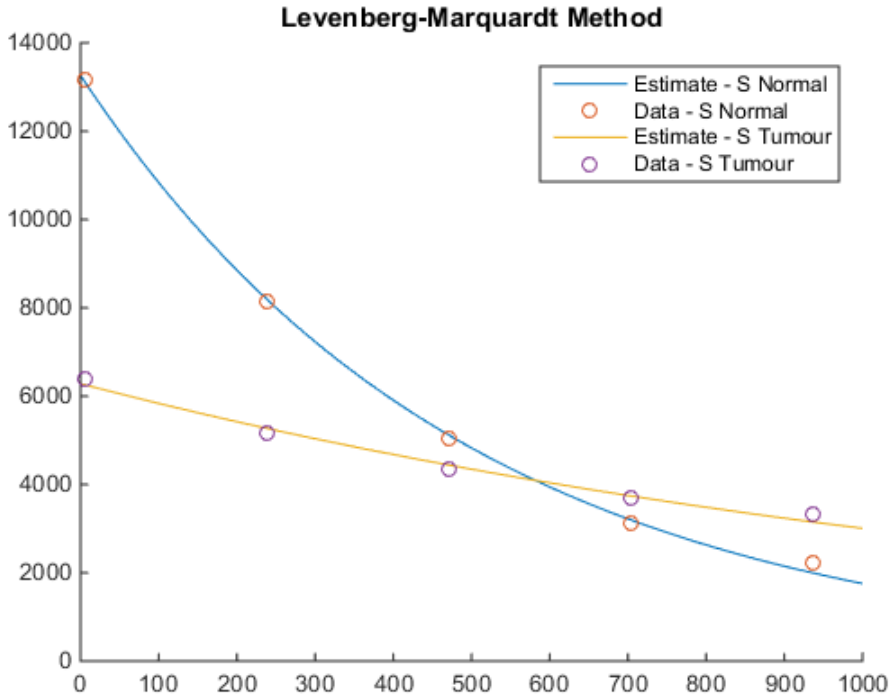

Figure 6: Levenberg-Marquardt fit onto the b-values below $1000 \frac{s}{mm^2}$

From figure 7 we can see that the fitting doesn't look any better but the relationship appears linear in the logarithmic case. Which makes sense looking at equation:

$$S(b) = S_0 e^{-bD} \tag{1}$$

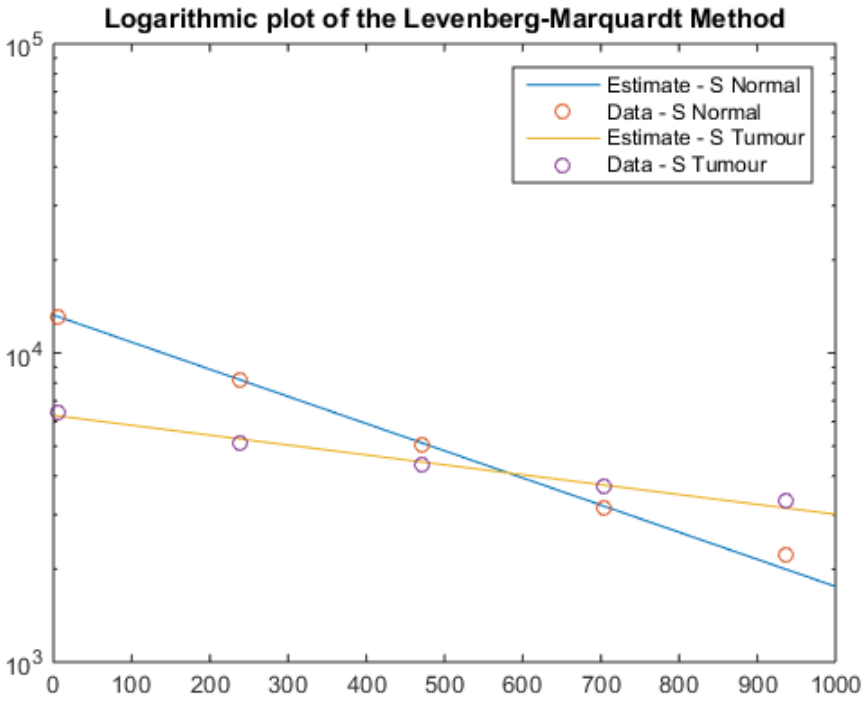

Figure 7: Logarithmic Levenberg-Marquardt fit onto the b-values below $1000\frac{s}{mm^2}$

The squares of the residuals are:

$$res1 = 1.97e6$$
$$res2 = 1.04e6$$

and the graph follows.

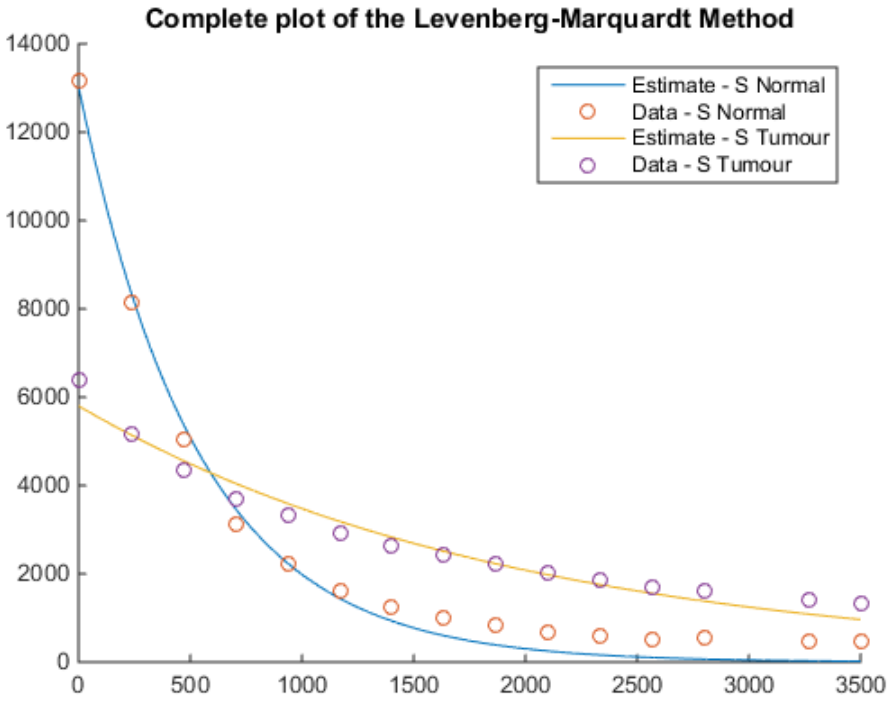

Figure 8: Levenberg-Marquardt fit onto all b-values

Part 5: Analysis of MRI Data: Diffusion Tensor Imaging (OPTIONAL)

1 FA and PDD

The fractional anisotropy (FA) was calculated according to the equation in figure 9. The principal direction of diffusion (PDD) is obtained by first obtaining the largest eigenvalue. Then the corresponding eigenvector is selected for the principal direction vector. The results was are stated below.

$$FA = 0.6266$$
$$PDD = [1\ 0\ 0]$$

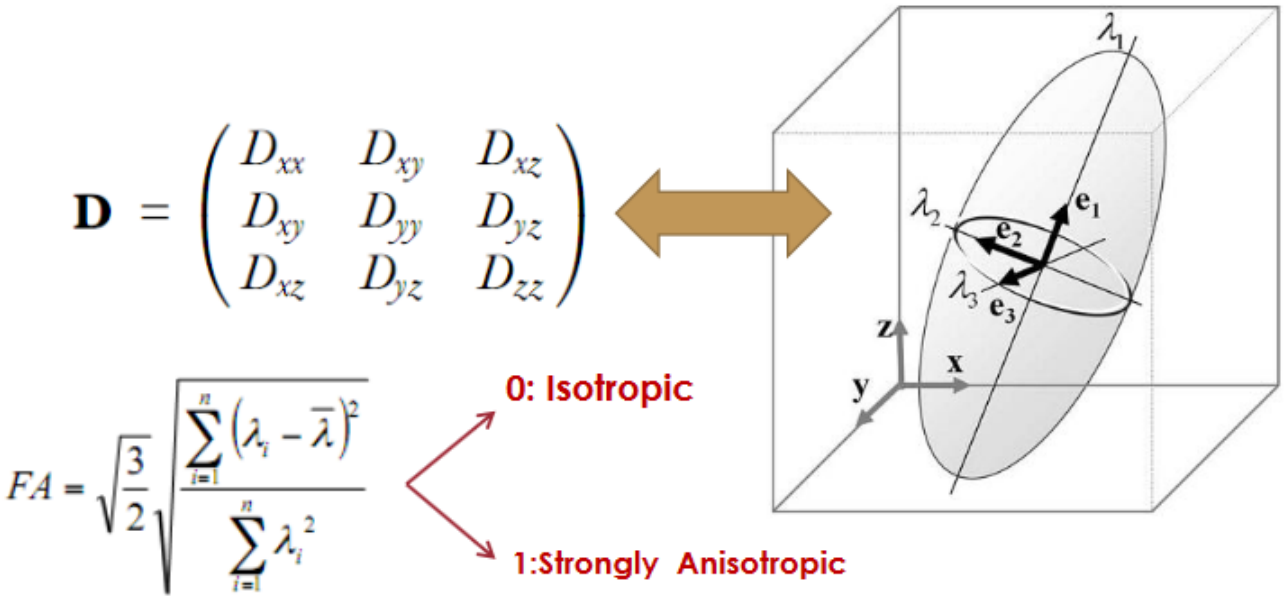

Figure 9: FA and PDD illustration [2]

2 Rician noise and $\hat{\mathbf{D}}$ estimation

Rician noise was generated according to the equation in the PM:

$$S_m = \sqrt{(S + n_1)^2 + (n_2)^2} \tag{2}$$

The variables n_1, n_2 are uncorrelated zero-mean Gaussian noise variables generated by the matlab code:

```
b = 1500; % diffusion weighting
s0 = 1000; % signal without diffusion sensitizing gradient

% Gaussian noise
SNR = 15;
var = s0/SNR;
n1 = var*randn(30,1);
n2 = var*randn(30,1);

% Rician noisy signal (Sm)
Sm = sqrt((s+n1).^2+n2.^2);
```

The 30 unique, 3 dimensional directions are generated by a MathWorks function called `random_unit_vector(arg)` where 'arg' represents the number of vectors.

$$g_x^2 D_{11} + g_y^2 D_{22} + g_z^2 D_{33} + 2g_x g_y D_{12} + 2g_x g_y D_{13} + 2g_y g_z D_{23} \tag{3}$$

In equation 2 the g terms are the 25 gradients generated by the `random_unit_vector(arg)` function. The D terms are the diffusion coefficients we are trying to find. The whole equation is equal to:

$$\mathbf{y} = -b^{-1} ln(S(\mathbf{g}/S_0)) \tag{4}$$

where $\mathbf{g}$ is the noisy signal at the respective b values and S_0. Then using equations 2 and 2 in their matrix form it follows:

$$\mathbf{Gt} = \mathbf{y} \tag{5}$$

where t represents the diffusion constants in a [6x1] vector form. The matrix is inverted and solved for $\mathbf{t}$ and $\hat{\mathrm{D}}$ is constructed from the tensor. The following is an example of the one of the resultant estimates of D.

$$D = \begin{bmatrix} 0.0016 & 0.0002 & -0.0003 \\ 0 & 0.0008 & 0.0001 \\ 0 & 0 & 0.0003 \end{bmatrix} \tag{6}$$

3 100 time repeat for AE and $\hat{\mathbf{F}}$A estimation

$$FA = 0.6109$$
$$AE = 0\,\mathrm{deg}$$

4 Repeat of 3, consistency check

No the results are not the same. The angular error remains the same but the numerical value of FA does change. This is expected because repeating the experiment you would get a new noise vector every time since noise is random by nature.

5 Improvement

As stated above the variation comes from the fact noise is random. The noise power is directly controlled by its variance, and therefore anything that reduces the variance will improve the results. An example of this is increasing the signal-to-noise (SNR).

6 Rotation-invariance

If the diffusion tensor is the same yet it is in a different orientation we expect the eigenvalues to remain the same, and therefore FA should not change, while obviously PPA will accordingly.

7 Unknown S_0

If S_0 is in fact missing we can try to find it using a non-linear curve fitting solution, like the Levenberg-Marquardt algorithm and proceed from there.

[2] http://www.sciencedirect.com/science/article/pii/S1361841505000976