Optimization issues in medical imaging and fiber-tracking

Ørjan Bergmann

Thesis for the degree of Philosophiae Doctor (PhD)
Department of Informatics
University of Bergen
Norway

February 2008

Preface

This dissertation is submitted as a partial fulfillment of the requirements for the degree of Doctor of Philosophy (PhD) at the University of Bergen. The dissertation consists of two parts. Part I provides a theoretical and methodological introduction to the papers in the second part. Part II contains the scientific contribution of the thesis.

Acknowledgments

First of all I would like to thank my supervisors at the University of Bergen; Prof. Trond Steihaug from the Department of Informatics and Prof. Arvid Lundervold from the Department of Biomedicine. They have both been my advisers since I started my masters degree, and I am deeply grateful for them agreeing to supervise my PhD as well.

I would like to express my sincere gratitude to my main supervisor Prof. Trond Steihaug who introduced me to the exciting field of optimization more than 7 years ago. His experience, patience, professionalism and extensive methodological and experimental knowledge both in optimization, and in science in general, has been inspiring and helpful in all phases of my research.

I would like to thank Prof. Arvid Lundervold for always being helpful, encouraging and supporting. His enthusiasm for the mathematical sciences coupled with a strong passion for the field of biomedical imaging has been highly motivating and his guidance has been a great help over the course of my PhD.

I would also like to thank Dr. Carl-Fredrik Westin, at the Laboratory for Mathematics in Imaging at Harvard Medical School, for hosting my stay at Harvard and for supervising my research there. I am deeply indebted for this great opportunity to work with one of the best medical imaging research groups in the world.

Dr. Gordon Kindlmann, thank you for a great collaboration on two of the publications and for making the beautiful pictures. I am especially grateful for the renderings that you made for this thesis, seen in figure 3.5 and 3.6.

I would also like to thank Dr. Johan Lie and Dr. Oddvar Christiansen for the countless hours we have worked together on our paper. We made a great team and I am really proud of our work together.

Thanks also goes to Dr. Lennart Frimannslund, Dr. Oddvar Christiansen and Dr. Johan Lie for reading this work and for providing comments and suggestions which have made the final thesis much better than I thought possible.

Finally, I would like to thank my wife Silje Bergmann for always being patient and for reading this thesis.

Ørjan Bergmann
June 29, 2007

Contents

Part I

Introduction

Overview

Fiber tracking is a relatively recent methodology, made possible by access to new highly advanced MR scanners able to produce high-quality diffusion tensor images (DTI), which promises clinicians a possibility to observe the actual trajectories of fibers and the connectivity of the living human brain. Analysis of such data provides clinicians with a unique tool for diagnosing and predicting disease which affect the “wiring” of the brain, such as stroke, tumors, multiple sclerosis and Alzheimer’s disease. The purpose of this PhD thesis has been to *explore ways to improve the quality of DTI as well as the accuracy of the fiber tractography which can be estimated from such data*. In order to achieve this purpose we have investigated two fundamentally different areas to improve DTI results; data *pre-processing* and more accurate data *descriptors*.

Topics

It has been shown that pre-processing of MR data to increase the relatively low signal-to-noise ratio in DTI can improve derived quantities such as anisotropy measures and estimated fiber tracts. Visually this low signal-to-noise ratio can manifests itself as random speckles of increased or lowered intensity scattered across the images. In terms of fiber tractography noise can cause estimated fibers to be terminated prematurely, or to veer off in unexpected directions. Improvements made in pre-processing of data will propagate itself through all quantities computed from the processed data including tensor invariants and tractography, and we therefore put emphasis on improving such methods.

In particular we have investigate data pre-processing in order to alleviate the following problems which are known to occur in MR data analysis

- Recovering of data which has been polluted by additive random noise.
- Recovering of data which has been degraded by blurring, as well as random additive noise.

Of course it is important that the pre-processing techniques we apply does not introduce any new artifacts into the data since these will propagate through the computation sequence as well, in a manner which may be hard to predict. This puts additional demands on the accuracy of the techniques used.

The analysis of diffusion tensor images is a well established area of research. Over the last decade the diffusion tensor model has shown itself to be a *descriptor* of local diffusion characteristics which provide a good trade-off between simplicity of the model and expressiveness in describing the actual data. Thousands of research papers has been published on diffusion tensor interpretation and visualization, and the general understanding of tensor analysis is good.

However, this research has also pointed out that there are regions of crossing fibers in the brain where the single diffusion tensor does not accurately describe the underlying physiology. These regions can be found e.g. near the corpus calosum and the corona radiata in the human brain. The DTI community has not agreed on how such cases should be handled. But since the diffusion tensor model has shown itself to be “accurate in most cases” it seems reasonable to consider generalizations of the single tensor model rather that replacing it with alternatives.

We will therefore investigate a more general tensor model which better describe the voxelwise diffusion in such cases, in order to compute more accurate fiber trajectories.

Contributions

Our contribution in order to investigate these research topics can be summarized as follows

- We have developed a denoising algorithm that can remove additive noise and thus improve the quality of captured DTI datasets [3]. This algorithm is outlined in chapter 1 and applied to DTI data in section 3.4.
- We have created an algorithm which can solve blurring problems which are known to occur in certain MRI applications [7]. This procedure is outlined in chapter 2.
- We have developed a conceptual model which describes all steps involved from DTI generation up to voxelwise diffusion tensor estimation [6]. This procedure can be used to generate synthetic DTI datasets for validation of algorithms. The steps involved are described in chapter 3.
- We have devised a simple and inexpensive way to estimate better descriptors of local diffusion than the traditional single diffusion tensor [4], and demonstrate cases in which it provides superior results. This *multi-tensor imaging* is described in chapter 4.
- We have created a new fiber-tractography algorithm [5] which can successfully estimate fibers through regions in which the traditional methods fail. This algorithm is outlined in chapter 4 by generalization of the standard method presented in section 3.5.

All these methods can be applied to improve the results of analysis performed on diffusion tensor images, thereby achieving the overall purpose of this thesis as stated above.

Important conclusions

Based on the research presented in this thesis the following conclusions are among the ones which may be the most significant

- By application of pre-processing algorithms to the raw DTI data it is possible to significantly increase the quality of quantities derived from the diffusion tensor. We present results which show that pre-processing of a DTI dataset which takes about 9 minutes to capture using an MR scanner can can give results comparable in quality to those which takes 45 minutes for the scanner to obtain.

- In some cases it is possible to calculate fiber trajectories in regions of crossing fibers from commonly available DTI data. We have done so on several datasets by application of our published multi-tensor techniques.

In chapter 5 we have included a more detailed list of conclusions.

Outline of the thesis

In the first two chapters we will present techniques for pre-processing in order to improve the quality of 2D and 3D images. We will start with a simple noise model in chapter 1 and describe a state-of-the-art algorithm for solving this problem in section 1.2. The examples we will use are not medically motivated to illustrate the generality of the method. The problem that we solve in chapter 1 is a special case of the more general problem of deconvolution, discussed in chapter 2. Solving deconvolution problems are in general more difficult as they may be ill-conditioned and the solution of these typically requires some form of regularization. In section 2.1 we discuss why these problems are hard and in section 2.3 we present an algorithm which can solve them with increased accuracy under certain assumptions. In order for easy comparison to similar methods and to simplify the discussion the examples are not taken from a medical application.

The following two chapters explores ways of improving the analysis of the medical images that arise in the field of DTI. In chapter 3 we give a brief introduction to DTI and to the quantities that often are estimated from the recorded data in order to examine or differentiate between health and disease in patient data. In section 3.4 the 3D techniques from section 1.2 are applied to matrix-valued DTI data in order to improve the quality of the datasets. The descriptors presented in chapter 3 may fail or provide inaccurate information in particular situations. In chapter 4 we will introduce more general models that may better capture the underlying anatomy. In section 4.1 we present a method that can be considered a generalization of the tensor-model described in section 3.2 and in section 4.3 we will use this model to improve existing fiber-tracking techniques for visualizing DTI data.

The final chapter of the thesis sumarize the main contributions of the thesis and lists some interesting problems for future investigation.

Chapter 1

Image Denoising

In this chapter we will start with a simple noise model that can describe the degradation of an initially undegraded image by noise. We will then briefly demonstrate how statistics of this noise can be estimated for later use during denoising. In section 1.1 we show how images can be transformed into an alternative form in which the noise often is easy to remove and we demonstrate a situation in which the approach is known to produce poor results. In section 1.2 of the chapter we present an algorithm which we show can can successfully solve the noise model, by avoiding the failings of the simpler transformation. The problem investigated in this chapter is a special case of a more general problem discussed in the next chapter, and will therefore also serve as an introduction to chapter 2. In section 3.4 we will utilize our 3D generalization of the technique developed in this chapter to perform denoising on multi-dimensional medical images.

Some degree of noise is always present in any electronic device that transmits or receives a signal. On televisions we may recognize this noise as “specs of snow” which fly across the screen, on the radio we might hear the noise as a weak background hiss when the volume is turned up loud. For digital images this noise appears as random speckles on an otherwise smooth surface. A reoccurring problem in digital image processing is the removal of this random noise from a digital image. This problem is commonly referred to as *image denoising* [13].

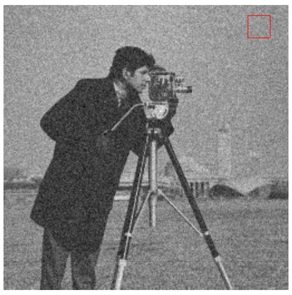
An example of image denoising can be seen in figure 1.1. Figure 1.1(a) shows the noisy image that we have available. This image is a mixture of both an *undegraded* image and *noise*. The undegraded image is seen in figure 1.1(b), and the noise we have used in this example is seen in figure 1.1(c). In actual applications both the undegraded image and the noise is unknown and we wish to approximate the undegraded image from the noisy image available.

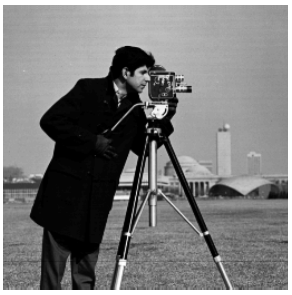
A common model for this problem is

$$b = x + \eta \tag{1.1}$$

where x denotes the undegraded image without noise, η is additive noise and b is our known noisy image. We wish to calculate the original unknown image x from the noisy observation b.

Note that the actual intensities of both x and η are unknown to us, and when this problem is solved without any additional knowledge it is referred to as *blind denoising*.


(a) The noisy image b. The square indicates a subregion Ω.


(b) The true undegraded image x.

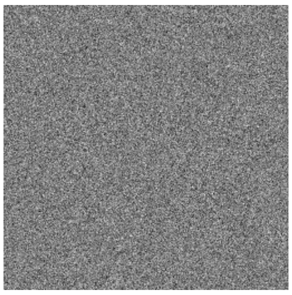

(c) Synthetically generated normal distributed noise η.

Figure 1.1: The components of equation (1.1) in the cameraman example. The images are individually normalized to improve visual contrast.

These types of problems are generally considered hard to solve as illustrated by the fact that there in are literally infinitely many ways to decompose a signal into the sum of two components. Therefore, in order to find a unique solution, various conditions need to be imposed on each of the two components.

In order to simplify these hard problems additional information about x or η is needed. Fortunately good estimates of the statistics of the noise η can often be found manually by searching for regions in which one would expect x to be constant. By an assumption of *spatially invariant* noise, the estimates from this region can be used for the whole image.

Let Ω be a set of connected pixels. The variance σ_Ω^2 of this region can be estimated as

$$\sigma_\Omega^2 = \frac{1}{|\Omega|-1}\sum_{p\in\Omega}[b(p)-\mu_\Omega]^2$$

where $|\Omega|$ denotes the number of pixels in the region Ω, $b(p)$ is the intensity of the noisy image b in pixel p and the mean of Ω is defined as

$$\mu_\Omega = \frac{1}{|\Omega|}\sum_{p\in\Omega} b(p)$$

The distribution of the noise η can similarly be estimated from the intensities of b in Ω. We also note that if the mean of the noise is non-zero it can, without loss of generality, be assumed to be zero by adding a scalar constant to the right-hand side of equation (1.1). Ignoring this constant will then have the effect of shifting the mean of the estimated denoised image. But if the intensities of the solution are normalized this shift will be of no consequence, and the assumption of zero-mean noise can safely be used.

In figure 1.1(a) we define Ω as the set of pixels in the red square box in the upper right hand corner of the image. The variance in this choice of Ω was estimated to be $\sigma_\Omega^2 = 0.0096$. This estimate corresponds well with the known noise shown in figure 1.1(c) which was generated to have zero-mean and a variance of $\sigma^2 = 0.01$. The size of the images in figure 1.1 are 256×256 and the intensities of the undegraded cameraman image are normalized between 0 and 1 with a mean of 0.45.

In the rest of this chapter we let the components of the noise η be taken from a normal distribution with zero-mean and known variance σ^2.

1.1 Frequency Domain Denoising

The images shown in figure 1.1 are examples of images in the *spatial domain*. In this domain each intensity represents a measurement at a given spatial position $p = (x, y)$.

By the application of a special transformation an image in the spatial domain can be transformed to an equivalent form in the *frequency domain* [32]. Each element of the image in the frequency domain is referred to as a *coefficient*. In the frequency domain each coefficient expresses a weight of a special *basis function*. Different transformations use different basis functions.

An important and often used transformation from the spatial to the frequency domain is called the *discrete cosine transform* (DCT). The basis function of the DCT is the cosine trigonometric function. The DCT differs from other well-known sinusoidal transformations, such as the *discrete Fourier transform*, in that it uses only real numbers and that different boundary conditions are implied in the frequency domain. The DCT is defined as

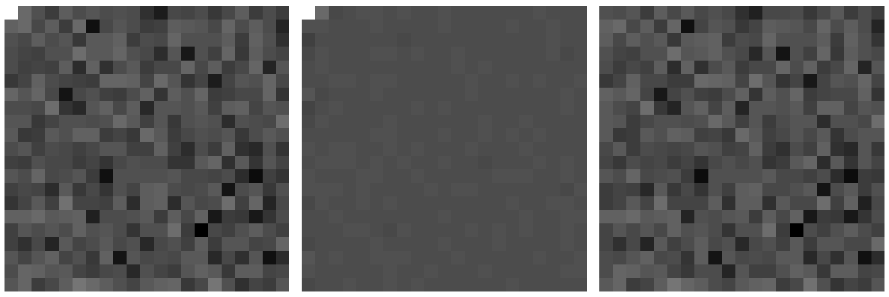
$$\hat{z}_k = c_k \sum_{i=0}^{M-1} \cos\left[\frac{\pi}{M}\left(i + \frac{1}{2}\right) k\right] z_i$$

where z is a vector with M elements of intensity in the spatial domain, $\hat{z}$ is the transformed z in the frequency domain and where $c_0 = \sqrt{\frac{1}{M}}$ and $c_k = \sqrt{\frac{2}{M}}, k \neq 0$ are constants which ensure that the transformation matrix is orthogonal. Since the DCT is defined simply as a weighted sum of intensities it is clear that it is a *linear transformation*. The DCT also has an inverse which transform a signal from the frequency domain back to the spatial domain. Multi-dimensional DCT transformations can be achieved by successively employing the presented one-dimensional transform along each coordinate axis direction [29].

It is important to point out that no information is lost by taking the DCT of an image, as can be observed by the fact that immediately applying its inverse will yield the original image. In this sense the image in the spatial and the frequency domain are equivalent. However, the DCT has a strong "energy compaction" property; most of the image information tends to be concentrated in a few low-frequency coefficients in the frequency domain. This can be observed in figure 1.2(b). Quite good approximations of the original image can be obtained even if many of the small frequency coefficients are truncated to zero before application of the inverse transform. This can for example be used for compression of images. This is the reason why the DCT is implemented in signal compression algorithms such as JPEG and MPEG [32].

By definition white Gaussian noise appears on all frequencies with the same probability. This can be observed in figure 1.2(c) in which the frequency domain coefficients are evenly distributed over the whole spectrum of frequencies. The magnitudes of this noise in the frequency domain is, as the name implies, Gaussian distributed and directly related to the noise variance σ^2 defined in the spatial domain.

As seen in figure 1.1(a), the intensities of the undegraded image is larger than those of the noise i.e. we have a high *signal-to-noise ratio*. In the frequency domain it also



(a) The coefficients of the noisy image $\hat{b}$.

(b) The true undegraded coefficients $\hat{x}$.

(c) Synthetically generated noise-coefficients $\hat{\eta}$.

Figure 1.2: A zoomed in view of the region Ω in the frequency domain, when Ω is defined as in figure 1.1(a). The images are shown on the same logarithmic scale with the origin in the upper left hand corner.

holds true that the magnitude of the main coefficients of $\hat{x}$ are larger than those of $\hat{\eta}$, as seen in figure 1.2.

Since the DCT is a linear operation the noisy image in the frequency domain $\hat{b}$ must be the sum of $\hat{x}$ and $\hat{\eta}$ also in the frequency domain, as seen in figure 1.2. Therefore it is a reasonable assumption that the large coefficients of $\hat{b}$ will come from $\hat{x}$ and most of the small coefficients will come from the noise $\hat{\eta}$. This suggests that we can remove the noise of b simply by thresholding to zero its small magnitude coefficients in the frequency domain.

The result of applying this procedure to the region Ω, after transformation back to the spatial domain can be seen in figure 1.3(c), with zoomed in details from figure 1.1 for reference in figure 1.3(a) and 1.3(b). Notice how the thresholding in the frequency domain has successfully made the noisy image smoother, resulting a denoised image significantly closer to the original undegraded image. Also observe that some of the irregularities found in the undegraded image has been removed by the denoising process. These were encoded in the small coefficients barely visible in 1.2(b), and were truncated to zero along with the noise.

In Ω no "important" information was encoded in the small magnitude coefficients of $\hat{b}$. However, figure 1.4 shows the same procedure applied to a different region of the same image. The region is the one surrounding the cameraman's face, as seen without noise in figure 1.4(b). The denoised image in 1.4(c) is hardly much better than the noisy one shown in 1.4(a). The denoised image is both *blurred* and it displays an "oscillatory pattern" commonly referred to as *Gibbs phenomenon* [32]. The blurriness has the effect of smoothing out the edges, such as the transition between the cameraman's hair and face, and the oscillations distorts otherwise near constant regions, such as the face. The result is an unsuccessful denoising which must come from the removal of coefficients containing "important" information.

The only differences between the successful denoising shown in figure 1.3 and the unsuccessful one shown in figure 1.4 was the region onto which we applied the method. Often edges and rapid changes in an image are encoded in low-magnitude high-frequency coefficients and the thresholding of these to zero will result in blurring

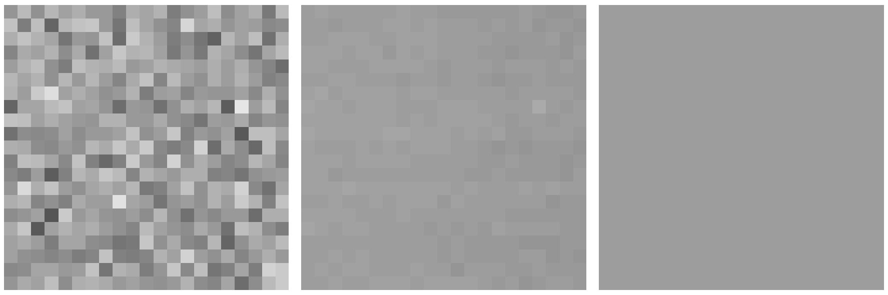

(a) The noisy image b. (b) The undegraded x. (c) The denoised image.

Figure 1.3: A view of the the region Ω visualized in figure 1.1(a) in the spatial domain. The images are on the same scale as in figure 1.1(b).

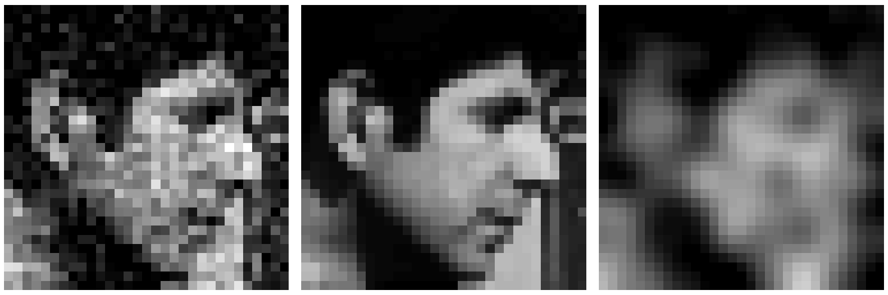

(a) The noisy image b. (b) The undegraded x. (c) The denoised image.

Figure 1.4: The denoising procedure applied to a region showing the cameraman's face. Again the images are on the same scale as figure 1.1(b).

and oscillations. However, if the region does not contain any edges then these problems are avoided. The basic idea of the method we present in the next section is to first apply some analysis in the spatial domain in order to find regions in which there are no edges. These regions can then safely be denoised using thresholding in the frequency domain, as we saw in figure 1.3.

1.2 Shape-Adaptive DCT

The Shape-Adaptive Discrete Cosine Transform (SA-DCT) for denoising is presented by Foi et al in [10] and later extended in [11, 12]. One of the main contributions of these works was the merging of the DCT with a *shape-adaptive* pre-processing step to find regions without any edges. In the following we will refer to such regions as *homogeneous* regions.

The pre-processing step consists of estimating weighted sums of neighboring pix-

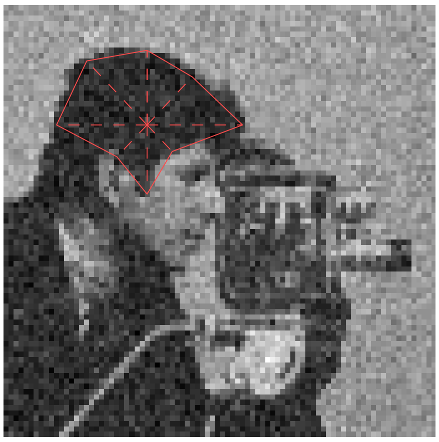
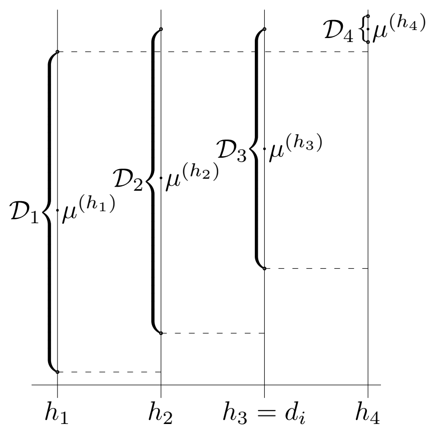


Figure 1.5: Details of a *shape-adaptive* region found by the LPA-ICI algorithm.


$\mathcal{D}_4$ $\mu^{(h_4)}$
$\mathcal{D}_3$ $\mu^{(h_3)}$
$\mathcal{D}_2$ $\mu^{(h_2)}$
$\mathcal{D}_1$ $\mu^{(h_1)}$
h_1
h_2
$h_3 = d_i$
h_4
Figure 1.6: Example of the *intersection of confidence intervals* (ICI) algorithm.

els, referred to as *local polynomial approximation* (LPA), followed by an *intersection of confidence intervals* (ICI) algorithm.

The LPA-ICI algorithm starts by considering a given pixel x, and try to span a region Ω_x around x so that the intensities in this region does not contain any discontinuities, apart from noise. This region is spanned in a predetermined set of directions θ_i. In 2D these are taken to be the four cardinal and the four intermediate compass directions. The output of the LPA-ICI algorithm is a distance $d_i \geq 0$ in each direction θ_i around the center-pixel x. Each neighboring *corner* $d_i\theta_i$ is then connected by lines and the region on the interior of this *polygonal hull* is referred to as Ω_x.

The 8 directions θ_i are in figure 1.5 visualized with dashed lines around the center pixel x. The length of each dashed line is the distance d_i from x to the corner where the region ends in that direction. The polygonal hull Ω_x is the region on the inside of the straight lines formed between neighboring corners.

In order to estimate the distance d_i in each direction θ_i from x the LPA considers weighted intensity averages $\mu^{(h)}$ of increasing length h along the direction. At length h the weighted average includes the intensities of the pixels $\{x, x+\theta_i, x+2\theta_i, \ldots, x+h\theta_i\}$. The weights applied to each of these pixel intensities has the property that they sum to one and the weight applied to pixel j in this set is greater or equal to the one applied to pixel $(j+1)$.

Intuitively this weighting has the effect of smoothing the data. If the pixels belong to a homogeneous region, we would expect the noise-level in the weighted estimate $\mu^{(h)}$ to decrease as the number of pixels h included in the average increases. We would also expect the weighted average $\mu^{(h)}$ to "not change much" within a homogeneous region. Conversely, if as the region is expanded in a given direction we observe "large changes" in $\mu^{(h)}$ this may be a sign that we have encountered pixels that should not be included in Ω_x.

The ICI algorithm formalizes this behavior. The local noise-level $\sigma_{\mu^{(h)}}$ in the average $\mu^{(h)}$ can be related to the global variance σ^2 and to the LPA weights. Together with the average $\mu^{(h)}$ the noise-level $\sigma_{\mu^{(h)}}$ defines a confidence-interval

$$\mathcal{D}_h = \mu^{(h)} \pm \Gamma\sigma_{\mu^{(h)}}$$

where $\Gamma > 0$ is a global parameter of the algorithm. When Γ is large more pixels will be included in each region and when Γ is small fewer will be accepted. The ICI rule states that the region should continue to be expanded as long as confidence interval $\mathcal{D}_h$ intersects all previous confidence-intervals $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_{h-1}$. The largest length h that this holds for is referred to as d_i.

Figure 1.6 illustrates this algorithm. We see the weighted averages $\mu^{(j)}$ on the y-axis plotted against the x-axis showing the lengths h_j in a given direction θ_i. The confidence intervals $\mathcal{D}_j$ are visualized with braces and we see the algorithm terminate when the intersection between all previous confidence intervals becomes empty. In this example this happens when $h = h_4$ giving $d_i = h_3$.

This LPA-ICI algorithm is applied to each of the (8 in 2D) directions and gives a homogeneous region Ω_x. This region can then be denoised by thresholding in the frequency domain as outlined in the previous section. Note however that Ω_x is not in general rectangular so therefore a slightly modified DCT algorithm is applied. In 1995 Sikora [29] presented a transform that is similar to the one presented in the previous section when the region is rectangular but will also give accurate results on non-rectangular support. We use a variant of this algorithm, although other alternatives also exists.

As the shape-adaptive DCT algorithm is applied to each pixel x observe that many pixels may be denoised in several (neighboring) regions. This provides us with multiple denoised estimates for most of the pixels in the input image and improves robustness in the denoising. A weighted average of each such overlapping pixel is then estimated giving the final denoised image. We let the weights applied to each region be inversely proportional to its size and how much noise we believe it to contain.

In our publication [3] we detail the SA-DCT denoising process. Our main contribution has been to extend the original 2D formulation to 3D and to apply this 3D algorithm to matrix-valued DTI data as described in section 3.4.

Figure 1.7 shows the results of applying the SA-DCT denoising algorithm to the cameraman example shown in figure 1.1. The undegraded and the noisy image are reproduced in figure 1.7(a) and 1.7(b) respectively for comparison. Figure 1.7(c) shows the denoised image estimated by the SA-DCT algorithm using $\Gamma = 1$. We observe that the SA-DCT has successfully managed to remove much of the noise as seen on the cameraman's coat while still preserving the edges between the coat and the background. We also observe that the details in the grass has been blurred out and that some of the buildings in the background seem a little vaguer compared to the undegraded image. This happens because the intensities in the undegraded image are on the same scale as the noise in these regions, i.e. we have locally low signal-to-noise ratio. Figure 1.7(d) shows the difference between the noisy image b and the recovered image. From equation (1.1) we recognize this difference as an estimate of the noise η. As expected by observing figure 1.1(c) this quantity shows very little structure indicating no systematic error in the denoising. Finally, the *root-mean-square* (RMS) error defined as

$$\sqrt{\frac{1}{mn}\sum_p (x(p) - b(p))^2} \tag{1.2}$$

between the undegraded image x and the noisy image b, both of size $m \times n = 256 \times 256$ is $\sigma = 0.100$, compared to an RMS error of 0.038 between the undegraded and the denoised image.

We implemented this algorithm in the Matlab programming language in order to produce the results from this chapter. The purpose of this 2D implementation was primarily to demonstrate the *feasibility* of the algorithm, and to reproduce previous results from literature.

It is often difficult to compare the running time of one algorithm implemented in Matlab to another, because of how the Matlab language is interpreted by the computer at runtime. For instance by going from for-loops to *vectorized* code, one can often achieve dramatic improvements in the running time. However, the SADCT algorithm is not very well suited for vectorization in Matlab and we therefore decided to implement some parts of the algorithm in C. By rewriting performance critical parts of our algorithm in C using the MEX interface we were able to reduce the running time of our implementation by many orders of magnitude, down to about 1.7 seconds for the code which produced the results shown in figure 1.7(c). We did not have access to such highly optimized code for other recently published works, and therefore opted not make any quantitative comparisons of the running time of our implementation compared to theirs.

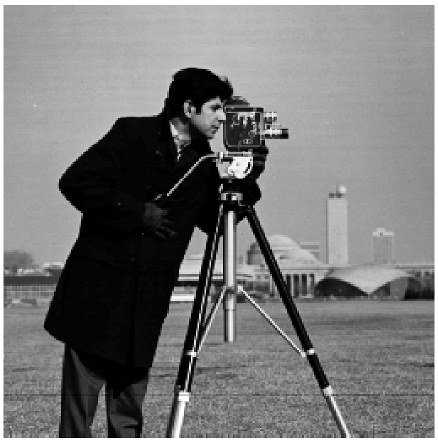
1.3 Conclusion

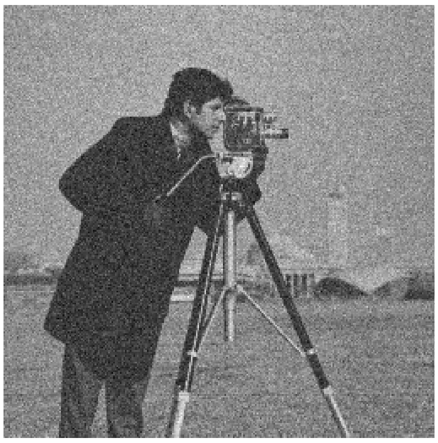
This chapter contains little new research per se; it should be considered more a distillation of previous works done in the context of SA-DCT denoising. In our work [3] we extend this 2D algorithm to a full 3D algorithm that we use to improve the quality of acquired DTI datasets. As expected from the 2D results presented in the previous section, our 3D algorithm can be shown to significantly improve signal-to-noise ratio in medical data as well, with relatively low cost in terms of computational expense.

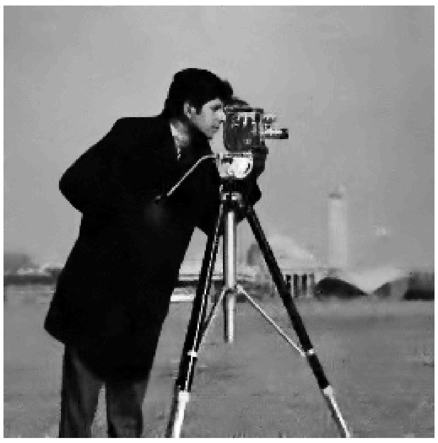
The fact that the algorithm is local in nature, meaning that only neighboring pixel information is needed to denoise each pixel, means the algorithm is *embarrassingly parallel* and simple to implement when multiple CPUs are available.

The increase in quality achieved by this algorithm is important since it implies that all derived quantities estimated from the denoised data should also be more accurate. This allows researchers and scientists to work with lower quality data, which is easier to acquire, while still obtaining accurate results. In chapter 3 we present examples of some of the improved derived quantities obtained using our algorithm.

Next we will show that the problem considered in this chapter is a simpler and special case of the more general deconvolution problem presented in chapter 2. This next chapter will serve to highlight that the main difficulty in solving deconvolution problems is not the randomness of the noise (which we have shown in this chapter can be overcome for instance by the method presented here) but rather the ill-posedness of the problems which makes the solution much more sensitive to noise.


(a) The undegraded x.


(b) The noisy image b.


(c) The SA-DCT denoised image.

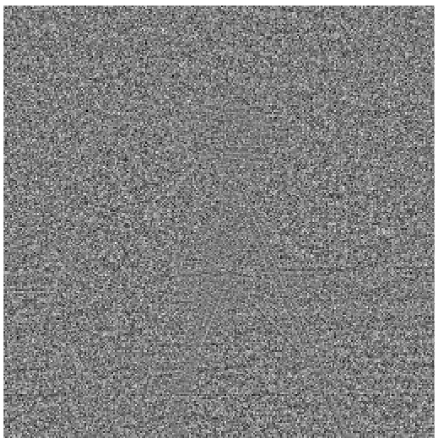

(d) The difference between image (b) and (c).

Figure 1.7: The SA-DCT denoising procedure applied to the whole cameraman image. Image (a) and (c) are shown on the same scale, image (b) and (d) are individually normalized.

Chapter 2

The Deconvolution Problem

In this chapter we will investigate a more general problem which is strongly related to the denoising problem we considered in the first chapter. Although originally an image processing problem, we state it here as a well known optimization problem and in section 2.1 investigate its properties in terms of numerical linear algebra. We show that this problem requires large-scale regularization and in section 2.2 we formulate a regularization model with known conditions for verifying a solution. We then give a brief overview of different standard algorithms that can solve the model. In section 2.3 we describe our main contribution to this chapter; a generalization of the regularized formulation, and we give some examples of the additional types of problems which can be solved using it. Finally we solve our model with help from one of the previously described standard algorithms, and give numerical results.

In order to demonstrate the generality of the method we will use non-medical test images in this chapter, as we also did in the first chapter. However, this does not imply that our presented method does not have applications in medical imaging. As pointed out e.g. by [30], the problem of deconvolution occurs in certain MRI applications where the acquisition time of the scanner is long compared to the rate of diffusion, resulting in a blurry MR image. In solving this MRI problem our deconvolution algorithm could have impact. Unfortunately, before application of our algorithm to such problems there are also quite a few non-trivial issues which must be resolved, including obtaining accurate estimates of the convolution kernel. This is known to be difficult, and we have therefore neglected this application in this chapter as well as in our deconvolution paper. Instead we have focused on well-known deblurring test datasets from astronomical imaging in which comparison to already existing methods is straightforward.

In the previous chapter we mentioned the concept of *blurring* when discussing figure 1.4(c) and we observed that when we made certain changes to an image in the frequency domain this resulted in a blurred image in the spatial domain. By the term “blurred image” we conceptually mean an image which is smoother, has less contrast and fewer sharp edges than an unblurred image, as seen in figure 2.1(a) compared to 2.1(c).

We can always generate a blurred image by removing the coefficients corresponding to the high-frequencies in the frequency domain. This procedure is commonly referred to as *low-pass filtering*, since only the low-frequency coefficients are preserved, while the high-frequency ones are discarded [32, 13].

In the previous chapter we performed denoising by removing all small *magnitude* coefficients, regardless of which frequency they occurred on. But since most of the large coefficients occurred in the low-frequency part of the spectrum we could have produced similar results with a low-pass filter. Indeed, in figure 1.2(a) there was only one single large coefficient found in the upper left-hand corner of the frequency-domain image, which encodes the lowest frequency. This particular coefficient corresponds to the weight of the basis function which (in the spatial domain) is constant everywhere giving the *mean* of the region. In this case the "large magnitude" filter of the previous chapter and a low-pass filter would give the same result.

Not surprisingly blurring can also be implemented in the spatial domain. When this is done it is sometimes referred to as *averaging* since each pixel p in the output image is then a weighted average of pixels in the input image. The pixels included in the average typically lie within a given distance of p and we will refer to them as the *neighbors* of p. When the neighbors of each pixel p are weighted in a consistent manner based on their position relative to p across the image this operation can be implemented as a mathematical operation called *convolution* [32].

Convolution between a kernel κ containing weights and an image X is defined as

$$Y(p) = (\kappa * X)(p) = \sum_q \kappa(q) X(p-q) \tag{2.1}$$

where p and q are pixels and where $*$ denotes the convolution operator. The sum over q is taken to include all pixels where $\kappa(q)$ is non-zero. A convolution can be interpreted as a *weighted moving average* where the weights are found in κ.

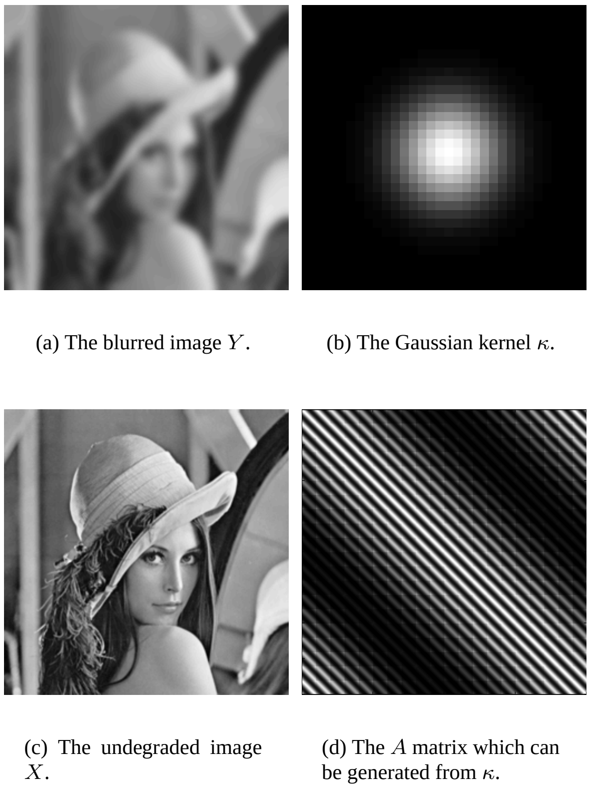
Blurring of images taken in the world around us can arise in many different situations. One source of blurring is if the camera is moved at the precise moment a picture is taken, or if the object being photographed is moving too fast compared to the time it takes the camera to acquire the image. Both of these cases results in what is known as *motion blur*. Another common source of blurring is when the lens of the camera is out of focus. In certain applications (such as those involving atmospheric turbulence e.g. in space imaging) blurring can be modeled as a *Gaussian blur*. When the kernel κ is a discretization of the 2D Gaussian distribution

$$\kappa(q) = \frac{1}{2\pi\sigma^2} e^{\frac{-\|q\|^2}{2\sigma^2}} \tag{2.2}$$

with variance σ^2 then equation (2.1) implements a low-pass filter termed a 2D Gaussian blur. An example of a discretized Gaussian blur kernel of size 32×32 with $\sigma^2 = 16$ can be seen in figure 2.1(b). In figure 2.1(c) we see the undegraded standard test image of former Playboy centerfold Lena Söderberg [1]. This normalized Lena image is of size 256×256 with a mean of 0.47. The result of the convolution between the Gaussian kernel in figure 2.1(b) and the image in figure 2.1(c) is seen in figure 2.1(a).

Notice that since a convolution is defined as a weighted sum of intensities from X, it can be expressed in a more general form using notation from linear algebra. If X is an image of size $m \times n$ then let $x = \mathrm{vec}(X)$ be a column-vector with mn elements obtained by stacking the columns of X on top of each other. In order to simplify notation we will throughout the rest of this chapter use upper-case letters to denote matrices and lower-case letters to denote vectors. Equation (2.1) can then be expressed equivalently as

$$y = Ax \tag{2.3}$$


(a) The blurred image Y. (b) The Gaussian kernel κ.

(c) The undegraded image X. (d) The A matrix which can be generated from κ.

Figure 2.1: Some components of equation (2.1) and (2.3) in an example. Images (a) and (c) and images (b) and (d) are on the same scale.

where $y = \mathrm{vec}(Y)$ and A is a matrix of size $mn \times mn$ created from κ. When κ is defined as in the previous example then A will have a near *block diagonal* structure similar to the one seen in figure 2.1(d).

As discussed in the previous chapter, all sampled images are subject to some degree of measurement noise. The degradation of an initially undegraded image x, first by blurring and then by measurement error, can then by substitution of equation (2.3) into equation (1.1) be modeled as

$$b = Ax + \eta \tag{2.4}$$

where b is our (vectorized) measured image and where η is additive unknown noise of the same dimensions as b. Again we wish to estimate the unknown undegraded image x from the noisy measurements b. When the matrix A is generated from a known kernel κ we will refer to the process of solving this model as *deconvolution* [13]. Also notice that when A is the identity matrix then this model reduces to that of equation (1.1). This implies that denoising is a special case of deconvolution, and that deconvolution in general is a more difficult problem.

The presence of the unknown noise η in equation (2.4) means that we cannot expect to find an exact solution to the problem. We can however attempt to estimate a solution which is "as close as as possible" to the exact solution x of the equation in some sense.

One way to do this is to "ignore the noise" and minimize the least-square difference between the right- and left-hand side of the equation. This yields the linear least-squares minimization problem

$$\min_x ||Ax - b||_2^2 \tag{2.5}$$

At this point it should be noted that the matrix A will be *very* large. When the image X is of size 256×256, as in figure 2.1(c) the vectorized image x will be a column-vector with $65\ 536$ elements. The matrix A will then be of size $65\ 536 \times 65\ 536$ and contain $4\ 294\ 967\ 296$ elements. If we attempt to store this matrix explicitly on a computer in double (8 byte) precision, this would require more than 30 gigabyte of storage capacity. Working with matrices of this size is rarely possible in practice. However, in order to gain some insight into the structure of this problem we will now examine a much smaller example: let X be of size 32×32 then A will be of size $1\ 024 \times 1\ 024$. Figure 2.1(d) shows this matrix A when κ is defined as in figure 2.1(b).

In the context of deconvolution the linear least-squares problem in equation (2.5) is difficult to solve as can be seen using the *singular-value decomposition* (SVD). Any real matrix A of size $m \times n$ can be written in terms of the SVD defined as

$$A = U\Sigma V^T \tag{2.6}$$

where U and V are orthogonal matrices and Σ is a diagonal matrix containing the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0 = \sigma_{r+1} = \cdots = \sigma_{\min\{m,n\}}$

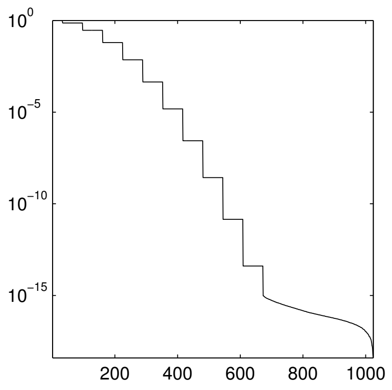
When A is defined as in figure 2.1(d) then its singular values are distributed as visualized in figure 2.2(a). The x-axis displays the index $i \in \{1, 2, \ldots, 1\ 024\}$, and the y-axis shows singular values σ_i on a logarithmic scale. The machine accuracy of the computer used to compute the singular values was on the order of 10^{-16}. Notice that the singular values decrease rapidly towards zero and that the difference between each consecutive singular value is small, especially when the values themselves are small. The latter observation implies that it is hard to determine the numerical rank r of A, and that when working in finite precision then A has a large null-space as illustrated in the example by the fact that about a third of the singular values are smaller than the machine accuracy.

In general we can always expect A in large deconvolution problems to be be singular in finite precision with large null-space. Unfortunately there exists no way to reformulate them into well-behaved problems either as the poor conditioning is a property of the problem itself, rather then our model [15]. Problems with this property are referred to as *ill-posed*.

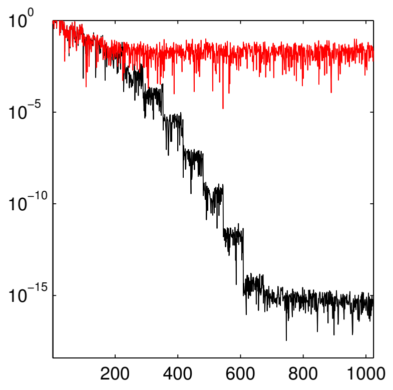
2.1 Regularization

The fact that the matrix A is ill-conditioned can be improved by employing *regularization* [23, 15] when solving the model. Assuming that we have determined some numerical rank r of A, using the truncated SVD we can compute a rank r approximate solution to equation (2.5) as

$$x^r = \sum_{i=1}^{r} \frac{U_i^T b}{\sigma_i} V_i \tag{2.7}$$



(a) The singular values σ_i of the A where the index i is on the x-axis. Notice the logarithmic scale on the y-axis.



(b) The coefficients $|U_i^T b|$. The black curve shows when b is without noise. In the other b is degraded by small amounts of additive noise η.

Figure 2.2: Logarithmic plot of the components of equation (2.7).

where U_i and V_i denotes column i in the matrices U and V respectively. The norm of this solution is

$$\|x^r\| = \sqrt{\sum_{i=1}^{r} \left[\frac{U_i^T b}{\sigma_i} \right]^2} \tag{2.8}$$

and we see that the norm will not be too large as long as $|U_i^T b| < \sigma_i$ when $i \in \{1, 2, \ldots, r\}$. The discrete Picard condition (DPC) [23] is said to be satisfied when $|U_i^T b|$ decay faster than σ_i

In figure 2.2(b) we see a plot of how $|U_i^T b|$ behaves in our example. The black curve shows the case when b is without noise, i.e. when $\eta = 0$. We observe that the coefficients decrease rapidly towards zero. Comparing to figure 2.2(a), they decrease at least as fast as the singular values down to machine accuracy. The red curve seen in the same plot shows the behavior when b is degraded by additive noise η with known standard deviation of 0.003. We observe that the coefficients converge to a relatively constant value on the order of 10^{-2} after about the 200 first vectors. It is clear that when the image is degraded by additive noise the DPC condition is not satisfied and the norm of the solution grows large.

Note in equation (2.7) that when i is large then the right singular vectors V_i are from the null-space of A. So if r is "too large" we see that we are adding weighted basis vectors from the null-space to the solution x^r. Although this may have little effect on the *solution error* defined as $\|Ax^r - b\|$, since $A(x^r + o^{(1)} + o^{(2)} + \cdots + o^{(k)}) = Ax^r$ when $o^{(i)}$ are nonzero vectors from the null-space of A, it will certainly increase the norm of the solution. Additionally the solution may look very different from the undegraded solution we are trying to approximate even if the solution error is quite small.

In the rest of this chapter we define regularization as the process of solving a problem while preventing the norm of the solution from becoming too large. We apply regularization hoping that the regularized solution will be “closer to the true solution” although not only in the sense that we wish to minimize the difference as in equation (2.5). Regularization therefore always implies that we have some *additional knowledge* that we wish to take into consideration and impose on the solution. There will always be some trade-off between fitting the model and limiting the norm of the solution. In regularized problems we can usually find a solution with smaller error but with larger norm or conversely, with smaller norm but larger error.

This trade-off (or additional knowledge) is often represented as a parameter called the *regularization parameter*. In equation (2.7) this parameter is the numerical rank r. As can be observed, choosing $r = 0$ results in the solution $x^r = 0$ with zero norm but with large error. A larger r will give a solution with smaller error but larger norm.

In the next section we will explore regularization algorithms which can be used to solve the large-scale deconvolution problems seen in figure 2.1 by regularizing equation (2.5).

2.2 Large-Scale Regularization

As we have seen, the size of the matrix A is prohibitively large even when working with images of relatively modest size such as those seen in figure 2.1. This makes the use of methods which work by factoring A, such as the truncated SVD seen in the previous section, impractical for most real problems. In this section we will use so called *matrix-free* algorithms to solve the deconvolution problem. These *iterative* algorithms are termed matrix-free because they do not require the explicit storage of the matrix A, nor do they rely on any complete factorizations of A. Instead they only require some way to evaluate the matrix-vector product Ax between A and a vector x.

Notice that the matrix-vector product of equation (2.3) by definition can be implemented using the convolution operator in equation (2.1). Specifically, we have

$$Ax = y = \mathrm{vec}(Y) = \mathrm{vec}(\kappa * X) = \mathrm{vec}(\kappa * \mathrm{vec}^{-1}(x)) \tag{2.9}$$

where vec^{-1} denotes the inverse of the vec operator, so that $\mathrm{vec}^{-1}(x) = X$ and where κ and the $*$ operator are as previously defined in equation (2.1) . This provides us with a way to evaluate the matrix-vector product Ax without ever forming the A matrix explicitly. We also point out that there exist efficient ways to evaluate this convolution by using some frequency domain manipulations when the size of the kernel is large [32].

We now turn our attention to how to regularize equation (2.5). Since our aim is to limit the norm of the solution, one possibility is to add a constraint to this problem which limits the norm. In certain applications good estimates for the norm Δ of the true solution is known a priori. This is for instance the case in the blurred images from the Hubble telescope seen in figure 2.6. By using this information we can add a constraint which ensures that the norm of our solution is no greater than that of the true solution. This yields the constrained minimization problem

$$\begin{aligned} &\min_x \frac{1}{2}\|Ax - b\|^2 \\ &\text{s.t. } \|x\| \leq \Delta \end{aligned} \tag{2.10}$$

where Δ is a scalar representing some upper bound on the norm of the solution. Writing out the norm of the objective function we arrive at the slightly more general form

$$\begin{aligned} \min_x \frac{1}{2}x^T Hx + g^T x \\ \text{s.t. } \|x\| \leq \Delta \end{aligned} \tag{2.11}$$

which is identical to equation (2.10) when $H = A^T A$ and $g = -A^T b$. This problem is commonly referred to as the *trust-region subproblem* (TRS) and is the topic of ongoing research in large-scale non-linear numerical optimization. The trust-region problem arise naturally in the context of unconstrained minimization of difficult non-linear problems and it is often used as an alternative to line-search methods. See [8] for more details about trust-region methods.

When solving the TRS there are two different classes of solutions that can be found. One of them is the *interior* solutions, which are found when the solution is unrestricted by the constraint, i.e. when $\|x\| < \Delta$. In such cases the estimated solution lies strictly in the interior of the feasible region and the solution is the same as that of the unconstrained linear least-squares problem in equation (2.5). This case therefore corresponds to a non-regularized solution.

The other class of solutions occur when the constraint is satisfied with equality so that $\|x\| = \Delta$. This type of solution is referred to as a *boundary solution*; the solution has limited norm, as we would expect in a regularized solution. In regularization problems we have pointed out that the unregularized solution will be dominated by elements from the null-space of A and that the norm will grow large. When Δ is chosen properly we hope that the TRS formulation will prevent elements of the null-space to enter into the solution while simultaneously ensuring that the solution error is minimal. We know that minimizing the solution error will increase the solution norm, so when Δ is adjusted to be small we know that we will find a boundary solution.

The TRS can be rewritten as an unconstrained minimization problem by formulating the *Lagrange relaxation*

$$\mathfrak{L}_\lambda(x) = \frac{1}{2}\|Ax - b\|^2 + \lambda(\|x\| - \Delta) \tag{2.12}$$

where $\lambda \geq 0$ is a Lagrange penalty parameter. When minimizing $\mathfrak{L}_\lambda(x)$ in terms of x the Δ term can be discarded as a constant and we arrive at the problem.

$$\min_x \mathfrak{L}_\lambda(x) = \frac{1}{2}\|Ax - b\|^2 + \lambda\|x\|$$

Notice that λ will impose a penalty on the norm of the solution. When $\lambda = 0$ we obtain an unconstrained solution with large norm, and when λ is large we obtain a solution which (because of the larger weight put on the norm of x in the minimization) will give a solution with smaller norm. It is clear that we can use λ as a regularization parameter. If we square the regularization term then the resulting regularization problem is referred to as *Tikhonov regularization* [37]. It can be shown that a solution to the TRS is also a solution to the Tikhonov regularization problem.

Returning now to the TRS; it can be shown that in order for x^* to be an optimal solution to the problem the following conditions are necessary and sufficient [8] :

$$(H + \lambda I)x^* = -g \tag{2.13a}$$

$$(H + \lambda I) \text{ is positive semidefinite} \tag{2.13b}$$

$$\lambda \geq 0 \tag{2.13c}$$

$$\lambda(\|x^*\| - \Delta) = 0 \tag{2.13d}$$

Notice that since in our application $H = A^T A$ we already know that H is a symmetric positive semidefinite matrix. This means that each of its eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ is non-negative. Since the eigenvalues of $H + \lambda I$ are $\lambda_i + \lambda$, we know that equation (2.13b) must be satisfied when (2.13c) is satisfied.

Equation (2.13a) states that we can obtain a solution by solving a linear set of equations. But, as opposed to the other linear systems of equations we have seen up until now, we know that the coefficient matrix is positive definite and has full rank when $\lambda > 0$. Solving large linear sets of equations with positive definite coefficient matrix is a classical theme from numerical analysis and can efficiently be implemented using iterative schemes. So for a given non-negative λ we can approximate the exact parameterized solution

$$\begin{aligned} x(\lambda) &= -(H + \lambda I)^{-1} g \\ &= -(E \Lambda E^T)^{-1} g \\ &= -E^T \Lambda^{-1} E g \\ &= -E^T \operatorname{diag}\left(\frac{1}{\lambda_i + \lambda}\right) E g \end{aligned} \tag{2.14}$$

where E are the orthogonal eigenvectors of $H + \lambda I$ and $\Lambda = \operatorname{diag}(\lambda_i + \lambda)$ is a diagonal matrix containing its eigenvalues.

Since we know that the matrix H is near singular we know that we need $\lambda > 0$ in order for $x(\lambda)$ to provide a meaningful regularized estimate for x^*. Equation (2.13d) then states that we must have

$$\begin{aligned} \Delta &= \|x(\lambda)\| \\ &= \left\| \operatorname{diag}\left(\frac{1}{\lambda_i + \lambda}\right) E g \right\| \\ &= \sqrt{\sum_{i=1}^{n} \frac{[Eg]_i^2}{(\lambda_i + \lambda)^2}} \end{aligned} \tag{2.15}$$

where $[Eg]_i$ denotes the ith component in the vector Eg. This equation implies that the solution we seek must lie on the boundary of the feasible region, as previously pointed out.

From the right-hand side of equation (2.15) we observe that when $[Eg]_i \neq 0$ then the norm of the solution will grow very large as λ approaches $-\lambda_i$. In this situation a plot of the right-hand side of the equation can, for various λ, be seen in figure (2.3). Here we have visualized λ on the first axis and the norm of the solution $x(\lambda)$ on the second axis. The asymptotes where the norm goes to infinity are found at $\lambda = -\lambda_i$, where $\lambda_i \geq 0$ are the eigenvalues of the positive semidefinite matrix H. Since these eigenvalues are non-negative we find all the asymptotes on the left-hand side of (or exactly at) the origin. Since we want $\lambda > 0$ we search for a positive choice of λ giving

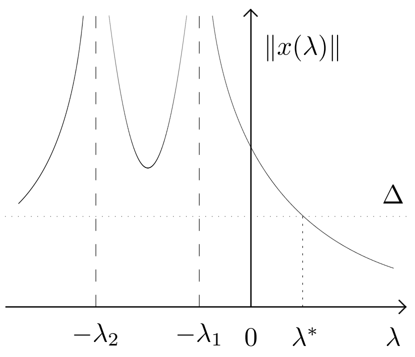


Figure 2.3: The simple case.

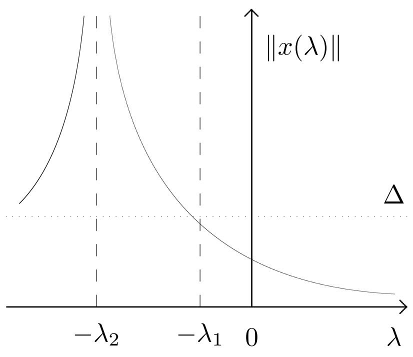


Figure 2.4: The hard case.

the prescribed norm Δ found on the second axis. In the figure we observe that $\lambda = \lambda^*$ is the value we seek.

Unfortunately the "simple" situation described above is usually *not* the case in regularization problems. We made the assumption that $[Eg]_i \neq 0$, i.e. that eigenvector E_i is *not* orthogonal to g. In certain applications (and among these, regularization) this is precisely the case for some small i [24]. As E_i becomes closer and closer to orthogonal compared to the vector g then the norm curve visualized in figure 2.3 becomes closer and closer to the asymptote at $-\lambda_i$. When the vectors are exactly orthogonal the asymptote is completely lost. This situation is visualized in figure 2.4, where we observe that we are unable to find an boundary solution for the prescribed Δ as the pole at $\lambda = -\lambda_1$ is lost. In literature this situation is often referred to as the *hard case*.

2.2.1 Computational methods for the TRS

In this section we will take a small detour and consider a few different ways to estimate a solution for the TRS. For completeness we will also consider briefly some of the "standard" approaches although direct application of them are not really appropriate for the large-scale regularization problems we have considered in this chapter. See [8] for references on each of the following methods.

As we saw in figure 2.3, a solution to the trust-region subproblem can in the simple case be found by solving $\|x(\lambda)\| = \Delta$ i.e. to find the roots of the function $\|x(\lambda)\| - \Delta = 0$. Newton's method is usually very efficient for root-finding, but does not perform well when the derivative of the function changes very quickly around a root. As seen in figure 2.3 we can expect this to be the case near the asymptotes and therefore implementing Newton's method directly may fail. Fortunately this problem can be avoided by inverting the function and instead finding the roots of the *secular* equation $\frac{1}{\|x(\lambda)\|} - \frac{1}{\Delta} = 0$. Newton's method approximates the secular equation with a quadratic Taylor polynomial involving the Hessian and gradient and it can be shown that the secular equation will in many regions be near linear. In such cases Newton's method excel. Differentiating the quadratic polynomial model and equating with zero yields a linear set of equations which can be solved using *Cholesky factorization*. It has been shown that this algorithm converges to a TRS solution either in a finite number of steps or, except in the hard case, ultimately at a Q-quadratic rate [8].

For our application however, the cost of calculating a Cholesky factorization of a large $n \times n$ matrix is prohibitively high in addition to the problems involving the hard case.

The dogleg method

A very different approach to solving equation (2.11) uses linear combinations of line-search directions to estimate a solution. In the TRS we observe that if Δ is large, then the minimizer of the problem is the unconstrained interior solution $x^u = -H^{-1}g$. Also notice that when Δ is very small the norm $\|x\| \leq \Delta$ will also be very small and the $x^T Hx$-term of the objective function should be small compared to $g^T x$. Then the constrained solution to the problem can be approximated as $x^c = -\frac{\Delta}{\|g\|}g$.

The *dogleg method* [19] assumes that the solution to the TRS problem is a linear combination of the two extremes x^c and x^u. In particular, the dogleg *path* is defined as

$$x(\tau) = \begin{cases} \tau x^c, & 0 \leq \tau \leq 1 \\ x^c + (\tau - 1)(x^u - x^c), & 1 < \tau \leq 2 \end{cases}$$

where $0 \leq \tau \leq 2$. We see that $\tau \leq 1$ gives the constrained linear solution x^c, and $\tau = 2$ corresponds to the unconstrained solution x^u. The dogleg solution $x^{\text{dogleg}} = x(\tau^*)$ is then calculated from the unique choice of τ^* such that $\|x(\tau^*)\| = \Delta$. Estimating τ^* here is straightforward and can be done analytically.

In our application, as was also the case in the previous approach, the cost of inverting the H matrix is prohibitive in addition to the previously mentioned problems involved with estimating an unregularized solution x^u.

Steihaug's approach

The *conjugate gradient* (CG) algorithm is a classical algorithm from numerical analysis which solves a linear set of equations with positive semidefinite coefficient matrix iteratively [19]. It can be implemented by first fixating all but the first variable and solving the resulting problem in one variable, in the second step by fixating all but the second variable, then solving again while keeping the solution components from the previous iteration, and so on. Each step will then only solve a problem in one variable and after i iterations i components of the iterate solution x_i should be set. Using this procedure it can be shown that when the H matrix of size $n \times n$ is diagonal then the algorithm will have solved the problem in at most n steps. When the matrix is not near diagonal then *preconditioners* can be applied to transform the problem into a form better suited for CG.

The CG algorithm has the important property of being “matrix free” in the sense that it does need to store or factorize the full $n \times n$ coefficient matrix but rather only requires a way to evaluate its product with a vector. Using this algorithm as a subprocedure for solving linear set of equations the previous two methods can be adapted to work in the large-scale case as well.

In the context of solving the TRS directly notice that minimizing equation (2.11) is equivalent (after differentiating and equating with zero) to solving the linear problem $Hx = -g$. Therefore CG can also be used directly for solving the TRS. Steihaug [8, 19] made the observation that if the CG algorithm starts out with an initial vector $x_0 = 0$, then the norm of the solutions generated in each consecutive iteration of the CG algorithm will be non-decreasing, i.e. $\|x_i\| \leq \|x_{i+1}\|$. When solving the TRS we can use this observation to terminated the algorithm as soon as we find $\|x_{i+1}\| > \Delta$ because then x_i is a TRS solution.

Steihaug's approach for solving TRS problems has shown itself to be very efficient when H is large and sparse but requires a positive semidefinite coefficient matrix. For our application, the hard case is however still a problem.

Eigenvalue-based techniques

In the mid 1990s some new techniques for solving the TRS were introduced in the literature [22, 33, 27]. These techniques differ from the hitherto presented approaches in that they solve the TRS as an eigenvalue problem as opposed to solving a linear systems of equations. An advantage with these methods is the fact that they provide a simple way to detect and handle the hard case which occurs in regularization.

Notice that equation (2.13a) can equivalently be expressed in matrix notation as

$$\begin{bmatrix} H & g \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} = -\lambda x \tag{2.16}$$

and that by augmenting the coefficient matrix we arrive at the eigenvalue problem

$$\begin{bmatrix} H & g \\ g^T & \theta \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} = -\lambda \begin{bmatrix} x \\ 1 \end{bmatrix} \tag{2.17}$$

where $\begin{bmatrix} x & 1 \end{bmatrix}^T$ is the eigenvector corresponding to the eigenvalue $-\lambda$. Since the asymptote we need in regularization is the one at the smallest eigenvalue $-\lambda_1$, we can use iterative methods which estimate good choices for λ_1 in relatively few operations.

The basic idea of the eigenvalue based approach is to solve this eigenvalue-problem for various choices of θ until a solution is obtained which can be normalized to $\begin{bmatrix} x & 1 \end{bmatrix}^T$ such that the solution satisfies equations (2.13b), (2.13c) and (2.13d).

In this framework the hard case manifests itself by an eigenvector which has zero in the last component. Obviously this vector cannot be normalized to be of the form we need, but in such cases it can be shown that there exists an eigenvalue slightly larger than λ_1 which has an eigenvector with non-zero last component. This can then be used to construct a solution x instead. Rojas et al [24] published in 2000 an efficient implementation which solves the large-scale TRS based on these ideas. Their implementation is publicly available in the Matlab package LSTRS [25].

In figure 2.5 we see some numerical examples of deconvolution using LSTRS. The original undegraded image shown in figure 2.5(a) is the same as in figure 2.1(c), and the deconvolution kernel is as defined on page 25 and seen in figure 2.1(b). In figure 2.5(b) we observe the convolution between the kernel and the undegraded image. No additive noise was added to the image i.e. $\eta = 0$. The result of the deconvolution can be seen in figure 2.5(e) using the regularization parameter $\Delta = \|x^*\|$. We observe that the deconvolution has successfully made the edges in the degraded image clearer and that details, for instance in the hair and eyes, are recovered. The reconstruction is not perfect however. It has an RMS error (defined in equation (1.2)) of 0.0577 compared to the original undegraded image, and we observe that the recovered image still looks somewhat blurred compared to the original. We also notice that the deconvolution process has introduced some ringing-artifacts. These are manifestations of the Gibbs phenomenon first encountered in chapter 1. These artifacts are particularly visible in the background around the hat, and along the dark band going down from the upper right-hand corner of the image.

Figures 2.5(c) and 2.5(d) shows the deconvolved image from figure 2.5(b) with Gaussian white noise added with variance 0.001 and 0.01 respectively. In figures 2.5(f) and 2.5(g) we visualize the deconvolution results. The RMS error in these reconstructions are 0.0741 and 0.0869 respectively, both compared to the original. We observe that the recovered images show some more detail, for instance in the eyes and in the feathers in the hat, but clearly the quality of the restorations quickly deteriorate as more and more additive noise is introduced. Also notice that we had to decrease Δ slightly in order to prevent LSTRS from converging to an interior solution in these cases.

2.3 The Linearly Constrained TRS

In figure 2.5 we observed that the solution to the regularization problem was quite sensitive to additive noise in the degraded image. We saw that as the variance of the noise η grows, the oscillations in the reconstructed solution also increased. In certain applications these oscillations can cause significant errors.

In 1990 the Hubble Space Telescope (HST), named after astronomer Edwin Hubble, was sent into orbit around Earth. The orbit of this satellite was outside the Earth's atmosphere so the images captured by it were expected to be of much higher quality compared to those of ground telescopes. But within weeks of the launch of the telescope, the images returned showed that there was a serious problem with the optical system. Although the first images appeared to be sharper than ground-based images, the telescope failed to achieve a final sharp focus, and the best image quality obtained was drastically lower than expected. The cause of the problems was identified as a faulty mirror which blurred the images.

In figure 2.6 we reproduce a zoomed in view[1] of an image taken by the HST. We have access to the true undegraded image, seen in figure 2.6(a), for reference. The minimum intensity of this undegraded image is on the order of 10^{-11}, the maximum is 31651.05 , with a mean of 6.36. We see that the image consists mainly of black background with a few bright stars of high intensity.

Figure 2.6(b) shows the observation available to us which has been degraded by a known blurring kernel and unknown additive noise. The statistics of the kernel were estimated from tests performed on the mirror prior to the launch of the satellite. At the time of the launch these tests were considered inaccurate, but they were later shown to be correct. The norm of the true solution is available, as this could be measured using other instruments on the satellite. Finally, the noisy image has an RMS error compared to the original of 181.11.

Solving this problem using the TRS approach described in the previous section we arrive at the solution seen in figure 2.6(c). However, inspection of this solution reveals some undesirable artifacts. We observe that about 49% of the pixels have negative intensity in the restoration; these have been marked in red in the figure. Particularly around the bright stars of high intensity we find negative intensities of strong magnitude; the minimum value of the restored image is -563.16 and this intensity is found next to a star. Compared to the undegraded image this reconstruction has an RMS error of 32.01.

Since each pixel intensity represent a measurement of the amount of light received by a sensor, negative intensities has no physical meaning in this context. A possible way to avoid this problem is simply to truncate negative intensities in the regularized

[1]The original image and the images used in the numerical experiments are of size 256×256. All statistics, such as mean, minimum and maximum intensity, as well as RMS errors are reported here are for the images of this size. But since these images do not look good on print we only visualize a small part (of the full image) of size 66×73 in figure 2.6.

solution to zero. It has however been shown that this might introduce significant errors [26].

In literature various methods have been proposed in order to avoid negative pixel intensities in the solution. Hanke et al [14] propose doing this by rewriting the variables of the regularization problem as $x = e^z$, and then solve the resulting problem in terms of the new variable z. The resulting problem is then solved using different regularization formulations and among these Tikhonov regularization. Rojas and Steihaug [26] perform regularization of astronomical images by solving the TRS with an additional non-negativity constraint $x \geq 0$. The solution is then implemented quite efficiently using the LSTRS algorithm of the previous section as a substep.

But sometimes even more prior knowledge about the properties of the true undegraded solution is known (in addition to the norm of x) , and we wish these imposed on the regularized solution as well, either in the whole image or in parts of it:

- Both *lower* as well as *upper bounds* are known.
- We may know something about the *mean* of the whole or in certain smaller regions of the image, possibly compared to other regions in the same image or a scalar quantity estimated from other sources.
- Some measure of *deviations* from the mean is known.
- The *Manhattan norm* $\|\cdot\|_1$ is known.
- The *infinity* or *maximum norm* $\|\cdot\|_\infty$ is known.
- The change in intensity (i.e. the *gradient*) from one pixel or region in the image to the next is known. This can for instance happen in regions which are known to be constant or where there is a strict increase in pixel intensity such as in a blue sky fading into white at the horizon.

Imposing such constraints is in some sense analogous to what we did in chapter 1 where we used a smaller region Ω to define local noise properties such as the noise mean and variance, and then used these properties to help estimate a global solution.

In our work [7] we propose a generalization of Rojas and Steihaug's formulation which can help model this additional knowledge that we wish to impose on the regularized solution. We do this by adding a linear constraint to the TRS model. This yields the *linearly constrained TRS*

$$\begin{aligned} \min_x \ & \frac{1}{2}x^T H x + g^T x \\ \text{s.t. } & Cx \leq d \\ & \|x\| \leq \Delta \end{aligned} \tag{2.18}$$

where C is a matrix and d is a vector.

The non-negativity constraint that we wish to impose in the HST deconvolution problem can then be represented by letting $C = -I$ and $d = 0$ where I is the identity matrix. Observe that the other constraints listed above can all be implemented as linear constraints by choosing C and d carefully as well as possibly introducing some additional variables where needed[2]. For the star-cluster problem however, we follow [26] and have only used lower bounds.

[2]The main difficulty in implementing $\|x\|_1$ and $\|x\|_\infty$ is to find a linear expression for $|x_i|$. This can be done however by introducing a helper variable $z_i = |x_i|$ and imposing the two linear constraints $z_i \geq x_i$ and $z_i \geq -x_i$ while adding the linear term $+\sum_i z_i$ to the objective function. Using a similar idea the $\max$ operator needed for the infinity norm can also be implemented.

In order to solve this linearly constrained TRS we use a log-barrier penalty approximation which yields the formulation

$$\min_x \frac{1}{2}x^T Hx + g^T x - \mu \sum_{i=1}^{n} \log(d - Cx)_i \qquad (2.19)$$
$$\text{s.t. } \|x\| \leq \Delta$$

where μ is a penalty parameter which impose a penalty on the last term in the objective as Cx approaches d. By second order Taylor approximation of the objective function at position $y = x + h$ a solution to this problem can be approximated by solving

$$\min_y \frac{1}{2}y^T H_\mu y + g_\mu^T y$$
$$\text{s.t. } \|y\| \leq \Delta$$

where H_μ and g_μ are a matrix and a vector, both using the Hessian and the gradient of the objective function of equation (2.19) at position x. Notice this problem is on the same form as equation (2.11). Our algorithm solves a series of subproblems on this form using LSTRS while driving the penalty parameter μ towards zero.

In figure 2.6(d) we see the non-negative TRS reconstruction that was estimated using our algorithm. The minimum intensity in the reconstruction is on the order of 10^{-6} and the maximum is 31810.39 . Compared to the undegraded original this image has an RMS error of 23.09. This RMS error indicates a significant improvement by imposing the non-negativity constraint, compared to the TRS solution.

2.4 Conclusion and Comments

In this chapter we have presented an algorithm which can solve the deconvolution problem with increased accuracy in certain cases. The details of our approach is presented in our paper [7].

The deconvolution problem arise in the context of single-shot magnetic resonance imaging protocols where the acquisition time of the scanner is long enough for a significant signal reduction from the first to the last echo to occur [30]. It can be shown that this signal loss has the effect of increasing the size of the point-spread function (and its discretized equivalent, the kernel κ), leading to blurring. Being able to solve this resulting deconvolution problem accurately has direct applications in these situations.

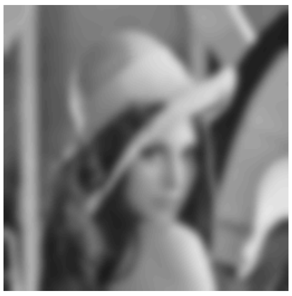
Since the algorithm we have presented in this chapter is more mathematical and theoretical in nature than the other work presented in this thesis, we have neglected the application to medical imaging for now. Instead our focus has been on deconvolution of standard datasets which makes comparison to previously published works easier. These other applications also highlight the strength of our approach: the incorporation of the linear inequality constraints. In our work [7] we show that on these datasets (in particular the HST example) our algorithm produce results which are comparable to some of the most popular of previously published works. The fact that our algorithm also can solve a large additional set of problems by using the general linear inequality constraints represents a significant theoretical contribution to the area of trust-region methods. Judging by the effectiveness and popularity of trust-region methods our algorithm could become an important tool for solving large-scale and ill-posed problems.

Application of the algorithm outlined in this chapter to medical data is one that we hope to investigate in future works. An additional difficulty when applying our algorithm to such applications is finding a good estimate for the point-spread function which in general is unknown and depend on specifics of the MR scanner used for the acquisition. Indeed, the estimation of the point-spread function from the available MR data is an interesting research-problem in its own.

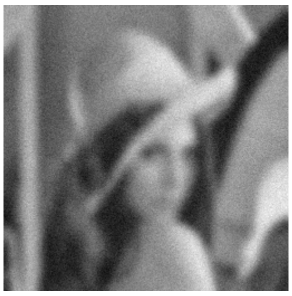
Both algorithms we have considered this far in the thesis are not limited to just medical applications and can be a applied directly to solve very fundamental problems in image processing. An important point in the presentation up until now has been to emphasis that these methods are general techniques and not very problem dependent, although they also have direct applications in MRI.

In the remaining chapters we will now focus on diffusion tensor imaging (DTI) and the specifics of this modality. These chapters will differ from the previous ones in that they are not immediately applicable to other areas of science, because they exploit the particulars of the modality to a greater extent.

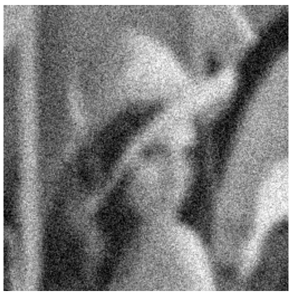
(a) The undegraded image X^*.



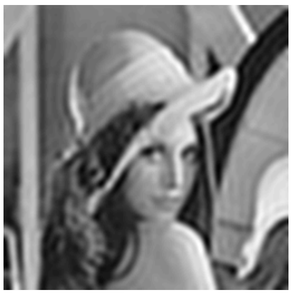
(b) The convoluted image $\kappa * X^* + \eta$ when $\eta = 0$.



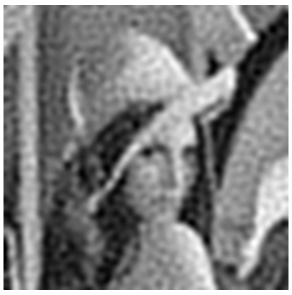
(c) η has $\sigma^2 = 0.001$.



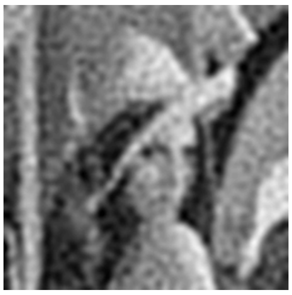
(d) η has $\sigma^2 = 0.01$.



(e) The recovered image using $\Delta = \|x^*\|$.

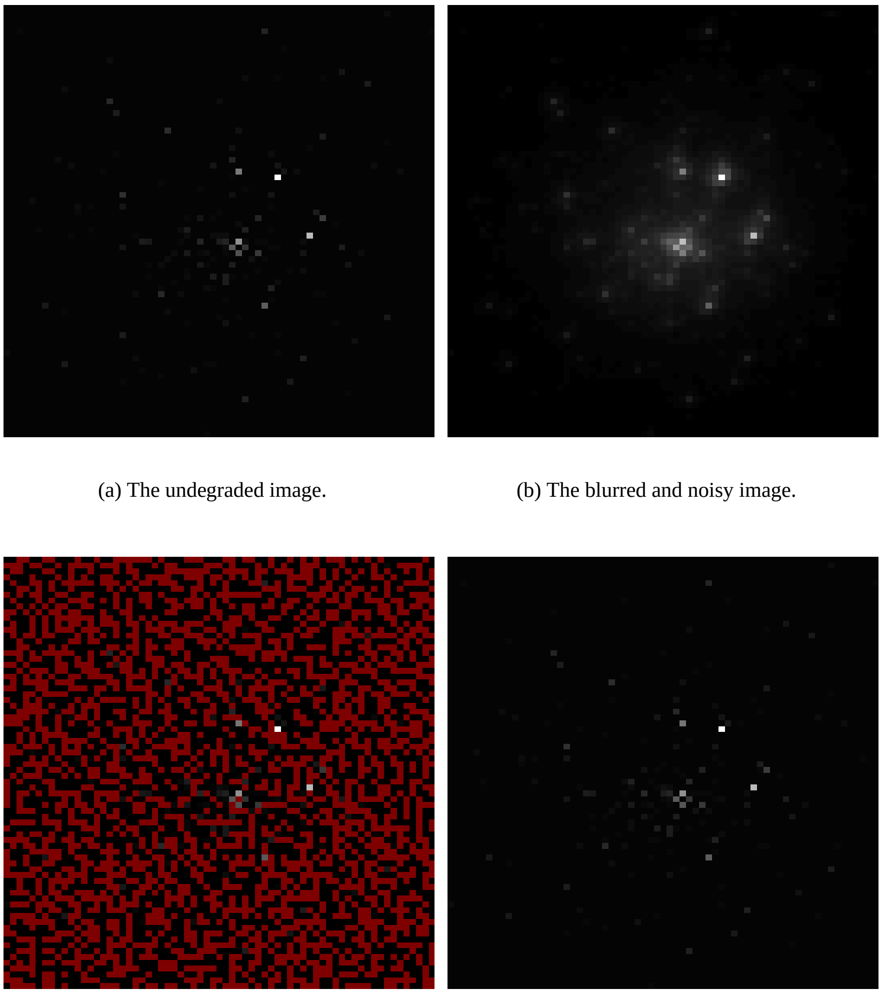


(f) $\Delta = 0.975\|x^*\|$.



(g) $\Delta = 0.95\|x^*\|$.

Figure 2.5: Deconvolution of the Lena image first seen in figure 2.1 using TRS regularization. All images are on the same scale.



(a) The undegraded image.

(b) The blurred and noisy image.

(c) The image recovered using TRS.

(d) The image recovered using TRS non-negativity constraint.

Figure 2.6: The deconvolution process applied to the star-cluster example. Image (a), (c) and (d) are shown on the same scale with negative intensities marked in red. Image (b) is normalized for visualization.

Chapter 3

Diffusion Tensor Imaging

In this chapter we will introduce the basics of *Diffusion Tensor Imaging* (DTI). We will start by outlining the more general field of *Magnetic Resonance Imaging* (MRI), and give a short explanation of the basic Magnetic Resonance (MR) physics involved. We then describe how DTI differs from and complements MRI. In section 3.1 we define precisely what constitutes a diffusion tensor dataset available to us from a scanner, and we introduce the diffusion *tensor* which summarizes the diffusion using a compact representation. In section 3.2 we show how the diffusion tensor can be estimated, and we present some of the derived quantities which can be estimated from it and which have medical significance. In section 3.4 we use the denoising algorithm developed in section 1.2 to improve the quality of the estimated tensors as well as the derived quantities. Finally, in section 3.5 we introduce fiber-tracking which offers an attractive way to visualize structure in DTI data. This chapter is intended to serve as a general introduction to the field of diffusion tensor imaging and to introduce the basic quantities needed for the more specialized topics covered in chapter 4.

Our main contribution to this chapter is the application of our 3D generalization of the originally 2D algorithm, presented in the first chapter, to medical data. The results from this denoising is presented in section 3.4. Our explicit formulation of the diffusion ellipsoid in equation (3.5) is also of importance since formulas describing this ellipsoid is often not detailed sufficiently.

Magnetic Resonance Imaging is a non-invasive method used to render images of the inside of an object. It is frequently used in medical imaging to investigate and diagnose pathological or physiological alterations in living tissues such as the brain. Using MRI diseases such as tumors, stroke, inflammation and Alzheimer's can be examined and monitored non-invasively in living patients.

As the name implies, MRI utilizes *magnetism* to capture images and is considered non-harmful to biological tissues, as opposed to other imaging modalities such as *computed tomography* (CT) *X-ray* which relies on potentially dangerous ionizing radiation. It is known that ionizing radiation may for instance increase the risk of malignancy especially in fetus. MRI is considered safe to use for all except patients with metal implants such as pacemakers, certain surgical prostheses or ferromagnetic foreign bodies like shell or bullet fragments.

An MRI dataset can be obtained using an MR *scanner*. The MR scanners used clinically today are physically quite large and typically allow a whole person to be placed inside of them. Usually however, the scan is limited to a certain organ, such as the head, torso, arms or legs. In this thesis we will only consider images of the brain.

Describing in details the principles involved in obtaining an MR image is beyond the scope of this dissertation, however for our applications the following simplified description will suffice: When protons, such as those found in water, are placed inside a strong magnetic field induced by an MR scanner the *spin* of their atomic nuclei arrange in a particular manner. Hydrogen nuclei are protons which have a large magnetic moment and they arrive at a spin which is either parallel or anti-parallel to the direction of the magnetic field. The majority of these spins will cancel each other out, but using Boltzman statistics it can be shown that there will be slightly more hydrogen atoms spinning in one direction compared to the other. The difference in population using a scanner of common field strength is about 1 or 2 in a million. By briefly exposing the magnetic field to a radio-frequency (RF) pulse which is specific only to hydrogen these extra atoms will absorb more energy that causes them to initiate a spin at a given tissue dependent frequency referred to as the *Larmour* frequency, and in a given different direction. When the RF pulse is turned off, the hydrogen protons begin to return to their natural alignment within the magnetic field and release their excess stored energy. When they do this, they give off, or *resonate*, a signal, with a tissue specific relaxation rate, that the scanner can measure. The strong magnetic field induced by the scanner can be manipulated by a set of much weaker *magnetic field gradients* which can be switched on and off very rapidly. By engaging very carefully chosen sequences of magnetic field gradient changes the main magnetic field is altered so that the process can be localized to a surprisingly small region. This allows sampling of tissue information at discretized spatial positions. This is accomplished by transforming the complex valued MR signal in the frequency domain to a magnitude image in the real domain.

Hydrogen in living tissues is mainly found in *water* and *lipids* such as fat and MRI measurements can thus be used to distinguish between such tissues and other substances such as bones or cartilage. Different tissues can also be distinguished by the time it takes for the tissue specific spin to relax after they have been excited (i.e. T_1, T_2, T_2^* relaxation times).

The resulting 2D or 3D MR image can be used to study the anatomy of the organ in question, or as a part of a more elaborate protocol involving other imaging techniques.

During the last decade the development of MRI has led to the design of numerous more specialized imaging techniques. One of these is *Diffusion Tensor MRI* (DT-MRI or simply DTI), which measures the thermic motion distribution of water molecules (i.e. water self diffusion) in 3D space.

Individual water molecules are constantly in motion, colliding with each other and with other molecules in tissues at high speed. These high-speed collisions cause the water molecules to spread out or diffuse. The phenomenon is commonly referred to as Brownian motion. Water in tissues containing a large number of fibers, like skeletal muscle or white matter in the brain, diffuses fastest along the direction that the fibers are pointing in and slowest at right angles to it. This restricted diffusion is because the various tissue components, such as membranes or cell walls, restrict the Brownian motion. When the diffusion is concentrated to a preferred direction in space it is referred to as *anisotropic diffusion*. In contrast, water diffuses in an almost spherical pattern in tissues that contain few fibers since the restriction (or rather, *lack* of restriction) is identical in every direction. This is referred to as *isotropic diffusion*. In the next sections we will describe how anisotropic diffusion can be used to infer brain connectivity.

An MR scanner programmed with a diffusion tensor protocol has the ability to sample water diffusion in a given set of encoded directions. As in previously described MRI a pulse sequence is used to resonate a spin in a small fraction of the proton population. This is done in DTI by first applying an RF pulse (referred to as *90 degree* pulse), immediately followed by a gradient magnetic pulse of duration δ. This ensures that the spin is sensitized in a particular preconfigured *gradient direction*. Then another RF pulse (referred to as a *180 degree* pulse, inverting the orientation of the spins) followed by another gradient magnetic pulse in the same gradient direction and duration is applied, refocusing the phase of the spins. The gradient direction pulses are applied at known time Δ apart. After the second gradient magnetic pulse is switched off the hydrogen protons will emit a measurable signal as their spin again realigns with the main magnetic field. As protons having diffused far in the spatial direction encoded by the gradient direction during the time Δ do not refocus completely following application of the second RF pulse, this results in a weaker signal.

The result is an image which encodes diffusion in the given gradient direction as a *loss* of signal intensity. There is however an additional difficulty: these diffusion images are also weighted with the amount of hydrogen at each discretized spatial position referred to as a *voxel*. Therefore a low intensity in a given voxel in a diffusion image can mean one of two things; either there was a high degree of diffusion in this voxel or there was little hydrogen here to begin with. Standard terminology takes this distinction into account and refers to the diffusion images as diffusion *weighted* images (DWI), indicating that they are essentially measurements of the hydrogen population *weighted* with a measure of diffusion. In order to resolve the actual diffusion from a DWI, a *baseline* reference image giving only the hydrogen population at each voxel needs to be estimated. Fortunately, this information is exactly what is represented and available in an MR image, and therefore a conventional MR baseline image always accompanies a DTI dataset.

3.1 DTI Data

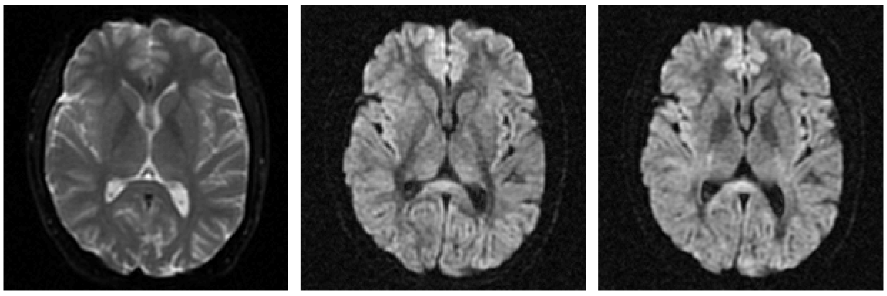
As mentioned, the DTI scanner is able to give some estimate of the apparent diffusion per voxel in a given set of n preconfigured gradient directions. Throughout the rest of this chapter we will refer to each of the normalized gradient directions as $r^{(i)} \in \mathbb{R}^3$ where $i \in \{1, 2, \ldots, n\}$. These n gradient directions are programmed into the scanner by the MR technician as a part of specification of the DTI protocol. Often it is assumed that the gradient directions are uniformly distributed on the unit sphere so that the scanner is equally sensitive to diffusion in all directions. A typical DTI scanner today has anywhere from $n = 6$ up to hundred gradient directions.

So in each voxel the scanner makes $n + 1$ measurements, one for each gradient direction $r^{(i)}$ and one for the baseline reference. We refer to the measurements in a given voxel captured in direction $r^{(i)}$ as S_i and denote the baseline MRI measurement as S_0. DTI data can therefore be considered 4 dimensional, with 3 spatial dimensions and one dimension for the sample number.

These voxelwise measured values S_i are related to the 3×3 *diffusion tensor* X as described by the Stejskal-Tanner equations [38]

$$S_i = S_0 \exp(-b{r^{(i)}}^T X r^{(i)}) \tag{3.1}$$

where $b = \gamma^2 \delta^2 (\Delta - \frac{\delta}{3})$ is an acquisition-specific constant (the "b-value") involving the


(a) A MRI baseline image S_0 (T2-weighted).

(b) The diffusion measurements S_1 in direction $r^{(1)}$.

(c) Measurements S_4 in direction $r^{(4)}$.

Figure 3.1: A DTI dataset showing mid-slice of the authors brain. Image (b) and (c) are on the same scale.

proton gyromagnetic ration $\gamma = 42$ MHz/Tesla, the time between the gradient pulses Δ and the duration of the gradient pulse δ, as described in the previous section.

The diffusion tensor X is the covariance matrix of the measurements with respect to the three spatial dimensions. All covariance matrices are positive semi-definite, and it can also be shown that *diffusion* tensors are symmetric (see [17] for references). By careful examination of the diffusion tensor a number of clinically useful quantities can be derived. The estimation of the diffusion tensor as well as these derived quantities will be the topic explored in the rest of this chapter.

The sample DTI datasets that we will be using for all examples in this chapter (except for the denoising in section 3.4) was acquired using a General Electric Signa 1.5 Tesla Echospeed MR scanner equipped with EPI measurement techniques at Haraldsplass Deaconess University Hospital. The data was collected from a 28 year old healthy volunteer; the author of this thesis. The size of the dataset is $256 \times 256 \times 24 \times 7$ and the diffusion measurements were obtained using a value of $b = 1000$. Total acquisition time for this DTI recording was 7 minutes.

Slice number 10 out of the total 24 axial slices available can be seen in figure 3.1, where 3.1(a) shows the baseline MRI S_0 image, and images 3.1(b) and 3.1(c) displays the measurements in direction $r^{(1)}$ and $r^{(4)}$, respectively. The visualized slice is physically located just about where the brim of a hat would run, and the view is as seen from above. The 6 gradient directions $r^{(i)}$ used in this dataset are given by column i in the matrix

$$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 & 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & 0 & 0 \end{bmatrix}$$

3.2 Tensor Estimation

When solving the Stejskal-Tanner equations in equation (3.1) for the unknown diffusion tensor X there are a number of different approaches, ranging from direct non-linear minimization of the squared difference between the right and left-hand side of the equations, to so-called *robust estimation* with outlier rejection [18]. When solving this equation we use the linear least-squares approach outlined in our publication [6] which has the advantage of being fast and relatively easy to implement.

Since X is a symmetric 3×3 matrix, we only have six unknowns in these equations. However, n is generally larger than six, and we therefore solve the voxelwise set of equations by least-squares minimization. Taking the logarithm on both sides and setting

$$d_i = \frac{\ln(S_0) - \ln(S_i)}{b} \tag{3.2}$$

we obtain the least-squares minimization problem

$$\min_X \sum_{i=1}^{n} ({r^{(i)}}^T X r^{(i)} - d_i)^2 \tag{3.3}$$

The key observation here is that the term inside the parenthesis is *linear* in terms of the components of X. We now define the six element column-vector x from the unique elements of the tensor X such that

$$X = \begin{bmatrix} x_1 & x_2 & x_3 \\ x_2 & x_4 & x_5 \\ x_3 & x_5 & x_6 \end{bmatrix}$$

and observe that if we let

$$R^{(i)} = \begin{bmatrix} {r_1^{(i)}}^2 & 2r_1^{(i)}r_2^{(i)} & 2r_1^{(i)}r_3^{(i)} & {r_2^{(i)}}^2 & 2r_2^{(i)}r_3^{(i)} & {r_3^{(i)}}^2 \end{bmatrix}$$

then the i'th element in the sum can be rewritten as $(R^{(i)}x - d_i)^2$. Finally letting R be the $n \times 6$ matrix whose row number i is $R^{(i)}$

$$R = \begin{bmatrix} R^{(1)} \\ R^{(2)} \\ \vdots \\ R^{(n)} \end{bmatrix}$$

equation (3.1) can be solved as the well-known linear least-squares problem

$$\min_x \|Rx - d\|^2 \tag{3.4}$$

It is also worth pointing out that the gradient directions $r^{(i)}$ are usually chosen in the MR protocol so that the R matrix has full rank. This least-squares can be solved using several standard methods such as the normal-equations, the QR algorithm or the singular-value decomposition. In our implementation we use the QR algorithm which has built-in support in the Matlab environment.

Notice that this diffusion tensor X is symmetric by construction, but that we have not made any attempts to ensure that it is positive semi-definite. In the absence of noise we know from the definition that this property is guaranteed, but as any real measurements invariably are degraded by noise the estimated diffusion tensor might be indefinite. In literature the problem of indefinite tensors is usually ignored as it can be shown to decrease when the signal-to-noise ratio of the dataset is increased. In DTI this can be accomplished by post-processing the data, as we do in section 3.4, or by altering the acquisition protocol to increase the number of diffusion measurements per voxel, to increase the voxel size or by adjusting the b-value of equation (3.1) [30].

3.2.1 The Diffusion Ellipsoid

One common way to visualize a symmetric positive definite diffusion tensor is to consider its *eigenvalue decomposition*. An eigenvalue decomposition of the tensor X is then

$$X = Q\Lambda Q^T$$

where Q is an orthogonal matrix whose column number i are the normalized and orthogonal eigenvectors $q^{(i)}$, and where $\Lambda = \mathrm{diag}(\lambda_i)$ is a diagonal matrix with the eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 > 0$ along the diagonal. This tensor can then be visualized by letting the orthogonal eigenvectors define the orientation of an ellipsoid *glyph* centered at the origin, whose scaling in each of the eigenvector directions is the corresponding eigenvalue. This *diffusion ellipsoid* then captures the main diffusion per voxel; it shows the mean squared distances traveled by molecules initially placed in the center of the voxel, after some diffusion time τ.[1]

We explore the relationship between the estimated diffusion tensor and this diffusion ellipsoid in our publication [6]. There we express the distance from the origin of the diffusion ellipsoid surface in a normalized arbitrary direction r as

$$f(r) = \frac{1}{\|\Lambda^{-1}Q^T r\|} \tag{3.5}$$

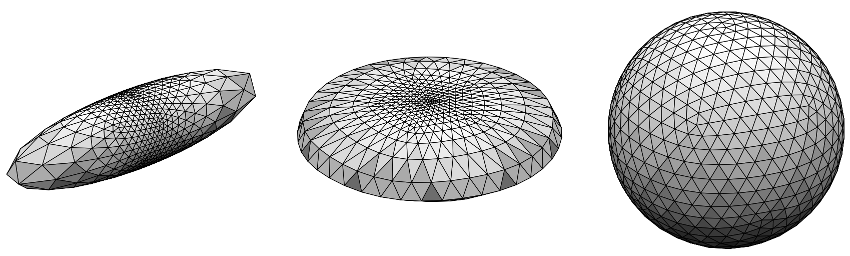
Figure 3.2 displays the diffusion ellipsoid for various values of λ_i giving $\Lambda^{-1} = \mathrm{diag}(\frac{1}{\lambda_i})$ and letting Q be the identity matrix. In order to display the surface we take 642 points $p^{(j)}$ uniformly distributed on the unit sphere and visualize the surface that the weighted points $f(p^{(j)})p^{(j)}$ span when $j \in \{1, 2, \ldots, 642\}$.

3.2.2 Tensor Invariants

I figure 3.2(a) we see an example when $\lambda_1 \gg \lambda_2 \approx \lambda_3$. This is referred to as *anisotropic* diffusion and in this case we believe the water to flow primarily in one main direction corresponding to the largest eigenvector. This shape of the diffusion ellipsoid is sometimes referred to as *cigar-shaped* and this is usually a sign that the samples are taken from a region containing strong structure such as muscle, fibers or fiber-bundles. When $\lambda_1 \approx \lambda_2 \gg \lambda_3$ the glyph is shaped as a disc, seen in figure 3.2(b). In this case we have unrestricted diffusion in two directions and a barrier in the last direction. Notice that we no longer have a single principal direction of diffusion. The final case is when $\lambda_1 \approx \lambda_2 \approx \lambda_3$ which is commonly referred to as *isotropic* (ball or sphere-shaped) diffusion. See figure 3.2(c). When water diffuses evenly in all directions it is assumed that there are few or no barriers constraining it. This will be the case in regions consisting of fat or inside tumors.

A number of quantities have since the inception of DTI been devised to summarize the shape of the diffusion ellipsoid [31]. Some of these quantities are defined only on the eigenvalues of the diffusion tensor (and not on the eigenvectors) and are thus rotational invariant. They are therefore commonly referred to as tensor *invariants*. The invariants we now consider will transform the estimated voxelwise diffusion tensor into

[1]Another similar diffusion ellipsoid is defined by the points r which satisfies the implicit equation $r^T X r = 1$. This ellipsoid will also have the orientation given by the eigenvectors of the tensor, but the scaling in each eigenvector direction will be proportional to the *square root* of the corresponding eigenvalues. This ellipsoid is an iso-surface of the probability density function of the diffusion model from equation (3.1); if releasing a drop of ink at the center of the voxel then the ink will disperse proportionally to this iso-probability surface.



(a) The linear case when $\lambda = \{1, \frac{1}{4}, \frac{1}{4}\}$.

(b) The planar case when $\lambda = \{1, 1, \frac{1}{4}\}$.

(c) The spherical case when $\lambda = \{1, 1, 1\}$.

Figure 3.2: The three basic cases of diffusion.

a *scalar* quantity which describe some particular aspect of the tensor. The fact that each tensor is reduced to a scalar value allows us to visualize tensor data as scalar intensity images, like those seen in figure 3.1.

Westin [39] defines three invariants

$$c_l = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2 + \lambda_3} \tag{3.6}$$

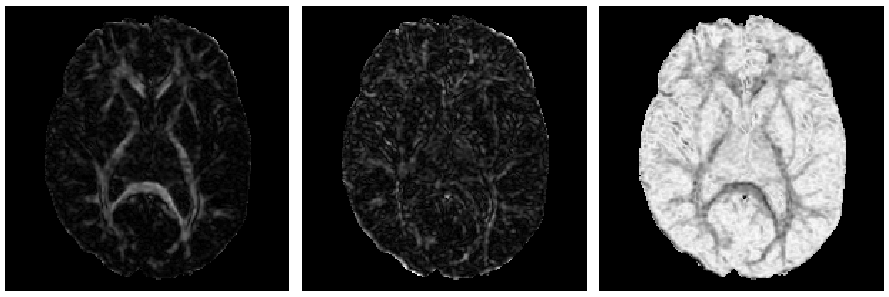
$$c_p = \frac{2(\lambda_2 - \lambda_3)}{\lambda_1 + \lambda_2 + \lambda_3} \tag{3.7}$$

$$c_s = \frac{3\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3} \tag{3.8}$$

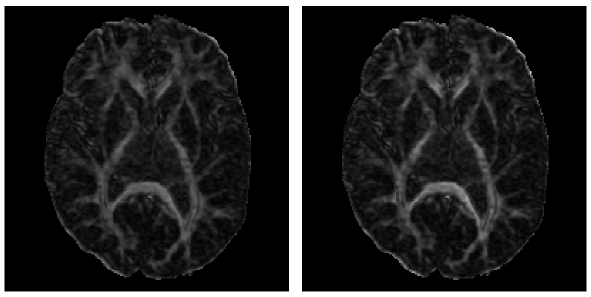
These are defined so that each invariant lies in the range 0 to 1 and so that $c_l + c_p + c_s = 1$. They describe how similar the diffusion ellipsoid is to the linear, planar and spherical case respectively. Specifically, if c_l is close to one then c_p and c_s will be close to zero, and we see from the numerator in equation (3.6) that the largest eigenvalue of the tensor is significantly larger than the second largest eigenvalue. This situation therefore corresponds to the case of linear diffusion described in the previous section and seen in figure 3.2(a). Similarly, when c_p or c_s is large then we have diffusion ellipsoids similar to those seen in figure 3.2(b) or 3.2(c) respectively.

Other invariants frequently used are the *fractional anisotropy* (FA) and *relative anisotropy* (RA) [38]. These two invariants describe the normalized variance of the eigenvalues and have the advantage that they can be calculated without explicitly estimating the eigenvalues of the diffusion tensor. They are defined as

$$\begin{aligned} \text{FA} &= \frac{1}{\sqrt{2}} \frac{\sqrt{(\lambda_1 - \lambda_2)^2 + (\lambda_2 - \lambda_3)^2 + (\lambda_1 - \lambda_3)^2}}{\sqrt{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}} \\ &= \frac{\sqrt{3}}{\sqrt{2}} \frac{|X - \frac{1}{3}\,\text{trace}(X)I|}{|X|} \end{aligned} \tag{3.9}$$



(a) The linear invariant c_l. (b) The planar invariant c_p. (c) The spherical invariant c_s.



(d) Fractional anisotropy (FA). (e) Rational anisotropy (RA).

Figure 3.3: The tensor invariants estimated from the data slice seen in figure 3.1.

and

$$\begin{aligned} \text{RA} &= \frac{1}{\sqrt{2}} \frac{\sqrt{(\lambda_1 - \lambda_2)^2 + (\lambda_2 - \lambda_3)^2 + (\lambda_1 - \lambda_3)^2}}{\lambda_1 + \lambda_2 + \lambda_3} \\ &= \frac{\sqrt{3}}{\sqrt{2}} \frac{|X - \frac{1}{3}\,\mathrm{trace}(X) I|}{\mathrm{trace}(X)} \end{aligned} \tag{3.10}$$

Again these are scaled so that they lie in the range 0 to 1.

In figure 3.3 we show the tensor invariants estimated from a *region of interest* (ROI) in the same slice as seen in figure 3.1. Notice from figure 3.1 that the measured S values are not exactly zero outside the brain because of noise. This is particularly visible in the diffusion weighted images. In these voxels outside of the brain we know there is no diffusion, and the estimation of a diffusion tensor therefore is meaningless. We therefore limit the estimation of the tensors to voxels inside the ROI, and define the ROI as all voxel where the baseline MRI is larger than a given threshold value, in this example larger than 150.

The linear invariant seen in figure 3.3(a) shows some of the “butterfly-shaped” anisotropic brain structures as the brighter region surrounding the darker ventricles in the center of the image. The inverted U-shaped structure seen in the center of the lower half is called the *corpus callosum*. The fibers running on both sides of the ventricles in the longitudinal direction above it are the *cortico-spinal* tracts. The corpus callosum connects the left and right hemispheres of the brain, and contains highly structured tissue where water primarily diffuse in one direction, explaining the high values in this plot. The same brain structure is also seen in figures 3.3(d) and 3.3(e), but notice that here the structures appear larger, as can be explained by the fact the FA and RA also have higher values in some of the regions of planar diffusion, seen in figure 3.3(b). Conversely this means that in the FA and RA plots we will find a high value where the diffusion is *not* isotropic. In this sense these two plots can be interpreted as some additive inverse of the spherical invariant seen in figure 3.3(c).

3.3 Noise in MRI

As pointed out in the first section of this chapter, the signal which is usually used in MRI is a magnitude image of the originally complex-valued data. It is known that the noise is Gaussian distributed in the complex domain, but by application of the non-linear magnitude operator the noise in the resulting resulting dataset becomes *Rician distributed*. A unfortunate property of this distribution is, as opposed to additive Gaussian noise, Rician noise is signal-dependent and consequently separating signal from noise is in principle a difficult task [28].

In particular, in regions of low magnitude signal-to-noise ratio (SNR), such as in the air in the background of an MRI, the Rician distribution behaves similarly to a Rayleigh distribution. In regions of high magnitude SNR the Rice distribution approaches a Gauss distribution.

Since we are concerned with the parts of an MRI where there is a significant signal, we make the approximation that the noise in DTI is Gaussian rather than Rician distributed. In the next section we will use this approximation and show that we obtain reasonable results.

3.4 Denoising of DTI

As described in chapter 1, all measured quantities are subject to some degree of degradation during acquisition. In MRI and DTI this is also true, but the source of the problem may not always be just hardware limitations.

There is usually a direct relationship between the time spent in the scanner during acquisition and the resolution of the acquired dataset: *high quality* datasets, meaning datasets with high resolution and many captured gradient directions, take a long time to capture. It is both uncomfortable for the patient, who has to lie perfectly still inside the noisy scanner for anywhere up to an hour, as well as expensive for the hospital, to obtain such datasets. However lower quality datasets, which can be acquired much faster, will have larger uncertainty associated with each measurement, and are thus more “noisy”. Faced with this trade-off between cost and image quality, hospitals may choose to use lower quality images.

In section 1.2 we outlined a 2D algorithm that can efficiently denoise a dataset containing normally distributed white noise. In our publication [3] we have extended this 2D formulation to 3D such that it can be used to denoise DTI data. Using this algorithm we show that we can, in the cases we have studied, denoise low quality datasets from the scanner and obtain denoised results which are comparable in quality to datasets which using the scanner takes much longer to obtain.

In order to achieve such results we start out with a DTI dataset S with a low number of gradient direction, but of high enough spatial resolution to be of interest. We then estimate a diffusion tensor at each voxel and apply our 3D denoising algorithm to each of the tensor elements separately. This means that we apply our denoising algorithm once for all 3D voxels using tensor element X_{11}, then once for X_{12} and so on until X_{33}. We will denote the denoised tensor dataset resulting from this procedure as $\bar{X}$. From this denoised dataset we estimate the FA tensor invariant outline in the previous section.

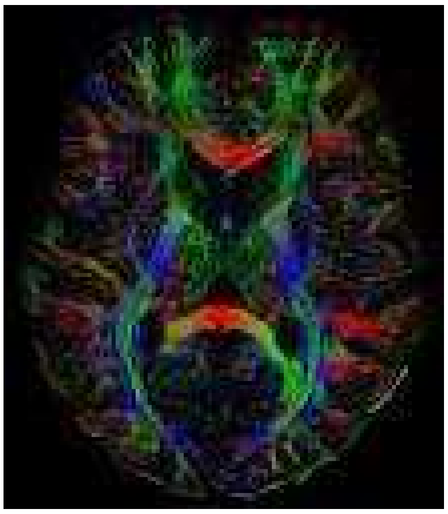
In figure 3.4 we see a visualization of the results of this procedure. The size of the dataset is $110 \times 126 \times 65 \times 7$ and we consider slice number 32 in the z-direction. In order to see the effect of noise and denoising on orientation we visualize color-coded main eigenvector plots weighted with FA. In these plots the intensity of each pixel is FA as described in the previous section, but we have replaced the gray intensity scale seen in figure 3.3 with a color scale which encodes the *principal direction of diffusion*, i.e. the eigenvector corresponding to the largest eigenvalue of the tensor per voxel. We let principal diffusion from ear-to-ear (the x-direction in the images) be represented by red, diffusion from the nose-to-the-back-of-the-head (the y-direction) with green, and the head-to-toe direction (the z-direction, out of the viewing plane) in blue. Intermediate directions are interpolated from these. A high FA value is thus represented by some color and a low FA value is visualized as black. In figure 3.4(a) we see the color-coded eigenvector plot estimated from the high quality DTI dataset S^*. The acquisition of this dataset took about 45 minutes using a 3 T Siemens scanner using $b = 1000$ with 6 gradient directions. Figure 3.4(b) displays the same calculated from a low quality scan S taking about 20% of the time to capture compared to the high quality S^* one. Notice the image degradation in the measurements seems to visually manifests itself in the image in a way similar to the noise seen in chapter 1. The final figure 3.4(c) shows the results of applying our 3D SA-DCT denoising procedure to the noisy tensors X, giving a denoised dataset $\bar{X}$ from which we calculate and visualize the FA weighted orientations. We observe that the image created from the low quality denoised samples seems visually to be comparable to that one estimated from the high quality one.

3.5 Fiber-Tracking

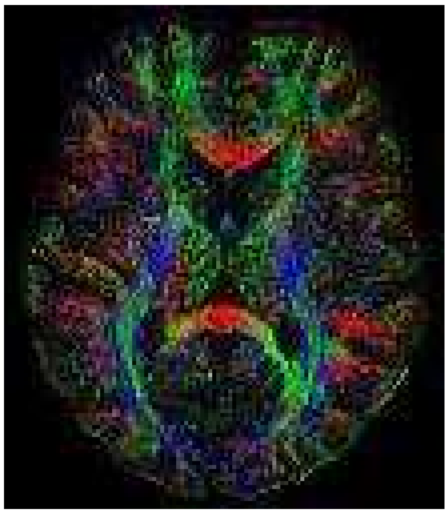
As we saw in section 3.2.1 the voxelwise principal direction of diffusion can be estimated from a diffusion tensor dataset by calculating the eigenvector corresponding to the largest eigenvalue. Fiber-tracking takes this idea one conceptual step further by trying to trace the movement of a virtual water-molecule from voxel to voxel by following the principal diffusion direction when the anisotropy is high. In this way DTI can be used to infer connectivity between various parts of the brain, which in turn can be used for diagnosis, exploration or classification of fiber and fiber-bundles in-vivo.

The measurements available from the DTI scanner are discretized at a given scanner-specific resolution. But in order to perform fiber-tracking a continuous tensor-field is needed so that a diffusion tensor can be estimated at arbitrary locations in the data. There exists a few different ways to *interpolate* in order to provide such a continuous diffusion tensor field

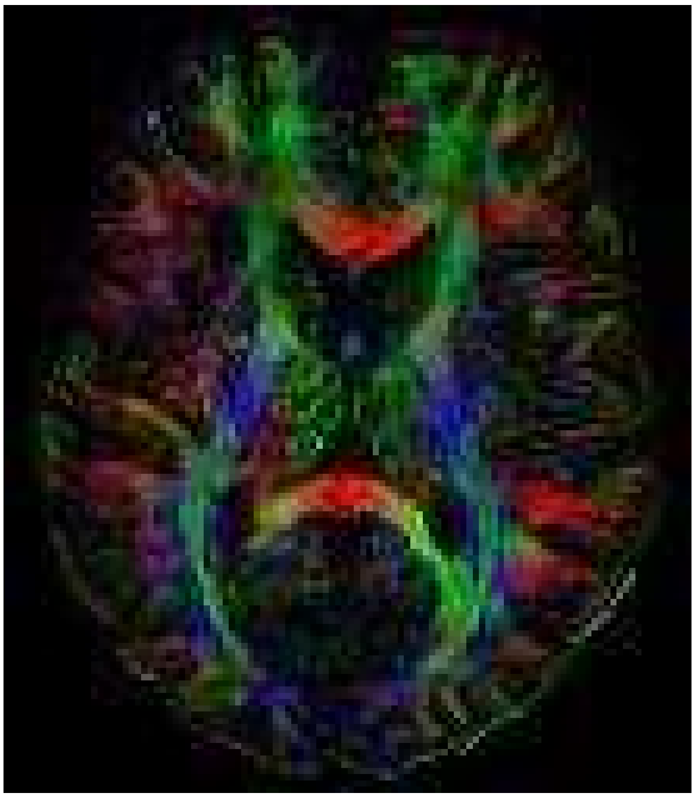
(i) Basser et al [2] precompute the diffusion tensors at each discrete voxel given by the resolution of the DTI scanner, and then performs trilinear interpolation



(a) A high quality dataset.



(b) A low quality dataset, taking about 20% of the time to capture.



(c) The denoised dataset restored from the data used in image (b).

Figure 3.4: Color-coded main eigenvector plots weighted with FA, from a DTI dataset.

on these tensors. The interpolation step will then consist of estimating a non-negatively weighted sum of tensors, where the weights are inversely proportional to the distance to each of the neighboring voxels where the tensors are located.

(ii) Another approach is to perform component-wise 3D interpolation on the n diffusion weighted images S_i when $i \in \{1, 2, \ldots, n\}$, giving a continuous n-dimensional dataset where a diffusion tensor can be evaluated at arbitrary locations.

Notice that since the estimation of a diffusion tensor by least-squares is a non-linear operation the tensors estimated by approach (i) and (ii) are not necessarily equal.

Approach (i) has the advantage that the diffusion tensor estimation only has to be performed once for each voxel, and that this can be done in advance potentially making the method faster. Additionally it is enough to perform interpolation on only the 6 elements of the precomputed diffusion tensors as opposed to the n elements in approach (ii). Because of this approach (i) is usually preferred in literature.

The principal direction of diffusion is given by the eigenvector $q^{(1)}$ corresponding to the largest eigenvalue λ_1 of the diffusion tensor at interpolated position $p = (x, y, z)$. Using simple Euler integration we can estimate a fiber trajectory by following the principal diffusion direction as

$$p_{i+1} = p_i + \alpha q^{(1)}(X(p_i))$$

where p_i is a given spatial position and $q^{(1)}(X(p_i))$ is the eigenvector corresponding to the largest eigenvalue calculated from the interpolated tensor at p_i. The scalar value α is the step length which must be chosen to be small in order for Euler integration to be accurate.

Tracking thus typically proceeds from a given seed position p_0 in which some anisotropy measure is larger than a given threshold value. A diffusion tensor is estimated at that position and the principal direction of diffusion is estimated. An updated position is found from this direction using some numerical procedure (e.g. Euler integration) and a diffusion tensor is calculated at the new position. This procedure is repeated until a position is encountered in the dataset in which the anisotropy drops below a given threshold value. The tract is then terminated.

The traditional measure of anisotropy used in fiber-tracking is FA [2]. As we saw in figure 3.3(d) compared to figure 3.3(a) however, this particular choice of measure may be less than ideal in certain cases as FA may have a larger value in regions of planar diffusion. Although not as common in the context of fiber-tracking, c_l may sometimes be a better measure of anisotropy. In the following examples we have used c_l to seed and terminate tracts.

In figure 3.5 and 3.6 we visualize diffusion tensor fiber-tracking results on the original dataset of the author's brain, also seen in figure 3.1 and 3.3. Figure 3.5 and 3.6 were created for this thesis by Dr. Gordon Kindlmann using utilities from his publicly available Teem software library [16].

The green fibers seen in the left hand foreground of figure 3.5 are the *inferior longitudinal fasciculus* which connects the occipital and temporal lobes. It has been shown that this structure is involved in human visual processing and object recognition. The blue fibers running upwards seen towards the center of the image belong to the *corona radiata* which connects the spinal cord (through the corticospinal tract) and the cerebral cortex. Finally, the red fibers barely visible in the center of the dataset in figure 3.5 but easily seen along the vertical midline in figure 3.6 are the *corpus callosum* which connect the left and right cerebral hemispheres. It is the largest white

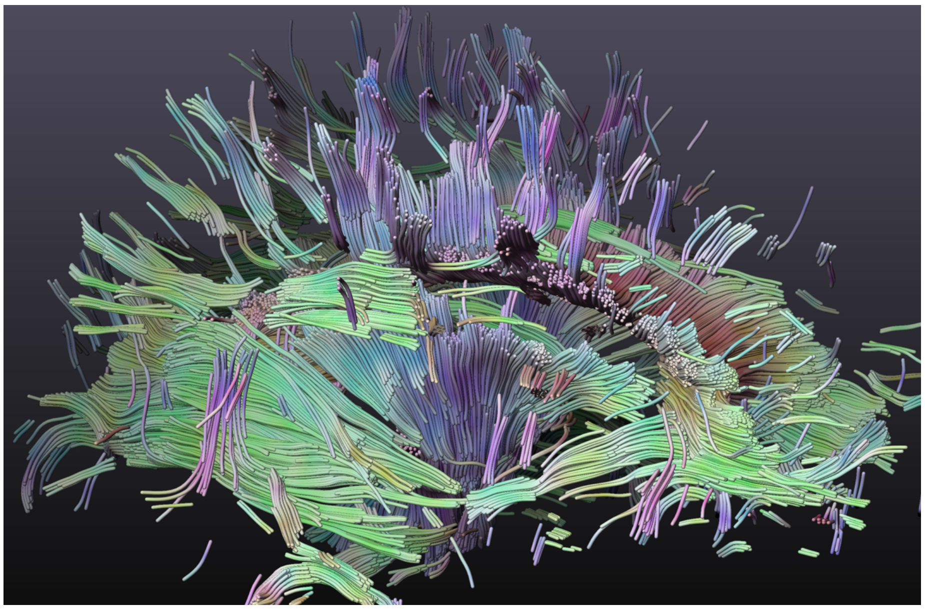


Figure 3.5: Diffusion tensor fiber-tracking results showing the corticospinal tract and superior corona radiata in the cranio-caudal direction, and the inferior fronto-occipital fasciculus and the inferior longitudinal fasciculus in the fronto-occipital direction. The colors on the fibers encode their directions as in figure 3.4.

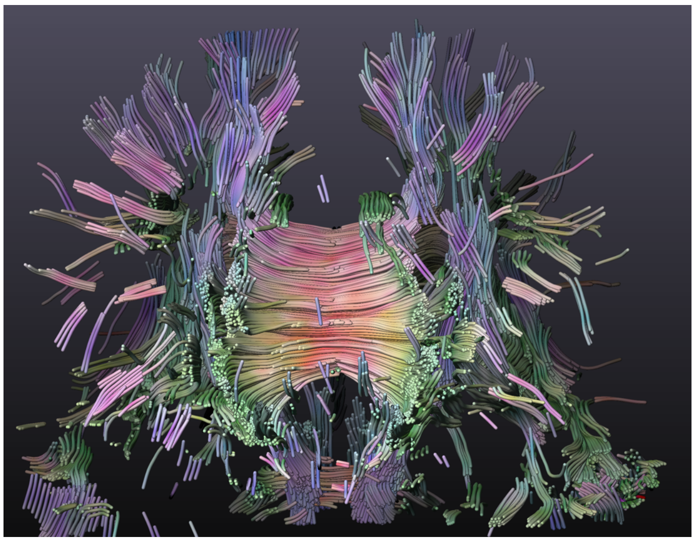
matter structure in the brain also clearly visible in the anisotropy plots visualized in figure 3.3 and 3.4.

3.6 Conclusion

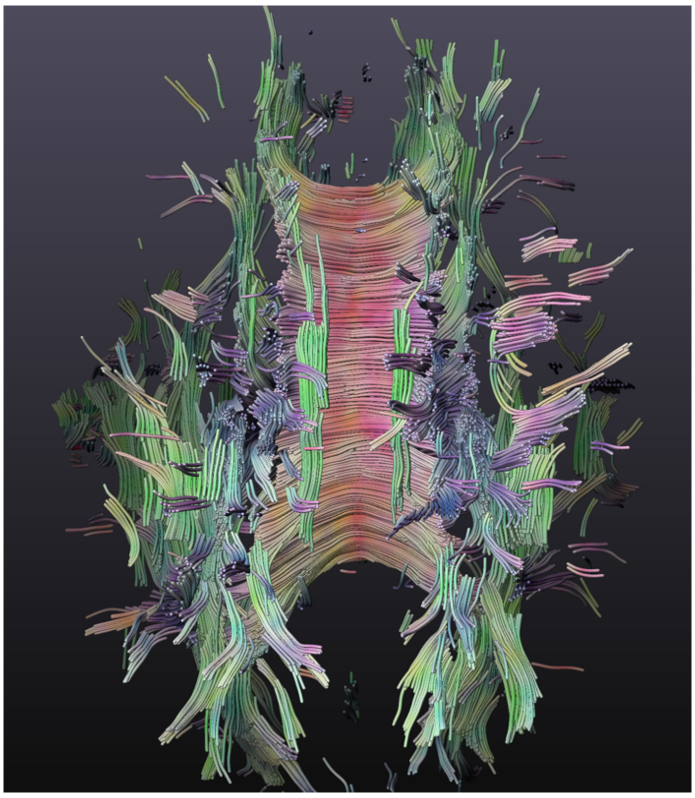
In this chapter we have presented a brief overview of “traditional” analysis performed in diffusion tensor images. This tour started with simpled descriptors of the voxelwise diffusion and culminated with state-of-the-art DTI tractography results. Much of the material in this chapter has been published in our publication [6].

Our main contribution to this chapter has been the 3D extension of the SA-DCT algorithm first outlined in chapter 1 to DTI data. We have shown that there is a significant improvement potential in terms of quality by applying pre-prosessing to DTI data. This way to improve image quality can be utilized for instance to allow researchers to work with scanner data of lower quality which are both faster and cheaper to acquire. By post-processing using our algorithm and conventional personal computer valuable scanner time can be freed up, while the post-processed data will be of comparable quality to data taking much longer to acquire. Therefore this algorithm has a significant potential for improving patient treatments clinically.

In the next and final chapter we will examine a situation in which the quantities presented in this chapter are known to fail, in particular where the diffusion tensor does not accurately represent the underlying physiology. We will then describe a more general framework which can remedy the situation, and our contributions to solve the resulting problems.



(a) Front view, looking towards the forehead.



(b) Top view, looking down from above.

Figure 3.6: Front and top view of results from figure 3.5 showing fiber bundles in the horizontal (mediolateral) direction representing the corpus callosum. Fiber bundles in the cranio-caudal direction are the corona radiata. In the postero-lateral region we see the visual radiations.

Chapter 4

Multi-Tensor Imaging

In this chapter we will investigate a generalization of the single tensor model from the previous chapter. In this new model the diffusion is described using multiple diffusion tensors per voxel. We will first motivate the new model by showing a pathological case in which the single tensor model is known to fail. In section 4.1 we will then present a generalization which can resolve the problem, and we briefly outline in section 4.1.1 some attempts in literature to solve it. In section 4.2 we outline our simplification which approximates the more general multi-tensor model, and describe how we solve it efficiently. Finally, in section 4.3 we present our multi-tensor tracking algorithm which can perform fiber-tracking in a multi-tensor dataset using our multi-tensor estimation procedure. The main new contributions of this thesis are the multi-tensor estimation procedure and the multi-tensor fiber-tracking algorithm presented in sections 4.2 and 4.3.

Diffusion tensor imaging as presented in the previous chapter has over many years proved itself to be a useful modality which today is employed for research and in clinical applications in hospitals around the world. Using diffusion tensors and their invariants a number of pathologies and diseases are investigated every day. Nevertheless it has been shown that the diffusion tensor model of the Stejskal-Tanner equation (3.1) has some drawbacks which in certain cases can lead to inaccuracies or errors in interpretation of the results.

On the spatial scale of DTI individual voxels may be quite large, in some datasets having dimensions of several millimeters in each direction. In such cases the voxelwise DTI measurements will be a composition of all diffusion within that voxel. This is true even if a single voxel contains diffusion in multiple directions. The term associated with this situation is *intra-voxel orientation heterogeneity* (IVOH).

A well known example of IVOH is when two fiber bundles cross in a voxel. Even though the diffusion in each one of the two trajectories flows in just one direction measurements taken over too large a region will contain information from both trajectories. When a diffusion tensor is calculated from this data the corresponding diffusion ellipsoid will obtain a disc-like shape [21]. This oblate diffusion is not an accurate description of the actual physiology but rather is an artifact of trying to fit a too simple convex model to the data, resulting in *under fitting*.

Figure 4.1 displays an example of this situation. In figure 4.1(a) we observe the true shape of the flow of diffusion at a given position in a synthetic DTI dataset. We observe that the distribution of water is limited to two preferred directions, rather than one as

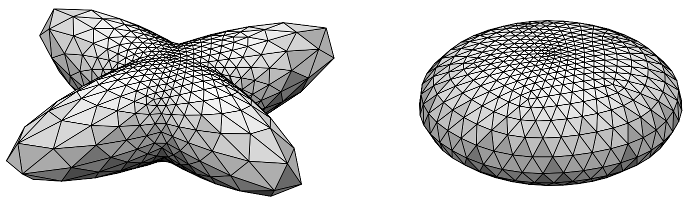

(a) The actual diffusion. (b) The estimated single tensor fit.

Figure 4.1: Pathological case for the single tensor.

in the case of cigar-shaped diffusion in the previous chapter. Attempting to estimate the diffusion tensor using the Stejskal-Tanner equation of the previous chapter yields the diffusion ellipsoid seen in figure 4.1(b). We observe that the model is unable to reproduce the two main directions of diffusion in the xy-plane, and instead yields a surface which in some sense is an average of the true diffusion surface; the result is a planar diffusion ellipsoid. In terms of the tensor invariants IVOH results in a falsely low anisotropy estimate, as can be seen for instance by estimating the linear invariant of this tensor.

This behavior motivates other models for describing the diffusion in certain cases. Since the tensor model of Stejskal-Tanner has proved itself over the last decade to be a both simple and accurate description of diffusion in most voxels in the brain, we will in the following consider a generalization of this model. The generalizations we consider will reduce to the Stejskal-Tanner model for certain choices of the free parameters, but also allow accurate descriptions of diffusion in crossing fibers for other choices. Therefore the generalization should theoretically be at least no worse than the current model, since the set of problems that can be resolved with the generalizations is superset of existing solvable problems.

An additional advantage of this approach is that it allows already existing software for diffusion analysis and fiber-tracking to be extended to also handle the crossing-fiber case without major rewriting to accommodate a different theoretical framework. Indeed, the implementation of our multi-tensor fiber-tracking algorithm in the open-source project Teem [16] demonstrates this point perfectly.

4.1 Multi-Tensor Estimation

An alternative to the single tensor model of chapter 3 is *multi-tensor estimation* [34, 35, 36, 20]. As we have seen modeling the diffusion with just one tensor will in some situations fail to describe the underlying anatomy accurately. Multi-tensor imaging attempts to rectify this situation by modeling the measured diffusion as a composition of multiple diffusion tensors.

Determining exactly how many diffusion tensors one should fit to the measurements in each voxel is a non-trivial problem. As we saw in chapter 3 a single tensor fit seemed to produce reasonable results compared with prior anatomical knowledge for most voxels. This suggest that we should use the single tensor model in *most* cases. When using multiple tensors there is also the risk of introducing too much complexity into the model. Although the number of gradient directions n used during acquisition effectively puts an upper bound on the total number of tensors per voxel, it may still be chosen too large leading to *over fitting* of the data and decrease of the model validity. In order to simplify the discussion throughout this chapter we will therefore assume that we know a priori some good estimate of the number of diffusion tensors k needed per voxel in order to accurately describe the diffusion measurements. Good estimates of k can often be obtained by examining relevant anatomical atlases, and manually inspecting the fit of the model. In the examples we will consider we let $k = 2$ in the regions of interest.

Tuch [34] has modeled the relationship between the k unknown tensors $X^{(j)}, j \in \{1, 2, \ldots, k\}$ in a given voxel and the n measured $S_i, i \in \{1, 2, \ldots, n\}$ values found there as

$$S_i = S_0 \sum_{j=1}^{k} f_j \exp(-b {r^{(i)}}^T X^{(j)} r^{(i)}) \tag{4.1}$$

where f_j are non-negative weights which sum to 1. The model lets $r^{(i)}$ be the normalized gradient directions and defines b as in the single tensor case. Note that this model reduces to that of equation (3.1) when $k = 1$. When $k > 1$ the measured diffusion is modeled as a weighted sum of multivariate Gaussian distributions.

An immediate observation from equation (4.1) is that there is some redundancy in the model. Notice that since the weights f_j are defined to be positive, they can without loss of generality be written on the form $f_j = e^{z_j}$ and the scalar z_j can be included in the exponent. Manipulation of the equation then yields an equivalent formulation involving a different tensor $X^{(j)} - \frac{z_j}{b} I$ in the exponent. Adjusting the weight f_j therefore corresponds to adding a constant to the eigenvalues of the estimated tensor. It is clear that the model loses no expressive power by always letting $f_j = 1$ giving $z_i = 0$. In our model we will later do this.

A disadvantage of model (4.1) is that the measured signal is modeled as a sum of statistics from all k tensors in that voxel. This implies that the tensor-invariants from section 3.2.2 change meaning when applied to each individual of the k tensors. Instead multi-tensor invariants analogous to their single tensor equivalents must be developed.

4.1.1 Solving for multiple tensors

As mentioned the problem of estimating multiple tensors per voxel using equation (4.1) can be interpreted as solving a Gaussian mixture problem. One common way of dealing with these types of problems is to use the *expectation maximization* algorithm. However, as Tuch [36] points out, we may want to impose other constraints on the solution which may not be easy to incorporate into this framework. He therefore employs another solution method, outlined below.

The relatively low number of gradient direction measurements available per voxel is a significant problem when working with real DTI data. When $k = 2$ there are in principle 12 unknowns (ignoring the fractional weights) in the set of multi-tensor equations. The data we will be analyzing later in this chapter has $n = 30$ measurements per voxel, giving slightly more than 2 measurements per variable. This makes the estimates sensitive to noise. In the following we will therefore first present some algorithms that will solve equation (4.1) by reducing the number of variables in the model based on various assumptions.

A priori knowledge

Tuch et al. [35, 36] solve a simplified version of equation (4.1) by means of least-squares minimization between the right and left-hand side. They rewrite the diffusion tensor using the eigenvalue decomposition $X^{(j)} = Q\Lambda Q^T$ and assumes that the linear diffusion in each of the tensor basis directions is known giving Λ. They then solve for the k orthogonal eigenvector matrices (each parameterized by the three Euler-angles) and the fractions f_j by minimizing the difference between the observed signal and the model in the least-squares sense. When $k = 2$ the resulting problem in 7 variables (using $f_2 = 1 - f_1$) is minimized using a "conventional gradient descent method" [36].

Using the single tensor fit

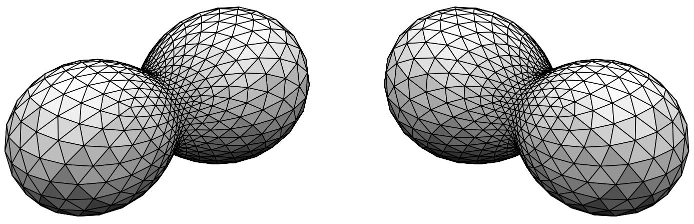
Peled et al. [20] solve the same problem in the $k = 2$ tensor case by examining a single tensor fit. As we have seen in figure 4.1, the resulting single tensor diffusion ellipsoid will obtain a disc-like shape when we have two crossing fibers. Since the two tensor ellipsoids of $X^{(1)}$ and $X^{(2)}$ to be estimated should lie in the plane of this disc, one eigenvector of each of the tensors is fixed to be orthogonal to this plane. The estimation of the remaining two orthogonal eigenvectors of each tensor is thus simplified to calculate an angle of rotation around this plane-normal. Additionally an assumption of *cylindrically symmetric* diffusion is made, implying that the two minor eigenvalues are equal for each tensor, and given by the the smallest eigenvalue λ_3 of the single tensor fit. The largest eigenvalue is also assumed identical for each of the two tensors. Thus the number of variables in the model is reduced to four; the in-plane rotation of each of the two tensors, the shared largest eigenvalue λ_1 and the fraction $0 \leq f_1 \leq 1$ giving $f_2 = 1 - f_1$. As in Tuch's approach, the least-squares difference between the model and the measurements are minimized, now using the Optimization toolbox in Matlab.

By using our observation about the fractional weights from the previous section it should be possible to reduce the number of unknowns even further, although we have not investigated the details of this approach any further.

4.2 Our approximate model

In our work [4, 5] which was presented at the MICCAI 2006 and ISBI 2007 conferences in Copenhagen, Denmark and Washington D.C., USA respectively, we present a new simplified solution method which can be used to estimate multiple tensors per voxel. These works represents some of the main contributions of this thesis and they were both well received when they were presented to the DTI community[1]. We will devote the rest of this chapter to these two new methods; the first which estimates multiple tensors per voxel, and the second which applies the first method iteratively to perform *multi-tensor fiber-tracking*.

[1]At the ISBI conference our publication was voted among the top student contributions.



(a) Tensor $X^{(1)}$.

(b) Tensor $X^{(2)}$.

Figure 4.2: The Rayleigh surfaces of the tensors seen in figure 4.1(a).

Our published multi-tensor solution method [4] can be interpreted as solving a model that approximates that of equation (4.1). Whereas equation (4.1) model each measurement as a weighted sum of exponentials, we make in our model the simplifying assumption that each measurement originates from just one of the exponentials of the k diffusion tensors. We will now justify this in the anisotropic $k = 2$ tensor case from the example in figure 4.1(a).

In figure 4.2 we have visualized the *Rayleigh surface* defined as

$$\left({r^{(i)}}^T X^{(j)} r^{(i)}\right) r^{(i)}$$

for each of the tensors $X^{(j)}$ from figure 4.1(a) using many normalized gradient directions $r^{(i)}$. This surface is an alternative way to visualize a tensor, similar to the one defined in section 3.2.1 and shown in figure 3.2. It is easy to prove that the Rayleigh surface must interpolate the same three points λq that define the diffusion ellipsoid when $q = r^{(i)}$ is an eigenvector of $X^{(j)}$ and λ is the corresponding eigenvalue. In this sense the diffusion ellipsoid and the Rayleigh surface are equivalent. In the figure 4.2(a) we notice that the main extents of the Rayleigh surface of $X^{(1)}$ correspond well with the first principal directions of diffusion seen in figure 4.1(a). We also see that the Rayleigh quotient of $X^{(2)}$ in the same direction seen in figure 4.2(b) has a comparatively low value. This is not surprising given that, if they were equal then both tensors would have the same principal direction of diffusion and a single tensor model would describe the situation accurately.

Now returning to equation (4.1), we recognize the Rayleigh quotient in the exponent. Since we now know that the exponent of the one term in the sum is larger than the second we also know that one of the exponentials will evaluate to a number significantly larger than the other. Since the measurement S_i in direction $r^{(i)}$ is the sum of these two exponentials, we hope that ignoring the smaller of the two terms does not introduce significant errors.

We therefore define the following model as an approximation to equation (4.1)

$$S_i = S_0 \exp(-b{r^{(i)}}^T X^{(j)} r^{(i)})$$

when $i \in Z_j$ and Z_j is the set of all measurement indexes that are approximated by tensor $X^{(j)}$ when $j \in \{1, 2, \ldots, k\}$. We will also require that the union of all sets Z_j contain all of the original measurements i exactly once. This way no measurements are discarded or included more than once in the model. When $k = 1$ we see that we only have one set $Z_1 = \{1, 2, \ldots, n\}$, and this model reduces exactly to the single tensor model from equation (3.1).

Our multi-tensor estimation procedure presented in our papers [4, 5] is a variant of the linear least-squares method shown in equation (3.4) adapted for solving our multi-tensor model. Again the idea is not to use all samples to estimate one diffusion tensor but rather to use disjunct subsets of the samples to estimate each of the k diffusion tensors per voxel. Thus the problem of estimating multiple tensors per voxel is reduced to a *segmentation problem*.

Given a segmentation of the measurements S_i into k disjunct sets $Z_j, j \in \{1, 2, \ldots, k\}$ equation (3.3) can be used to estimate a diffusion tensor $X^{(j)}$ by letting the sum run over $i \in Z_j$. By the same reasoning used in section 3.2 this problem can be rewritten in an equivalent weighted form analogous to equation (3.4) as

$$\min_{x^{(j)}} \|W^{(j)}(Rx^{(j)} - d)\|^2 \tag{4.2}$$

where $W^{(j)}$ is a diagonal 0-1 matrix so that $\sum_{j=1}^{k} W^{(j)}$ equals the identity matrix. The weighting matrix $W^{(j)}$ then enforces the segmentation Z_j by weighting to zero those measurements that are not in the set Z_j. Specifically, we define element (i, i) in the diagonal matrix as

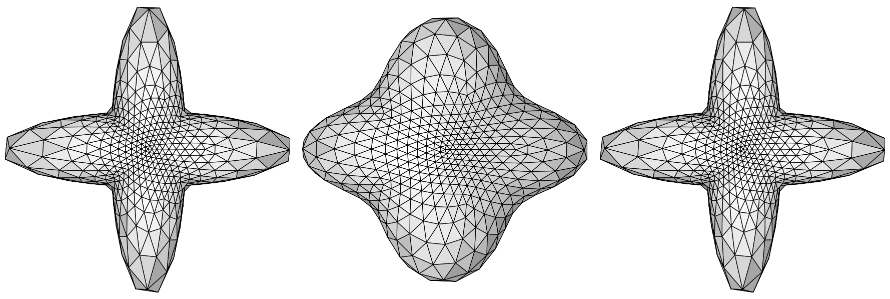
$$W_{ii}^{(j)} = \begin{cases} 1, \text{ iff } i \in Z_j \\ 0, \text{ otherwise} \end{cases} \tag{4.3}$$

Note that when $k = 1$ all samples are segmented into the same set $Z_1 = \{1, 2, \ldots, n\}$ and therefore $W^{(1)}$ is the identity matrix. The approach then reduces exactly to the standard single tensor estimation of equation (3.4).

An additional observation can be made by remembering the fact that there is only six unknowns per diffusion tensor in the classical Stejskal-Tanner equation (3.1). When exactly six measurements are available then a diffusion tensor can be created that fits the data perfectly (i.e. with zero difference between the right hand and left hand side in the equation). If we assume that there is no noise in the samples and that there are more than six measurements available, then there the same diffusion tensor can still be calculated with zero error. Finally, any subset containing six samples would uniquely define the same diffusion tensor with zero error, in the absence of noise.

From this reasoning it is clear that our multi-tensor procedure which works by estimating k tensors from k subsets of the available data should produce k equal diffusion tensors if applied to “clean data” originating from a voxel where the single tensor model would be appropriate.

Although we know that the assumption of zero noise is not justified in DTI we obtain good results using our procedure on the real (and noisy) datasets presented at the end of this chapter. It is reasonable to expect these tensors to be as sensitive to noise as single tensor estimates would be, when computed from the same number of samples as in each subset.



(a) The two tensors seen in figure 4.1(a).

(b) The corresponding q-ball profile.

(c) The recovered two tensors using our procedure.

Figure 4.3: Visualization of the q-ball profile and the recovered multi-tensors.

Applying Q-ball Imaging

This approach hinges on finding a good segmentation of the samples. In our publications [4, 5] this is done using *q-ball imaging* [34, 35, 9]. The basic idea of q-ball imaging is to transform the measured diffusion signal to a orientation distribution function directly on the sphere. The q-ball can thus be considered an alternative model to the tensor.

Considering each point $r^{(i)}$ on the sphere as a *pole*, the q-ball transform assigns the value at the pole q_i to be the integral over the associated equator [35]. This integral can either be approximated numerically [35] or analytically, typically using spherical harmonics [9]. We approximate this integral simply as a weighted sum of discrete samples S_i yielding

$$q_i = \frac{1}{S_0}\sum_{j=1}^{n} S_j \cos(\frac{\pi}{2} {r^{(i)}}^T r^{(j)})^p \tag{4.4}$$

where p is a user specified parameter. When $r^{(i)}$ and $r^{(j)}$ are orthogonal the weighting cosine-term evaluates to 1 and when they are parallel it evaluates to 0. The parameter p controls how quickly the weights go to zero in the other cases.

Figure 4.3(b) shows an example where the q-ball is evaluated using this approximation. The surface displayed is $q_i r^{(i)}$ for a large number of directions $r^{(i)}$. We observe that the main extents of the q-ball corresponds well with the two principal directions of diffusion seen in figure 4.1(a), also reproduced in figure 4.3(a) using the same view for comparison.

In our publication [4] the main extents of the q-ball is captured by estimating the k lines that best approximate it, by solving a binary optimization formulation. Our publication [5], which is an continuation of the previous work [4], approximates a solution to this optimization problem using a *k-means clustering* algorithm. This algorithm is outlined in the next subsection. In terms of computational complexity the latter approach is much more efficient.

Since each vertex, as seen in figure 4.3(b), corresponds to a gradient direction $r^{(i)}$ and also to a sample S_i, segmentation of the q-ball vertexes also implicitly segments the samples S. We segment the vertexes into k sets Z_j according to which of the k lines each vertex is the closest to. Each set Z_j is then used to estimate a diffusion tensor $X^{(j)}$ as outlined before. The result of applying our multi-tensor estimation procedure to the measurements in the example from figure 4.3(a) is seen in figure 4.3(c) using the q-ball approximation from figure 4.3(b).

Clustering of Q-ball vertexes

The result of the Q-ball procedure is a large number of vertexes distributed in 3D space. We now wish to find the k lines which best approximate the data. In order to do so we use an heuristic based on the k-means clustering algorithm.

Our modified heuristics starts with k random lines in $\mathbb{R}^3$, all going through the origin. Each Q-ball vertex is then clustered according to which of the k lines it is the closest to. This defines k sets of vertexes. For each of these sets the corresponding line is moved so that it goes through the origin and the mean of the points in each set. The process is now once again repeated by clustering vertexes to the closets of the current lines, then again moving the lines, and so on until no more vertexes change cluster. The algorithm usually converges to a steady state within a few iterations.

The final clustering of Q-ball vertexes belonging to the same lines implicitly gives the segmentation of samples that we use to estimate each of the k diffusion tensors.

In figure 4.4 we visualize the results of our multi-tensor estimation procedure in a larger synthetic example. The dataset consists of a set of diffusion weighted images S generated from equation (4.1) where the diffusion tensors $X^{(j)}$ are known and given by a torus model. The torus model defines two diffusion tensors per voxel, one with its main diffusion direction parallel to the tube, along the larger radius, and the other goes around the tube, tangential to the circular cross section. The details of how such a synthetic DTI dataset can be generated is outlined in our publication [6]. A cutting plane samples the field in figure 4.4(a) on which glyphs indicate the single-tensor fit. The diffusion is disc-shaped indicating the expected crossing fiber trajectories. Also note that there is no well defined principal direction of diffusion, so standard tractography would fail to reveal the underlying anisotropic structure in this example. Figure 4.4(b) displays the two tensors estimated per voxel using our multi-tensor estimation procedure, visualized on the same slice through the same data as in figure 4.4(a). Each glyph has been colored using the same colors as in section 3.4 to help distinguish the principal diffusion direction of each tensor. The figure shows that the estimated diffusion tensors seem to correspond well with the two orthogonal principal diffusion directions in the original data.

4.3 Multi-Tensor Fiber-Tracking

Having generalized the single tensor model of chapter 3 we now turn our attention to how we can adapt the single tensor fiber-tracking procedure outlined in section 3.5 to work in a multi-tensor framework. This work is detailed in our publication [5] and represents our second main contribution in the field of multi-tensor imaging.

In section 3.5 we outlined how single tensor fiber-tracking usually is implemented; starting from a given seed point, then by interpolation of the diffusion tensors finding the principal direction of diffusion and finally, taking a step in that direction.

Although in principle similar to this procedure, there are a couple of additional complications that arise in the context of multi-tensor fiber-tracking. One of the major ones is that the *ordering* of the multiple diffusion tensors estimated at each position is not well defined. By this we mean that the "first" tensor $X^{(j=1)}$ estimated at a given position does not necessarily have a principal direction of diffusion similar to the "first" tensor at any of the neighboring positions. This is because the principle flow of diffusion from one position to the next may be represented by any of the k tensors available, regardless of the number j we have labeled them with. Solving this *correspondence problem* to find out which tensor $X^{(j_1)}$ describes the same flow of diffusion as neighboring tensor $X^{(j_2)}$ is fundamental to calculating multi-tensor fiber-tracts.

In single-tensor tracking interpolation is usually done by estimating a weighted sum of pre-calculated neighboring diffusion tensors. In multi-tensor interpolation the correspondence problem makes this approach difficult, since it is non-trivial to determine which one of the k tensors found at each neighboring voxel position should be included in the weighted sum. In particular, if a given position p there are 8 neighboring voxels, each with $k = 2$ tensors then there is $2^8 = 256$ possible ways of choosing which one tensor to include in the 8 element weighted sum which gives the tensor at p. Because of this problem we use approach (ii) from section 3.5 when interpolating to evaluate a multi-tensor dataset.

Another difference between tracking in the single and multi-tensor case lies in how the principal direction of diffusion is calculated at each step of the algorithm. In single tensor fiber-tracking the main direction of diffusion is given by the eigenvector corresponding to the largest eigenvalue. When we estimate multiple tensors per position we also get multiple largest eigenvectors and determining which of these (if any) a trajectory should follow requires careful consideration. In our work [5] we solve the correspondence problem here by comparing each of the k principal diffusion directions with the direction taken in the previous step of the procedure. In particular, for each step of the fiber tractography algorithm, we choose the diffusion tensor which has the principal eigenvector which is the *most similar* to the principal eigenvector calculated in the previous step. In our work we define "most similar" as the one in which the angular difference is the smallest[2].

In this way we have resolved the principal diffusion direction for each step of the algorithm but the first. In the first step we have no previous direction defined and we let our algorithm estimate one trajectory for each of the k tensors estimated there. Therefore the output of our multi-tensor fiber-tracking algorithm is k trajectories per seed point. As in the single tensor case, each of these tracts are terminated when the trajectory being followed encounters a position in which the corresponding tensor has FA lower than a given threshold value. One interpretation of this choice is that we use prior information to determine which of the k tensors describe the "flow of diffusion" the algorithm is currently following. The tract is then terminated once the FA of this tensor becomes too small.

In figure 4.4(c) we show the results of applying our multi-tensor tracking algorithm to the estimated multi-tensor dataset visualized in figure 4.4(b) and described on page 62. We start tracking in three seed positions chosen such that the $k = 2$ tensors defined there each have relatively high FA. We then visualize the k trajectories estimated for each of these seed positions. Since these dataset is defined to have constant FA for each

[2]A consequence of this choice is that the algorithm will tend to favor producing fibers which do not change direction. This is obviously problematic in terms of the situation of "kissing fibers" (discussed in paper C) in which fibers with higher degree of curvature would more accurately describe the actual diffusion. In the absence of an *a priori* model (such as an atlas giving probabilities of crossing vs. kissing) it is not known how the case of kissing fibers should be handled.

point in the fiber-trajectory we terminate the paths in this example after a predefined number of steps of the tractography algorithm. We observe that the estimated fibers seem reasonable compared to the principal direction of diffusion given by the multi-tensor glyphs seen in figure 4.4(b).

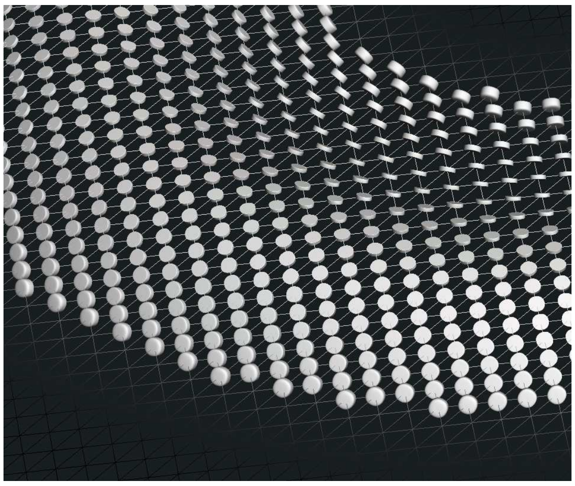
Figure 4.5 shows results from a DTI dataset captured from a healthy human volunteer. The scanning was performed using a 3T GE system with 51 gradient directions and with $b = 700$. The total scan time was 17 minutes. The ROI is centered around the corpus callosum and the ascending fibers of the internal capsule and corona radiata. Prior anatomical knowledge indicates that crossing or kissing fibers can be found in the intersection of these structures. The single tensor glyphs in figure 4.5(a) indicate a region of high and consistently oriented planar anisotropy, suggesting fiber crossing. Indeed, single-tensor fiber tractography results seeded in the area (figure 4.5(b)) fail to pass through the area, due to the single tensor fit dropping below the FA $= 0.15$ threshold. The two-tensor fiber-tracking shown in figure 4.5(c) uses the same FA $= 0.15$ threshold, with tracts passing through the region of single-tensor planar anisotropy. As expected from prior knowledge of anatomy, some of the fibers from the internal capsule successfully extend upwards into the corona radiata.

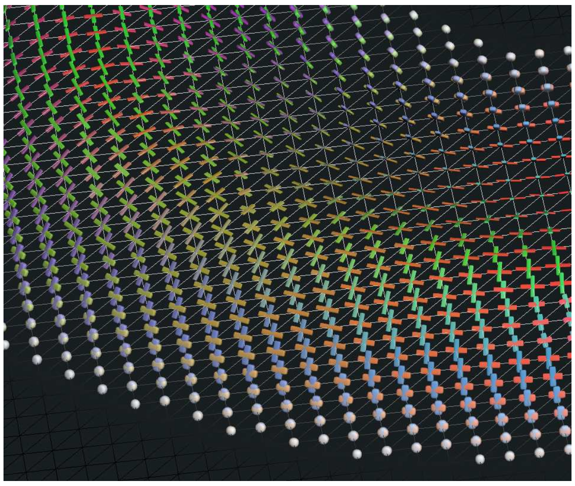
4.4 Conclusion

In this chapter we have extended the diffusion tensor framework developed in chapter 3 so that it can better represent the actual physiology in certain cases. These new methods are fast, relatively easy to implement and quite similar to their well-known and well-understood single tensor equivalents. Therefore they provide a vital complement to, and natural extension of, the standard single-tensor model.

It is important to point out that these new quantities allows the investigation of regions of the brain where there are crossing fibers, and that tensor methods previously has failed to compute accurate descriptors and trajectories in these regions. Therefore our algorithms has the potential to lead to significant clinical applications. The fact that the approach is relatively fast and inexpensive to compute suggests that it can be useful e.g. for interactive examination of patient data. In such an application switching between standard single tensor and our multi-tensor method would be simple, only requiring a change of the “number of tensors” parameter to visualize the difference between the different approaches.

A reference implementation of our k-tensor estimation routine is freely available in the open-source project Teem. This should make quantitative comparison to other methods, such as those outlined in section 4.1.1, relatively easy. Our initial experiments seems to indicate that our method performs favorably since it does not impose simplifying constraints, such as cylindrical diffusion or the same amount of diffusion in each of the two principal directions, which may not generally be justified in real data. Our algorithm is also simpler to compute as it does not require non-trivial non-linear minimization.


(a) The single tensor fit.


(b) Our 2-tensor fit.

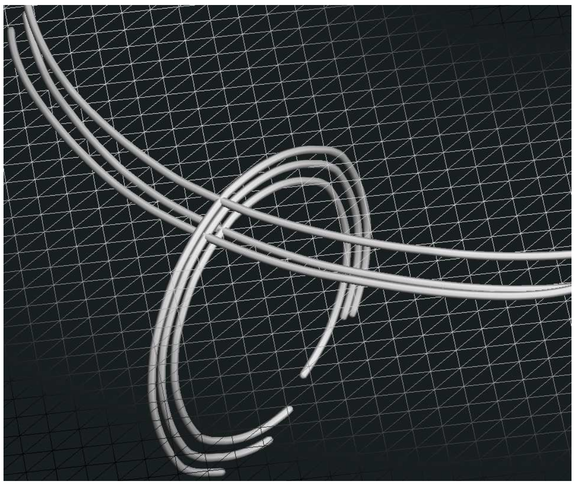
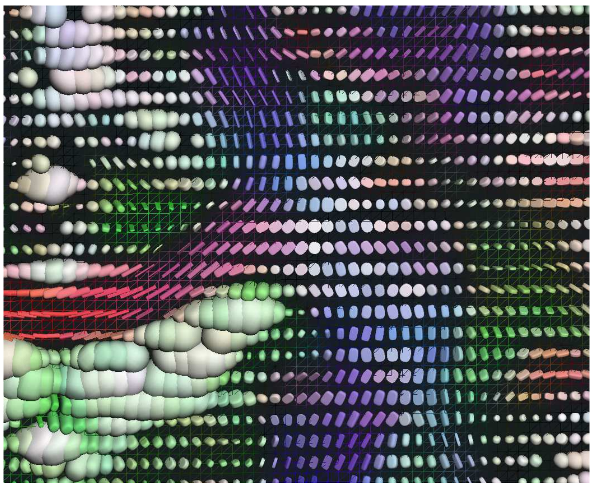
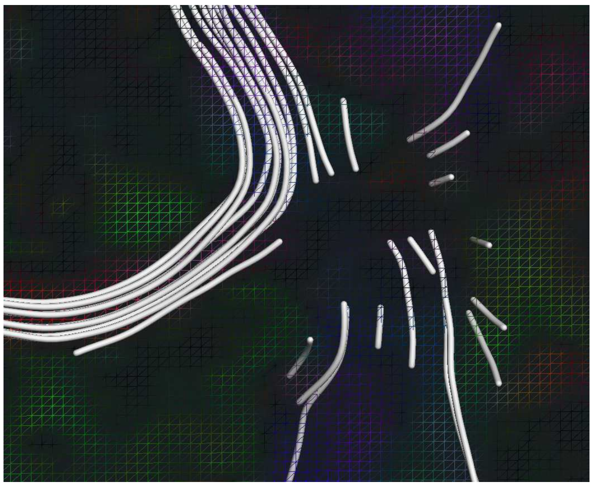

(c) Our multi-tensor fiber-tracking procedure.

Figure 4.4: The synthetic diffusion tensor torus dataset.


(a) Single tensor estimation.


(b) Single-tensor tractography seeded in region.

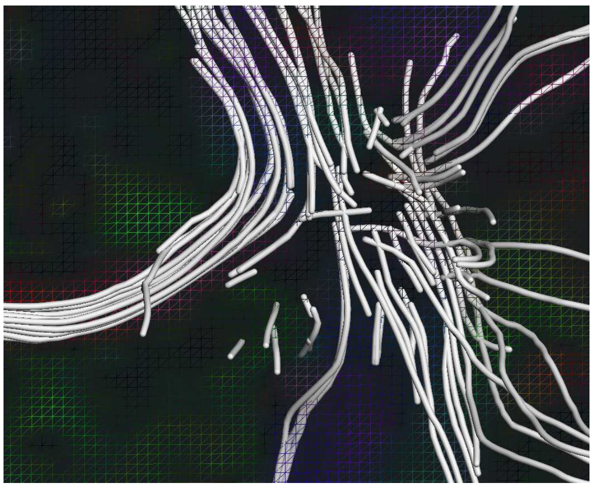

(c) 2-tensor tractography with same seed points.

Figure 4.5: Results in a coronal slice near corpus callosum and corona radiata.

Chapter 5

Conclusion

In this chapter we will summarize the main contributions of this thesis. As stated in the overview section, the overall purpose has been to investigate and improve algorithms which can be applied to improve the the results of DTI analysis, including specifically, fiber tracking.

The research problems that we have investigated to fulfill the overall objective can naturally be grouped into two separate types: those that work by improving the quality of the sampled data, and those that provide a more accurate description of actual diffusion. In chapter 1, 2 and 3 we presented our contributions for improving sampled data, and in chapter 4 we presented ways to more accurately estimate the diffusion in the special case of crossing fibers.

Our main contributions in each of these two approaches can be summarized as follows

- Quality improvement
  - We have created an algorithm which can improve the signal-to-noise ratio in DTI data significantly.
  - We have created an algorithm which can solve blurring problems with increased accuracy in certain cases.
  - We have developed a conceptual model which can be used to generate synthetic datasets for the testing of e.g. noise removal algorithms or multi-tensor estimation procedures.
- Generalized DTI descriptors
  - We have created a way to estimate multi-tensors in DTI data for more accurate description of actual diffusion.
  - We have generalized fiber-tracking so that this can be performed in multi-tensor datasets.

Out of these the contributions the ones which may have the most immediate applications, as discussed in sections 3.6 and 4.4, are the 3D SA-DCT denoising algorithm and the multi-tensor estimation and fiber-tracking algorithms respectively.

5.1 Future work

Being a PhD student means that at some point you have to decide that there is no more time to do more research. This means that there are interesting questions that for the moment must be left unanswered. If I had more time to continue my work, the following is a non-exhaustive list of problems that I would have liked to look more into:

- Investigate the details of how the deconvolution algorithm should be applied to MRI data.
- Generalize the SA-DCT algorithm to also explicitly handle other noise distributions than the normal distribution.
- Quantitative comparisons with other multi-tensor estimation procedures should be performed, including performance on noisy data.
- Find criteria for determining from DTI data how many diffusion tensors are needed locally to describe the actual diffusion, i.e. to determine automatically when multi-tensor imaging should be applied.
- Investigate how the multi-tensor fiber-tracking algorithm should be extended to perform tracking with different descriptors, such as the Q-ball, directly.

Bibliography

[1] The USC-SIPI image database, Signal & Image processing Institute, Electrical Engineering Department, University of Southern California.

[2] PJ Basser, S Pajevic, C Pierpaoli, J Duda, and A Aldroubi. In vivo fiber tractography using DT-MRI data. *Magnetic Resonance in Medicine*, (44):625–632, 2000.

[3] Ø Bergmann, O Christiansen, J Lie, and A Lundervold. Shape adaptive DCT for denoising of 3D scalar and tensor valued images. *Journal of Digital Imaging*, 2008.

[4] Ø Bergmann, G Kindlmann, A Lundervold, and C-F Westin. Diffusion k-tensor estimation from q-ball imaging using discretized principal axes. In *9th International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI)*, pages 268–275, October 2006.

[5] Ø Bergmann, G Kindlmann, S Peled, and C-F Westin. Two-tensor fiber-tractography. In *2007 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pages 796–799, April 2007.

[6] Ø Bergmann, A Lundervold, and T Steihaug. Generating a synthetic diffusion tensor dataset. In *Eighteenth IEEE Symposium on Computer-Based Medical Systems*, pages 277–281. IEEE Computer Society Press, June 2005.

[7] Ø Bergmann and T Steihaug. Solving a TRS which has linear inequality constraints. *Submitted to Optimization Methods and Software*, 2007.

[8] AR Conn, NIM Gould, and PL Toint. *Trust-region Methods*. MPS-SIAM Series on Optimization. SIAM, 2000.

[9] M Descoteaux, E Angelino, S Fitzgibbons, and R Deriche. A fast and robust ODF estimation algorithm in q-ball imaging. In *Symposium Proceedings: From Nano to Macro*, IEEE International Symposium on Biomedical Imaging, 6–9 April 2006.

[10] A Foi, V Katkovnik, and K Egiazarian. Pointwise shape-adaptive dct as an overcomplete denoising tool. Number 5 in The International Workshop on Spectral Methods and Multirate Signal Processing. IEEE SP/CAS and Tampere International Center for Signal Processing, June 2005.

[11] A Foi, V Katkovnik, and K Egiazarian. Pointwise shape-adaptive DCT denoising with structure preservation in luminance-chrominance space. Number 2 in The

International Workshop on Video Processing and Quality Metrics for Consumer Electronics, January 2006.

[12] A Foi, V Katkovnik, and K Egiazarian. Shape-adaptive DCT for denoising and image reconstruction. Proceedings of SPIE Electronic Imaging 2006, Image Processing: Algorithms and Systems V, 2006.

[13] RC Gonzalez and RE Woods. *Digital Image Processing*. Prentice Hall, 2002.

[14] M Hanke, JG Nagy, and C Vogel. Quasi-newton approach to nonnegative image restoration. *Linear Algebra and its applications*, 316:223–236, 2000.

[15] PC Hansen. *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*. Monographs on Mathematical Modeling and Computation. SIAM, 1987.

[16] G Kindlmann. Teem: Tools to process and visualize scientific data and images. Software, http://teem.sourceforge.net/.

[17] G Kindlmann. *Visualization and Analysis of Diffusion Tensor Fields*. PhD thesis, School of Computing, University of Utah, Salt Lake City, UT 84112 USA, September 2004.

[18] JF Mangin, C Poupon, C Clark, D Le Bihan, and I Bloch. Distorion correction and robust tensor estimation for MR diffusion imaging. *Medical Image Analysis*, 6:191–198, 2002.

[19] J Nocedal and SJ Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer Verlag, August 27 1999.

[20] S Peled, O Friman, F Jolesz, and C-F Westin. Geometrically constrained two-tensor model for crossing tracts in DWI. *Magnetic Resonance Imaging*, (24):1263–1270, 2006.

[21] S Peled and C-F Westin. Geometric extraction of two crossing tracts in DWI. In *Proceedings of 13th ISMRM*, 2005.

[22] F Rendl and H Wolkowicz. A semidefinite framework for trust region subproblems with applications to large scale minimization. *Mathematical Programming*, 2:273–299, 1997.

[23] M Rojas. *A large-scale trust-region approach to the regularization of discrete ill-posed problems*. PhD thesis, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 1998. Technical Report TR98-19.

[24] M Rojas, SA Santos, and DC Sorensen. A new matrix-free algorithm for the large-scale trust-region subproblem. *SIAM Journal on Optimization*, 11(3):611–646, 2000.

[25] M Rojas, SA Santos, and DC Sorensen. LSTRS: Matlab software for large-scale trust-region subproblems and regularization. Technical Report 2003-4, Department of Computational and Applied Mathematics, Rice University, Houston, TX, USA, 2003.

[26] M Rojas and T Steihaug. An interior-point trust-region-based method for large-scale non-negative regularization. *Inverse Problems*, (18), 2002.

[27] SA Santos and DC Sorensen. A new matrix-free algorithm for the large-scale trust-region subproblem. Technical Report TR95-20, Department of Computational and Applied Mathematics, Rice University, Houston, TX, USA, 1995.

[28] J Sijbers. *Signal and noise estimation from Magnetic Resonance Images*. PhD thesis, University of Antwerpen, 1998.

[29] T Sikora. Low complexity shape-adaptive DCT for coding of arbitrarily shaped image segments. *Signal Processing: Image Communication*, 7:381–395, 1995.

[30] S Skare. *Optimisation strategies in diffusion tensor MR imaging*. PhD thesis, MR Center, Department of Clinical Neuroscience, Karolinska Institutet, Stockholm, Sweden, 2002.

[31] S Skare, T-Q Li, B Nordell, and M Ingvar. Noise considerations in the determination of diffusion tensor anisotropy. *Magnetic Resonance Imaging*, 18:659–669, 2000.

[32] SW Smith. *Digital Signal Processing. A Practical Guide for Engineers and Scientists*. Demystifying Technology Series. 2003.

[33] DC Sorensen. Minimization of a large-scale quadratic function subject to a spherical constraint. *SIAM Journal on Optimization*, 7(2):141–161, 1997.

[34] DS Tuch. *Diffusion MRI of Complex Tissue Structure*. PhD thesis, Massachusetts Institute of Technology, January 2002.

[35] DS Tuch. Q-ball imaging. *Magnetic Resonance in Medicine*, 52:1358–1371, 2004.

[36] DS Tuch, TG Reese, MR Wiegell, N Makris, JW Belliveau, and VJ Wedeen. High angular resolution diffusion imaging reveals intravoxel white matter fiber heterogeneity. *Magnetic Resonance in Medicine*, (48):577–582, 2002.

[37] AN Tychonoff. Solution of incorrectly formulated problems and the regularization method. *Soviet Math*, Dokl 4:1035–1038, 1963.

[38] C-F Westin, SE Maier, H Mamata, A Nabavi, FA Jolesz, and R Kikinis. Processing and visualization for diffusion tensor MRI. *Medical Image Analysis*, 6:93–108, June 2002.

[39] C-F Westin, S Peled, H Gudbjartsson, R Kikinis, and FA Jolesz. Geometrical diffusion measures for MRI from tensor basis analysis. In *ISMRM '97*, page 1742, Vancouver Canada, April 1997.