Compressed Sensing in Low-Field MRI

Using Multiplicative Regularization

M.A. Verseput

Delft University of Technology

TU Delft

Compressed Sensing in Low-Field MRI

Using Multiplicative Regularization

by

M.A. Verseput

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Friday December 1, 2023 at 14:00.

Student number:	4603796
Master program:	Electrical Engineering
Project duration:	January, 2023 – December, 2023
Thesis committee:	Dr. R.F. Remis
	Ir. B. de Vos
	Dr. O.A. Krasnov

Abstract

Magnetic resonance imaging (MRI) is a non-invasive tool to image the body's anatomy and physiology, but suffers from long scan times. Compressed Sensing (CS) is used to accelerate MRI scans by incoherently taking fewer measurements and using a nonlinear optimization algorithm to image the undersampled data. Convex optimization techniques are generally used for image reconstruction. Minimizing a data fidelity term along with two regularization terms, which are a total variation (TV) based and a wavelet transform based function, is a standard procedure in CS-MRI. Regularization parameters are needed to balance the different terms, but it is impossible to know upfront what the optimal regularization parameters are to get the desired output. A consequence is that the algorithm of choice needs to be executed many times for many different values of the regularization parameters, which is a time-consuming process requiring knowledge of the algorithm.

In this work we rewrite and implement the regularization functions in a multiplicative manner by multiplying the data fidelity term with the regularization terms, thereby eliminating the need to tune the regularization parameters. Moreover, we include a region of support (ROS) mask to further accelerate reconstruction. The performance of different combinations of regularization functions and reconstruction algorithms are validated on a simulation study and various experiments on a low-field MRI scanner. This also shows the capability of CS applied to low-field MRI, which has lower signal-to-noise ratio compared to conventional MRI. Of all proposed methods, a nonlinear conjugate gradient method applied to the fully multiplicatively regularized objective function shows the most robust performance.

Preface

The thesis that lies before you is the final chapter of my Master of Science in Electrical Engineering. For the past ten months I have been working on this joint project between the Delft University of Technology and the Leiden University Medical Center under the supervision of my main supervisor Rob Remis and my daily supervisor Bart de Vos. This project gave me the wonderful opportunity to study and experiment with reconstruction methods in (low-field) magnetic resonance imaging, which I found to be a perfect fit for my interests. Despite the occasionally challenging times and non-linear progress, I was ever motivated and would say I am happy with how everything turned out in the end. I hope you enjoy reading it!

Of course, I would like to take this opportunity to thank the people who supported me during this process.

Rob, for introducing me to this project and supervising me the past months. Your enthusiasm was inspiring and the courses you lectured were interesting, which is why I approached you for a potential master thesis project. I always looked forward to our informal meetings.

Bart, for always taking the time to help me with my struggles and ideas and keeping the practical relevance in mind. This report would not look half as good without your help. Also, thanks for babysitting me during the experiments with the scanner.

Dr. Oleg Krasnov, for taking the time to be in my thesis committee.

The Gorter MRI group, for the much needed coffee and lunch breaks. A special thanks to Prof. Andrew Webb for giving me the project and the low-field group for allowing me to perform my experiments on the scanner.

The Signal Processing Systems section, for allowing me to do my project in their group and the inspiring seminars.

My study friends Jurgen, Florens, Sheng, Bob, and Patrick, for the most random lunch break conversations to always look forward to.

My roommates, for all the life and laughter everyday when I come home.

Tess and my family, for your love, patience, and support.

M.A. Verseput
Delft, November 2023

Contents

Mathematical Notation

The following mathematical notation is used throughout the report, unless indicated otherwise:

x	scalar
$\mathbf{x}$	vector
$\mathbf{X}$	matrix
$\mathbf{I}$	identity matrix
$\mathbf{1}$	all ones matrix
$\mathbf{0}$	all zeros matrix
$(\cdot)^T$	transpose
$(\cdot)^*$	complex conjugate
$(\cdot)^H$	Hermitian
$\langle \cdot, \cdot \rangle$	inner product
diag$(\mathbf{x})$	matrix with elements of $\mathbf{x}$ on the diagonal
$\lVert \cdot \rVert_p$	l_p-norm
$*$	convolution
$\odot$	Hadamard / elementwise matrix product
$\otimes$	Kronecker product
$\oplus$	Morphological dilation
$\ominus$	Morphological erosion
$\bullet$	Morphological closing
$\circ$	Morphological opening

1 Introduction

This chapter gives an introduction to this master thesis. The motivation for low-field magnetic resonance imaging (MRI) and the work discussed in this thesis is given in Section 1.1. Section 1.2 discusses the problem of long scan time in MRI and how this process can be accelerated. The research objective is stated in Section 1.3 and the outline of the remaining part of this thesis is given in Section 1.4.

1.1. Motivation

Magnetic resonance imaging is a useful tool to image the body's anatomy and physiology without harmful radiation. Patients lie in a large magnet which creates a uniform magnetic field. The signal in MRI comes from Hydrogen nuclei in the body, which get aligned by a uniform magnetic field. A number of electromagnetic pulses are transmitted by an RF coil to push the nuclei out of alignment and as these nuclei return to equilibrium they also emit radio-waves which get measured. A varying secondary magnetic field is created by gradient coils to allow spatial encoding such that we know where each signal exactly comes from. The fields are constructed such that the relation between our measured data and the desired image is a Fourier transform. For a more detailed explanation, see for example [1].

Modern MRI units utilize high magnetic field strengths which result into images with high signal-to-noise ratio (SNR). This however, comes at a cost of millions of euros of initial and maintenance costs. Moreover, operating MRI machines requires trained personnel, has a large physical footprint, and requires active cooling. Many countries, i.e. most developing countries in Africa, do not have access to these kind of resources. This is what originally motivated the research into low-field systems. The low-field MRI project, which started as a collaboration between the Delft University of Technology and the Leiden University Medical Center (LUMC) and is now further developed at the LUMC, set out to construct a system which is able to image hydrocephalus in children in developing countries [2]. The target patient group was set at children up to 8 years old with hydrocephalus. Nowadays, the use of this technology is extended to find purpose in point-of-care (POC).

The magnet used is a Halbach array [2, 3, 4, 5, 6], shown in Fig. 1.1, with the purpose of performing brain scans on both children and adults. There are a number of advantages by utilizing this design. The array has a length of 50 cm and a diameter of 30 cm [7] and consists mainly of permanent magnets. This makes the construction lightweight, portable, and low-cost. It also does not need high power or cooling, unlike the coils in conventional MRI scanners which dissipate a lot of heat during operation due to the resistance of the wires and the current that flows through it. The array can also be configured such that an approximately uniform magnetic field is produced inside the array while producing a minimal external field, reducing any safety concerns during POC use and transport. Although the experiments in this thesis are performed using a Halbach array construction, research into other magnet structures is being conducted as well [8]. As one might expect, the new design also comes with disadvantages. Conventional MRI scanners operate with a main magnetic field strength B_0 of 1.5 or 3 T, but the Halbach array scanner at the LUMC has a main magnetic field of about 50 mT within a 15 cm diameter spherical volume in the center of the magnet [3]. As the SNR is proportional to a fractional power of B_0 [9, 10] the SNR in the low-field system is significantly lower compared to conventional scanners. Furthermore, it is inevitable that there are small manufacturing inaccuracies among magnets and the rings which hold them. The magnets and rings are produced individually and the field can vary due to the temperature dependence of permanent magnets [2] making the B_0 field less uniform as well as creating field drift during scans. To compensate for the lower SNR we can increase scantime. However, a fundamental disadvantage also present in conventional MRI scanners, is the already long time to perform a scan. Patients have to lie still in a confined space for possibly tens of minutes, which is especially hard for children who were the initial target group, resulting in a uncomfortable experience. Therefore, accelerating the scan process in (low-field) MRI is an important research topic.

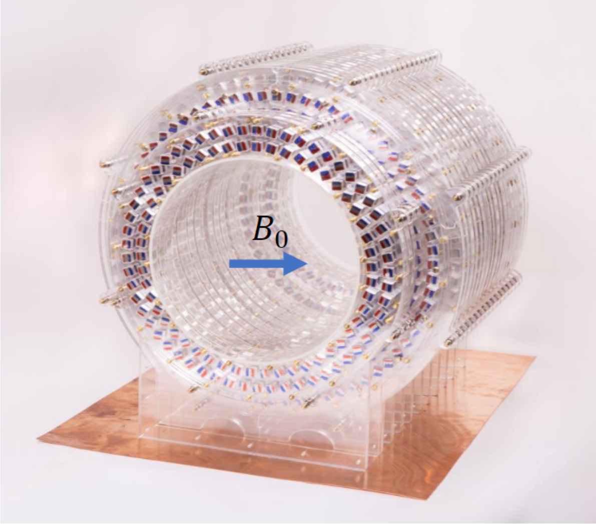

Figure 1.1: The constructed Halbach array at the LUMC. *T. O'Reilly, et al., Journal of magnetic resonance* (2019) [2]

1.2. Accelerating MRI Scans

Making MR images takes a long time due to the manner in which data are gathered. A gridded data space called k-space needs to be filled and the number of k-space points which can be sampled per unit of time is limited by different physical phenomena, such as gradient amplifier strength, slew rate and peripheral nerve stimulation. One way to cut into the scanning time is to use parallel imaging (PI) [11]. Parallel imaging uses multiple receive coils instead of one and by making use of the information on placement and sensitivity of the coils fewer k-space points need to be sampled. This does however come with the downside that a lot of extra hardware is required and PI is also more difficult to implement in low-field due to coil coupling. Another way to accelerate the scan time, which does not necessitate additional hardware, is to use partial Fourier [1]. This technique makes use of the conjugate symmetry property k-space in theory holds, making it possible to scan up to twice as fast. In practice however, k-space always contains phase errors due to a number of different sources, e.g. inhomogeneity of the main magnetic field or motion of the patient within the scanner, resulting in an acceleration factor of less than 2 [1].

The technique researched in this thesis is compressed sensing (CS) [12, 13]. This technique does not require any additional hardware, in contrast to PI and allows for higher acceleration factors than 2, in contrast to partial Fourier. Compressed sensing, or compressive sampling, relies on the fact that there is redundancy in the data. If there is a lot of redundant information in the data it is possible to compress the amount of stored data such that less memory is needed while the quality of the image is maintained. We can take advantage of this compressibility, or sparsity, when scanning. If the data are randomly undersampled and the image is sparse, either in itself or when transformed to another mathematical domain, it might be possible to fully recover the desired image using an iterative reconstruction algorithm. The reconstruction problem can be cast as a convex optimization problem in which we minimize the difference between the measured data and the obtained solution and add constraints to ensure that the (transformed) image remains sparse. A great abundance of convex optimization tools is available to solve the abovementioned problem efficiently if all terms, the data consistency term and any regularization terms, can be formulated as convex functions [14, 15]. All functions can be combined into a minimization problem as

$$\min_{\mathbf{x}} F(\mathbf{x}) = F^{\text{data}}(\mathbf{x}) + \sum_i \lambda_i F_i^{\text{reg,add}}(\mathbf{x}) \ , \tag{1.1}$$

where $\mathbf{x}$ is the image, $F^{\text{data}}(\mathbf{x})$ is the data consistency term, $F_i^{\text{reg,add}}(\mathbf{x})$ are regularization terms and λ_i are regularization parameters. A regularization parameter weighs how important a regularization function is, i.e. a high value for the regularization parameter means that the corresponding regularization function is more important and vice versa. A downside of this method is that it is impossible to know upfront how to balance the different terms to get the desired output. The consequence of not knowing the optimal values for the regularization parameters is that the algorithm of choice needs to be executed many times for many different values of the regularization parameters, which is a time-consuming process. An alternative of including the regularization functions in an additive way, is to transform the regularization functions such that they can be used in a multiplicative way as

$$\min_{\mathbf{x}} F(\mathbf{x}) = F^{\text{data}}(\mathbf{x}) \prod_i F_i^{\text{reg,mult}}(\mathbf{x}) \ , \tag{1.2}$$

where $F_i^{\text{reg,mult}}(\mathbf{x})$ is a regularization term unequal to $F_i^{\text{reg,add}}(\mathbf{x})$. While a lot of theory and algorithms are developed for the general minimization problem of (1.1), not as many are developed for the problem of (1.2). Moreover, Eq. (1.2) has been researched for denoising and deblurring while using 1 multiplicative regularization term with a particular transform called total variation (TV) [16, 17, 18], but not yet in a compressed sensing setting, using a mix of additive and multiplicative regularization terms, or using multiple multiplicative regularization terms.

1.3. Research Objective

The goal of this research project is to make a compressed sensing pipeline for the low-field MRI system at the LUMC, which requires as little human interaction and tuning of parameters as possible. The additional input parameters, next to the standard required input parameters to perform a scan, are the option to turn compressed sensing on and how much data have to be sampled. Based on the output data, an image needs to be reconstructed that comes close to the quality of the image if fully sampled. It is desirable to include as few free parameters as possible, such that the user does not have to tune them themselves. We do this by choosing algorithms which introduce a small number of free parameters and by adopting multiplicative regularization.

To further improve reconstruction speed it might be possible to outline the object of interest from the pre-reconstructed image [19]. This may help the reconstruction algorithm in narrowing the solution space to just within the image domain mask. Preferably we do not make any additional scans and base this mask purely on the undersampled data. In this way, the patient does not have to be inside the scanner for a longer time than necessary. Another constraint is that the algorithm performing the image segmentation should be fast. By the time we perform image segmentation the patient does not have to lie in the scanner anymore, which is the most important part, but the image domain mask is practically useless if constructing the mask takes longer than the time we gain by incorporating the mask within the reconstruction algorithm. The main research question can be formulated as follows:

How can we make a compressed sensing pipeline for the LUMC low-field MRI system which requires minimal human interaction?

A block diagram of the compressed sensing pipeline is shown in Fig. 1.2. The user enters the scan parameters in the GUI of the pulse sequence program and has the option to turn on compressed sensing, pick an undersampling ratio and the sampling rate in k-space center. The (undersampled) k-space data are sent back to the pulse sequence program during scanning. The data are sent to MATLAB/Python to first create an image domain mask. Lastly, both the k-space data and the image domain mask are sent to the reconstruction algorithm to obtain an output image.

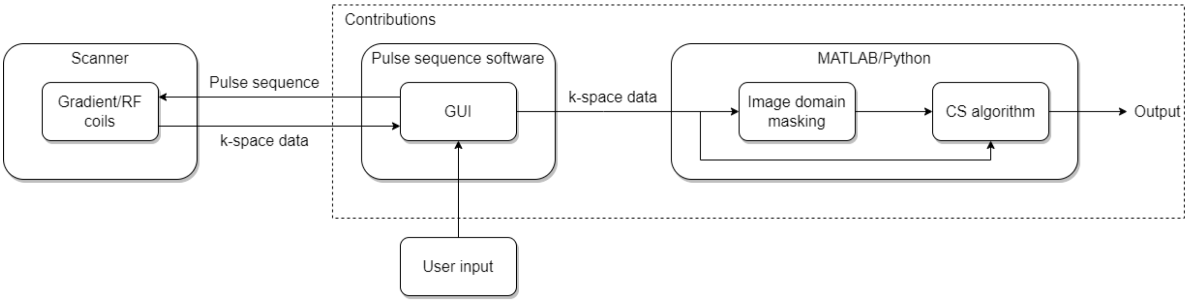

Figure 1.2: Block diagram of the compressed sensing pipeline.

1.4. Thesis Outline

In Chapter 2 general theory on k-space and compressed sensing are given. How an undersampling mask for k-space and a mask in image domain are created is discussed in Chapter 3. In Chapter 4 a number of variations on the optimization problem for reconstruction including algorithms to solve them are analyzed. Simulation studies and reconstructions from experimental data are shown and discussed in Chapter 5 and the conclusions and future recommendations are given in Chapter 6.

2 Background

In this chapter we introduce the basics of k-space needed to create a proper undersampling mask in Section 2.1. In Section 2.2 sparsity and incoherence, important concepts for CS, are further explained and two of the most used transformations for CS in MRI are given, namely total variation and the wavelet transform.

2.1. k-Space

The raw data acquired during scanning are gathered in k-space. k-Space is a gridded space, equal in dimensions to the desired image, of which each gridpoint represents a unique spatial frequency. The intensity of each gridpoint resembles the extent to which the corresponding spatial frequency is found in the image. The location of the gridpoint determines which spatial frequency it represents. If the dimensions of a three-dimensional image are represented by x, y, and z, then k-space is also arranged such that points along k_i represent how much a spatial frequency along i is present in the image, for $i \in \{x, y, z\}$. The further a point is located from the origin of k-space, the higher the frequency the point represents is. This means that the center of k-space contains information on lower frequencies, i.e. general shapes, and the outer parts of k-space contains information on higher frequencies, thus the finer details [1].

To acquire an image from the measured k-space, one can simply perform a discrete Fourier transform (DFT). The DFT is computed efficiently using the fast Fourier transform (FFT) method. An inverse FFT (IFFT) is performed to get the image from k-space data and conversely, an FFT is performed on the image to acquire its k-space data.

To acquire the k-space points, different types of trajectories are possible. A few of the most popular trajectories are Cartesian, radial, and spiral sampling [1]. This report will focus on Cartesian sampling, as this is the most used trajectory at the low-field scanner at the LUMC. Cartesian sampling is also the most intuitive when considering undersampling patterns. When using a 3D Cartesian sampling pattern, two phase direction steps followed by a frequency encoded data acquisition can be distinguished. The phase encoding steps are time consuming, so if we want to combine 3D Cartesian sampling with an undersampling pattern, we can simply ommit points in the phase encoding plane [13]. The acceleration factor is then inversely proportional to the fraction of number of read out lines sampled.

A last, but important aspect is the energy distribution in k-space. Most of the energy can be found in the center of k-space. When undersampling k-space it is critical to emphasize sampling in the center of k-space to keep enough signal, while also sampling enough outer k-space points to recover the details of the image.

2.2. Fundamental Principles of Compressed Sensing

Conventional sampling in MRI requires a sampling rate which satisfies the Nyquist-Shannon sampling theorem to obtain all information of the desired signal. Trying to reconstruct k-space, which is undersampled at a constant rate, through an inverse Fourier transform will result in aliasing, so both another undersampling pattern and reconstruction method are needed. Compressed sensing theory states that it is possible to reconstruct the signal in an underdetermined system when three criteria are met. The first two of these criteria are sparsity and incoherence. The last criterion is applying a nonlinear/iterative reconstruction method, which will be further discussed in Chapter 4.

2.2.1. Sparsity and Incoherence

A signal is sparse when it can be represented with relatively few coefficients. For example, an image consisting of many zero-valued pixels and only a few nonzero pixels, is sparse. An MR image may not be sparse in itself, but it may be transformed to any other basis Ψ in which it does have a sparse representation. If the signal indeed has a sparse representation in Ψ we call this transform sparsity. In case $\Psi = I$ (the identity matrix) the transform domain is the image domain itself, but as most MR images are not sparse in itself other transforms like total variation and the wavelet transform are more regularly used in CS-MRI. Also, in practice we always have noise in our images. This means that in practice we have approximate sparsity, i.e. there are relatively many coefficients close to 0. The implication of sparsity in Ψ is that many small coefficients may be dismissed without losing much or any information about the signal. This thus makes the signal compressible when the exact values and locations of the few significant coefficients are known, but does not enable nonadaptive data acquisition on its own. This is because the values and locations of significant coefficients are signal dependent and not known in advance.

Next to sparsity we need incoherent sampling. In [20] the coherence μ between orthonormal bases Φ and Ψ both in $\mathbb{R}^n$ is defined as

$$\mu(\Phi, \Psi) = \sqrt{n} \cdot \max_{1 \leq k, j \leq n} |\langle \phi_k, \psi_j \rangle|, \tag{2.1}$$

where $\langle \cdot\, , \cdot \rangle$ is the inner product. Coherence measures the correlation between elements of Φ and Ψ. If Φ and Ψ contain elements which correlate highly, there is high coherence/low incoherence and vice versa. Because $\mu \in [1, \sqrt{n}]$, maximal coherence corresponds to $\mu(\Phi, \Psi) = \sqrt{n}$ and maximal incoherence corresponds to $\mu(\Phi, \Psi) = 1$. It turns out that random matrices are incoherent with any basis Ψ. The coherence between Ψ and a uniformly randomly selected Φ is $\sqrt{2 \log n}$ and likewise, any Φ with independent identically distributed (i.i.d.) entries will have very high incoherence [20]. For compressed sensing we are interested in incoherent bases, e.g. by randomly undersampling our data, because the resulting artifacts will appear noise-like in the sparse domain.

2.2.2. Total Variation

As most MR images are not sparse themselves, we need a mathematical basis in which they are. One such basis popular in MRI is total variation [21, 22], which is based on the gradient of the signal. The total variation of a continuous signal is defined as

$$\text{TV}(F(\mathbf{x})) = \int_a^b \nabla F(\mathbf{x}) d\mathbf{x} \ , \tag{2.2}$$

where $F(\mathbf{x})$ is a continuous signal defined on the interval $[a, b]$. In the discrete case however, for example an image with a finite number of pixels, there are different definitions using finite differences. This is because there is no one absolute definition of the gradient of a discrete signal. Two well known variants of the total variation are the isotropic total variation and the anisotropic total variation [22]. The isotropic TV is defined as

$$\begin{aligned} \text{TV}(x) = \sum_{n_1=1}^{N_1} \sum_{n_2=1}^{N_2} \sum_{n_3=1}^{N_3} & \left(x(n_1+1, n_2, n_3) - x(n_1, n_2, n_3)\right)^2 + \left(x(n_1, n_2+1, n_3) - x(n_1, n_2, n_3)\right)^2 \\ & + \left(x(n_1, n_2, n_3+1) - x(n_1, n_2, n_3)\right)^2 \Big)^{1/2} \ , \end{aligned} \tag{2.3}$$

where x is an image of $N_1 \times N_2 \times N_3$ voxels. We use the isotropic TV, because the anisotropic TV favours structures which are directed along the major axes during minimization. The definition for the isotropic TV above is given using forward differences, but can also be implemented using backward differences or a mix of both. The TV is used in for example denoising, deblurring, and compressed sensing for its edge preserving capabilities [22]. An example of a finite differences image of an MRI image of a brain is shown in Fig. 2.1.b.

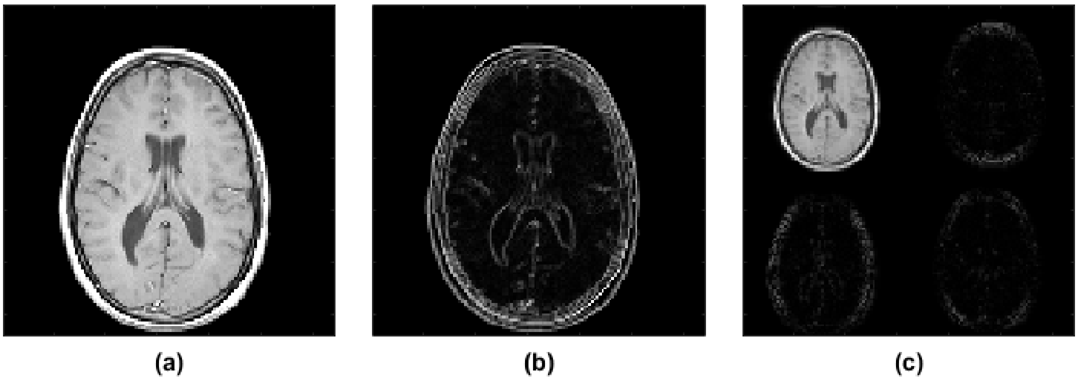

Figure 2.1: Different representation bases of the same image. (a) MRI image slice of a brain, (b) finite differences, (c) wavelet transform.

2.2.3. Wavelet Transform

With a wavelet transform, the signal gets convolved with a set of orthogonal basis functions, the wavelets [23]. The wavelets are characterized by a scale and a shift parameter. The scale parameter represents the frequency range and the shift parameter represents the location within the signal. Coarse-scale coefficients represent the lower frequency content and fine-scale coefficients represent the higher resolution content. Figure 2.1.c shows a level 1 wavelet decomposition, which means that the original image is decomposed in an approximate image, i.e. the coarse-scale content, and in the horizontal, vertical, and diagonal fine-scale details. Another wavelet decomposition can be performed on the approximate image in the same way, making it a level 2 wavelet decomposition. Each time, increasingly more details are extracted from the approximate image making the representation sparser but also requiring more computations. Different wavelet transforms exist, such as the Haar wavelets and Coiflets [23]. We will use the Daubechies wavelets as these are the most widely used both in CS-MRI and in general signal processing.

It is evident from Fig. 2.1 that the gradient image and the wavelet transform of an MR image look significantly sparser than the original MR image, because they contain significantly more black pixels compared with the original image. Additionally, Fig. 2.2 shows the normalized mean squared error (MSE) of the image shown in Fig. 2.1.a with both the image itself and its wavelet transform when an increasing number of the smallest coefficients is omitted. We can see that the wavelet transform needs a significantly smaller number of coefficients to keep a more accurate representation compared to the image itself, which confirms the induced sparsity of the wavelet transform.

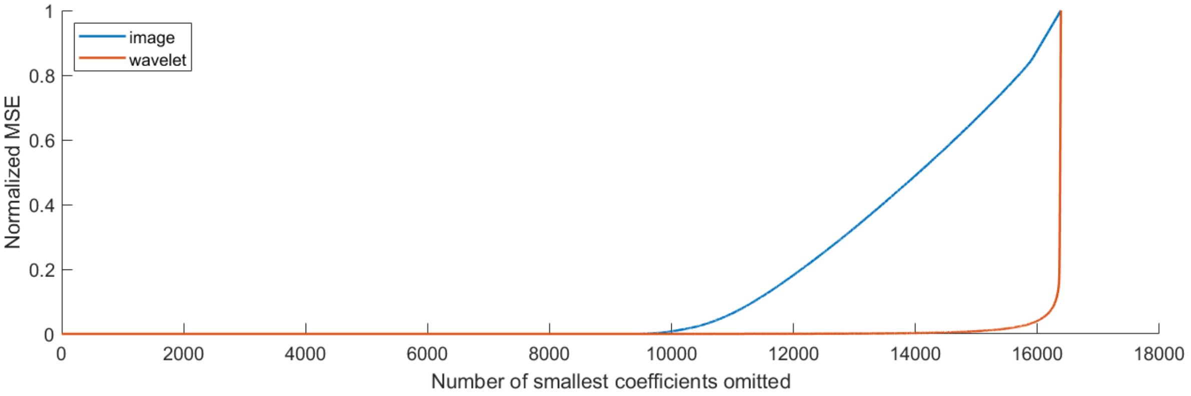

Figure 2.2: Normalized mean squared error with the image in Fig. 2.1.a of the image itself and the wavelet transform shown in Fig. 2.1.c when deleting an increasingly larger number of smallest coefficients. The wavelet transform needs significantly less coefficients (8% for $\mathrm{MSE} = 0.1$) to keep an accurate representation compared to the image itself (31% for $\mathrm{MSE} = 0.1$).

3
Image and k-Space Masking

In this chapter we state a method to undersample k-space in Section 3.1 based on the properties of k-space introduced in the previous chapter. We also give an algorithm based on low-complexity operations to create an image domain mask in Section 3.2.

3.1. Masking in k-Space

Even when confining ourselves to Cartesian k-space sampling, there are many ways to undersample k-space. At the moment, there is no universally best undersampling pattern, but we know for example that most of the information lies in the center of k-space [1]. Therefore, a uniform random distribution is almost certainly not optimal for MRI. Instead, a distribution with variable density focusing on the center of k-space is preferred. Two valid distributions for CS-MRI are the Gaussian distribution and the variable density Poisson disc. The variable density Poisson disc inherently spaces samples out over all of k-space, but is also a slow process and has a more complex implementation when compared to a Gaussian distribution. Therefore, we opt for a Gaussian undersampling mask which samples the phase encoding plane in a Gaussian manner with which the user may choose the acceleration factor, which is proportional to the rate of undersampling the phase encoding lines. Additionally, we enable the user to deterministically choose a proportion of samples which is always sampled in the center of k-space to ensure enough signal energy is captured and heavy aliasing is avoided. This results in a fully sampled circle when the number of phase encoding steps are equal to each other, or an ellipse with equal relative axis lengths to the number of phase encoding steps when they differ.

3.1.1. Gaussian k-Space Masking Algorithm

To make an undersampling mask in k-space for three-dimensional imaging, we can use the bivariate Gaussian probability density function (PDF), which for variables X_1 and X_2 is defined as

$$\begin{aligned} f_{X_1,X_2}(x_1,x_2) =& \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\left\{ -\frac{1}{2(1-\rho^2)} \left[\left(\frac{x_1-\mu_{X_1}}{\sigma_{X_1}}\right)^2 + \left(\frac{x_2-\mu_{X_2}}{\sigma_{X_2}}\right)^2 \right.\right. \\ & \left.\left. -2\rho \left(\frac{x_1-\mu_{X_1}}{\sigma_{X_1}}\right)\left(\frac{x_2-\mu_{X_2}}{\sigma_{X_2}}\right) \right] \right\} , \end{aligned} \tag{3.1}$$

where μ_{X_1} and μ_{X_2} are the respective means of X_1 and X_2, σ_{X_1} and σ_{X_2} are the respective standard deviations of X_1 and X_2 and ρ is the correlation between X_1 and X_2. We want to sample the two phase encoding lines according to the bivariate Gaussian PDF. As we do not want correlation between the samples of the two phase encoding directions, we can set $\rho = 0$. If we define the coordinates of the center of the phase encoding plane to be $(k_{pe1}, k_{pe2}) = (0,0)$, it makes sense to have mean $(\mu_{K_{pe1}}, \mu_{K_{pe2}}) = (0,0)$ as the center contains the most signal. The expression for the PDF reduces to

$$f_{K_{pe1},K_{pe2}}(k_{pe1},k_{pe2}) = \frac{1}{2\pi\sigma_{K_{pe1}}\sigma_{K_{pe2}}} \exp\left\{ -\frac{1}{2} \left[\left(\frac{k_{pe1}}{\sigma_{K_{pe1}}}\right)^2 + \left(\frac{k_{pe2}}{\sigma_{K_{pe2}}}\right)^2 \right] \right\} . \tag{3.2}$$

Due to the fact that low-field systems have poor SNR it may be advantageous to take a percentage c to fully sample the center and sample the remainder of k-space according to a Gaussian distribution. Additionally, fully sampling the center of k-space gives a low resolution approximation of the desired image which may aid in creating an image domain mask. We cannot however sample too much only in the center as this comes at the expense of losing details. The standard deviation in both directions can be defined proportional to the maximum index in their corresponding direction. This essentially leaves two parameters which can be tuned: the percentage of samples in the center and the standard deviation in both phase encoding directions.

When the number of phase encoding steps N_{pe1} and N_{pe2} are chosen, a Gaussian PDF is constructed according to the PDF in Eq. (3.2). Thereafter a threshold is determined using the bisection method and subtracted from the constructed PDF such that the expected number of samples is equal to the desired number of samples. When a mask is created by sampling from this PDF it is not guaranteed that we get exactly the desired number of samples, so we can create new masks until we are satisfied with the accuracy in number of samples and for instance the maximum spacing between samples. The complete sampling procedure is summarized in Algorithm 4.1. See Appendix A for details of the full k-space masking procedure.

An example of a Gaussian k-space undersampling mask for different sampling ratios is shown in Figure 3.1. The standard deviation in both directions is chosen as $\sigma_{pe} = (1-r)N_{pe}$ and the 20% points (points in the phase encoding plane are readout lines) closest to the center are sampled.

Algorithm 1: Gaussian k-space undersampling mask

```
Initialize: $r, c, N_{pe1}, N_{pe2}, \sigma_{K_{pe1}}, \sigma_{K_{pe2}}, x = -2, y = 2$
Create grid $p$ with Gaussian distribution
Until $\lfloor \sum p \rfloor = rN_{pe1}N_{pe2}$
    $z = \frac{x+y}{2}$
    $p = p + z\mathbf{1}_{\{N_{pe1} \times N_{pe2}\}}$
    $p_{i,j} = 0$ if $p_{i,j} < 0$
    $p_{i,j} = 1$ if $p_{i,j} > 1$
    $p_{i,j} = 1$ if $\sqrt{\left(\frac{i}{N_{pe1}}\right)^2 + \left(\frac{j}{N_{pe2}}\right)^2} < c$
    if $\lfloor \sum p \rfloor > rN_{pe1}N_{pe2}$
        $y = z$
    else
        $x = z$
Create mask = $\mathbf{0}_{\{N_{pe1} \times N_{pe2}\}}$
Until $\lfloor rN_{pe1}N_{pe2} - \sum \mathrm{mask} \rfloor < 5$ and $\sum_{i,j} \mathrm{mask}_{5(i-1):5i,5(j-1):5j} > 0$
    Create $N_{pe1} \times N_{pe2}$ grid $u$ with $u_{i,j} \sim U(0,1)$
    mask$_{i,j} = 1$ if $p_{i,j} > u_{i,j}$
```

3.2. Masking in Image Domain

Creating a mask in the image domain reduces the solution space during reconstruction. The location of the object is already accessible from the unprocessed undersampled data as the IFFT of the undersampled data gives a noisy, possibly aliased version of the desired image. Two aspects are important when creating the image domain mask. First, how accurately the object in the image is captured within the mask. It is preferable to outline the mask as close as possible to the actual borders of the object of interest. It is acceptable if a small number of voxels which only contain noise are added to the mask. It is however problematic to ommit any voxels from the mask which actually do contain signal. Another important aspect is the time it takes to perform the masking algorithm. The goal of CS is to accelerate the scanning procedure, but if the image domain masking step takes a significantly long time, it might not be worth it to include this step. Additionally, the reconstruction algorithms discussed in Chap. 4 require an image domain mask for proper functioning.

3.2.1. Noise Estimation and Thresholding

Low-field MR images already suffer from significantly lower SNR with respect to conventional high-field MR images. The SNR is further reduced by undersampling as we acquire less signal compared to a full

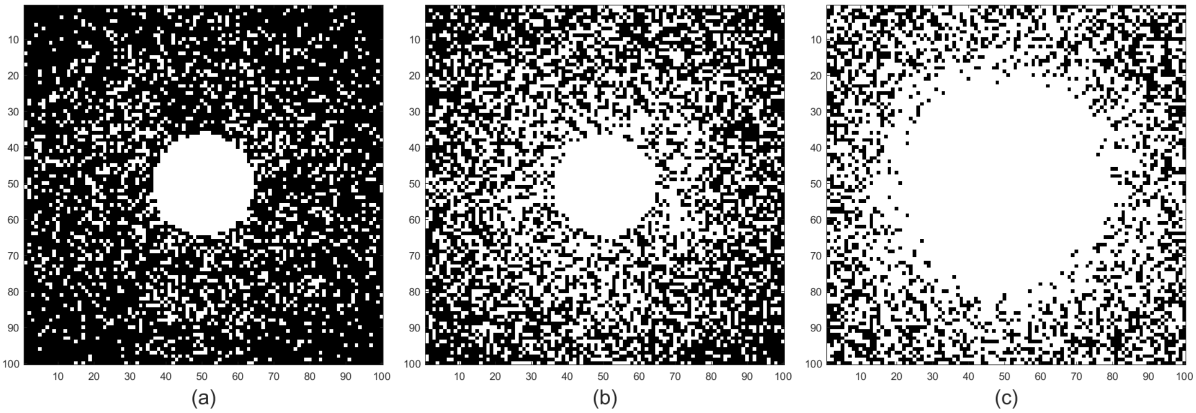

Figure 3.1: Different sampling ratios of a 100x100 phase encoding plane in k-space with 20% of the center fully sampled. (a) 25% sampled. (b) 50% sampled. (c) 75% sampled. The white dots are the readout lines which do get sampled.

scan. We can remove noise by thresholding the image. Hard-tresholding an image $\mathbf{X}$ with threshold value τ is defined as

$$\mathbf{Y} = \mathbf{X} \odot \mathbf{1}_{|\mathbf{X}|>\tau} \ , \tag{3.3}$$

where $\mathbf{X}$ is typically a 2D or 3D image, $\mathbf{Y}$ is the hard-thresholded image, $\odot$ is the elementwise matrix product and the element on position $\{i,j,k\}$ of $\mathbf{1}_{|\mathbf{X}|>\tau}$ is

$$\left(\mathbf{1}_{|\mathbf{X}|>\tau}\right)_{\{i,j,k\}} = \begin{cases} 1 \text{ if } |\mathbf{X}_{\{\mathbf{i},\mathbf{j},\mathbf{k}\}}| > \tau \\ 0 \text{ otherwise} \end{cases} . \tag{3.4}$$

A natural question that follows is what the value of τ should be. Picking τ too small may leave almost all noise in the image, while a too large value may result into destroying too much signal of interest. An example of both cases is shown in Fig. 3.2.

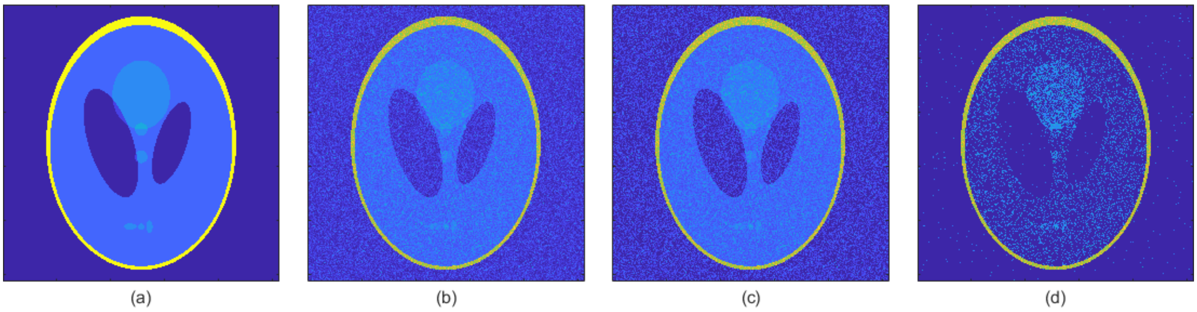

Figure 3.2: Results of thresholding with different threshold levels to remove noise from an image. (a) Noiseless image. (b) Image with additive Gaussian noise. (c) Noisy image thresholded with a too small threshold level. (d) Noisy image thresholded with a too large threshold level.

There are a number of methods to find a threshold level [24]. One method is to estimate the noise level in the image and use a multiple of the noise level as a threshold level. The median absolute deviation (MAD) of wavelet detail coefficients could be used to estimate the noise standard deviation σ_n [25]. First, a DWT is performed on the IFFT of the undersampled k-space data. Then, the median value of the detail coefficients is computed. The MAD is the median of the absolute distance of each detail coefficient value to the median. The threshold level would be $\tau = c\,\sigma_n$ with constant $c > 0$.

3.2.2. Morphological operations

Morphological operations in image processing are pixelwise/voxelwise operations on images using a smaller structure called a structuring element [26]. The structuring element is a predefined shape of at most a few pixels, like a square, diamond, or disk, and the shape, form, and size of objects in the input image are altered based on the structuring element and the specific operation. We shortly introduce the most basic morphological operations on 2D binary images below. See Appendix B for more details on the operations used in this thesis, or [26] for a complete analysis on morphological operations.

The four most basic operations are dilation, erosion, closing, and opening. With dilation and erosion we respectively expand or shrink an object in the input image. This is seen in Fig 3.3, where the input image is shown in Fig 3.3.a, a structuring element is shown in Fig 3.3.b and the output after dilation and erosion are shown in Fig 3.3.c and 3.3.d, respectively. The closing operation can be used to fill small holes and notches in objects as shown in Fig 3.3.e and the opening operation can be used to open up any small holes and notches, as shown in Fig 3.3. Lastly, a flood-fill operation can be performed using morphological operations to fill holes in an image [26]. For a binary image, this operation simply seeks and fills all areas of zero-valued voxels which are fully enclosed by non-zero voxels.

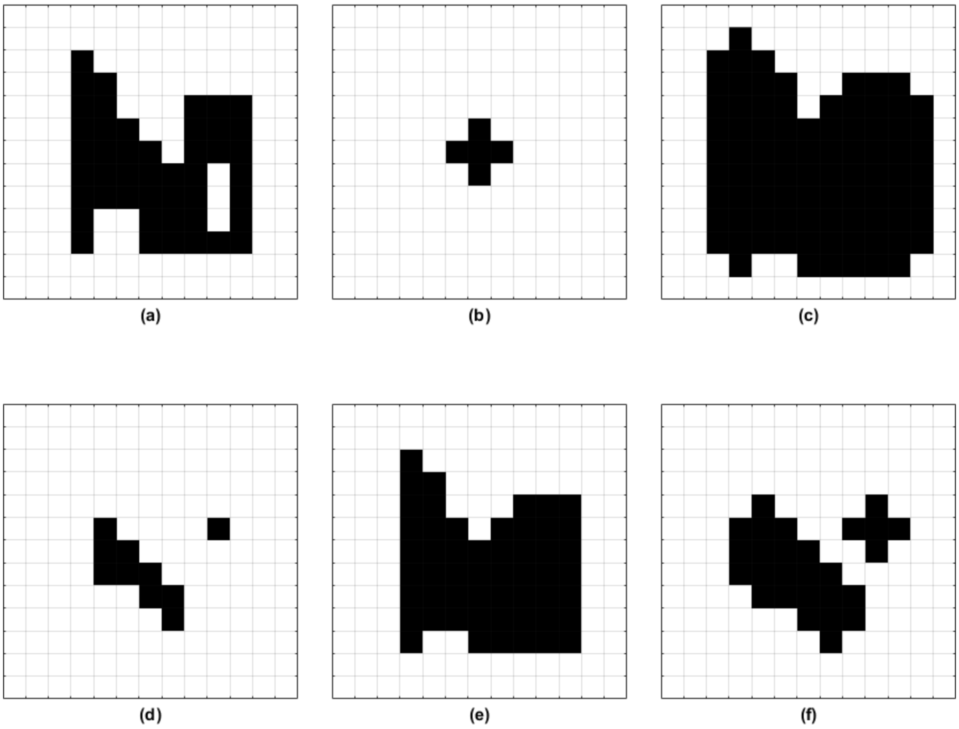

Figure 3.3: Morphological operations on an input image with a diamond shaped structuring element. (a) input image, (b) structuring element, (c) dilated output image, (d) eroded output image, (e) closed output image, (f) opened output image.

3.2.3. Image Domain Masking Algorithm

The image domain masking procedure consists of the following steps:

1. Estimate noise
2. Hard-threshold smoothed image
3. Morphological operations

The first step is to estimate the noise standard deviation using the MAD method on the detail coefficients of the DWT of the input image. Taking x as input image and denoting $\mathbf{W}$ as the DWT operator, the wavelet coefficients of the image are $\mathrm{w} = \mathbf{W}\mathrm{x}$. We can express the detail coefficients of the wavelet transform as $\mathrm{w}_{\text{detail}}$. The MAD is then defined as

$$\mathrm{MAD}(\mathbf{w}_{\text{detail}}) = \mathrm{median}(|\mathbf{w}_{\text{detail}} - \mathrm{median}(|\mathbf{w}_{\text{detail}}|) \cdot \mathbf{1}|) \ . \tag{3.5}$$

We set the noise standard deviation as $\sigma_n = \mathrm{MAD}(\mathbf{w}_{\text{detail}})$ and the threshold level for the hard-thresholding step as $\tau = c\,\sigma_n$. Before we threshold, we can first smooth the image. In this way, higher valued pixels outside of the object of interest are smoothed out such that there is a higher probability that these pixels fall below the threshold level. In addition, any holes within the object of interest get a nonzero value and have a higher probability to not get thresholded away. Denoting $\mathcal{K}$ as a kernel of choice, i.e. a Gaussian kernel, the smoothed image is the convolution between the image and the kernel, that is $\mathcal{K} * \mathbf{x}$. The hard-thresholded smoothed image is then $\mathbf{x}_\tau = (\mathcal{K} * \mathbf{x}) \odot \mathbf{1}_{\{|\mathcal{K}*\mathbf{x}|>\tau=c\,\sigma_n\}}$ and the corresponding binary mask is $\mathbf{m} = \mathbf{1}_{|\mathbf{x}_\tau|>0}$. The mask $\mathbf{m}$ resulting from the hard-thresholding step is further improved by morphological operations. We use a sequence of different morphological operations, which we will denote as $\mathrm{MORPH}(\cdot)$. The details on the exact sequence are discussed in Appendix B. The image domain masking algorithm is shown in Algorithm 2.

```
Algorithm 2: Image domain masking
Initialize: $\mathbf{x} = \mathbf{F}^H\mathbf{b}, \mathbf{w} = \mathbf{W}\mathbf{x}, \mathcal{K}, c, \mathbf{y}_1, \mathbf{y}_2, t$
$\sigma_n = \mathrm{MAD}(\mathbf{w}_{\text{detail}})$
$\mathbf{x}_\tau = (\mathcal{K} * \mathbf{x}) \odot \mathbf{1}_{\{|\mathcal{K}*\mathbf{x}|>\tau=c\,\sigma_n\}}$
$\mathbf{m} = \mathrm{MORPH}(\mathbf{1}_{|\mathbf{x}_\tau|>0})$
```

4

The Compressed Sensing Problem

This chapter shows how compressed sensing can be formulated mathematically in Section 4.1, creating a convex optimization problem. In Sections 4.1.1-4.1.3 cost functions different from the state-of-the-art are introduced and analyzed which have fewer or even no free regularization parameters. Different reconstruction algorithms are given in Section 4.2 as well as a method to select the regularization parameter if relevant. Additionally, in Section 4.3 we find stopping criteria for the reconstruction algorithms based on the convergence behaviour of the functions.

4.1. Convex Optimization Problem Formulation

To reiterate, we want a solution which is sparse, either in itself or when transformed to another mathematical domain, and remains close to the measured data. Let us denote the vectorized image x, the vectorized data b and the Fourier transform $\mathbf{F}$, such that $\mathbf{F}\mathrm{x}$ is the Fourier transform of x and $\mathbf{F}^H$ indicates the inverse Fourier transform. To indicate which coefficients get sampled in k-space, we can use the diagonal matrix $\mathbf{S}_k$ which has ones on the diagonal positions corresponding to a sampled coefficient and zeros elsewhere. We can then write the undersampled image as $\mathbf{S}_k\mathbf{F}\mathrm{x}$. We can also use a diagonal matrix $\mathbf{S}_x$ to indicate which voxels of x are within the image domain mask, such that $\mathbf{S}_x\mathrm{x}$ is the masked image.

To enforce sparsity we can use the l_1-norm as a regularization function, which is defined as

$$\|\mathbf{x}\|_1 = \sum_i |x_i| \ . \tag{4.1}$$

If we want a solution which is sparse in both TV and wavelet domain, we can solve the minimization problem

$$(\mathrm{P1}) \quad \min_{\mathbf{x}} \ \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda_1 \|\nabla\mathbf{x}\|_1 + \lambda_2 \|\mathbf{W}\mathbf{x}\|_1 \ . \tag{4.2}$$

where $\mathbf{A} = \mathbf{S}_k\mathbf{F}\mathbf{S}_x$, $\mathbf{W}$ denotes the discrete wavelet transform and λ_1 and λ_2 are regularization parameters which balance respectively the TV term and the wavelet term with the other terms. A number of fast iterative algorithms exist which can efficiently minimize Eq. (4.2), even though the l_1-norm is non-differentiable, but the problem in practice is that it is not known a priori what the optimal values for λ_1 and λ_2 are as they are problem specific. As a consequence, either fixed (sub-optimal) values need to be used, or the algorithm has to be run for a lot of possible combinations of λ_1 and λ_2 values which greatly increases reconstruction time up to the point that it might not even be worth it to use CS.

4.1.1. Multiplicative TV Regularization

To partly mitigate the problem of finding optimal regularization parameters, one can replace the additive TV regularization term by a multiplicative TV regularization term, as has already been done for denoising and deblurring applications in [16, 17, 18], where only a single additive TV regularization term was included in the cost function. Multiplying the data consistency term by the new multiplicative TV regularization term, gives us the general minimization problem

$$\min_{\mathbf{x}} \ F^{\mathrm{data}}(\mathbf{x})F^{\mathrm{TV}}(\mathbf{x}) + \lambda F^{\mathrm{wavelet}}(\mathbf{x}) \ , \tag{4.3}$$

where we only need to tune 1 regularization parameter. To make use of a number of suitable reconstruction algorithms it is desirable that the $F^{\text{data}}(\mathbf{x})F^{\text{TV}}(\mathbf{x})$ term is convex and differentiable. In [18] a weighted l_2-norm TV regularization term with a normalized data consistency term is used to overcome the non-differentiability of the l_1-norm function. The l_2-norm in itself is not a sparse regularizer, but a weighted l_2-norm can mimic the behaviour of the l_1-norm. In e.g. [18] the following weighted l_2-norm TV regularization term is used

$$F^{\text{TV}}_{k-1}(X) = \int \frac{|\nabla X|^2 + \delta^2_{\text{TV},k-1}}{|\nabla X_{k-1}|^2 + \delta^2_{\text{TV},k-1}} dV \ , \tag{4.4}$$

where X is the continuous counterpart of $\mathbf{x}$, k is the iteration number and $\delta^2_{\text{TV},k-1}$ is a purely data-dependent parameter such as

$$\delta^2_{\text{TV},k-1} = F^{\text{data}}(\mathbf{x}_{k-1})^2 \int |\nabla X_{k-1}|^2 dV. \tag{4.5}$$

Thus, at each iteration the TV l_2-norm is weighted by the TV l_2-norm of the previous iteration. Note that $F^{\text{TV}}_{k-1}(X_{k-1}) = 1$. Weighting by the TV of the previous iteration automatically incorporates information of the location of edges in the image. Assuming that during the reconstruction algorithm at each iteration the image quality improves slightly, the edge locations are more easily distinguished and the weighting gets better. The definitions in Eq. (4.4) and (4.5) are continuous, so following the derivations in [18] we get the following discretization.

We can first introduce a weighting parameter

$$w_{k-1} = \frac{1}{|\nabla X_{k-1}|^2 + \delta^2_{\text{TV},k-1}} \ , \tag{4.6}$$

such that Eq. (4.4) can be written as

$$F^{\text{TV}}_{k-1}(X) = \delta^2_{k-1} \int w_{k-1} dV + \int w_{k-1} |\nabla X|^2 dV. \tag{4.7}$$

The continuous gradients are discretized in the x-, y-, and z-direction, using finite differences where the forward and backward differencing matrices are respectively defined as

$$\mathbf{D}_{\zeta;f} = \frac{1}{\Delta\zeta} \begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \\ & & & & -1 \end{bmatrix} \tag{4.8}$$

and

$$\mathbf{D}_{\zeta;b} = \frac{1}{\Delta\zeta} \begin{bmatrix} 1 & & & & \\ -1 & 1 & & & \\ & \ddots & \ddots & & \\ & & -1 & 1 & \\ & & & -1 & 1 \end{bmatrix} , \tag{4.9}$$

with $\zeta = \{x, y, z\}$. Using $\mathcal{W}_{\text{TV},k-1} = \text{diag}(\mathbf{w}_{k-1})$ we can write Eq. (4.7) as

$$F^{\text{TV}}_{k-1}(X) \approx \delta^2_{\text{TV},k-1} V_\Delta \mathbf{e}^T \mathcal{W}_{\text{TV},k-1} \mathbf{e} + V_\Delta \sum_{i=x,y,z} \mathbf{g}^H_{\text{TV},i} \mathcal{W}_{\text{TV},k-1} \mathbf{g}_{\text{TV},i}, \tag{4.10}$$

where V_Δ is the voxel volume, $\mathbf{e}$ is the all-ones vector and $g_{\text{TV},i}$ is the discretized gradient defined as

$$\begin{aligned} \mathbf{g}_{\text{TV},x} &= \left(\mathbf{I}_R \otimes \mathbf{I}_Q \otimes \mathbf{D}_x\right)\mathbf{x} = \mathcal{D}_x \mathbf{x}, \\ \mathbf{g}_{\text{TV},y} &= \left(\mathbf{I}_R \otimes \mathbf{D}_y \otimes \mathbf{I}_P\right)\mathbf{x} = \mathcal{D}_y \mathbf{x}, \\ \mathbf{g}_{\text{TV},z} &= \left(\mathbf{D}_z \otimes \mathbf{I}_Q \otimes \mathbf{I}_P\right)\mathbf{x} = \mathcal{D}_z \mathbf{x}, \end{aligned} \tag{4.11}$$

where $\mathbf{I}_i$ is the order i identity matrix, $\otimes$ is the Kronecker product, and $\mathbf{D}_i$ may be either the forward or backward differencing matrix. Introducing the weighted approximate Laplacian given by

$$\begin{aligned}\mathbf{L}_\mathbf{w} &= \left(\mathbf{I}_R \otimes \mathbf{I}_Q \otimes \mathbf{D}_x^T\right) \mathcal{W}_{k-1} \left(\mathbf{I}_R \otimes \mathbf{I}_Q \otimes \mathbf{D}_x\right) \\ &+ \left(\mathbf{I}_R \otimes \mathbf{D}_y^T \otimes \mathbf{I}_P\right) \mathcal{W}_{k-1} \left(\mathbf{I}_R \otimes \mathbf{D}_y \otimes \mathbf{I}_P\right) \\ &+ \left(\mathbf{D}_z^T \otimes \mathbf{I}_Q \otimes \mathbf{I}_P\right) \mathcal{W}_{k-1} \left(\mathbf{D}_z \otimes \mathbf{I}_Q \otimes \mathbf{I}_P\right) \ , \\ &= \mathcal{D}_x^T \mathcal{W}_{\mathrm{TV},k-1} \mathcal{D}_x + \mathcal{D}_y^T \mathcal{W}_{\mathrm{TV},k-1} \mathcal{D}_y + \mathcal{D}_z^T \mathcal{W}_{\mathrm{TV},k-1} \mathcal{D}_z \ ,\end{aligned} \tag{4.12}$$

we get the compact expression

$$F_{k-1}^{\mathrm{TV}}(X) \approx F_{k-1}^{\mathrm{TV}}(\mathbf{x}) = \delta_{\mathrm{TV},k-1}^2 V_\Delta \mathbf{e}^T \mathcal{W}_{\mathrm{TV},k-1} \mathbf{e} + V_\Delta \mathbf{x}^H \mathbf{L}_\mathbf{w} \mathbf{x}. \tag{4.13}$$

Additionally, a mixed finite difference Laplacian may be defined which uses both forward and backward differencing matrices and leads to faster convergence. The mixed finite difference Laplacian is defined as

$$\mathbf{L}_{\mathbf{w},\mathrm{mixed}} = \frac{1}{2} \left(\mathbf{L}_\mathbf{w}|_{\mathbf{D}=\mathbf{D}_f} + \mathbf{L}_\mathbf{w}|_{\mathbf{D}=\mathbf{D}_b} \right). \tag{4.14}$$

The resulting minimization problem then becomes

$$(\mathrm{P2}) \quad \min_{\mathbf{x}} \ \frac{\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{b}\|_2^2} \left(\delta_{\mathrm{TV},k-1}^2 V_\Delta \mathbf{e}^T \mathcal{W}_{\mathrm{TV},k-1} \mathbf{e} + V_\Delta \mathbf{x}^H \mathbf{L}_\mathbf{w} \mathbf{x} \right) + \lambda \|\mathbf{W}\mathbf{x}\|_1 \ , \tag{4.15}$$

where from now on we define the data functional as

$$F^{\mathrm{data}}(\mathbf{x}) = \frac{\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{b}\|_2^2} \ , \tag{4.16}$$

which is the energy of the data mismatch normalized by the energy of the data.

4.1.2. Double Multiplicative Regularization

The minimization problem in Eq. (4.15) obviously still contains 1 tunable regularization parameter for the wavelet regularization term. Therefore, we remove the additive wavelet regularization term and introduce a multiplicative wavelet regularization term. Incorporating the wavelet regularization in a multiplicative manner gives the general minimization problem

$$\min_{\mathbf{x}} \ F^{\mathrm{data}}(\mathbf{x}) F^{\mathrm{TV}}(\mathbf{x}) F^{\mathrm{wavelet}}(\mathbf{x}) \ , \tag{4.17}$$

which does not contain any free regularization parameters at all. The multiplicative wavelet regularization term is implemented in the same way as the TV term, namely a weighted l_2-norm with a fully data-dependent regularization parameter. We can define the continuous weighted l_2-norm wavelet regularizer as

$$F_{k-1}^{\mathrm{wavelet}}(X) = \int \frac{|\Psi X|^2 + \delta_{\mathrm{w},k-1}^2}{|\Psi X_{k-1}|^2 + \delta_{\mathrm{w},k-1}^2} dV \ , \tag{4.18}$$

where we use Ψ to denote the continuous wavelet transform and the regularization parameter is now defined as

$$\delta_{\mathrm{w},k-1}^2 = F^{\mathrm{data}}(\mathbf{x}_{k-1})^2 \int |\Psi X_{k-1}|^2 dV \ . \tag{4.19}$$

We need to discretize the regularizer for which we follow similar steps as with the TV regularizer. We can define the weighting parameter as

$$w_{k-1} = \frac{1}{|\Psi X|^2 + \delta_{\mathrm{w},k-1}^2} \ , \tag{4.20}$$

and write the regularizer as

$$F_{k-1}^{\mathrm{wavelet}}(X) = \delta_{\mathrm{w},k-1}^2 \int w_{k-1} dV + \int w_{k-1} |\Psi X_{k-1}|^2 dV \ . \tag{4.21}$$

We now substitute the continuous wavelet transform Ψ by the discrete wavelet transform $\mathbf{W}$ and write the discretized function as

$$F_{k-1}^{\text{wavelet}}(X) \approx F_{k-1}^{\text{wavelet}}(\mathbf{x}) = \delta_{\text{w},k-1}^2 V_\Delta \mathbf{e}^T \mathcal{W}_{k-1}\mathbf{e} + V_\Delta \mathbf{x}^H \mathbf{W}^H \mathcal{W}_{\text{w},k-1}\mathbf{W}\mathbf{x} \ , \tag{4.22}$$

where $\mathbf{W}^H$ is the inverse discrete wavelet transform. The new optimization problem then becomes

$$\begin{aligned}(\text{P3}) \quad \min_{\mathbf{x}} \ & \frac{\|\mathbf{b}-\mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{b}\|_2^2} \left(\delta_{\text{TV},k-1}^2 V_\Delta \mathbf{e}^T \mathcal{W}_{\text{TV},k-1}\mathbf{e} + V_\Delta \mathbf{x}^H \mathbf{L}_\mathbf{w}\mathbf{x}\right) \left(\delta_{\text{w},k-1}^2 V_\Delta \mathbf{e}^T \mathcal{W}_{\text{w},k-1}\mathbf{e}\right. \\ & \left. + V_\Delta \mathbf{x}^H \mathbf{W}^H \mathcal{W}_{\text{w},k-1}\mathbf{W}\mathbf{x}\right) \ .\end{aligned} \tag{4.23}$$

4.1.3. Incorporating Directional TV

The abovementioned weighted l_2-norm implementation of TV uses information on the location of edges in the image, as the weights are essentially the edge locations of the previous iteration. Instead of merely using information on the edge locations, we can also use the information of edge directions. The underlying assumption is that the direction of edges in the image locally are approximately equal. Many implementations of dTV use the structure tensor to incorporate the directional information [27, 28]. These methods are based on computing eigenvalue decompositions at each iteration, which are computationally expensive. A method of incorporating directional information which is less computationally heavy, is proposed in [29]. The directionality is implemented by projecting the gradient onto the orthogonal complement of the gradient field of a similar image. This is useful when taking multiple MRI scans of the same patient where the gradient information of the first scan can be used during reconstruction of any subsequent scans. In our case, we do not have prior scan information, but we can use the directional information at one iteration for the computation of the next iteration similar to the edge location weights.

The implementation of dTV is as follows. The projection matrix defined for a unit vector $\mathbf{u}$ is

$$\mathbf{P}_\mathbf{u} = \mathbf{u}\mathbf{u}^H \ . \tag{4.24}$$

So if we have a vector $\mathbf{v}$ which consists of a component $\mathbf{v}_\parallel$ parallel to $\mathbf{u}$ and a component $\mathbf{v}_\perp$ perpendicular to $\mathbf{u}$, then

$$\mathbf{P}_\mathbf{u}\mathbf{v} = \mathbf{P}_\mathbf{u}(\mathbf{v}_\parallel + \mathbf{v}_\perp) = \mathbf{v}_\parallel \ . \tag{4.25}$$

From this it follows that the projection onto the orthogonal complement of $\mathbf{u}$ is given by $\mathbf{I} - \mathbf{P}_\mathbf{u}$, where $\mathbf{I}$ is the identity matrix, because

$$(\mathbf{I} - \mathbf{P}_\mathbf{u})\mathbf{v} = (\mathbf{v}_\parallel + \mathbf{v}_\perp) - \mathbf{P}_\mathbf{u}(\mathbf{v}_\parallel + \mathbf{v}_\perp) = (\mathbf{v}_\parallel + \mathbf{v}_\perp) - \mathbf{v}_\parallel = \mathbf{v}_\perp \ . \tag{4.26}$$

Now let us define $\mathbf{P}_\mathbf{u}^\perp = \mathbf{I} - \mathbf{P}_\mathbf{u}$ and the vector operation $|\cdot|_\eta$ defined elementwise as

$$|\mathbf{x}|_\eta = \begin{bmatrix} \dfrac{x_1}{\sqrt{x_1^* x_1 + \eta^2}} & \dfrac{x_2}{\sqrt{x_2^* x_2 + \eta^2}} & \cdots & \dfrac{x_2}{\sqrt{x_2^* x_2 + \eta^2}} \end{bmatrix}^T \ , \tag{4.27}$$

for a real scalar $\eta > 0$. We can apply the projection onto orthogonal complement and the newly defined vector function $|\cdot|_\eta$ to the gradient of the image. Instead of a weighted l_2-norm implementation of the gradient we get a weighted l_2-norm implementation of the gradient projected on the orthogonal complement of the gradient at the previous iteration. If we replace $\mathbf{u}$ and $\mathbf{v}$ by $\xi = |\nabla\mathbf{x}_{k-1}|_\eta$ and $\nabla\mathbf{x}_k$, respectively, we get

$$\begin{aligned}\mathbf{P}_\xi^\perp \nabla\mathbf{x}_k &= \left(\mathbf{I} - \xi\xi^H\right)\nabla\mathbf{x}_k \\ &= \left(\mathbf{I} - |\nabla\mathbf{x}_{k-1}|_\eta |\nabla\mathbf{x}_{k-1}|_\eta^H\right)\nabla\mathbf{x}_k \\ &= \nabla\mathbf{x}_k - \langle|\nabla\mathbf{x}_{k-1}|_\eta, \nabla\mathbf{x}_k\rangle |\nabla\mathbf{x}_{k-1}|_\eta \ .\end{aligned} \tag{4.28}$$

For very small values of η the above expression approximately gives the projection of $\nabla\mathbf{x}_k$ onto the orthogonal complement of $\nabla\mathbf{x}_{k-1}$, illustrated in Fig. 4.1. The conventional additively regularized formulation uses the l_1-norm to induce sparsity in number edges and Sec. 4.1.1 and 4.1.2 discussed the weighted l_2-norm, weighting the gradient voxelwise by the gradient of the previous iteration, to

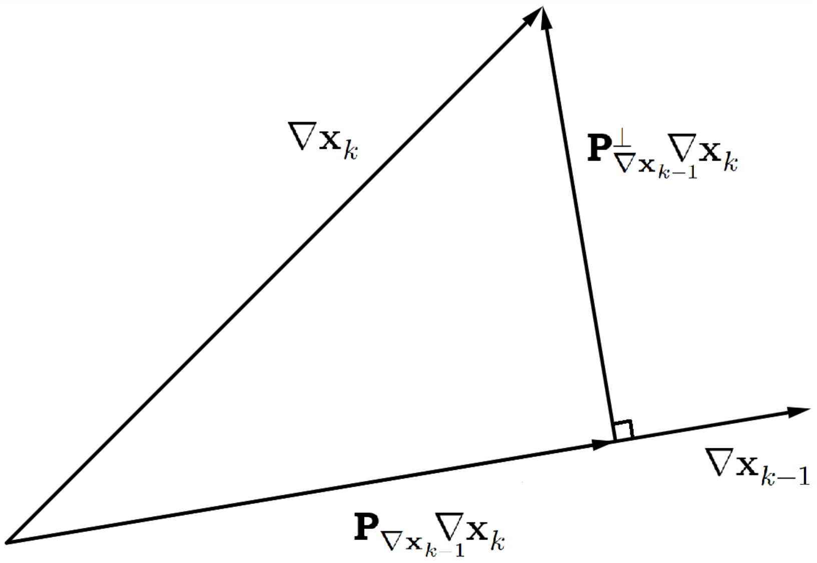
$\nabla \mathbf{x}_k$
$\mathbf{P}^{\perp}_{\nabla \mathbf{x}_{k-1}} \nabla \mathbf{x}_k$
$\nabla \mathbf{x}_{k-1}$
$\mathbf{P}_{\nabla \mathbf{x}_{k-1}} \nabla \mathbf{x}_k$

Figure 4.1: Parallel and orthogonal projection of $\nabla \mathbf{x}_k$ onto $\nabla \mathbf{x}_{k-1}$.

also induce sparsity in number of edges. Our implementation of incorporating dTV weights $\mathbf{P}^{\perp}_{\nabla \mathbf{x}_{k-1}} \nabla \mathbf{x}_k$ voxelwise by $\mathbf{P}^{\perp}_{\nabla \mathbf{x}_{k-2}} \nabla \mathbf{x}_{k-1}$. Hereby not only the number of edges is sparse, but the local changes in direction of the gradients are sparse.

To derive the multiplicative directional TV regularization functional from the multiplicative TV regularization functional, we can start from the discretized gradient expressions in Eq. (4.11). If we set $\xi = |\mathbf{g}_{\mathrm{TV},k-1}|_\eta$, the relation between the dTV and TV discretized gradient at the k'th iteration can be defined as

$$\mathbf{g}_{\mathrm{dTV}} = \mathbf{P}^{\perp}_{\xi} \mathbf{g}_{\mathrm{TV}} = \mathbf{g}_{\mathrm{TV}} - \langle |\mathbf{g}_{\mathrm{TV},k-1}|_\eta, \mathbf{g}_{\mathrm{TV}} \rangle |\mathbf{g}_{\mathrm{TV},k-1}|_\eta \, . \tag{4.29}$$

Now introducing $\delta^2_{\mathrm{dTV},k-1}$ as

$$\delta^2_{\mathrm{dTV},k-1} = F^{\mathrm{data}}(\mathbf{x}_{k-1})^2 \sum_{i=x,y,z} \mathbf{x}^H \mathcal{D}_i^T \mathbf{P}^{\perp}_{|\mathbf{g}_{\mathrm{TV},k-1}|_\eta} \mathcal{D}_i \mathbf{x} \, , \tag{4.30}$$

and the diagonal matrix $\mathcal{W}_{\mathrm{dTV},k-1}$ as

$$\mathcal{W}_{\mathrm{dTV},k-1} = \mathrm{diag}(\mathbf{w}_{k-1}) = \mathrm{diag}\left(|\mathbf{g}_{\mathrm{dTV}}|^2 + \delta^2_{\mathrm{dTV},k-1} \right)^{-1} \, , \tag{4.31}$$

we get an expression for the directional TV functional

$$F^{\mathrm{dTV}}_{k-1}(\mathbf{x}) = \delta^2_{\mathrm{dTV},k-1} V_\Delta \mathbf{e}^T \mathcal{W}_{k-1} \mathbf{e} + V_\Delta \sum_{i=x,y,z} \mathbf{g}^H_{\mathrm{dTV},i} \mathcal{W}_{\mathrm{dTV},k-1} \mathbf{g}_{\mathrm{dTV},i} \, . \tag{4.32}$$

To keep notation compact again we can introduce a modified approximate Laplacian $\mathbf{L}_{\mathrm{dTV,w}}$ as

$$\mathbf{L}_{\mathrm{dTV},\mathbf{w}} = \sum_{i=x,y,z} \mathcal{D}_i^T \mathbf{P}^{\perp}_{|\mathbf{g}_{\mathrm{TV},k-1}|_\eta} \mathcal{W}_{\mathrm{dTV},k-1} \mathbf{P}^{\perp}_{|\mathbf{g}_{\mathrm{TV},k-1}|_\eta} \mathcal{D}_i \, , \tag{4.33}$$

resulting in the final expression of the minimization problem including directional TV regularization

$$\begin{aligned} (\mathrm{P4}) \quad \min_{\mathbf{x}} \; & \frac{\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{b}\|_2^2} \left(\delta^2_{\mathrm{dTV},k-1} V_\Delta \mathbf{e}^T \mathcal{W}_{\mathrm{dTV},k-1} \mathbf{e} + V_\Delta \mathbf{x}^H \mathbf{L}_{\mathrm{dTV,w}} \mathbf{x} \right) \left(\delta^2_{\mathrm{w},k-1} V_\Delta \mathbf{e}^T \mathcal{W}_{\mathrm{w},k-1} \mathbf{e} \right. \\ & \left. + V_\Delta \mathbf{x}^H \mathbf{W}^H \mathcal{W}_{\mathrm{w},k-1} \mathbf{W} \mathbf{x} \right) \, . \end{aligned} \tag{4.34}$$

4.2. Reconstruction Algorithms

As mentioned before, a number of fast iterative algorithms exist which can efficiently minimize problem (P1). Examples are the alternating direction method of multipliers (ADMM) [30] and the split-Bregman algorithm [31]. These algorithms make use of shrinkage operations to deal with the non-differentiability of the l_1-norm, which are computationally cheap resulting in fast iterations. Even though these algorithms are very efficient, they do introduce additional free regularization parameters and finding the optimal values for the regularization parameters is a cumbersome and time-consuming process. Instead we will focus on algorithms which introduce less or even no additional free parameters for problems (P2), (P3) and (P4), thereby minimizing the need for manual tuning of the reconstruction process.

4.2.1. FISTA

For the problem formulation (P2), which uses a multiplicative TV and an additive wavelet regularization term, we can make use of the fast iterative shrinkage thresholding algorithm (FISTA) [32]. FISTA is a specialized algorithm for functions of the general form

$$F(\mathbf{x}) = F_1(\mathbf{x}) + F_2(\mathbf{x}) \,, \tag{4.35}$$

where $F_1(\mathbf{x})$ is a differentiable convex function with a Lipschitz continuous gradient $g_1(\mathbf{x})$ and $F_2(\mathbf{x})$ is a convex function which is possibly non-differentiable. The function of (P2) is of this form if we define

$$\begin{aligned} F_1(\mathbf{x}) &= \frac{\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{b}\|_2^2} \left(\delta^2_{\mathrm{TV},k-1} V_\Delta \mathbf{e}^T \mathcal{W}_{\mathrm{TV},k-1} \mathbf{e} + V_\Delta \mathbf{x}^H \mathbf{L}_\mathbf{w} \mathbf{x} \right) \,, \\ F_2(\mathbf{x}) &= \lambda \|\mathbf{W}\mathbf{x}\|_1 \,, \end{aligned} \tag{4.36}$$

where the gradient of $F_1(\mathbf{x})$ is

$$\nabla F_1(\mathbf{x}) := g_1(\mathbf{x}) = 2 \left[- \|\mathbf{b}\|_2^{-2} \mathbf{A}^H \mathbf{r}_{k-1} + V_\Delta F^{\mathrm{data}}(\mathbf{x}) \mathbf{L}_\mathbf{w} \mathbf{x} \right] \,, \tag{4.37}$$

where $\mathbf{r}_{k-1} = \mathbf{b} - \mathbf{A}^H \mathbf{x}_{k-1}$. The idea of FISTA is as follows: each iteration of the algorithm a quadratic approximation of the cost function is computed and minimized. This quadratic approximation is minimized instead of the original cost function as it has a unique minimizer, which is easily computed. The next iteration is then computed as a linear combination of the past two iterations such that faster convergence is achieved. For more details on the derivations see Appendix C. The quadratic approximation $Q_L(\mathbf{x}, \mathbf{y})$ can be written as

$$Q_L(\mathbf{x}, \mathbf{y}) = F_1(\mathbf{y}) + (\mathbf{x} - \mathbf{y})^H g_1(\mathbf{y}) + \frac{L}{2} \|\mathbf{x} - \mathbf{y}\|_2^2 + F_2(\mathbf{x})) \,, \tag{4.38}$$

where L is the Lipschitz constant of g_1. The minimizer of $Q_L(\mathbf{x}, \mathbf{y})$ is the proximal operator $p_L(\mathbf{y})$ defined as

$$\begin{aligned} p_L(\mathbf{y}) &= \arg\min_{\mathbf{x}} \{Q_L(\mathbf{x}, \mathbf{y})\} \\ &= \mathbf{W}^H \mathcal{T}_{\lambda/L} \left(\mathbf{W}(\mathbf{y} - \frac{1}{L} g_1(\mathbf{y})) \right) \,, \end{aligned} \tag{4.39}$$

where we make use of the shrinkage operator defined as

$$\mathcal{T}_a(\mathbf{x})_i = (|x_i| - a)_+ \mathrm{sgn}(x_i) \,, \tag{4.40}$$

where a is a real constant, $(\cdot)_+ = \max\{0, \cdot\}$ and $\mathrm{sgn}(\cdot)$ is the sign function [12], which can be defined for both real and complex numbers as

$$\mathrm{sgn}(z) = \begin{cases} \dfrac{z}{|z|} \text{ for } z \neq 0 \\ 0 \text{ for } z = 0 \end{cases} . \tag{4.41}$$

So, each iteration k we update $\mathbf{x}_k = p_L(\mathbf{y}_k)$. Image domain support could also be incorporated in the update equation of $\mathbf{x}_k$ if available. This results in

$$\mathbf{x}_k = \mathbf{S}_\mathbf{x} p_L(\mathbf{y}_k) = \mathbf{S}_\mathbf{x} \mathbf{W}^H \mathcal{T}_{\lambda/L} \left(\mathbf{W}(\mathbf{y}_k - \frac{2}{L} \left[- \|\mathbf{b}\|_2^{-2} \mathbf{A}^H \mathbf{r}_{k-1} + V_\Delta F^{\mathrm{data}}(\mathbf{y}_k) \mathbf{L}_\mathbf{w} \mathbf{y}_k \right]) \right) . \tag{4.42}$$

Thereafter, $\mathbf{y}_k$ gets updated as a linear combination of $\mathbf{x}_k$ and $\mathbf{x}_{k-1}$ as

$$\mathbf{y}_{k+1} = \mathbf{x}_k + \frac{t_k - 1}{t_{k+1}} (\mathbf{x}_k - \mathbf{x}_{k-1}) \, , \tag{4.43}$$

where t_{k+1} gets updated as

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2} \, , \tag{4.44}$$

with $t_1 = 1$.

The drawback of FISTA is that it is not easy to compute the optimal Lipschitz constant L. Although the optimal L can be overestimated by a very large margin, it would slow down the convergence of the algorithm significantly. For this reason, we use a modified form of FISTA with a backtracking stepsize rule instead [32].

The last issue of this method is the fact that we still have to choose λ. A number of post-processing methods to compute an optimal value for a regularization parameter are available, e.g. the L-curve method [33], Morozov's discrepancy principle [34], and the generalized cross-validation method [35]. The problem with these methods is that the user needs to run the algorithm a large number of times for different values of λ to find an optimal value. Alternatively, λ can be computed adaptively each iteration. We will compute λ adaptively based on Stein's unbiased risk estimate (SURE) [36]. Each iteration, λ_k is the argument which minimizes the optimization problem

$$\lambda_k = \arg\min_{\lambda} \; N - 2 \cdot \sum_i \mathbf{1}_{|(\tilde{\mathbf{W}}\mathbf{x})_i| \leq \lambda} + \sum_i (\min\{|(\tilde{\mathbf{W}}\mathbf{x})_i|, \lambda\})^2 \, , \tag{4.45}$$

where N is the number of voxels in $\mathbf{x}$ and $\tilde{\mathbf{W}}\mathbf{x}$ is the discrete wavelet transform of $\mathbf{x}$ normalized to standard deviation 1. As all operations in the above optimization problem are cheap to compute, this step will not introduce significant extra computational time. Moreover, this method does not introduce any extra free parameters.

Adaptive FISTA with backtracking is summarized in Reconstruction Algorithm 1.

```
Reconstruction Algorithm 1: adaptive FISTA with backtracking for (P2)
Initialize: $\mathrm{x}_0 = \mathbf{A}^H \mathrm{b} = \mathbf{S}_\mathbf{x} \mathbf{F}^H \mathbf{S}_\mathbf{k} \mathrm{b}$, $\mathbf{y}_1 = \mathrm{x}_0, t_1 = 1, L_0 > 0, \eta > 1$
Until convergence or $k > k_{max}$
  $\lambda_k = \arg\min_{\lambda} \; N - 2 \cdot \sum_i \mathbf{1}_{|(\tilde{\mathbf{W}}\mathbf{x})_i| \leq \lambda} + \sum_i (\min\{|(\tilde{\mathbf{W}}\mathbf{x})_i|, \lambda\})^2$
  Find smallest integer $i_k$ s.t. with $L_k = \eta^{i_k} L_{k-1}$
    $F(p_L(\mathbf{y}_k)) \leq Q_L(p_L(\mathbf{y}_k), \mathbf{y}_k))$
  $\mathbf{x}_k = \mathbf{S}_\mathbf{x} \mathbf{W}^H \mathcal{T}_{\lambda_k/L} \left( \mathbf{W}(\mathbf{y}_k - \frac{2}{L} \left[ - \|\mathbf{b}\|_2^{-2} \mathbf{A}^H \mathbf{r}_{k-1} + V_\Delta F^{\mathrm{data}}(\mathbf{y}_k) \mathbf{L}_\mathbf{w} \mathbf{y}_k \right] ) \right)$
  $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
  $\mathbf{y}_{k+1} = \mathbf{x}_k + \frac{t_k - 1}{t_{k+1}} (\mathbf{x}_k - \mathbf{x}_{k-1})$
```

4.2.2. Conjugate Gradient Method

The double multiplicatively regularized functions in minimization problems (P3) and (P4) are differentiable, meaning we do not need any soft-thresholding operations to mitigate the indifferentiability of the l_1-norm of additive regularization functions. Instead, we can fall back to a conjugate gradient (CG) method [37] in which the image is reconstructed with update formula

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \beta_k \mathbf{d}_k, \tag{4.46}$$

where β_k is the stepsize and the update direction $\mathbf{d}_k$ is given by

$$\mathbf{d}_k = \mathbf{g}_k + \frac{\mathrm{Re}(\mathbf{g}_k^H (\mathbf{g}_k - \mathbf{g}_{k-1}))}{\|\mathbf{g}_{k-1}\|_2^2} \mathbf{d}_{k-1}, \tag{4.47}$$

with $\mathbf{g}_k$ the gradient of the cost function evaluated at $\mathbf{x}_{k-1}$. We can give a single general expression of the cost functions in (P3) and (P4) as

$$\begin{aligned} F_{k-1}(\mathbf{x}) &= F^{\text{data}}(\mathbf{x}) F^{\mathrm{T}}_{k-1}(\mathbf{x}) F^{\text{wavelet}}_{k-1}(\mathbf{x}) \\ &= \frac{\|\mathbf{b}-\mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{b}\|_2^2} \left(\delta^2_{\mathrm{T},k-1} V_\Delta \mathbf{e}^T \mathcal{W}_{\mathrm{T},k-1}\mathbf{e} + V_\Delta \mathbf{x}^H \mathbf{L}_{\mathrm{T},\mathbf{w}}\mathbf{x}\right) \left(\delta^2_{\mathrm{w},k-1} V_\Delta \mathbf{e}^T \mathcal{W}_{\mathrm{w},k-1}\mathbf{e} \right. \\ &\quad \left. + V_\Delta \mathbf{x}^H \mathbf{W}^H \mathcal{W}_{\mathrm{w},k-1} \mathbf{W}\mathbf{x}\right) \ , \end{aligned} \tag{4.48}$$

where T is either TV or dTV and $\mathcal{P}_{\mathrm{T}} = \mathbf{I}$ for $\mathrm{T} =$ TV and $\mathcal{P}_{\mathrm{T}} = \mathbf{P}^{\perp}_{|\mathbf{g}_{\mathrm{TV},k-1}|_\eta}$ for $\mathrm{T} =$ dTV. The gradients of the three individual functionals are

$$\begin{aligned} g_k^{\text{data}} &= -2\,\|\mathbf{b}\|_2^{-2}\,\mathbf{A}^H \mathbf{r}_{k-1}, \\ g_k^{\mathrm{T}} &= 2V_\Delta \mathbf{L}_{\mathrm{T},\mathbf{w}}\mathbf{x}, \\ g_k^{\text{wavelet}} &= 2V_\Delta \mathbf{W}^H \mathcal{W}_{\mathrm{w},k-1}\mathbf{W}\mathbf{x}_{k-1}, \end{aligned} \tag{4.49}$$

where $\mathbf{r}_{k-1} = \mathbf{b} - \mathbf{A}\mathbf{x}_{k-1}$ is the data residual. Keeping in mind that $F^{\mathrm{T}}_{k-1}(\mathbf{x}_{k-1}) = 1$ and $F^{\text{wavelet}}_{k-1}(\mathbf{x}_{k-1}) = 1$, the gradient of $F_{k-1}(\mathbf{x}_{k-1})$ is given by

$$\begin{aligned} g_k &= g_k^{\text{data}} F^{\mathrm{T}}_{k-1}(\mathbf{x}_{k-1}) F^{\text{wavelet}}_{k-1}(\mathbf{x}_{k-1}) + F^{\text{data}}(\mathbf{x}_{k-1}) g_k^{\mathrm{T}} F^{\text{wavelet}}_{k-1}(\mathbf{x}_{k-1}) \\ &\quad + F^{\text{data}}(\mathbf{x}_{k-1}) F^{\mathrm{T}}_{k-1}(\mathbf{x}_{k-1}) g_k^{\text{wavelet}} \\ &= g_k^{\text{data}} + F^{\text{data}}(\mathbf{x}_{k-1})\left(g_k^{\mathrm{T}} + g_k^{\text{wavelet}}\right) \\ &= 2\left[-\|\mathbf{b}\|_2^{-2}\mathbf{A}^H\mathbf{r}_{k-1} + V_\Delta F^{\text{data}}(\mathbf{x}_{k-1})\left(\mathbf{L}_{\mathrm{T},\mathbf{w}} + \mathbf{W}^H \mathcal{W}_{\mathrm{w},k-1}\mathbf{W}\right)\mathbf{x}_{k-1}\right] \ . \end{aligned} \tag{4.50}$$

The update coefficient β_k is found by solving

$$\left.\frac{\partial F_{k-1}(\mathbf{x}_{k-1} + \beta\mathbf{d}_k)}{\partial\beta}\right|_{\beta=\beta_k} = 0 \ . \tag{4.51}$$

Writing out the cost function in terms of β we get

$$\begin{aligned} F_{k-1}(\mathbf{x}_{k-1} + \beta\mathbf{d}_k) &= F_{k-1}(\mathbf{x}_{k-1} + \beta\mathbf{d}_k) F^{\mathrm{TV}}_{k-1}(\mathbf{x}_{k-1} + \beta\mathbf{d}_k) F^{\text{wavelet}}_{k-1}(\mathbf{x}_{k-1} + \beta\mathbf{d}_k) \\ &= (a_0 + a_1\beta + a_2\beta^2)(b_0 + b_1\beta + b_2\beta^2)(c_0 + c_1\beta + c_2\beta^2) \ , \end{aligned} \tag{4.52}$$

with coefficients

$$\begin{aligned} &a_0 = F^{\text{data}}(\mathbf{x}_{k-1}), && a_1 = -2\frac{\mathrm{Re}\left(\mathbf{r}_{k-1}^H \mathbf{A}\mathbf{d}_k\right)}{\|\mathbf{b}\|_2^2}, && a_2 = \frac{\|\mathbf{A}\mathbf{d}_k\|_2^2}{\|\mathbf{b}\|_2^2}, \\ &b_0 = 1, && b_1 = 2V_\Delta\,\mathrm{Re}\left(\mathbf{x}_{k-1}^H \mathbf{L}_{\mathrm{T},\mathbf{w}}\mathbf{d}_k\right), && b_2 = V_\Delta \mathbf{d}_k^H \mathbf{d}_k, \\ &c_0 = 1, && c_1 = 2V_\Delta\,\mathrm{Re}\left(\mathbf{x}_{k-1}^H \mathbf{W}^H \mathcal{W}_{\mathrm{w},k-1}\mathbf{W}\mathbf{d}_k\right), && c_2 = V_\Delta \mathbf{d}_k^H \mathbf{W}^H \mathcal{W}_{\mathrm{w},k-1}\mathbf{W}\mathbf{d}_k \ . \end{aligned} \tag{4.53}$$

We can then explicitly write out

$$\begin{aligned} \frac{\partial F_{k-1}(\mathbf{x}_{k-1} + \beta\mathbf{d}_k)}{\partial\beta} &= a_0c_1 + a_0b_1 + a_1 + 2(a_0b_2 + a_0c_2 + a_0b_1c_1 + a_1b_1 + a_1c_1 + a_2)\beta \\ &\quad + 3(a_0b_1c_2 + a_0b_2c_1 + a_1b_1c_1 + a_1b_2 + a_1c_2 + a_2b_1 + a_2c_1)\beta^2 \\ &\quad + 4(a_0b_2c_2 + a_1b_1c_2 + a_1b_2c_1 + a_2b_1c_1 + a_2b_2 + a_2c_2)\beta^3 \\ &\quad + 5(a_1b_2c_2 + a_2b_1c_2 + a_2b_2c_1)\beta^4 + 6(a_2b_2c_2)\beta^5 \ . \end{aligned} \tag{4.54}$$

We solve the polynomial using a built-in root-finding algorithm, but it can be solved analytically as well. The algorithm is summarized in Reconstruction Algorithm 2.

```
Reconstruction Algorithm 2: CG for (P3) and (P4)
Initialize: $\mathbf{x}_0 = \mathbf{A}^H\mathbf{b},\ \mathbf{d}_0 = \mathbf{0}, \mathscr{P}_{\mathrm{T}}$
Until convergence or $k > k_{max}$
    $g_k = 2\left[-\|\mathbf{b}\|_2^{-2}\mathbf{A}^H\mathbf{r}_{k-1} + V_\Delta F^{\mathrm{data}}(\mathbf{x}_{k-1})\left(\mathbf{L}_{\mathrm{T},\mathbf{w}} + \mathbf{W}^H\mathcal{W}_{k-1}^{\mathrm{wavelet}}\mathbf{W}\right)\mathbf{x}_{k-1}\right]$
    $\mathbf{d}_k = \mathbf{g}_k + \frac{\mathrm{Re}\left(\mathbf{g}_k^H(\mathbf{g}_k - \mathbf{g}_{k-1})\right)}{\|\mathbf{g}_{k-1}\|_2^2}\mathbf{d}_{k-1}$
    $\beta_k = \arg\min_\beta F_{k-1}(\mathbf{x}_{k-1} + \beta\mathbf{d}_k)$
    $\mathbf{x}_k = \mathbf{x}_{k-1} + \beta_k\mathbf{d}_k$
```

4.3. Convergence Behaviour and Stopping Criteria

First off, we emphasize the importance of having an image domain mask $\mathbf{S_x} \neq \mathbf{I}$ for correct initialization in both FISTA and the CG method. If we look at the gradient used in FISTA in Eq. (4.37) and the gradient used in the CG method in Eq. (4.50) they are both dependent on the data residual $\mathbf{r}_{k-1}$ and the data functional $F^{\mathrm{data}}(\mathbf{x}_{k-1}) = \frac{\|\mathbf{r}_{k-1}\|_2^2}{\|\mathbf{b}\|_2^2}$. If we initialize the algorithms without an image domain mask we get $\mathbf{x}_0 = \mathbf{A}^H\mathbf{b} = \mathbf{F}^H\mathbf{S_k}\mathbf{b}$, which results in

$$\begin{aligned} \mathbf{r}_0 &= \mathbf{b} - \mathbf{A}\mathbf{x}_0 \\ &= \mathbf{b} - \mathbf{S_k}\mathbf{F}\mathbf{F}^H\mathbf{S_k}\mathbf{b} \\ &= \mathbf{b} - \mathbf{S_k}\mathbf{S_k}\mathbf{b} \\ &= \mathbf{b} - \mathbf{S_k}\mathbf{b} = \mathbf{b} - \mathbf{b} = \mathbf{0}\ . \end{aligned} \tag{4.55}$$

If $\mathbf{r}_0 = \mathbf{0}$ then the gradient in both algorithms will also be $\mathbf{g} = \mathbf{0}$ and the algorithms immediately stop after 1 iteration. For this reason it is not only advantageous, but necessary to have an image domain mask.

Next, the convergence behaviour of the multiplicative regularization functions is different from that of the additive l_1-norm regularization functions. Normally, while minimizing a cost function with an additive regularization function the regularization function itself gets minimized. However, this is not the case for the multiplicative regularization functions. We can analyze for example the multiplicative continuous TV regularization function

$$F_{k-1}^{\mathrm{TV}}(X) = \int \frac{|\nabla X|^2 + \delta_{\mathrm{TV},k-1}^2}{|\nabla X_{k-1}|^2 + \delta_{\mathrm{TV},k-1}^2} dV\ ,\ \text{with}\ \delta_{\mathrm{TV},k-1}^2 = F^{\mathrm{data}}(\mathbf{x}_{k-1})^2 \int |\nabla X_{k-1}|^2 dV\ . \tag{4.56}$$

In the limit we get

$$\lim_{k\to\infty} X_{k-1} = \lim_{k\to\infty} X_k = X\ , \tag{4.57}$$

hence

$$\lim_{k\to\infty} \nabla X_{k-1} = \lim_{k\to\infty} \nabla X_k = \nabla X \tag{4.58}$$

and the same goes for $\delta_{\mathrm{TV},k-1}^2$. Thus, in the limit we have

$$\lim_{k\to\infty} F_{k-1}^{\mathrm{TV}}(X) = \lim_{k\to\infty} \int \frac{|\nabla X|^2 + \delta_{\mathrm{TV},k-1}^2}{|\nabla X_{k-1}|^2 + \delta_{\mathrm{TV},k-1}^2} dV = \int \frac{|\nabla X|^2 + \delta_{\mathrm{TV}}^2}{|\nabla X|^2 + \delta_{\mathrm{TV}}^2} dV = 1\ . \tag{4.59}$$

The same conclusions can be made for the multiplicative directional TV and wavelet regularization functions. We can use this convergence behaviour to formulate stopping criteria by setting a tolerance level *Tol* such that the algorithms stop when the multiplicative regularization functions are within tolerance level from 1. This means that we stop FISTA with single multiplicative regularization function when $|1 - F^{\mathrm{TV}}(\mathbf{x}_k)| \leq$ *Tol* and we stop the CG method with double multiplicative regularization when both $|1 - F^{\mathrm{TV}}(\mathbf{x}_k)| \leq$ *Tol* and $|1 - F^{\mathrm{wavelet}}(\mathbf{x}_k)| \leq$ *Tol*. Additionally, we set a maximum number of iterations and only start checking the stopping criteria when $|1 - F^{\mathrm{TV}}(\mathbf{x}_n)| \geq$ *Tol* and for the CG methods when also $|1 - F^{\mathrm{wavelet}}(\mathbf{x}_n)| \geq$ *Tol* for any $n \in 1, 2, ..., k$. This is because the regularization functions sometimes already start very close to 1 and only start to deviate from 1 after a number of iterations. If this check is not done, the algorithm might stop immediately without doing anything.

5

Results

We set up a simulation study to make both a quantitative and qualitative comparison of the reconstruction algorithms introduced in Chap. 4. We also show reconstructions of experimental data of scans performed on the low-field MRI scanner at the LUMC. We first introduce the different experiments in Section 5.1. The simulation results are discussed in Section 5.2. The results from the scanner data for a two-dimensional phantom, a postmortem brain scan, and an in-vivo scan are given in Section 5.3, Section 5.4, and Section 5.5, respectively.

5.1. Experimental Setups

A number of experiments are conducted to analyze and show the performance of the reconstruction methods from Chap. 4, which are listed again in Table 5.1. We will also compare the reconstruction performance with the plain IFFT of the undersampled data.

Table 5.1: Overview of reconstruction algorithms and the corresponding cost functions they minimize.

Algorithm	General form of $F(\mathbf{x})$	
FISTA	$F^{\text{data}}(\mathbf{x})F^{\text{TV}}(\mathbf{x}) + \lambda F^{\text{wavelet}}(\mathbf{x})$	(P2)
CG (TV)	$F^{\text{data}}(\mathbf{x})F^{\text{TV}}(\mathbf{x})F^{\text{wavelet}}(\mathbf{x})$	(P3)
CG (dTV)	$F^{\text{data}}(\mathbf{x})F^{\text{dTV}}(\mathbf{x})F^{\text{wavelet}}(\mathbf{x})$	(P4)

Before going into detail with each experiment, a short summary of the most important data is shown in Table 5.2. All experimental data are acquired using a turbo spin echo (TSE) sequence [1], of which the acquisition parameters are reported under the reconstruction figures.

Table 5.2: Overview of the conducted experiments and their corresponding data sizes, resolutions, and acceleration factor R for which we performed reconstructions. We also state whether the data in the experiment is simulated or from the real low-field scanner at the LUMC.

Experiment	Experiment type	Data size	Res (mm)	R
Shepp-Logan phantom	Simulation	$[256, 256]$	-	$1.5,\ 2,\ 4$
2D brain phantom	Scanner	$[280, 280]$	$[2, 2, 3]$	$1.4,\ 2,\ 3.3$
Postmortem brain	Scanner	$[120, 120, 40]$	$[2, 2, 5]$	$2,\ 4$
In-vivo	Scanner	$[167, 120, 58]$	$[1.5, 1.5, 3]$	$1,\ 2$

There are still a few free parameters which we need to choose. The backtracking parameter in FISTA is set to $\eta = 1.5$ and the initial Lipschitz constant is set to $L_0 = 1000$. The weighting parameter in the $|\cdot|_\eta$ operator used in CG (dTV) is set to $\eta = 0.01 \cdot \max|\mathrm{x}_0|$. The tolerance for all algorithms is set to *Tol* $= 0.015$ and the maximum number of iterations is $k_{\max} = 300$, unless indicated otherwise. For the image domain masking algorithm the default value of the hard-thresholding parameter was set to $\tau = 2$, which resulted in satisfactory masks in most cases. The only exceptions are for the fully sampled in-vivo scan for which we set the parameter to $\tau = 1.5$ and the R = 4 case of the postmortem brain for which we set the parameter to $\tau = 5$.

The first experiment is a simulation study on the Shepp-Logan phantom, which is a standard test image used in image processing, introduced in [38]. We undersample k-space data of a complete image of the Shepp-Logan phantom for different acceleration ratios and add complex Gaussian noise such that we have an SNR $= 12$ dB before undersampling. In this way, we can make a quantitative comparison, because we have the ground truth. We compare the computational time and the reconstruction performance of the different algorithms against each other and the plain IFFT of the noisy undersampled data. If we normalize all images such that the maximum pixel/voxel intensity has magnitude 1, we can compare the magnitude images of the ground truth $\mathbf{x}_{\mathrm{ref}}$ and any reconstruction $\mathbf{x}_{\mathrm{rec}}$ by peak signal-to-noise ratio (PSNR), which is defined as

$$\mathrm{PSNR}(\mathbf{x}_{\mathrm{rec}}, \mathbf{x}_{\mathrm{ref}}) = 10 \cdot \log_{10} \left(\frac{\max\{\mathbf{x}_{rec}\}^2}{\mathrm{MSE}(\mathbf{x}_{\mathrm{rec}}, \mathbf{x}_{\mathrm{ref}})} \right) , \tag{5.1}$$

where $\max\{\mathbf{x}_{\mathrm{rec}}\}$ is the maximum pixel intensity of the reconstructed image and $\mathrm{MSE}(\mathbf{x}_{\mathrm{rec}}, \mathbf{x}_{\mathrm{ref}})$ is the mean squared error between $\mathbf{x}_{\mathrm{rec}}$ and $\mathbf{x}_{\mathrm{ref}}$. Additionally we can compare the reconstructions by means of the structural similarity index measure (SSIM), which is defined as

$$\mathrm{SSIM}(\mathbf{x}_{\mathrm{rec}}, \mathbf{x}_{\mathrm{ref}}) = l(\mathbf{x}_{\mathrm{rec}}, \mathbf{x}_{\mathrm{ref}}) \cdot c(\mathbf{x}_{\mathrm{rec}}, \mathbf{x}_{\mathrm{ref}}) \cdot s(\mathbf{x}_{\mathrm{rec}}, \mathbf{x}_{\mathrm{ref}}) , \tag{5.2}$$

where $l(\mathbf{x}_{\mathrm{rec}}, \mathbf{x}_{\mathrm{ref}})$ is the luminance term, $c(\mathbf{x}_{\mathrm{rec}}, \mathbf{x}_{\mathrm{ref}})$ is the contrast term, and $s(\mathbf{x}_{\mathrm{rec}}, \mathbf{x}_{\mathrm{ref}})$ is the structural term. Details on how the different terms are calculated from local image statistics can be found in [39]. Although the PSNR is a well recognized quantitative measure, the SSIM might be better able to capture the reconstruction quality as perceived by humans [39]. Next to these quantitative measures we plot difference images and reconstructions along a line (line profiles) to have a more in-depth look at what parts of the images are reconstructed properly. Lastly, we inspect the convergence behaviour of the regularization functionals and the performance over a broad range of undersampling factors and input SNR scenarios.

In the subsequent sections we successively look at reconstructions of a 2D slab brain phantom, a postmortem brain, and in-vivo scans, which were scanned with the low-field scanner at the LUMC. We do not have a ground truth like in the simulation study, but we can still make qualitative comparisons. For the in-vivo experiment, we additionally use our algorithms as a pure denoising algorithm for the fully sampled data. The in-vivo images contain more noise introduced by the body and due to the fact that the scanner cannot be fully closed to form a Faraday cage as with the other experiments. We set the k-space mask in the purely denoising case as $\mathbf{S}_k = \mathbf{I}$, resulting in a measurement matrix $\mathbf{A} = \mathbf{S}_k\mathbf{F}\mathbf{S}_x = \mathbf{F}\mathbf{S}_x$.

5.2. Shepp-Logan Simulation

The reconstructions of the noisy undersampled Shepp-Logan phantom along with the ground truth are shown in Fig. 5.1 with the corresponding k-space undersampling patterns in Fig. 5.2. All noise outside of the object is removed, also due to the image domain mask, except for some noise very near the object in the FISTA reconstructions. The CG (TV) method seems to perform best visually, which is also reflected in the SSIM shown in Table 5.3. Both the noise and the ghosting pattern are almost completely removed except for the lower part of the object where the ghosting pattern was the most severe. It is also evident that the FISTA reconstructions do not outperform the CG method reconstructions quantitatively according to the PSNR and SSIM, but also qualitatively. The ghosting after FISTA reconstructions are visible for all undersampling factors, which is not the case for the other reconstruction methods. This is further emphasized by the difference images shown in Fig. 5.3. The difference images also show that on average for R = 1.5 and R = 2 the CG (dTV) seems to outperform the other methods as the values seem to be closer to 0 on average, but the ghosting artefacts are more present than in the CG (TV) reconstructions. We can also see from Table 5.3 that the CG (dTV) is consistently faster than the CG (TV) method.

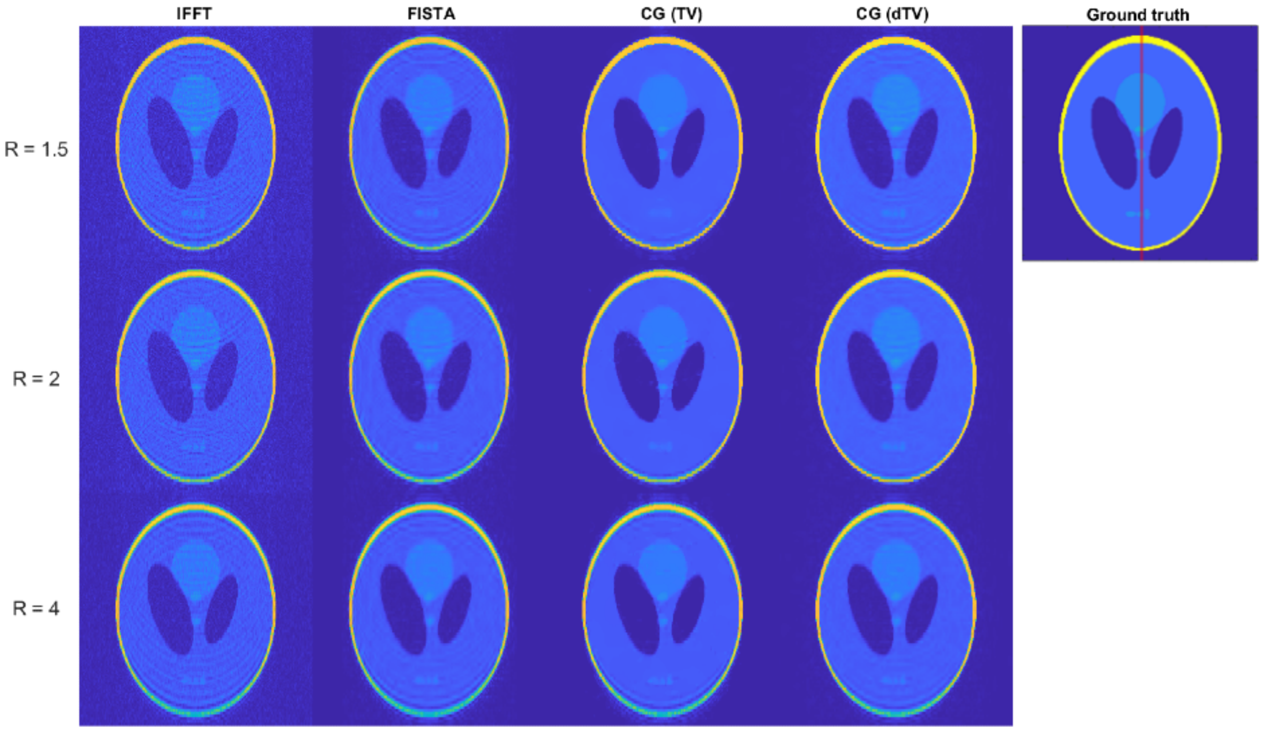

Figure 5.1: Noisy undersampled Shepp-Logan phantom reconstructions of the different algorithms for different sampling ratios. Rows from top to bottom: reconstructions for R = 1.5, R = 2, R = 4. Columns from left to right: IFFT, FISTA, CG method with isotropic TV, CG method with directional TV reconstructions. The ground truth image is shown to the right with the red line indicating where the line profiles are taken.

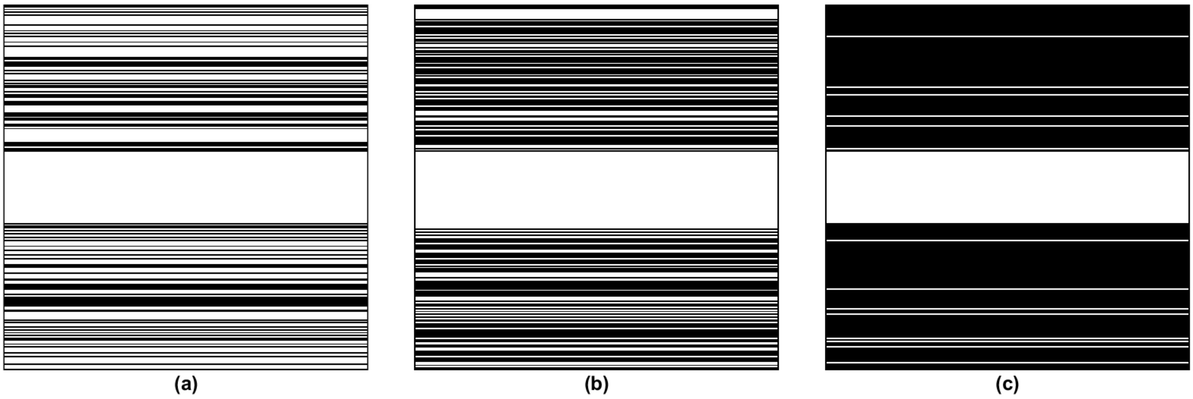

Figure 5.2: k-space undersampling patterns for the Shepp-Logan phantom with 20% sampling in the center for R = 1.5 (a), R = 2 (b) and R = 4 (c). The white lines represent the sampled lines.

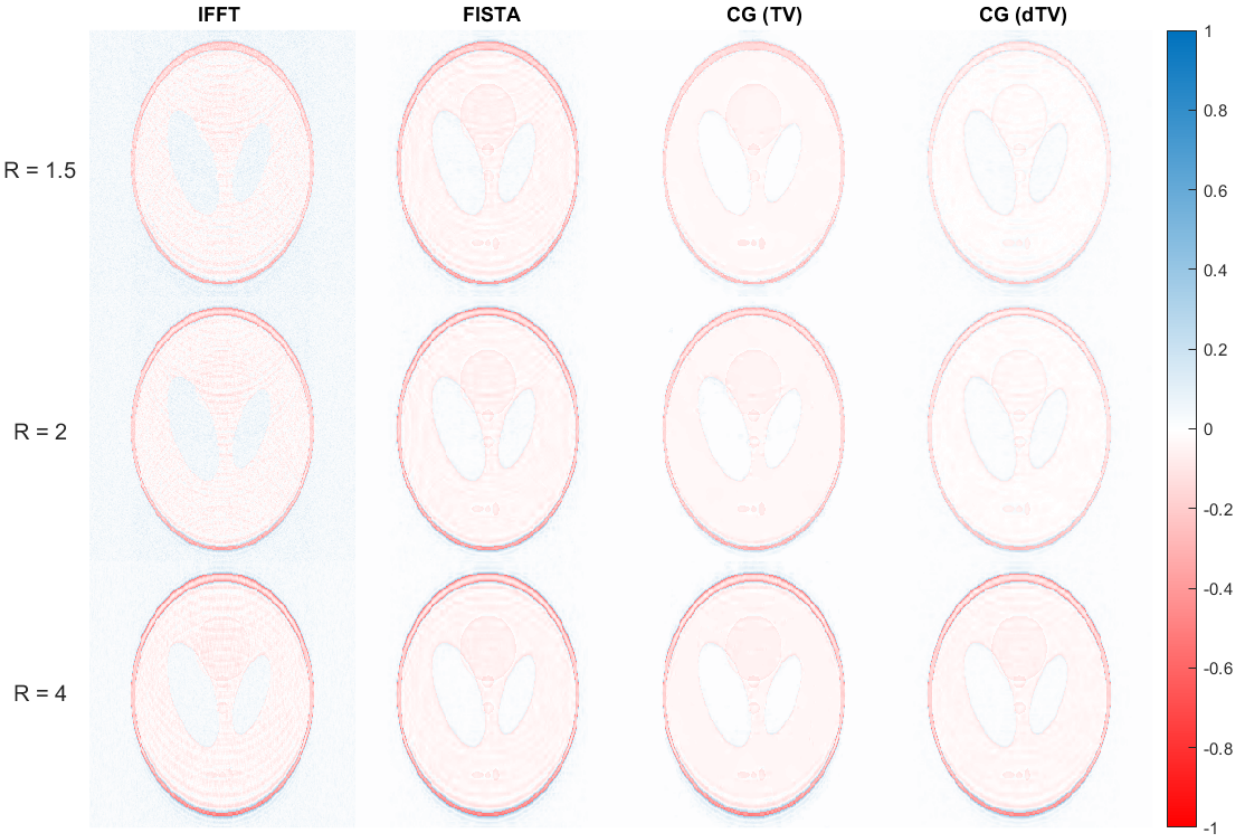

Figure 5.3: Difference images of the reconstructions shown in Fig. 5.1 with the ground truth. A blue pixels indicates that the reconstruction absolute pixel value is higher than the ground truth and vice versa for red pixels. Rows from top to bottom: difference images of the reconstructions for R = 1.5, R = 2, R = 4. Columns from left to right: difference images of IFFT, FISTA, CG method with isotropic TV, CG method with directional TV reconstructions.

Table 5.3: The PSNR (dB), SSIM, and reconstruction times (s) for the IFFT, FISTA, and the CG methods (TV and dTV) for the Shepp-Logan phantom reconstructions for acceleration factors R = 1.5, 2 and 4. The reconstruction time of the IFFT reconstructions are taken as instantly and the best performance for each acceleration factor and measure is indicated in bold numbers.

Acceleration factor	Reconstruction method	PSNR (dB)	SSIM	Reconstruction time (s)
1.5	IFFT	23.50	0.28	-
	FISTA	23.58	0.73	6.71
	CG (TV)	27.12	**0.94**	8.83
	CG (dTV)	**29.29**	0.80	**4.97**
2	IFFT	23.08	0.30	-
	FISTA	22.99	0.77	9.54
	CG (TV)	25.28	**0.93**	11.43
	CG (dTV)	**27.19**	0.83	**6.84**
4	IFFT	21.80	0.37	-
	FISTA	21.92	0.76	**11.38**
	CG (TV)	22.57	**0.87**	21.24
	CG (dTV)	**23.19**	0.79	16.49

5.2.1. Line Profiles of Reconstructions

The line profiles for R = 1.5 are shown in Fig. 5.4 and the line profiles for R = 2 and R = 4 are shown in Appendix D.1. These line profiles are taken along the red line shown on the ground truth image in Fig. 5.1. It is clear that the IFFTs are denoised as the spikes are smoothed out. These plots again show that the CG (TV) method performs best at smoothing out the parts of the image which should be constant. The FISTA and CG (dTV) line profiles look similar to each other, but the CG (dTV) reconstructions seem less volatile and consistently closer to the ground truth. We can also see that at the outer parts of the image, where the ground truth image is perfectly 0, there is still some ghosting/noise left. Lastly, we see that if the IFFT does not capture a part of the structure well, such as the narrow peak on the right, the reconstruction algorithms will not be able to retrieve it either. This is especially noticeable for the R = 4 case.

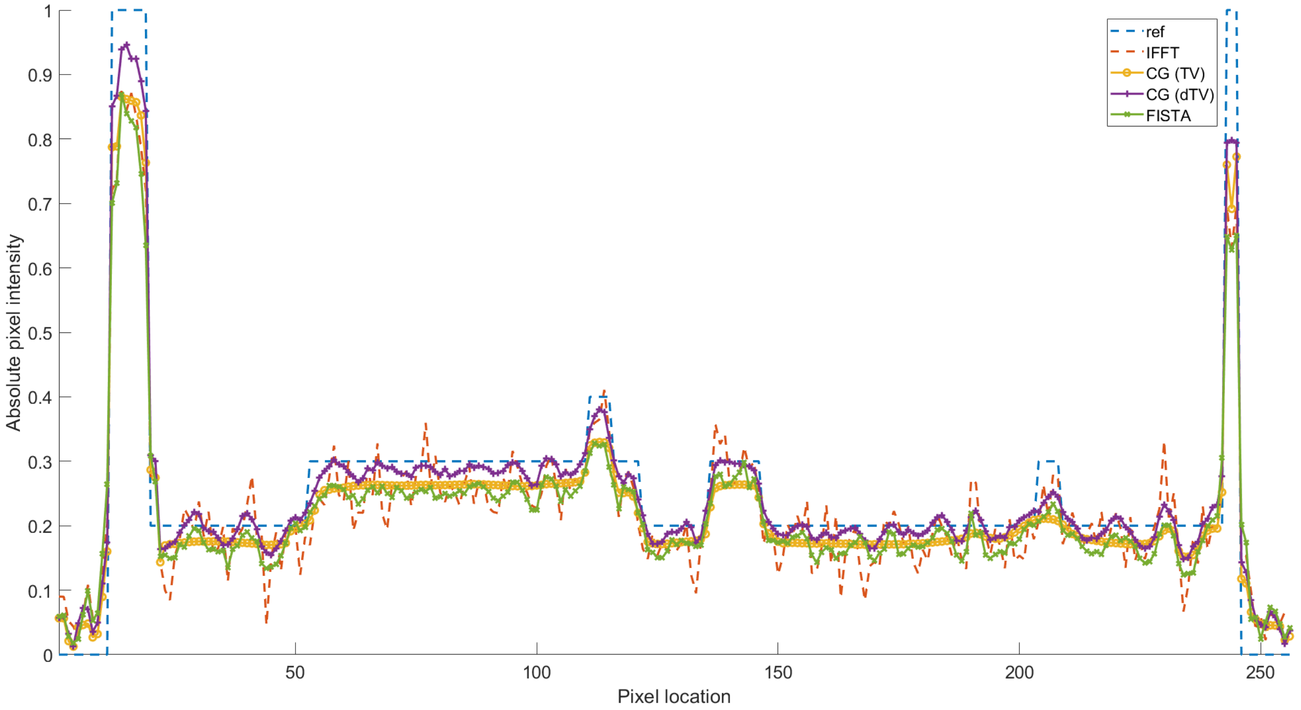

Figure 5.4: Line profiles of the ground truth and the IFFT, CG (TV), CG (dTV), and FISTA reconstructions along the red line indicated on the ground truth image in Fig. 5.1 for R = 1.5.

5.2.2. Convergence Behaviour

To show the validity of using the multiplicative regularization functionals as stopping criteria we plot the corresponding multiplicative functionals for all three algorithms for the R = 4 case in Fig. 5.5 (the plots look similar for the R = 1.5 and R = 2 cases and are therefore left out). The TV functional for FISTA in (a) and the dTV functional for CG (dTV) in (c) both overshoot in the first two iterations due to initialization and then stay close to 1. The TV and wavelet for the CG (TV) in (b) already start very close to 1, so it is beneficial to only start checking when the functionals have sufficiently moved away from 1. After a few iterations all graphs slowly converge to 1.

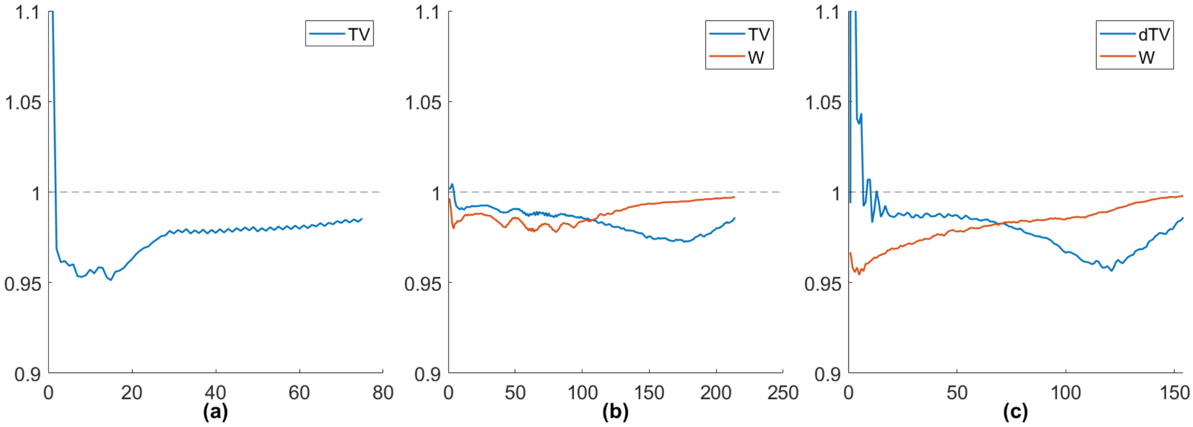

Figure 5.5: Convergence for the multiplicative (d)TV and wavelet regularization functionals for the R = 4 case. (a) TV functional for FISTA on multiplicatively regularized TV, additively regularized wavelet function, (b) TV and wavelet functional for CG (TV) method on double multiplicatively regularized function, (c) dTV and wavelet functional for CG (dTV) method on double multiplicatively regularized function.

5.2.3. Performance vs Input SNR and Acceleration Factor

We inspect the performance of all reconstruction methods with respect to input SNR over a range of 1 to 10 dB and acceleration factors ranging from R = 1.5 to R = 10. The maximum number of iterations is set to 2500 as the lower SNR tests need significantly more iterations to converge. The corresponding reconstruction PSNR and SSIM are shown in Fig. 5.6. All reconstructions are also shown in Fig. D.3 for different acceleration factors and in Fig. D.4 for different input SNR values.

From Fig. 5.6 we can immediately see the significant improvement compared with the IFFT for all reconstruction algorithms. The CG (TV) method shows the most robust performance as its consistently performs best in terms of SSIM and rivals the performance of CG (dTV) in terms of PSNR. Both CG methods tend to perform better than FISTA. The CG (dTV) however, seems to struggle with lower SNR if we look at the SSIM in Fig. 5.6.d. This could be because it is increasingly more difficult to extract the edge information from the input image for lower SNR values.

Lastly, each reconstruction algorithm has been run 1 time for each test. To be more certain of the performance we could run a Monte Carlo simulation in which we run the algorithms in each test for a large number of times and change the k-space undersampling mask and noise for each run. The results would be averaged, which could for example alleviate the performance drop we see in the SSIM values for the CG (dTV) method in the input SNR test between an input SNR of 6 and 8 dB.

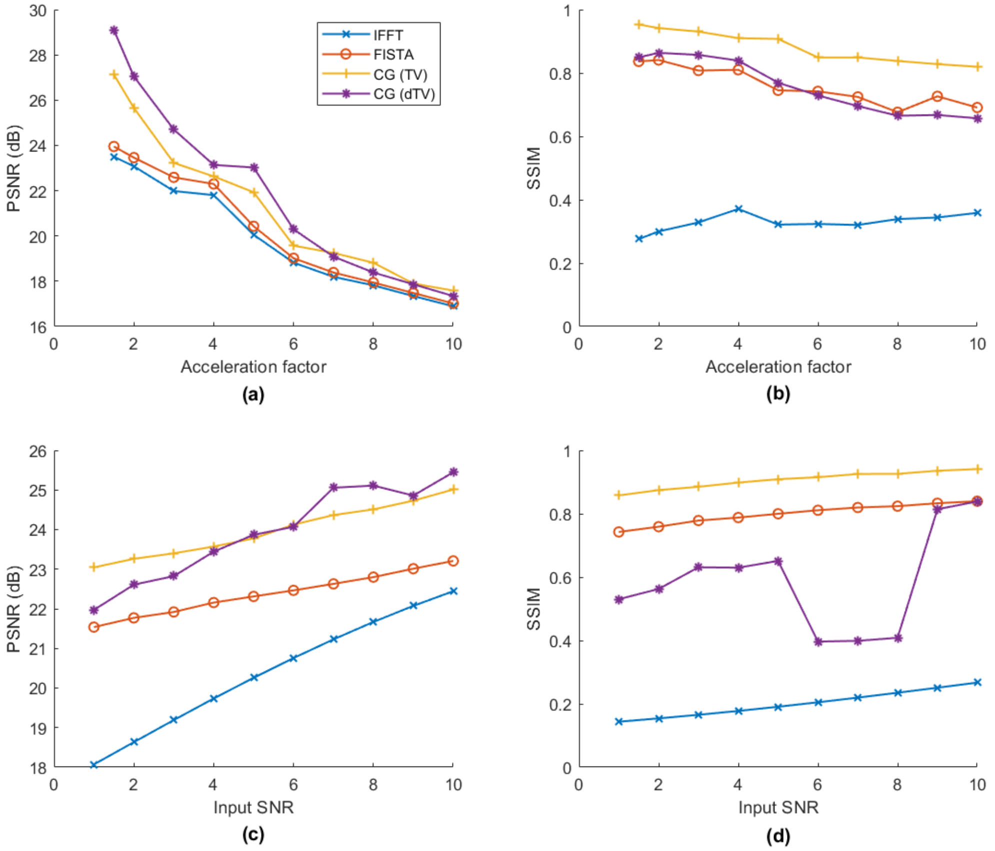

Figure 5.6: Performance of the IFFT, FISTA, CG (TV), and CG (dTV) reconstructions with respect to varying acceleration factors and input SNR values. (a) Reconstruction PSNR vs acceleration factor. (b) Reconstruction SSIM vs acceleration factor. (c) Reconstruction PSNR vs input SNR (d) Reconstruction SSIM vs input SNR.

5.3. Two-Dimensional Phantom Scans

Next, we reconstruct images from the actual scanner of a two-dimensional brain phantom. The reconstructions can be found in Fig. 5.7 with the corresponding k-space undersampling patterns shown in Fig. 5.8 and the image domain mask shown in Fig. D.5.

It seems like the FISTA reconstructions did not do much as the noise present within the phantom seems as severe as with the IFFT reconstructions. Also, none of the FISTA reconstructions terminated before the maximum number of iterations was reached. In general, the CG methods performed well for all acceleration factors, except for the CG (dTV) reconstruction of the R = 3.3 scan which terminated early. The CG (TV) again results in a piecewise smoother reconstruction than the CG (dTV) and has the visually best reconstruction for the R = 3.3 case. In terms of computational time the CG (dTV) is clearly the fastest in all cases as is shown in Table 5.4.

When comparing the CG reconstructions in the R = 1.4 and R = 2 case with the fully sampled IFFT, one might argue that the undersampled CG reconstruction look better. These reconstructions are visually completely denoised and only some of the line structures at the bottom of the phantom are slightly blurred.

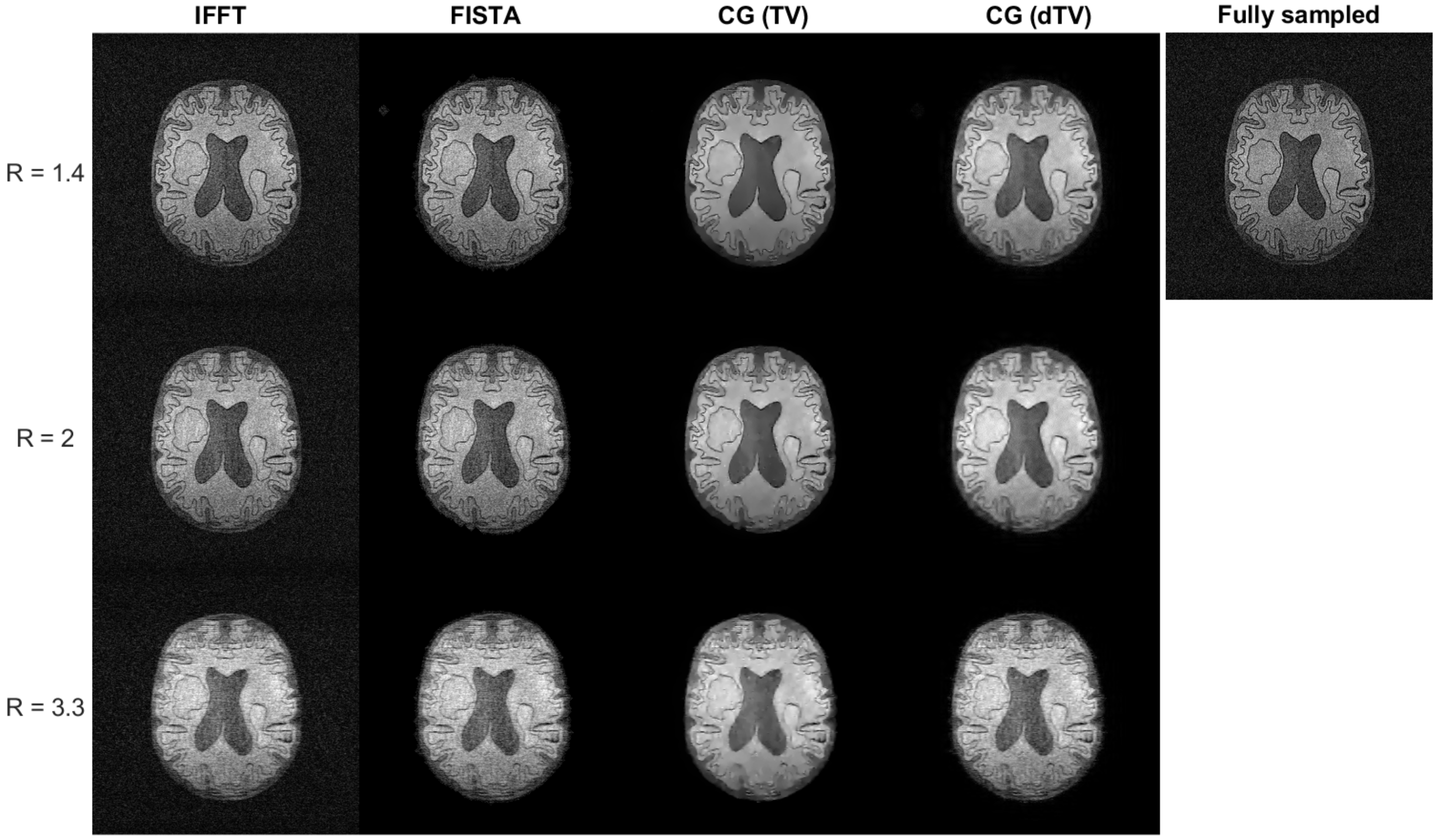

Figure 5.7: Two-dimensional brain phantom reconstructions of the different algorithms for different sampling ratios. Rows from top to bottom: reconstructions for R = 1.4, R = 2, R = 3.3. Columns from left to right: IFFT, FISTA, CG method with isotropic TV, CG method with directional TV reconstructions. The fully sampled image is shown to the right. Data acquired using a TSE sequence with the following scan parameters: TR/TE: 600/25 ms, echo train length: 1.

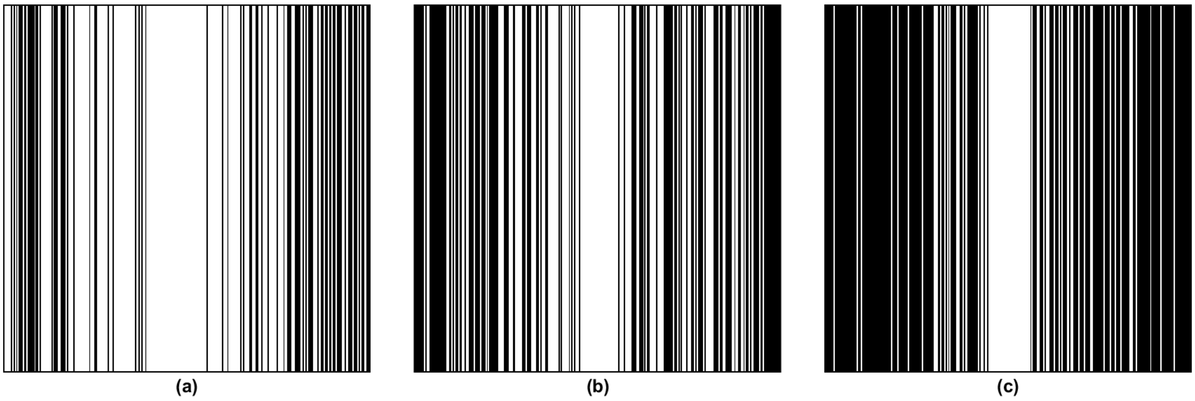

Figure 5.8: k-space undersampling patterns for the two-dimensional brain phantom with 10% sampling in the center for R = 1.4 (a), R = 2 (b) and R = 3.3 (c). The white lines represent the sampled lines.

Table 5.4: Reconstruction times for the FISTA and both CG method reconstructions of the brain phantom data for acceleration factors R = 1.4, 2 and 3.3. The fastest reconstruction time for each acceleration factor is indicated in bold numbers.

Acceleration factor	Reconstruction method	Reconstruction time (s)
1.4	FISTA	26.39
	CG (TV)	4.10
	CG (dTV)	**1.22**
2	FISTA	29.92
	CG (TV)	6.12
	CG (dTV)	**1.66**
3.3	FISTA	29.87
	CG (TV)	8.65
	CG (dTV)	**0.85**

5.4. Postmortem Brain Scans

A slice of the reconstructions of the postmortem brain scans are shown in Fig. 5.9 with the phase-encoding planes of the corresponding k-space undersampling patterns shown in Fig. 5.10 and the image domain mask shown in Fig. D.6. Note that at the moment of performing the scans the fully sampled center was a square. It is immediately clear that it is tougher to denoise these images while keeping the details, as many details are more subtle compared with the Shepp-Logan and brain phantom. It is also clear that many details are completely lost, especially in the R = 4 case, such that the reconstructions of the R = 4 case do not look significantly better than the IFFT. Compared with the R = 4 scan, the R = 2 scan maintains more detail therefore its reconstructions do as well. Again the CG (dTV) method is the fastest as can be seen in Table 5.5, but all reconstruction methods perform qualitatively comparable to each other for this experiment.

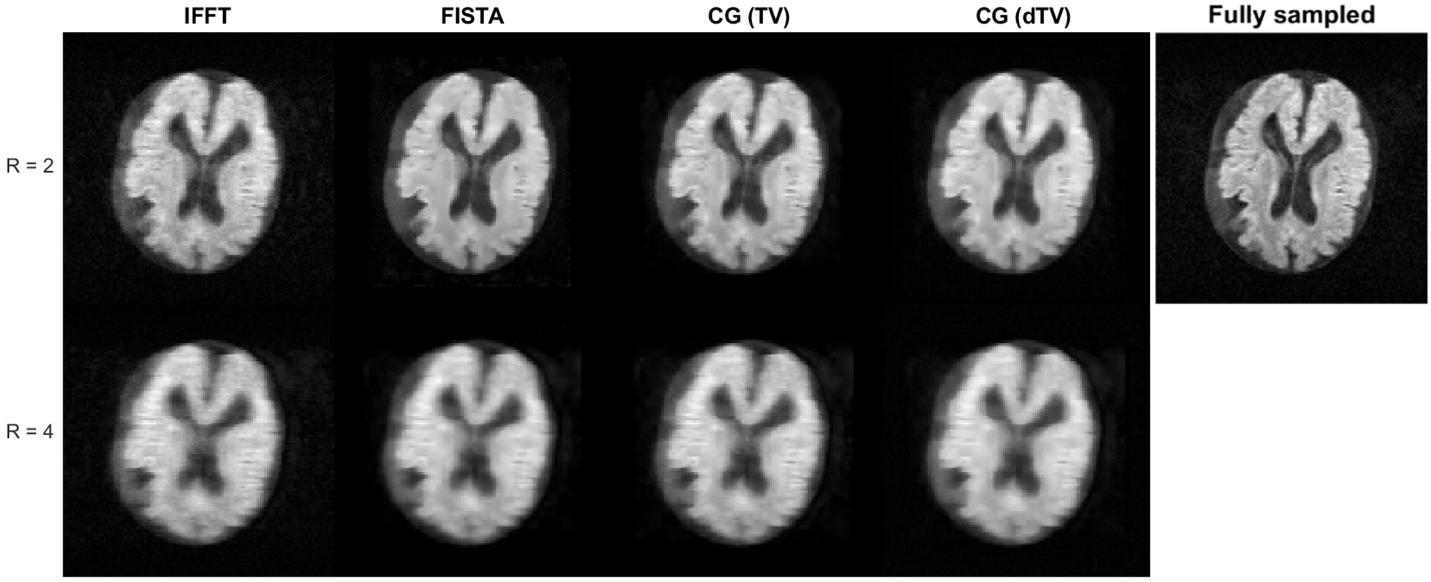

Figure 5.9: A slice of three-dimensional postmortem brain reconstructions of the different algorithms for different sampling ratios. Rows from top to bottom: reconstructions for R = 2, R = 4. Columns from left to right: IFFT, FISTA, CG method with isotropic TV, CG method with directional TV reconstructions. The fully sampled image is shown to the right. Data acquired using a TSE sequence with the following scan parameters: TR/TE: 1200/18 ms, echo train length: 6.

Table 5.5: Reconstruction times for the FISTA and both CG method reconstructions of the postmortem brain data for acceleration factors R = 2 and 4.

Acceleration factor	Reconstruction method	Reconstruction time (s)
2	FISTA	16.26
	CG (TV)	69.28
	CG (dTV)	**7.76**
4	FISTA	18.00
	CG (TV)	115.96
	CG (dTV)	**15.61**

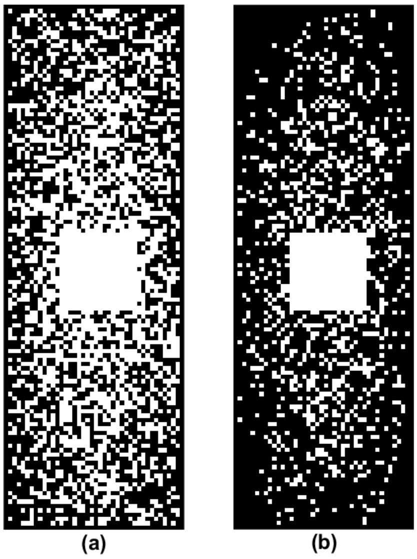

Figure 5.10: Phase-encoding plane of k-space undersampling patterns for the postmortem brain scans with 5% sampling in the center for R = 2 (a) and R = 4 (b). The white points represent the sampled lines.

5.5. In-Vivo Scans

Figure 5.11 shows slices of both fully sampled and 50% undersampled in-vivo brain slices and the CG (TV) reconstructions. We can see that the fully sampled scan already has low SNR, also lower compared with the fully sampled brain phantom or postmortem brain scan, thus requiring a denoising algorithm. The reconstruction images of the fully sampled data are well denoised and most structures which were present in the IFFT images are well preserved. The images from the undersampled data clearly have lower SNR, thus some of the details are already lost before our reconstruction algorithm. Nevertheless, much of the structure that did remain in the IFFT are also still present in the CG (TV) reconstructions. We also see that in both the fully sampled and undersampled cases, there is still some noise left at some spots just outside the head. Generally, the noise level was already relatively high at these points in the IFFT images, such that the noise did not get removed but it rather got smoothed out. This is more severe in the undersampled case as the SNR is lower due to the undersampling.

Figure 5.12 shows the phase-encoding plane of the k-space sampling pattern for the undersampled scan. In this case, a rectangle was in implemented in the center of k-space instead of the square in the previous scans. The rectangle is scaled with the dimensions of the phase-encoding plane. The image domain mask is shown in Fig. D.7.

Table 5.6: Reconstruction times for the CG (TV) method reconstructions of the in-vivo data for acceleration factors R = 1 and 2.

Acceleration factor	Reconstruction method	Reconstruction time (s)
1	CG (TV)	38.16
2	CG (TV)	79.59

Both the FISTA and CG (dTV) method do not perform well in this experiment. It turns out that the adaptively computed λ in the first iteration of FISTA is too large as is shown in Fig. D.8. The threshold level for the shrinkage operation is larger than approximately 99.7% of the wavelet coefficients. This means that 99.7% of coefficients are set to 0 after the first iteration thereby destroying too much significant information. The image after 1 iteration of FISTA for the undersampled in-vivo data set is shown in Fig. D.9 in Appendix D.

The CG (dTV) method also diverges after a couple of iterations. We hypothesize that the SNR in the in-vivo data sets is too low to accurately extract the directional edge information used in the CG (dTV) method, but more testing is required to verify our hypothesis. The resulting images for the first 9 iterations of the CG (dTV) method are shown in Fig. D.10.

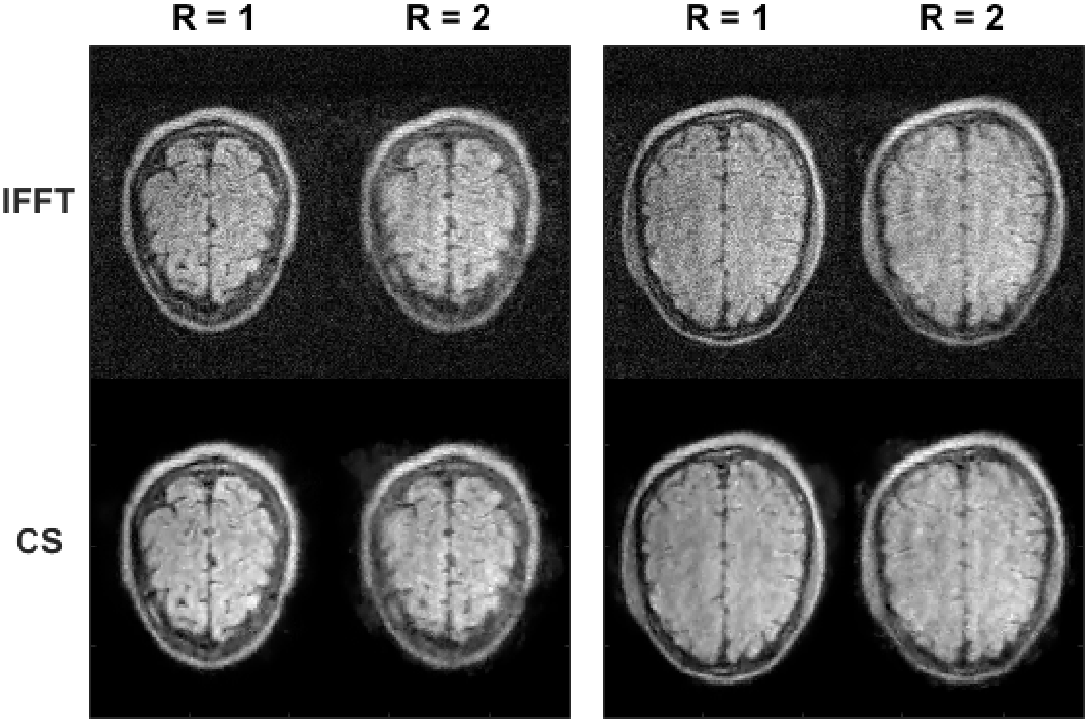

Figure 5.11: A slice of fully sampled and undersampled three-dimensional in-vivo reconstructions. Rows from top to bottom: reconstructions for R = 1, R = 2. Columns from left to right: IFFT, CG method with isotropic TV reconstructions. Data acquired using a TSE sequence with the following scan parameters: TR/TE: 700/20 ms, echo train length: 6.

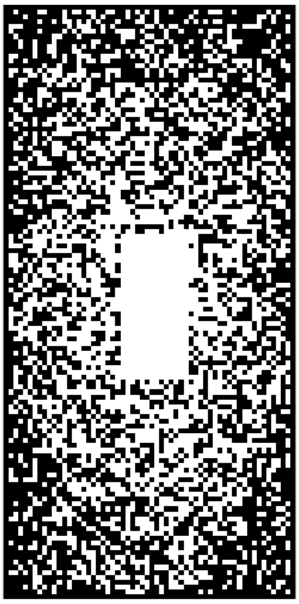

Figure 5.12: Phase-encoding plane of k-space undersampling pattern for the undersampled in-vivo scan with 5% sampling in the center. The white points represent the sampled lines.

6 Conclusion

Low-field MRI is a great low-cost, lightweight, portable alternative to the conventional high-field MRI units. High-field MRI scanners use various techniques, like parallel imaging and compressed sensing, to accelerate scantime, but these were not yet present at the low-field scanner at the LUMC. Compressed sensing is an attractive technique as it does not require any additional hardware, yet it may lead to significant acceleration of scantime by undersampling k-space and using an iterative algorithm for reconstruction. A popular reconstruction approach is to use convex optimization techniques to minimize an additively regularized cost function. A downside of convex optimization techniques used during compressed sensing reconstruction is the fact that we need a number of free parameters to converge to an acceptable reconstruction. As the optimal parameter values are not known a priori and differ for each scan we want to avoid choosing these parameters as much as possible. In this thesis we thus tried to answer the following research question:

How can we make a compressed sensing pipeline for the LUMC low-field MRI system which requires minimal human interaction?

To undersample k-space we implemented a Gaussian k-space undersampling mask for Cartesian k-space trajectories. With this mask we sample the phase-encoding lines according to a Gaussian distribution. The mask is implemented in the pulse sequencing program of the scanner such that the user only has to specify the percentage of undersampling. The user is also able to specify a percentage of k-space to deterministically sample in the center of k-space which holds most of the signal energy. The option to use CS during a scan, the undersampling percentage and the percentage sampled in the center are incorporated in the existing GUI.

The undersampled k-space data are passed to MATLAB for reconstruction where we first create an image domain mask, which shrinks the solution space thereby aiding our reconstruction algorithms. This image domain mask is created by thresholding based on the estimated noise level and performing a sequence of morphological operations. All operations used in creating the image domain mask have computationally low-complexity such that creating the mask does not add significant computational time to our reconstruction process.

Both the undersampled k-space data and image domain mask are passed to the actual reconstruction algorithm. The cost function conventionally used in CS-MRI which uses additive TV and wavelet l_1-norm regularization were rewritten using multiplicative regularization functions. We first implemented the TV regularization as a multiplicative weighted l_2-norm regularization function, while leaving the wavelet regularization as an additive l_1-norm which still uses a single artificial regularization parameter. To reconstruct this new cost function we have used FISTA with backtracking and an adaptive linesearch method based on SURE to estimate the optimal regularization parameter for each iteration. We have also rewritten the additive wavelet function to a multiplicative weighted l_2-norm function such that we have a fully multiplicative cost function. The fully multiplicative cost function is reconstructed using a CG method. Additionally, we incorporated directional TV in the fully multiplicative function by iteratively projecting the image gradient onto its previous iteration.

We have used the abovementioned methods to undersample and reconstruct both a simulated image and multiple real MRI scans of a phantom, a postmortem brain and a healthy volunteer. In the end, the CG method with isotropic TV implementation proved to be the most robust reconstruction method proposed in this thesis, reconstructing typically sized three-dimensional datasets in less than 2 minutes.

6.1. Future Work

There are a number of topics which could be further investigated. We make a number of recommendations which are either related to testing of the current implementations, or related to more theoretical work.

6.1.1. Recommendations for Testing

Optimal free parameters Each algorithm still has a few free parameters left, like the stopping tolerance level, for which the optimal values are also scan specific. If certain scans are to be performed more often in the future we may be able to optimize the remaining parameters for these scans, or at least limit the search range for the free parameters.

k-space undersampling Scans with different acceleration factors are performed in this thesis, but we can test for each type of experiment, e.g. the brain phantom or in-vivo scans, how far we can go in accelerating these scans. Also, the full sampling in k-space center and the variance of the Gaussian distribution may be varied and assessed based on the trade-off between SNR and details. Moreover, non-Cartesian trajectories and other sampling distributions may be tested as well.

Comparison with state-of-the-art This report did not include a comparison with a benchmark state-of-the-art algorithm. This could be done by implementing for example FISTA, ADMM, or split-Bregman on the fully additively regularized function, i.e. data-mismatch term with an additive TV function and an additive wavelet function. We then have to tune more parameters ourselves which adds to the post-processing time and should be taken into account when comparing the results.

Other applications In this thesis we have investigated the possibility of using mixed additively multiplicatively regularized and multiple multiplicatively regularized functions with the application of compressed sensing reconstruction and pure denoising of MR images. It may be interesting to see how these techniques perform (against state-of-the-art algorithms) on other types of images/data, or even other general image processing problems like deblurring or image completion.

6.1.2. Recommendations on Theoretical Work

δ-parameter throughout the whole thesis we have worked with the δ-parameter of the form chosen in [18]. Other forms may be analyzed as well, such as the ones introduced in [16, 17].

Incorporate different priors In this thesis we incorporated (directional) TV and wavelet regularization. Other implementations of both TV and wavelet regularization could be explored as well, such as the dual-tree complex wavelet transform instead of the standard wavelet transform, or other implementations of directional TV such as total generalized variation. Most of the more sophisticated implementations do require more computations or more free parameters, so a trade-off vs improved performance must be made. Next to TV/wavelet regularization we could incorporate parallel imaging or motion correction.

Preconditioning A preconditioning step can be added to the algorithms to reduce the number of iterations during reconstruction. The computation of the preconditioner should be efficient such that the reduced number of iterations outweights the computation of the preconditioner.

Adaptive λ computation in FISTA In the adaptive FISTA for the multiplicative TV additive wavelet regularized cost function we attempt to compute an optimal λ each iteration using SURE. Instead it might be possible to modify and use some of the post-processing methods, e.g. the L-curve method, generalized cross-validation, or unbiased predictive risk estimation, to adaptively compute lambda.

Bibliography

[1] R. W. Brown et al., eds. *Magnetic resonance imaging : physical principles and sequence design*. Second. New Developments in NMR. Wiley-Blackwell, 2014.

[2] T. O'Reilly, W.M. Teeuwisse, and A.G. Webb. "Three-dimensional MRI in a homogenous 27 cm diameter bore Halbach array magnet". In: *Journal of Magnetic Resonance* 307 (2019), p. 106578.

[3] T O'Reilly et al. "In vivo 3D brain and extremity MRI at 50 mT using a permanent magnet Halbach array". In: *Magnetic Resonance in Medicine* 85.1 (2021), pp. 495–505.

[4] K. Halbach. "Design of permanent multipole magnets with oriented rare earth cobalt material". In: *Nuclear Instruments and Methods* 169.1 (1980), pp. 1–10.

[5] M.L. Johns et al., eds. *Mobile NMR and MRI. Developments and Applications*. New Developments in NMR. The Royal Society of Chemistry, 2016, P001–352.

[6] H. Raich and P. Blümler. "Design and construction of a dipolar Halbach array with a homogeneous field from identical bar magnets: NMR Mandhalas". In: *Concepts in Magnetic Resonance Part B: Magnetic Resonance Engineering* 23B.1 (2004), pp. 16–25.

[7] B. de Vos et al. "Design, Characterisation and Performance of an Improved Portable and Sustainable Low-Field MRI System". In: *Frontiers in Physics* 9 (2021).

[8] M. Sarracanie and N. Salameh. "Low-Field MRI: How Low Can We Go? A Fresh View on an Old Debate". In: *Frontiers in Physics* 8 (2020).

[9] D.I. Hoult and R.E. Richards. "The signal-to-noise ratio of the nuclear magnetic resonance experiment". In: *Journal of Magnetic Resonance* 213.2 (2011). Magnetic Moments, pp. 329–343.

[10] J.P. Marques, F.F.J. Simonis, and A.G. Webb. "Low-field MRI: An MR physics perspective". In: *Journal of Magnetic Resonance Imaging* 49.6 (2019), pp. 1528–1542.

[11] K. P. Pruessmann et al. "SENSE: sensitivity encoding for fast MRI". In: *Magnetic resonance in medicine* 42.5 (1999), pp. 952–962.

[12] E.J. Candes, J. Romberg, and T. Tao. "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information". In: *IEEE Transactions on Information Theory* 52.2 (2006), pp. 489–509.

[13] M. Lustig, D.L. Donoho, and J.M. Pauly. "Sparse MRI: The application of compressed sensing for rapid MR imaging". In: *Magn. Reson. in Med.* 58.6 (2007), pp. 1182–1195.

[14] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[15] B. Deka and S. Datta. *Compressed Sensing Magnetic Resonance Imaging Reconstruction Algorithms*. Springer Series on Bio- and Neurosystems. Springer, 2019.

[16] A. Abubakar and P.M. van den Berg. "A multiplicative weighted L2-norm total variation regularization for deblurring algorithms". In: *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 4. 2002, pp. IV-3545-IV–3548.

[17] P.M. van den Berg, A. Abubakar, and J.T. Fokkema. "Multiplicative Regularization for Constrast Profile Inversion". In: *Radio Science* 38.2 (2003).

[18] M.L. de Leeuw den Bouter, M.B. van Gijzen, and R.F. Remis. "Low-field magnetic resonance imaging using multiplicative regularization". In: *Magnetic Resonance Imaging* 75 (2021), pp. 21–33.

[19] Y. Zhang et al. "Energy Preserved Sampling for Compressed Sensing MRI". In: *Computational and Mathematical Methods in Medicine* (2014).

[20] E.J. Candes and M.B. Wakin. "An Introduction To Compressive Sampling". In: *IEEE Signal Process. Mag.* 25.2 (2008), pp. 21–30.

[21] L.I. Rudin, S. Osher, and E. Fatemi. “Nonlinear total variation based noise removal algorithms”. In: *Physica D: Nonlinear Phenomena* 60.1 (1992), pp. 259–268.

[22] L. Condat. “Discrete Total Variation: New Definition and Minimization”. In: *SIAM Journal on Imaging Sciences* 10.3 (2017), pp. 1258–1290.

[23] I. Daubechies. *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics, 1992.

[24] N.R. Pal and S.K. Pal. “A Review on Image Segmentation Techniques”. In: *Pattern Recognition* 26.9 (1993), pp. 1277–1294.

[25] I.M. Johnstone and B.W. Silverman. “EbayesThresh: R Programs for Empirical Bayes Thresholding”. In: *Journal of Statistical Software* 12.8 (2005), pp. 1–38.

[26] P. Soille. *Morphological Image Analysis. Principles and Applications*. Second. Springer, 2004.

[27] V. Estellers, S. Soatto, and X. Bresson. “Adaptive Regularization with the Structure Tensor”. In: *IEEE Transactions on Image Processing* 24.6 (2015), pp. 1777–1790.

[28] Stamatios Lefkimmiatis et al. “Convex Generalizations of Total Variation Based on the Structure Tensor with Applications to Inverse Problems”. In: *Scale Space and Variational Methods in Computer Vision*. Springer Berlin Heidelberg, 2013, pp. 48–60.

[29] M.J. Ehrhardt and M.M. Betcke. “Multicontrast MRI Reconstruction with Structure-Guided Total Variation”. In: *SIAM Journal on Imaging Sciences* 9.3 (2016), pp. 1084–1106.

[30] S. Boyd et al. “Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers”. In: *Foundations and Trends in Machine Learning* 3.1 (2010), pp. 1–122.

[31] T. Goldstein and S. Osher. “The Split Bregman Method for L1-Regularized Problems”. In: *SIAM Journal on Imaging Sciences* 2.2 (2009), pp. 323–343.

[32] A. Beck and M. Teboulle. “A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems”. In: *SIAM Journal on Imaging Sciences* 2.1 (2009), pp. 183–202.

[33] K. Miller. “Least Squares Methods for Ill-Posed Problems with a Prescribed Bound”. In: *Siam Journal on Mathematical Analysis* 1.1 (1970), pp. 52–74.

[34] V.A. Morozov. *Methods for Solving Incorrectly Posed Problems*. Springer, 1984.

[35] G.H. Golub and U von Matt. “Generalized Cross-Validation for Large-Scale Problems”. In: *Journal of Computational and Graphical Statistics* 6.1 (1997), pp. 1–34.

[36] D.L. Donoho and I.M. Johnstone. “Adapting to Unknown Smoothness via Wavelet Shrinkage”. In: *Journal of the American Statistical Association* 90.432 (1995), pp. 1200–1224.

[37] J. Nocedal and S.J. Wright. *Numerical Optimization*. Second. Springer Series in Operations Research. Springer, 2006.

[38] L. A. Shepp and B. F. Logan. “The Fourier reconstruction of a head section”. In: *IEEE Transactions on Nuclear Science* 21.3 (1974), pp. 21–43.

[39] Z. Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612.

[40] N. Parikh and S. Boyd. “Proximal Algorithms”. In: *Foundations and Trends in Optimization* 1.3 (2013), pp. 123–231.

A

k-Space Mask

This appendix discusses the details of creating a k-space undersampling mask.

When the number of phase encoding steps N_{pe1} and N_{pe2} are known, a grid of equal number of points can be constructed ranging from -1 to 1 in both directions with N_{pe1} and N_{pe2} steps in their corresponding directions. First, with this grid the desired k-space center points are computed from their Euclidean distance from the center. So, if we define c as the portion of the grid which get sampled in the center, then all points (n_{pe1}, n_{pe2}) for which $\sqrt{n_{pe1}^2 + n_{pe2}^2} \leq c$ holds, are sampled. Next, all points in the grid are assigned a value following the Gaussian distribution. The key idea is to add or subtract a constant to all points and create a grid of standard uniform random variables which gets subtracted from the Gaussian distributed grid. Each point which stays positive after subtraction gets sampled. The constant is chosen such that the right sampling ratio r is achieved. To compute the constant, the bisection method is used [13]. For a grid p with points $p_{i,j}$ where i and j are integers in the range $[1, N_{pe1}]$ and $[1, N_{pe2}]$, respectively, the bisection method is as follows

0. Choose scalars x and y such that $x < y$ and the desired constant z^* will be in the interval $[x, y]$.
1. Add $z = (x + y)/2$ to all $p_{i,j}$.
2. Set all $p_{i,j}$ which are smaller than 0 or larger than 1, equal to 0 or 1, respectively and set all centerpoints equal to 1.
3. If the rounded sum of all $p_{i,j}$, $\lfloor \sum p \rfloor$ where $\lfloor \cdot \rfloor$ is the floor function, is larger than the desired number of samples $rN_{pe1}N_{pe2}$, then $y = z$. Otherwise, $x = z$. In this way, the interval to search over gets halved.
4. Repeat steps 1-3 until $\lfloor \sum p \rfloor = rN_{pe1}N_{pe2}$.

After the bisection method for determining the grid values of grid p an equal sized grid is created with each point sampled from a standard uniform distribution. The new grid values get subtracted from p. All positive values get sampled and all negative values will be left out. Two additional checks can be performed to refine the final mask. We can check whether the final number of sampled points is close enough to the desired number of samples and we can check whether large parts of the phase encoding plane are not skipped, such that we do not have large ”holes” in our phase encoding plane.

B

Morphological Operations

This appendix gives a more detailed explanation on the morphological operations used in this thesis and the exact sequence of morphological operations used in the image domain masking algorithm.

The definitions and examples below again apply to 2D binary images, but morphological operations can also be defined for 3D images as well as greyscale images. The two fundamental operations are dilation and erosion and by combining these operations sequentially, new operations are defined, such as closing and opening.

With dilation, objects in the image get expanded. The dilation operation on a binary input image X with a binary structuring element Y is defined as

$$X \oplus Y = \bigcup_{y \in \mathcal{D}_Y} X_y = Z \,, \tag{B.1}$$

where $\mathcal{D}_Y$ is the domain of the structuring element, X_y is the image X translated by y, and Z is the dilated output image. This means that for each pixel x_{ij} in X, the structuring element Y is placed on top of it and if the structuring element overlaps with a part of the object, the corresponding output pixel z_{ij} is set to 1. The result is that objects get expanded at their boundaries and small holes within the object are filled, as is shown in Fig. B.1.c.

Erosion is the opposite of dilation as it shrinks objects in the image. The erosion operation can be written as

$$X \ominus Y = \{Z | Y_z \subseteq X\} \,. \tag{B.2}$$

The pixels z_{ij} in the eroded output image Z which are set to 1 correspond to the pixels x_{ij} on which the center of the structuring element can be overlaid such that the whole structuring element sits in the object. An example using the same input image and structuring element as before, is shown in Fig. B.1.d. The erosion operation removes any isolated or thin lines and only larger structures will remain which will also appear smaller.

Closing is the sequential operation of dilation followed by erosion and can be written as

$$X \bullet Y = (X \oplus Y) \ominus Y = Z \,. \tag{B.3}$$

An example is shown in Fig. B.1.e. First, the dilation operation is applied which fills holes and enlarges the object. Next, the erosion operation is applied on the dilated image, which shrinks the object back to approximately the original size and shape. The result thus fills small holes within objects, while preserving the old size and shape.

Opening is the counterpart of closing, defined as

$$X \circ Y = (X \ominus Y) \oplus Y = Z \,. \tag{B.4}$$

Objects are first reduced in size and any isolated or smaller structures are removed. Then the image is dilated such that any remaining objects are approximately restored in size. An example is shown in Fig. B.1.f.

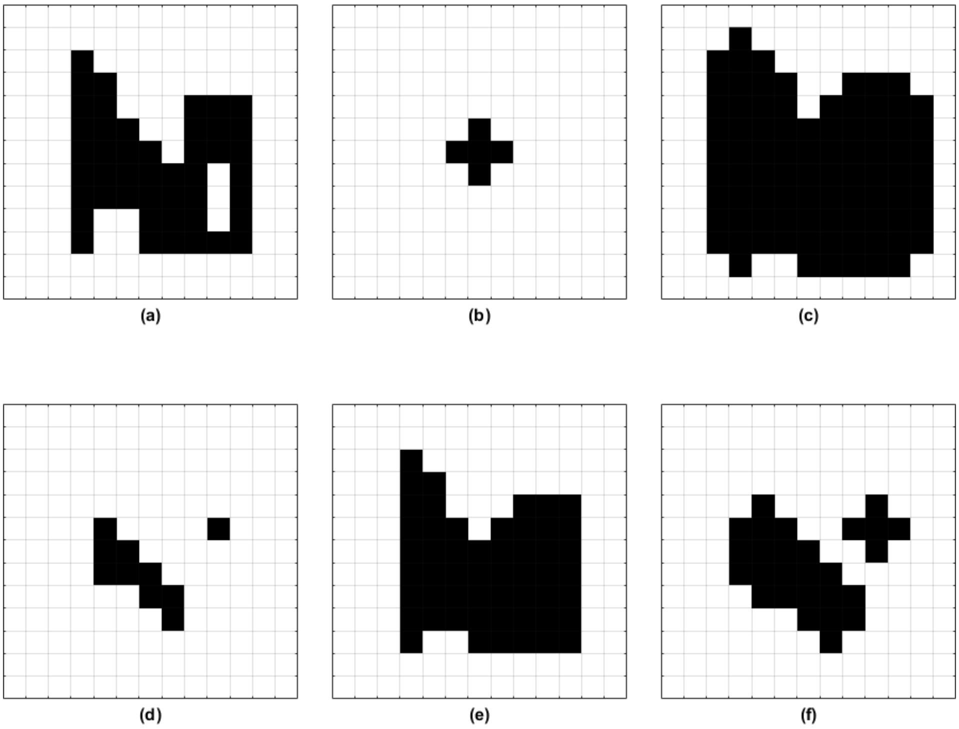

Figure B.1: Morphological operations on an input image with a diamond shaped structuring element. (a) input image, (b) structuring element, (c) dilated output image, (d) eroded output image, (e) closed output image, (f) opened output image.

As discussed in Section 3.2, a sequence of morphological operations, which we called $\mathrm{MORPH}(\cdot)$, is performed on the binary thresholded input image to obtain an image domain mask. In this sequence we use two diamond-shaped structuring elements. The first one has a 3×3 neighbourhood like in Fig. B.1.b and the second one has a 7×7 neighbourhood. We call these structuring elements SE1 and SE2, respectively. Let us call the input mask IN, the output mask OUT, and the morphological operations OPERATION$_{\mathrm{SE}}$. The sequence we use in our image domain masking algorithm, resulting from extensive numerical testing, is as follows

$$\begin{aligned} &\mathrm{IN} \Rightarrow \mathrm{CLOSE}_{\mathrm{SE2}} \Rightarrow \mathrm{FILL} \Rightarrow \mathrm{ERODE}_{\mathrm{SE2}} \Rightarrow \mathrm{CLOSE}_{\mathrm{SE1}} \ldots \\ &\quad \Rightarrow \mathrm{DILATE}_{\mathrm{SE2}} \Rightarrow \mathrm{FILL} \Rightarrow \mathrm{DILATE}_{\mathrm{SE2}} \Rightarrow \mathrm{FILL} \Rightarrow \mathrm{OUT}\,. \end{aligned} \tag{B.5}$$

C

FISTA

This appendix discusses the details of FISTA applied to our model using multiplicative total variation regularization and additive wavelet regularization which were left out of Chap. 4.

Derivations

For convenience we first restate some of the definitions. The function which we consider is of the form

$$F(\mathbf{x}) = F_1(\mathbf{x}) + F_2(\mathbf{x}) \ , \tag{C.1}$$

where $F_1(\mathbf{x})$ is a differentiable convex function with a Lipschitz continuous gradient $g_1(\mathbf{x})$ and $F_2(\mathbf{x})$ is a convex function which is possibly non-differentiable. The function of (P2) is of this form if we define

$$\begin{aligned} F_1(\mathbf{x}) &= \frac{\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{b}\|_2^2} \left(\delta^2_{\mathrm{TV},k-1} V_\Delta \mathbf{e}^T \mathcal{W}_{\mathrm{TV},k-1} \mathbf{e} + V_\Delta \mathbf{x}^H \mathbf{L}_\mathbf{w} \mathbf{x} \right) \ , \\ F_2(\mathbf{x}) &= \lambda \|\mathbf{W}\mathbf{x}\|_1 \ , \end{aligned} \tag{C.2}$$

where the gradient of $F_1(\mathbf{x})$ is

$$g_1(\mathbf{x}) = 2 \left[- \|\mathbf{b}\|_2^{-2} \mathbf{A}^H \mathbf{r}_{k-1} + V_\Delta F^{\mathrm{data}}(\mathbf{x}) \mathbf{L}_\mathbf{w} \mathbf{x} \right] \ , \tag{C.3}$$

where $\mathbf{r}_{k-1} = \mathbf{b} - \mathbf{A}^H \mathbf{x}_{k-1}$. The quadratic approximation $Q_L(\mathbf{x}, \mathbf{y})$ can be written as

$$Q_L(\mathbf{x}, \mathbf{y}) = F_1(\mathbf{y}) + (\mathbf{x} - \mathbf{y})^H g_1(\mathbf{y}) + \frac{L}{2} \|\mathbf{x} - \mathbf{y}\|_2^2 + F_2(\mathbf{x})) \ , \tag{C.4}$$

where L is the Lipschitz constant of g_1. The minimizer of $Q_L(\mathbf{x}, \mathbf{y})$ is the proximal operator $p_L(\mathbf{y})$ defined as

$$\begin{aligned} p_L(\mathbf{y}) &= \arg\min_{\mathbf{x}} \{ Q_L(\mathbf{x}, \mathbf{y}) \} \\ &= \arg\min_{\mathbf{x}} \left\{ F_1(\mathbf{y}) + (\mathbf{x} - \mathbf{y})^H g_1(\mathbf{y}) + \frac{L}{2} \|\mathbf{x} - \mathbf{y}\|_2^2 + F_2(\mathbf{x}) \right\} \ . \end{aligned} \tag{C.5}$$

The proximal operator is strongly convex, hence it has a unique minimum. We can replace $F_1(\mathbf{y})$ by $\frac{1}{2L}\left\|g_1(\mathbf{y})\right\|_2^2$ without changing the minimization solution as this term is independent of $\mathbf{x}$. This gives

$$\begin{aligned}
p_L(\mathbf{y}) &= \arg\min_{\mathbf{x}} \left\{ \frac{1}{2L}\left\|g_1(\mathbf{y})\right\|_2^2 + (\mathbf{x}-\mathbf{y})^H g_1(\mathbf{y}) + \frac{L}{2}\left\|\mathbf{x}-\mathbf{y}\right\|_2^2 + F_2(\mathbf{x}) \right\} \\
&= \arg\min_{\mathbf{x}} \left\{ \frac{L}{2}\frac{1}{L}(g_1(\mathbf{y}))^H \frac{1}{L} g_1(\mathbf{y}) \right. \\
&\left. + \frac{L}{2} 2(\mathbf{x}-\mathbf{y})^H \frac{1}{L} g_1(\mathbf{y}) + \frac{L}{2}(\mathbf{x}-\mathbf{y})^H(\mathbf{x}-\mathbf{y}) + F_2(\mathbf{x}) \right\} \\
&= \arg\min_{\mathbf{x}} \left\{ \frac{L}{2}(\mathbf{x}-\mathbf{y}+\frac{1}{L}g_1(\mathbf{y}))^H(\mathbf{x}-\mathbf{y}+\frac{1}{L}g_1(\mathbf{y})) + F_2(\mathbf{x}) \right\} \\
&= \arg\min_{\mathbf{x}} \left\{ \frac{L}{2}\left\|\mathbf{x} - \left(\mathbf{y} - \frac{1}{L}g_1(\mathbf{y})\right)\right\|_2^2 + F_2(\mathbf{x}) \right\} \; .
\end{aligned} \tag{C.6}$$

This term can be further reduced by making use of the shrinkage operator, which is defined as

$$\mathcal{T}_a(\mathbf{x})_i = (|x_i| - a)_+ \operatorname{sgn}(x_i) \; , \tag{C.7}$$

where a is a real constant, $(\cdot)_+ = \max\{0, \cdot\}$ and $\operatorname{sgn}(\cdot)$ is the sign function which can be defined for both real and complex numbers as

$$\operatorname{sgn}(z) = \begin{cases} \dfrac{z}{|z|} \text{ for } z \neq 0 \\ 0 \text{ for } z = 0 \end{cases} \; . \tag{C.8}$$

By substituting $F_2(\mathbf{x}) = \lambda\left\|\mathbf{x}\right\|_1$ into the last line of Eq. (C.6), the expression reduces to

$$\begin{aligned}
p_L(\mathbf{y}) &= \arg\min_{\mathbf{x}} \left\{ \frac{L}{2}\left\|\mathbf{x} - \left(\mathbf{y} - \frac{1}{L}g_1(\mathbf{y})\right)\right\|_2^2 + \lambda\left\|\mathbf{x}\right\|_1 \right\} \\
&= \mathcal{T}_{\lambda/L}\left(\mathbf{y} - \frac{1}{L}g_1(\mathbf{y})\right) \; .
\end{aligned} \tag{C.9}$$

However, we have $F_2(\mathbf{x}) = \lambda\left\|\mathbf{W}\mathbf{x}\right\|_1$. We can use the precomposition property of proximal operators [40], which states that if $f(\mathbf{x}) = \varphi(\mathbf{Q}\mathbf{x})$, where $\mathbf{Q}$ is an orthogonal or unitary matrix, then

$$p_{L;f}(\mathbf{x}) = \mathbf{Q}^H p_{L;\varphi}(\mathbf{Q}\mathbf{x}) \; . \tag{C.10}$$

We can use this property as $\mathbf{W}$ is unitary. We plug Eq. (C.9) into Eq. (C.10) and use the expression in Eq. (C.3) for $g_1(\mathbf{y})$ to get the final expression for the proximal operator

$$\begin{aligned}
p_L(\mathbf{y}) &= \mathbf{W}^H \mathcal{T}_{\lambda/L}\left(\mathbf{W}(\mathbf{y} - \frac{1}{L}g_1(\mathbf{y}))\right) \\
&= \mathbf{W}^H \mathcal{T}_{\lambda/L}\left(\mathbf{W}(\mathbf{y} - \frac{2}{L}\left[-\left\|\mathbf{b}\right\|_2^{-2}\mathbf{A}^H\mathbf{r}_{k-1} + V_\Delta F^{\text{data}}(\mathbf{y})\mathbf{L}_{\mathbf{w}}\mathbf{y}\right])\right) \; .
\end{aligned} \tag{C.11}$$

Lipschitz continuity of the gradient

The gradient function g_1 is Lipschitz continuous if

$$\left\|g_1(\mathbf{x}) - g_1(\mathbf{y})\right\|_2 \leq L\left\|\mathbf{x}-\mathbf{y}\right\|_2 \;, \forall\, \mathbf{x}, \mathbf{y} \in \mathbb{R}^N \; , \tag{C.12}$$

holds for a real finite value L [14]. Although we may not explicitly use the Lipschitz constant L when we use FISTA with backtracking, we still need to know whether F_1 has a Lipschitz continuous gradient.

Filling g_1 into the Lipschitz condition gives

$$\begin{aligned}
\|g_1(\mathbf{x}) - g_1(\mathbf{y})\|_2 &= \Bigg\| \left[2V_\Delta \frac{\|\mathbf{b}-\mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{b}\|_2^2} \mathbf{L}_\mathbf{w}\mathbf{x} - \frac{2}{\|\mathbf{b}\|_2^2}\mathbf{A}^H(\mathbf{b}-\mathbf{A}\mathbf{x}) \right] \\
&\quad - \left[2V_\Delta \frac{\|\mathbf{b}-\mathbf{A}\mathbf{y}\|_2^2}{\|\mathbf{b}\|_2^2} \mathbf{L}_\mathbf{w}\mathbf{y} - \frac{2}{\|\mathbf{b}\|_2^2}\mathbf{A}^H(\mathbf{b}-\mathbf{A}\mathbf{y}) \right] \Bigg\|_2 \\
&= \Bigg\| \left[2V_\Delta \frac{\|\mathbf{b}-\mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{b}\|_2^2} \mathbf{L}_\mathbf{w}\mathbf{x} + \frac{2}{\|\mathbf{b}\|_2^2}\mathbf{A}^H\mathbf{A}\mathbf{x} \right] \\
&\quad - \left[2V_\Delta \frac{\|\mathbf{b}-\mathbf{A}\mathbf{y}\|_2^2}{\|\mathbf{b}\|_2^2} \mathbf{L}_\mathbf{w}\mathbf{y} + \frac{2}{\|\mathbf{b}\|_2^2}\mathbf{A}^H\mathbf{A}\mathbf{y} \right] \Bigg\|_2 \\
&= \Bigg\| \frac{2V_\Delta}{\|\mathbf{b}\|_2^2} \left(\|\mathbf{b}-\mathbf{A}\mathbf{x}\|_2^2 \mathbf{L}_\mathbf{w}\mathbf{x} - \|\mathbf{b}-\mathbf{A}\mathbf{y}\|_2^2 \mathbf{L}_\mathbf{w}\mathbf{y} \right) \\
&\quad + \frac{2}{\|\mathbf{b}\|_2^2}\mathbf{A}^H\mathbf{A}(\mathbf{x}-\mathbf{y}) \Bigg\|_2 \ .
\end{aligned} \tag{C.13}$$

If we assume that $\|\mathbf{b}-\mathbf{A}\mathbf{x}\|_2$ is always smaller than $\|\mathbf{b}-\mathbf{0}\|_2 = \|\mathbf{b}\|_2$, which means that $\mathbf{A}\mathbf{x}$ is close to the data, we get the following inequality

$$\begin{aligned}
\|\nabla F_1(\mathbf{x}) - \nabla F_1(\mathbf{y})\|_2 &= \Bigg\| \frac{2V_\Delta}{\|\mathbf{b}\|_2^2} \left(\|\mathbf{b}-\mathbf{A}\mathbf{x}\|_2^2 \mathbf{L}_\mathbf{w}\mathbf{x} - \|\mathbf{b}-\mathbf{A}\mathbf{y}\|_2^2 \mathbf{L}_\mathbf{w}\mathbf{y} \right) \\
&\quad + \frac{2}{\|\mathbf{b}\|_2^2}\mathbf{A}^H\mathbf{A}(\mathbf{x}-\mathbf{y}) \Bigg\|_2 \\
&\le \Bigg\| \frac{2V_\Delta}{\|\mathbf{b}\|_2^2} \left(\|\mathbf{b}\|_2^2 \mathbf{L}_\mathbf{w}\mathbf{x} - \|\mathbf{b}\|_2^2 \mathbf{L}_\mathbf{w}\mathbf{y} \right) \\
&\quad + \frac{2}{\|\mathbf{b}\|_2^2}\mathbf{A}^H\mathbf{A}(\mathbf{x}-\mathbf{y}) \Bigg\|_2 \\
&= \left\| 2V_\Delta \mathbf{L}_\mathbf{w} (\mathbf{x}-\mathbf{y}) + \frac{2}{\|\mathbf{b}\|_2^2}\mathbf{A}^H\mathbf{A}(\mathbf{x}-\mathbf{y}) \right\|_2 \\
&= \left\| \left(2V_\Delta \mathbf{L}_\mathbf{w} + \frac{2}{\|\mathbf{b}\|_2^2}\mathbf{A}^H\mathbf{A} \right) (\mathbf{x}-\mathbf{y}) \right\|_2 \ .
\end{aligned} \tag{C.14}$$

By making use of the sub-multiplicative property of norms and the triangle inequality, we can further reduce the expression to

$$\begin{aligned}
\|\nabla F_1(\mathbf{x}) - \nabla F_1(\mathbf{y})\|_2 &= \left\| \left(2V_\Delta \mathbf{L}_\mathbf{w} + \frac{2}{\|\mathbf{b}\|_2^2}\mathbf{A}^H\mathbf{A} \right) (\mathbf{x}-\mathbf{y}) \right\|_2 \\
&\le \left\| 2V_\Delta \mathbf{L}_\mathbf{w} + \frac{2}{\|\mathbf{b}\|_2^2}\mathbf{A}^H\mathbf{A} \right\|_2 \|\mathbf{x}-\mathbf{y}\|_2 \qquad \text{(sub-mult.)} \\
&\le \left(\|2V_\Delta \mathbf{L}_\mathbf{w}\|_2 + \left\| \frac{2}{\|\mathbf{b}\|_2^2}\mathbf{A}^H\mathbf{A} \right\|_2 \right) \|\mathbf{x}-\mathbf{y}\|_2 \, \text{(triangle ineq.)} \\
&= \left(2V_\Delta \|\mathbf{L}_\mathbf{w}\|_2 + \frac{2}{\|\mathbf{b}\|_2^2} \|\mathbf{A}^H\mathbf{A}\|_2 \right) \|\mathbf{x}-\mathbf{y}\|_2 \ ,
\end{aligned} \tag{C.15}$$

which is of the form of Eq. (C.12). This gives an upperbound on L equal to

$$L \leq 2V_{\Delta} \left\|\mathbf{L}_{\mathbf{w}}\right\|_2 + \frac{2}{\left\|\mathbf{b}\right\|_2^2} \left\|\mathbf{A}^H\mathbf{A}\right\|_2 = 2V_{\Delta} \left\|\mathbf{L}_{\mathbf{w}}\right\|_2 + \frac{2}{\left\|\mathbf{b}\right\|_2^2} \lambda_{\max}\left(\mathbf{A}^H\mathbf{A}\right) \ , \tag{C.16}$$

where $\lambda_{\max}\left(\mathbf{A}^H\mathbf{A}\right)$ is the maximum eigenvalue of $\mathbf{A}^H\mathbf{A}$. This proves that F_1 has a Lipschitz continuous gradient and FISTA may be used.

D

Supplementary Results

This appendix shows supplementary results which were left out of Chap. 5.

D.1. Line Profiles of the Shepp-Logan Phantom Reconstructions

This section shows the remaining line profiles for the Shepp-Logan phantom reconstructions.

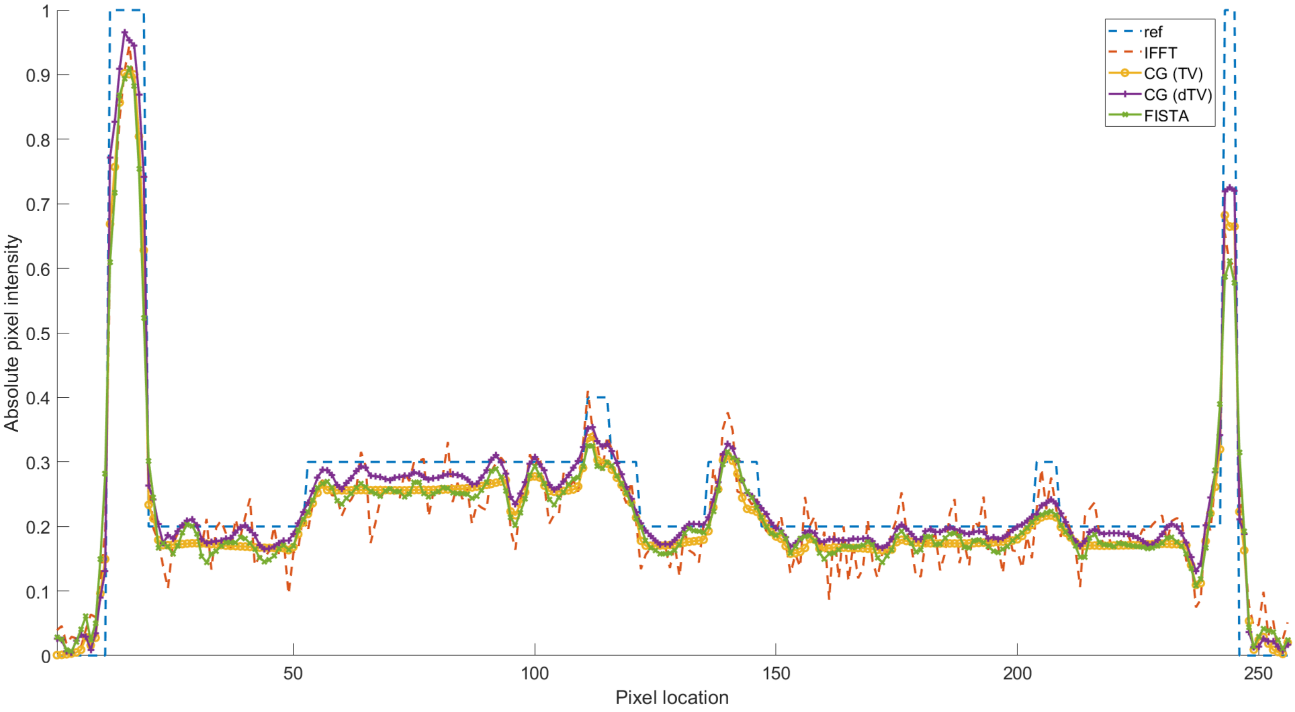

Figure D.1: Line profiles of the ground truth and the IFFT, CG (TV), CG (dTV), and FISTA reconstructions along the red line indicated on the ground truth image in Fig. 5.1 for R = 2.

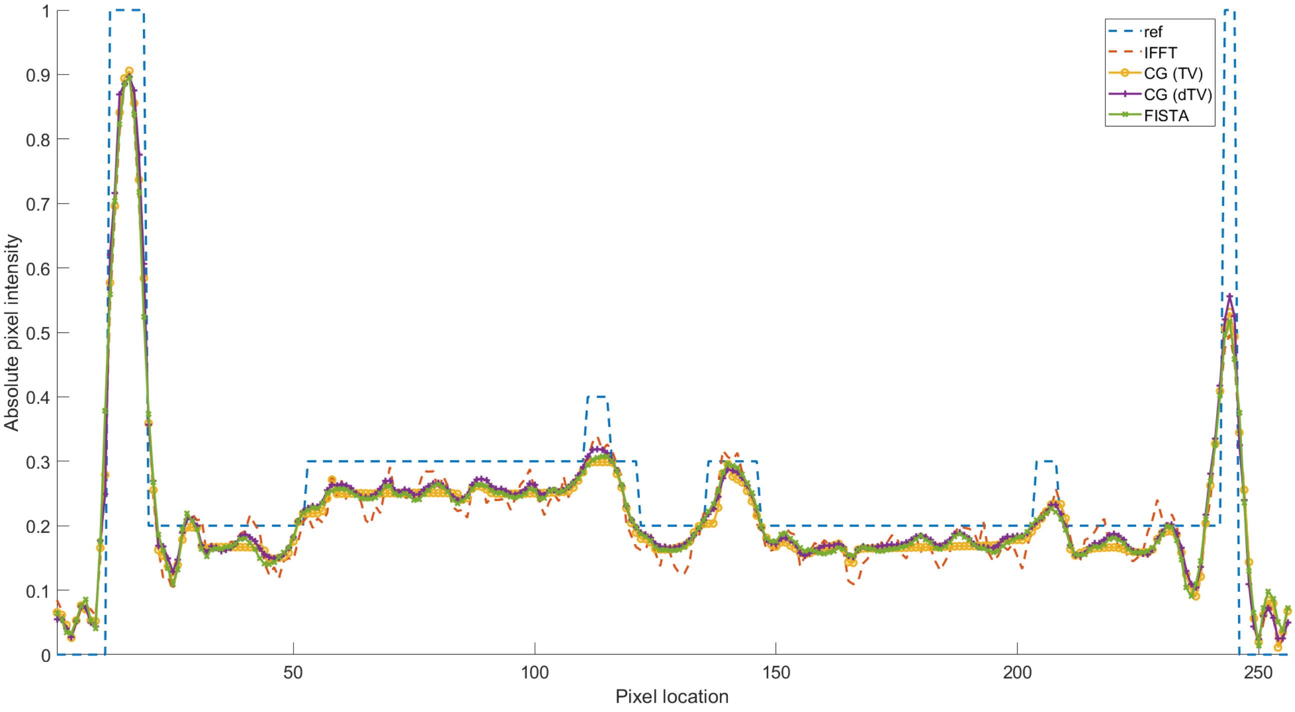

Figure D.2: Line profiles of the ground truth and the IFFT, CG (TV), CG (dTV), and FISTA reconstructions along the red line indicated on the ground truth image in Fig. 5.1 for R = 4.

D.2. Shepp-Logan Reconstructions vs Input SNR and Acceleration Factors

This section shows the IFFT, FISTA, CG (TV), and CG (dTV) reconstructions for the range of acceleration factors and input SNR values tested in Section 5.2.3.

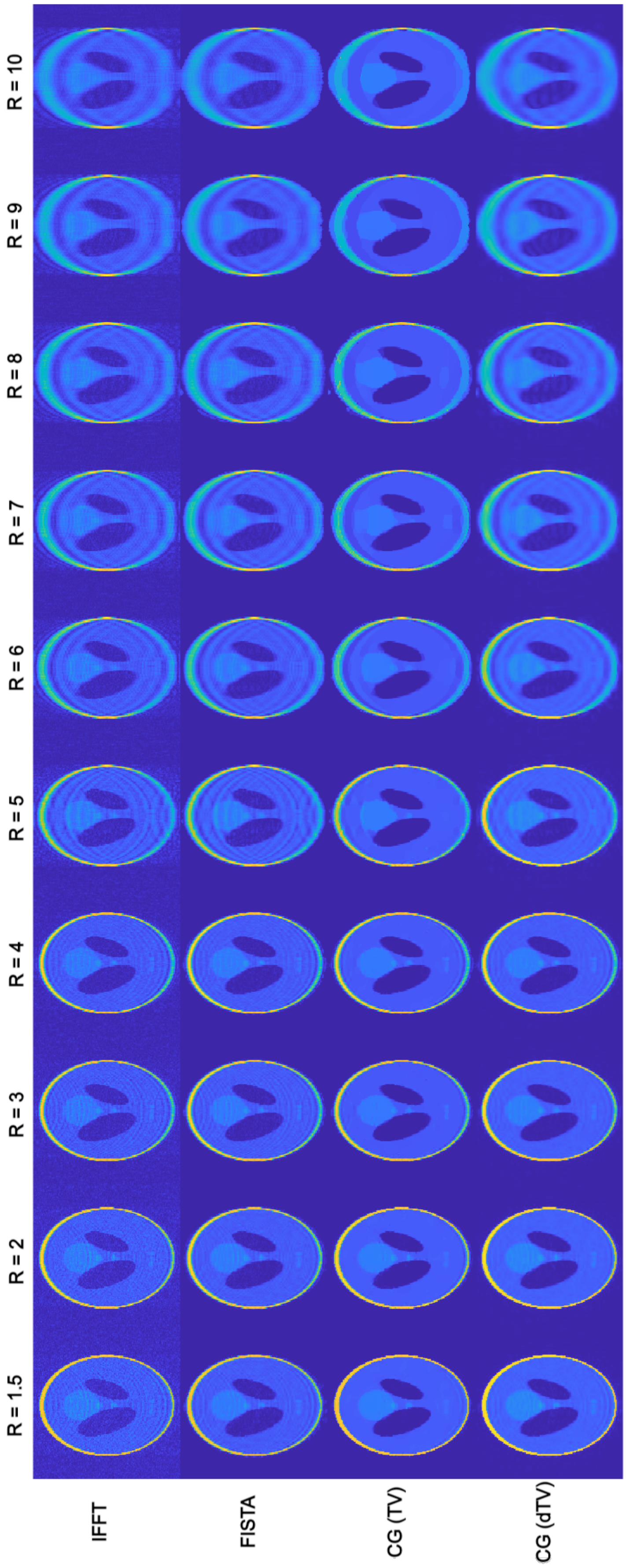

Figure D.3: Reconstructions vs different acceleration factors. Columns from left to right: acceleration factors from 1.5 to 10. Rows from top to bottom: IFFT, FISTA, CG (TV), CG (dTV) reconstructions.

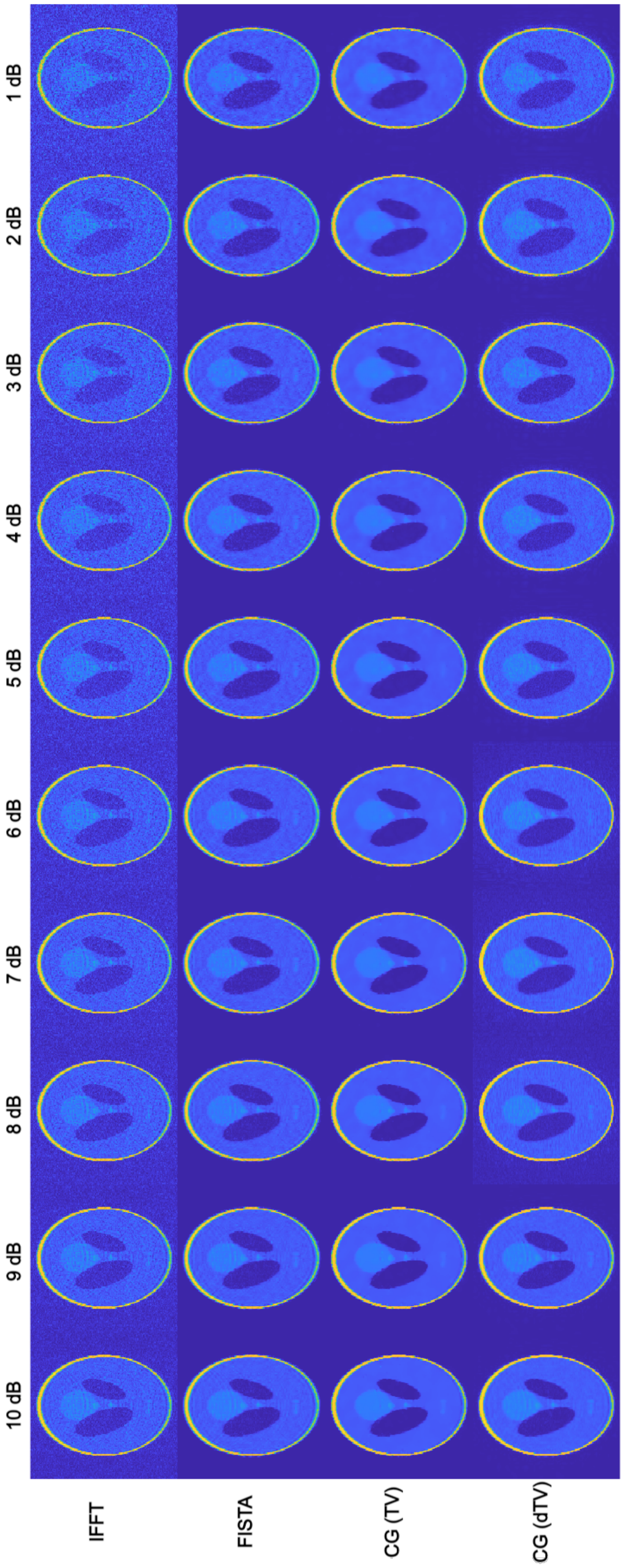

Figure D.4: Reconstructions vs different input SNR values. Columns from left to right: input SNR in dB from 10 dB to 1 dB. Rows from top to bottom: IFFT, FISTA, CG (TV), CG (dTV) reconstructions.

D.3. Image Domain Masks

This section shows the image domain masks of a selection of the experiments. We show an image domain mask for the 50% undersampled 2D brain phantom scan, slices of the 50% undersampled postmortem brain scan, and slices of the 50% undersampled in-vivo scan.

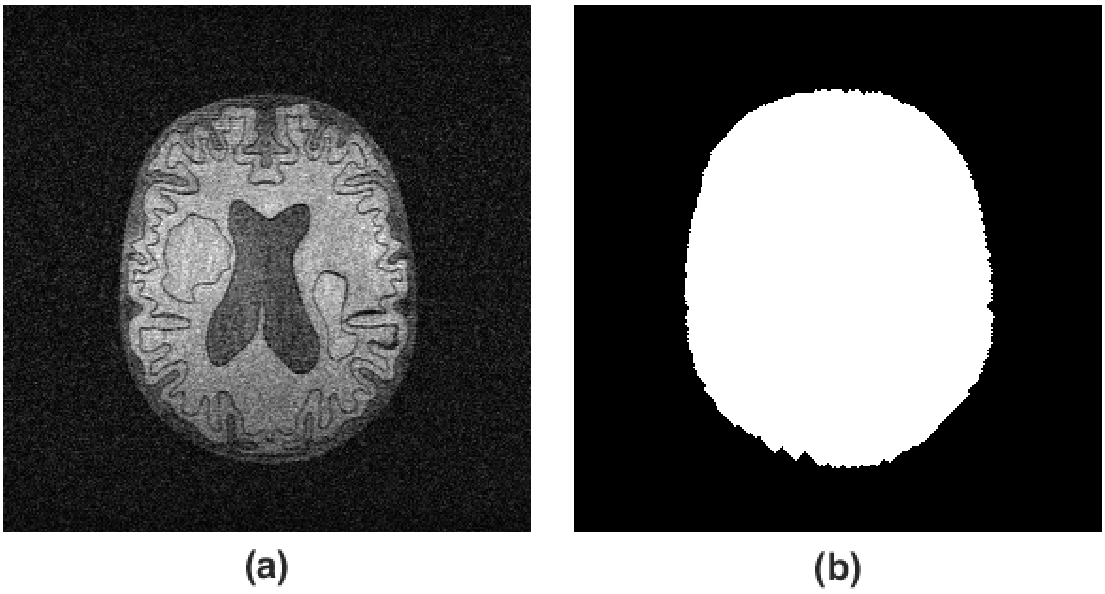

Figure D.5: (a) The 50% undersampled brain phantom and (b) its corresponding image domain mask.

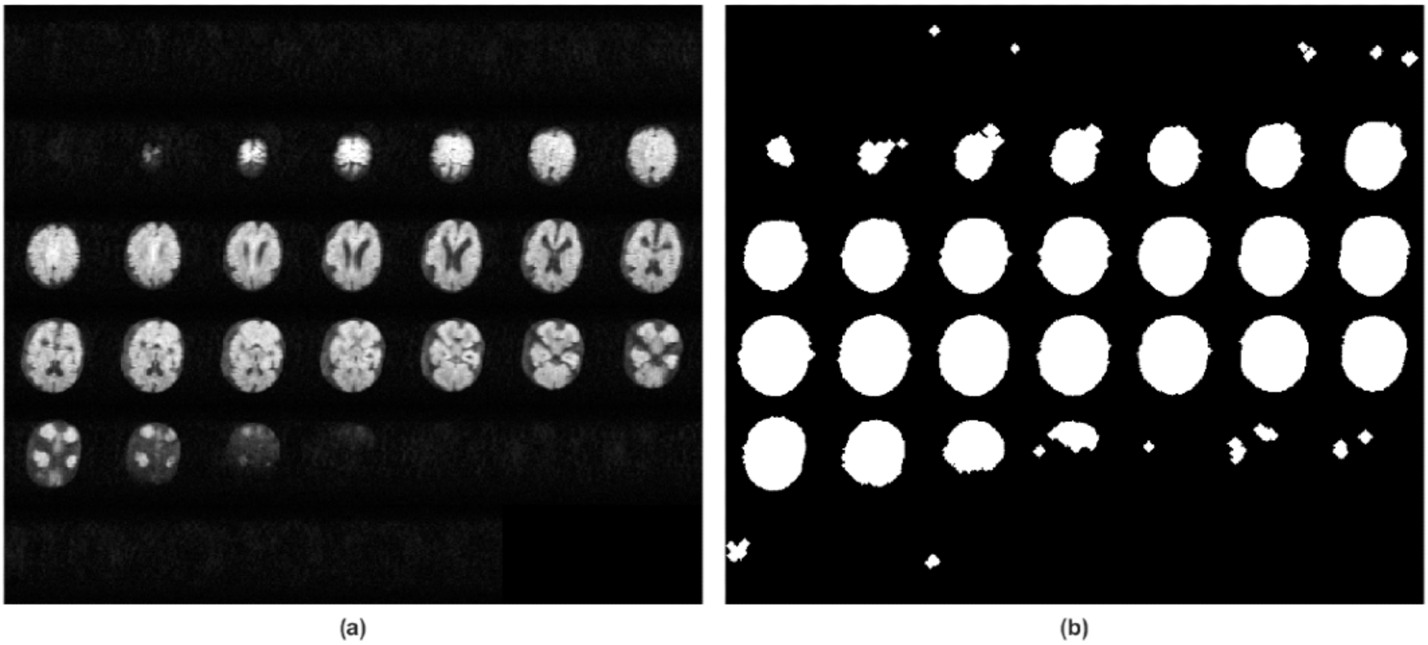

Figure D.6: (a) Slices of the 50% undersampled postmortem brain images and (b) slices of the corresponding image domain mask.

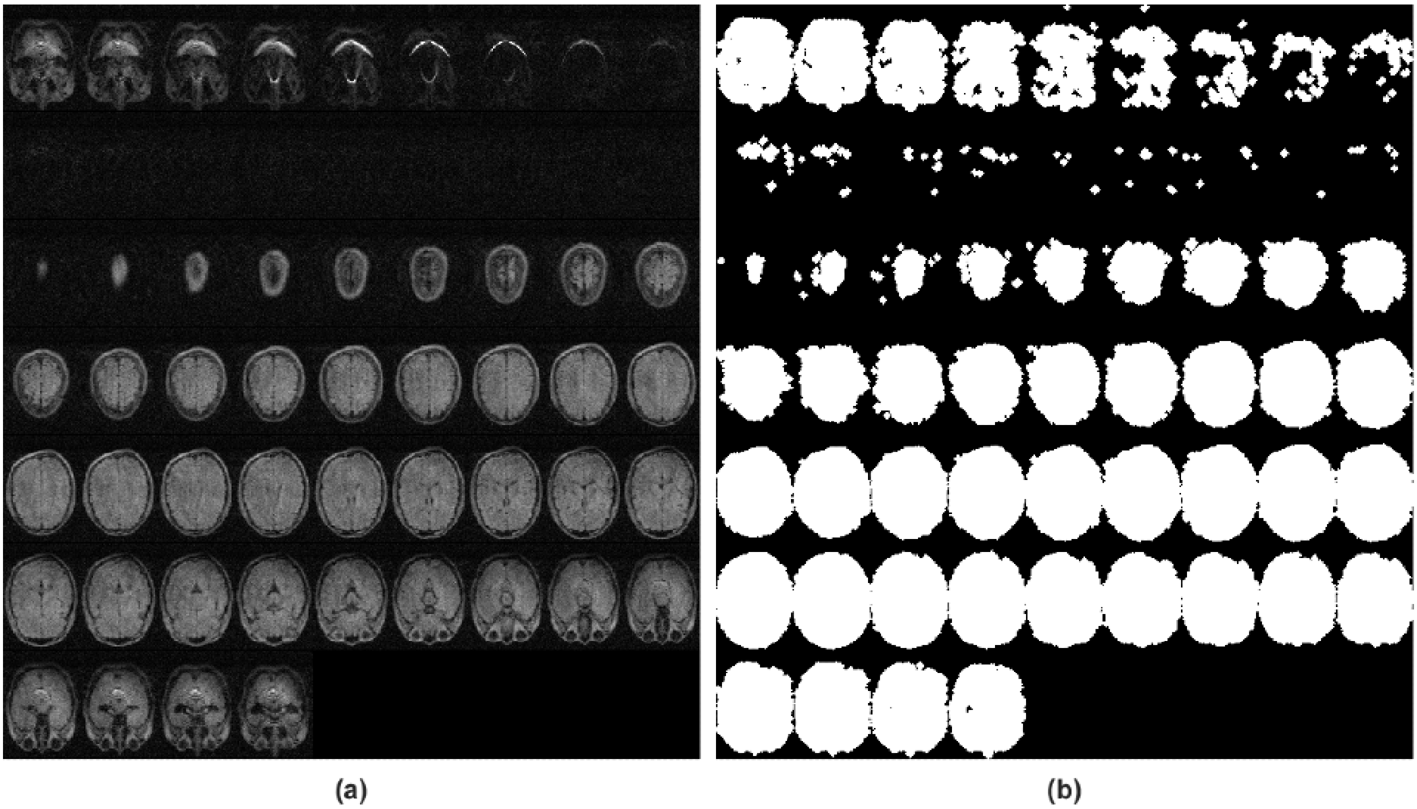

Figure D.7: (a) Slices of the 50% undersampled in-vivo images and (b) slices of the corresponding image domain mask.

D.4. FISTA and CG (dTV) On In-Vivo Data

This section shows the behaviour of the FISTA and CG (dTV) reconstruction methods for the in-vivo data. The thresholding in the first iteration of FISTA is too aggressive as indicated in Fig. D.8 and results in an image which has lost most details, as seen in Fig. D.9. The CG (dTV) slowly starts to diverge already in the first couple of iterations as shown in Fig. D.10.

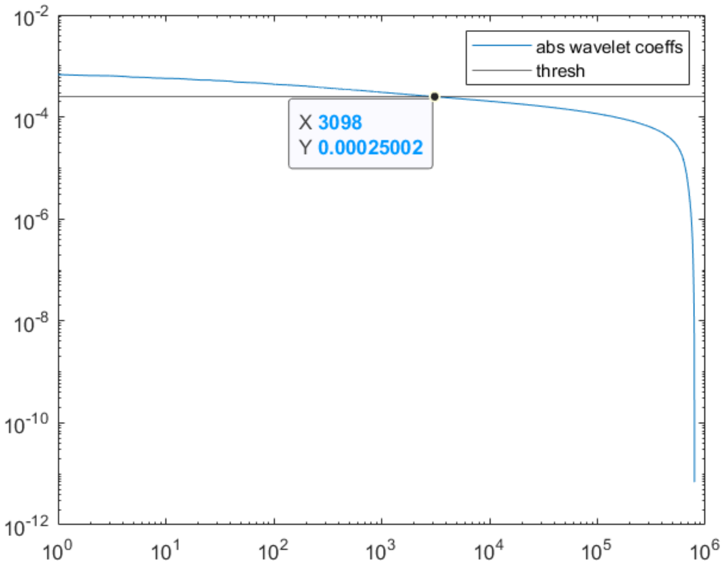

Figure D.8: The absolute value of the wavelet coefficients of the undersampled in-vivo image and the threshold level of the shrinkage operator of FISTA computed in the first iteration. Approximately 99.7% of the wavelet coefficients fall below the computed threshold.

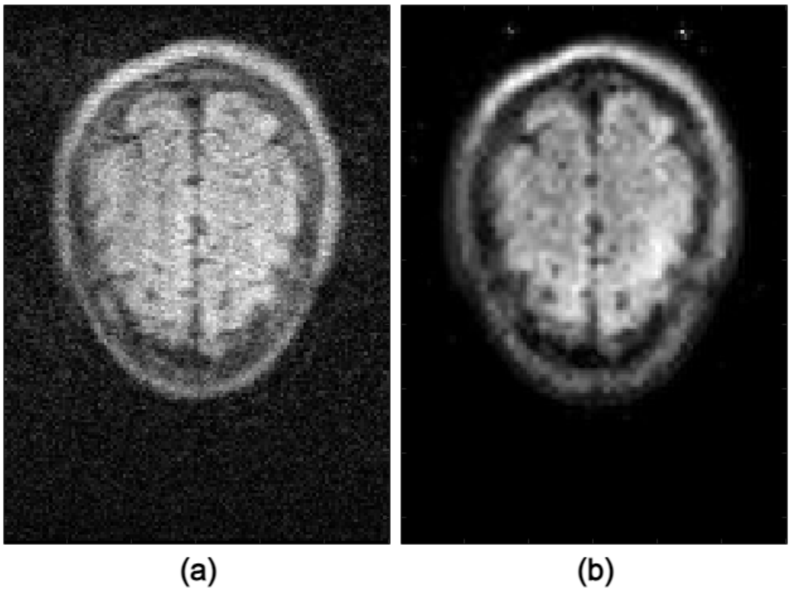

Figure D.9: (b) First iteration of the adaptive FISTA algorithm on the undersampled in-vivo data set (a).

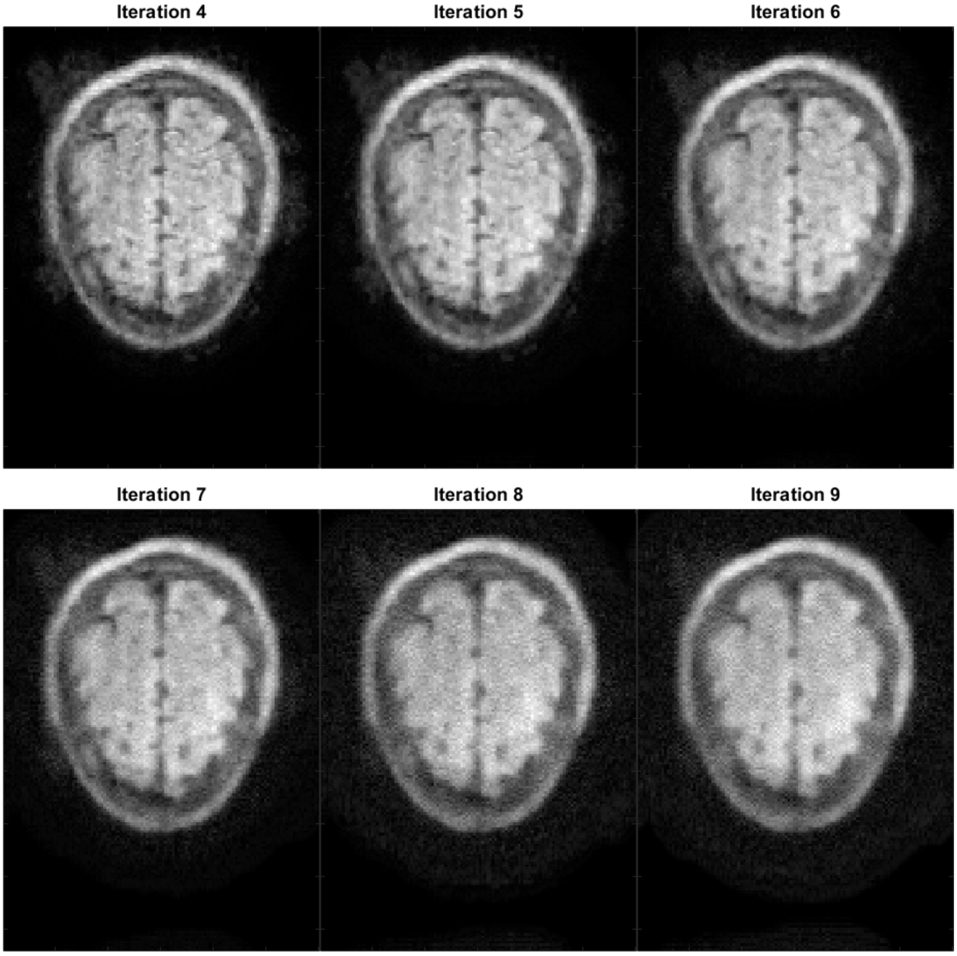

Figure D.10: Iterations 4 through 9 of the CG dTV method on the undersampled in-vivo data set exhibiting the diverging behaviour for this particular data set. The reconstruction continues to worsen as the algorithm advances.