Practical imaging and analyses for qMRI

by

Macy M. Payne

A.G.S., Front Range Community College, 2016
B.S., University of Northern Colorado, 2018

AN ABSTRACT OF A DISSERTATION

submitted in partial fulfillment of the requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Chemistry
College of Arts and Sciences

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2022

Abstract

Magnetic resonance imaging (MRI) is a powerful tool to gain insight into physiological structures and processes, molecular behavior, and psychological conditions. It has a broad range of applications, including, but not limited to, medical imaging, such as the insight it can provide towards medicinal chemistry regarding the appropriate targeting and efficiency of new drug candidates. While the fundamentals of MRI are thoroughly understood at this point of its development, the ability to extrapolate results beyond the qualitative data often seen with anatomical imaging to reach new levels of insight regarding the aforementioned uses has gained burgeoning attention from physicists, chemists, and mathematicians. Through exploitation of the meta-data and manipulation of multi-image acquisition, it has been demonstrated that ultra-high-resolution imaging, quantitative imaging, and real-time data analyses are possible. This provides critical information for clinicians and fellow researchers towards understanding disease progression, treatment efficacy, and treatment behavior in infinitesimal steps, providing new methods to investigate mode of action in combination with other monitoring methodology. Quantitative MRI (qMRI) requires the application of rigorous acquisition methods, sophisticated post-processing techniques, and fastidious analysis to be relevant and replicable. However, should all these qualifications be executed appropriately, it can provide verifiable, repeatable, and pivotal information regarding enigmatic maladies and treatment pathways.

Investigation into the quantitative mapping of longitudinal relaxation times to investigate tumor structures and attempted treatment methods through T_1-mapping will provide insight into the specificity and sensitivity of qMRI. The use of T_1-mapping has been shown to be a non-invasive methodology to gain deeper insight and understanding into cellular level changes as the result of inflammation and stromal barrier formation. It is sensitive enough to distinguish

between ascites and tumor formations, can provide information regarding tumor heterogeneity, and is a quick non-invasive method in which to do so. Sensitivity is demonstrated by the differences of T_1-mapping behavior based upon different cell lines compared to normal tissue, and its quick acquisition time makes it a promising method for clinical applications.

Diffusion tensor imaging, one of the leading modalities under study with qMRI, will be assessed for faster image acquisition. Through prioritizing different parameters in order to monitor the impact alterations have on image quality and information available, a template of key characteristics that can be monitored in varying degrees of depth will be investigated. This will provide insight into how alteration of parameters affect final data acquisition and potential remedies to condense imaging time that can be altered based upon study design. Furthermore, the inverse of this investigation will determine which key parameters will yield the highest quality data possible and examine the impacts of obtaining ultra-high-direction diffusion tensor images and optimal resolutions, as well as the consequences to signal when resolutions are too high. Additionally, a new method of analysis through an integrated pipeline that will accommodate artifact correction, tensor calculation, and yield multiple diffusion metrics will be presented that is compatible with various forms of instrumentation in order to correct for limited compatibility, un-intuitive application, and difficulty of use seen in many commercially available platforms available today.

Finally, investigation into next generation steps for qMRI which look to further increase the depths of information gained through imaging will be examined through iterative back projection to calculate ultra-high-resolution images at a fraction of the time cost seen with traditional imaging. This provides insight into the application of super resolution techniques to gain higher level information regarding diffusion tensors and anatomical structures without

suffering from signal loss due to nominal voxel sizes. The ultimate goal is to generate data sets which no longer hinge upon subjective interpretation and progress the field towards machine learning techniques which can be used to identify anomalies present post-imaging.

Practical imaging and analyses for qMRI

by

Macy M. Payne

A.G.S., Front Rang Community College, 2016
B.S., University of Northern Colorado, 2018

A DISSERTATION

submitted in partial fulfillment of the requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Chemistry
College of Arts and Sciences

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2022

Approved by:

Co-Major Professor
Dr. Christine Aikens

Approved by:

Co-Major Professor
Dr. Stefan H. Bossmann

Copyright

Abstract

Magnetic resonance imaging (MRI) is a powerful tool to gain insight into physiological structures and processes, molecular behavior, and psychological conditions. It has a broad range of applications, including, but not limited to, medical imaging, such as the insight it can provide towards medicinal chemistry regarding the appropriate targeting and efficiency of new drug candidates. While the fundamentals of MRI are thoroughly understood at this point of its development, the ability to extrapolate results beyond the qualitative data often seen with anatomical imaging to reach new levels of insight regarding the aforementioned uses has gained burgeoning attention from physicists, chemists, and mathematicians. Through exploitation of the meta-data and manipulation of multi-image acquisition, it has been demonstrated that ultra-high-resolution imaging, quantitative imaging, and real-time data analyses are possible. This provides critical information for clinicians and fellow researchers towards understanding disease progression, treatment efficacy, and treatment behavior in infinitesimal steps, providing new methods to investigate mode of action in combination with other monitoring methodology. Quantitative MRI (qMRI) requires the application of rigorous acquisition methods, sophisticated post-processing techniques, and fastidious analysis to be relevant and replicable. However, should all these qualifications be executed appropriately, it can provide verifiable, repeatable, and pivotal information regarding enigmatic maladies and treatment pathways.

Investigation into the quantitative mapping of longitudinal relaxation times to investigate tumor structures and attempted treatment methods through T_1-mapping will provide insight into the specificity and sensitivity of qMRI. The use of T_1-mapping has been shown to be a non-invasive methodology to gain deeper insight and understanding into cellular level changes as the result of inflammation and stromal barrier formation. It is sensitive enough to distinguish

between ascites and tumor formations, can provide information regarding tumor heterogeneity, and is a quick non-invasive method in which to do so. Sensitivity is demonstrated by the differences of T_1-mapping behavior based upon different cell lines compared to normal tissue, and its quick acquisition time makes it a promising method for clinical applications.

Diffusion tensor imaging, one of the leading modalities under study with qMRI, will be assessed for faster image acquisition. Through prioritizing different parameters in order to monitor the impact alterations have on image quality and information available, a template of key characteristics that can be monitored in varying degrees of depth will be investigated. This will provide insight into how alteration of parameters affect final data acquisition and potential remedies to condense imaging time that can be altered based upon study design. Furthermore, the inverse of this investigation will determine which key parameters will yield the highest quality data possible and examine the impacts of obtaining ultra-high-direction diffusion tensor images and optimal resolutions, as well as the consequences to signal when resolutions are too high. Additionally, a new method of analysis through an integrated pipeline that will accommodate artifact correction, tensor calculation, and yield multiple diffusion metrics will be presented that is compatible with various forms of instrumentation in order to correct for limited compatibility, un-intuitive application, and difficulty of use seen in many commercially available platforms available today.

Finally, investigation into next generation steps for qMRI which look to further increase the depths of information gained through imaging will be examined through iterative back projection to calculate ultra-high-resolution images at a fraction of the time cost seen with traditional imaging. This provides insight into the application of super resolution techniques to gain higher level information regarding diffusion tensors and anatomical structures without

suffering from signal loss due to nominal voxel sizes. The ultimate goal is to generate data sets which no longer hinge upon subjective interpretation and progress the field towards machine learning techniques which can be used to identify anomalies present post-imaging.

Table of Contents

List of Figures

List of Tables

Acknowledgements

It is my greatest joy to express my sincerest thanks to the multitude of people who have made this journey possible. Shannon Kershaw was an influential person during my formative years and helped pave my initial venture into college, assisting me as I navigated difficult waters throughout my education and embarked on a new adventure. Her dedication to helping her students is unparalleled and has made a lasting impact on many. My venture into the sciences would not have been possible without her help.

I owe a great deal of gratitude to Dr. Spencer Bagley who played an influential role in my education at the University of Northern Colorado. He is a dedicated teacher whose emphasis on learning from our mistakes made upper-level calculus one of my most memorable classes throughout my undergraduate career. Dr. Bagley went on to assist me throughout my undergraduate courses in understanding the intricacies of the mathematics for physical chemistry, helped me navigate graduate school applications, and gave me confidence as I left to continue my studies.

Perhaps one of the most influential people in my education was my PI throughout my undergraduate studies, Dr. Richard Schwenz from the University of Northern Colorado Chemistry Department. His passion for physical chemistry, ability to apply the concepts towards real-world scenarios, and openness towards questions regarding all forms of chemistry made him a favorite among many students. Throughout my undergraduate research he played a key role in fostering my passion for independent investigation into the various research questions presented within the department and was one of my strongest supporters in my quest for continued education.

I would like to express my gratitude to the Department of Chemistry at Kansas State, PLU, and the donors responsible for the NOTICXE Fellowship. It was amazing to be considered for the fellowship, and I am grateful for the time it granted me to focus on my research throughout the summer.

I sincerely thank Dr. Simon Sham for all he has done for me throughout my time at Kansas State University. He has always offered his guidance and helpful tips as I initially began my journey with MRI. To this day he remains a key partner in troubleshooting the various complications we face when we inevitably start our new projects, and I am incredible grateful for his calm, pragmatic approach to any difficulty I faced throughout my research.

I owe a great deal of gratitude to Dr. Tej B. Shrestha from Kansas State University. He was an influential mentor during my years of graduate school who introduced me to the complexities and excitement of MR spectroscopy. I greatly appreciated his candor, encouraging me to be sure of myself, and always challenging me to know more.

I must extend my sincerest thanks to my PI, Dr. Stefan H. Bossmann. I have been incredibly fortunate to have a mentor such as Dr. Bossmann as a sounding board through the intricacies faced throughout research, graduate school, and life. Throughout my time here I have been continually remined how fortunate I am to have such an intellectually generous mentor who has continually supported my ventures, encouraged a deeper understanding of my research, and rallied my spirits when frustration crept in.

Finally, my ultimate support system and assistance has come from those that are closest to me. My sincerest gratitude to Vitaly, whose support and motivation as I pursue an advanced degree has never gone unnoticed. He has often served as my rock during the largest moments of my graduate degree from preparation for cumulative exams to writing a dissertation. He has been

the most supportive partner I could have ever asked for, and I am so incredibly thankful for all he has done. Additionally, I wish to acknowledge the support and dedication my Mother and brother have given as I have completed my studies throughout the years. From the countless revisions of documents and drafts, the limitless hours on the phone for a listening ear, and endless support, I owe them an immense amount of gratitude.

Dedication

To my Mother and Brother for their endless support, guidance, and comradery throughout all stages of life.

It’s always been Us.

Chapter 1 - Introduction

What is MRI

Magnetic resonance imaging (MRI) is an imaging modality that utilizes radiofrequency pulses to perturb the spins of protons to generate grayscale images. It is a non-invasive, high-resolution technique capable of revealing not only anatomical structures but exploring micro-tissue structures and properties. Magnetic resonance imaging is a safe and repeatable modality with no exposure to radiation and few minor temporary health concerns, generally lasting only as long as the patient is within the scanner being actively imaged.[1-3] This modality has faced rapid growth since its genesis with the first patient being scanned in 1977.[4] Current techniques have focused on enhancing pulse program protocol, enabling more rapid image acquisition, obtaining higher resolution images, and advancing computational methods to gain the maximal amount of information from each imaging technique.

MRI utilizes the attenuation of protons during the application of a series of radio frequency (RF) pulses to produce grayscale images denoting the structures and composition of the sample being imaged.[5] In both clinical and research settings there has been a call to utilize ultra-high field MRI in order to elucidate the finer structures previously unknown without the use of histological studies post mortem.[6] Through the use of ultra-high field MRI at 14 to 21 Tesla, it is possible to dramatically increase the signal to noise ratio, decrease voxel sizes, and increase slice configurations in order to provide more detailed images than previously possible without signal dropout, a common consequence among applications of similar imaging parameters on lower strength instruments.[5,7] This becomes a particular asset when discussing quantitative and low signal imaging techniques such as diffusion tensor imaging. Due to the application of various selective gradients, the presence of protons possible to excite in a particular direction

decreases. By increasing the Boltzmann distribution through the initial excitation pulse of high field MRI, it is possible to increase the distribution of protons not previously possible with preceding instrumentation. This leads to increased signal with highly selective gradients which has been erstwhile impossible, thus limiting the overall usefulness, and potential, of diffusion tensor imaging. Various studies have been underway since the genesis of ultra-high field MRI to set a precedent for the overall safety and parameter control necessary to generate these images.[1,2,7,8]

MRI Basics

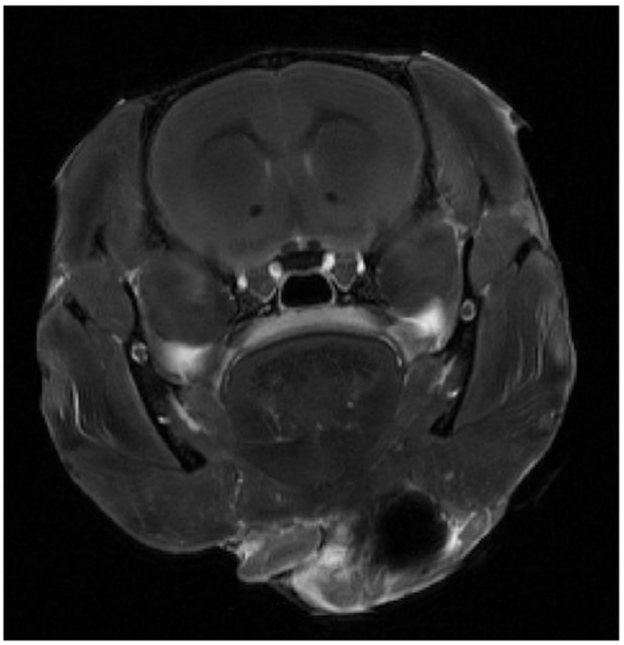

Figure 1.1: A T_2-RARE anatomical image of a rat brain imaged in Fomblin with a 14.1T Bruker Avance III in a 30 mm probe**.**

In broad terms, the MRI consists of a main magnet, a series of shims and coils, and a shielding component. These control the direction and strength of the application of the radio frequency (RF) pulses in order to generate the precessional frequencies of the protons which result in an echo formation. The echo is the signal recorded to generate the grayscale images

commonly thought of when discussing MRI, as seen in Figure 1.1.[5] Most ultra-high-field magnetic resonance instruments consist of a superconducting magnet due to their high field, high stability, and unparalleled field uniformity. They generally require a water-cooled gradient, helium cooling chamber, and cooling liquid nitrogen in order to prevent the liquid helium from boiling off.[5,9] Throughout the following sections different components of MRI, their hand in the application and acquisition of imaging, and potential implications of various modifications to pulse parameters and instrumentation will be investigated.

Mechanical Principles of MRI

The MRI referred to throughout this dissertation is a Bruker 14.1T Avance III with a compatible micro 2.5/MICWB40 probe.[9] This instrumentation utilizes a superconducting magnet embedded in a copper matrix, surrounded by a helium chamber, which is further immersed into a liquid nitrogen chamber. The liquid nitrogen chamber is used to prevent the boil off of liquid helium due to the heat generated through instrument use. Superconducting magnets are known for high levels of field uniformity and ultra-high-field strengths.[5] This leads to the possibility of shorter scan times which can make it easier for subjects during imaging, due to less time required within the scanner, and the ability to generate large volumes of data in condensed time frames in research settings. The probe referenced above, used in both *in vivo* and *ex vivo* imaging, is compatible with a variety of RF coil inserts to adapt to sample size and maximize the signal-to-noise ratios. It is available in a range of sizes from 5 to 30 mm diameters.

The shims of the MRI can be used in various ways. Fixed shimming is in reference to solid pieces of steel which are positioned throughout the magnet to improve the main field homogeneity. Further shimming can be performed through the use of the gradient coils, which can be repositioned to further improve the field homogeneity by accounting for and

accommodating the inhomogeneity that is introduced when a sample is placed in the imaging probe. Generally, the gradients are placed in three orthogonal directions and are parallel to the main magnetic field (B_0). The point which is at the center of the gradient is termed the "iso-center" and is the ideal location for samples once loaded into the instrumentation.

Gradient amplifiers are used to generate the current and voltages needed to be applied to the gradient coils and produce the gradient-induced magnetic fields. They are capable of inducing large electrical currents through the coils with short rise and fall times, thus allowing the current to rapidly rise from zero and reach its maximal value and then suddenly decline, supplying enough voltage to overcome back possible back EMF, and apply gradients in rapid succession.

Aside from these larger components, the radio frequency system is responsible for the transmitter, coil, and receiver. The transmitter generates the radio frequency pulse at a low frequency with specified bandwidth, amplitude, and phase. It is used to excite the nuclei within the desired slices/slabs of the image. The transmit coil, commonly referred to as just the coil, receives and detects the MR signals. Additionally, the coil protects the receiver circuitry from the very high voltages that are applied during the transmit and prevents the loss of small signals. A pre-amplifier, usually mounted to the coil assembly, enhances the small MR signal through a signal boost, generally with negligible noise being added. Finally, the receiver retrieves and demodulates the signal. Here the signal is maximized and the noise is minimized through separation of high frequency side bands.[5] For maximal signal to noise ratios, the coil size for the transmit coil must be as small as possible compared to the size of the sample, in that the sample must occupy the maximal volume possible of the coil itself.

MRI's have become increasingly complex and as such require an increasing number of computer systems in order to maximize their potential and increase their overall usability. They generally consist of a host computer, a pulse programmer, and an array processor.[5] The computer where the user inputs the desired parameters for a particular sequence such as slice number, slice thickness, repetition times (TR), echo times (TE), and other factors such as field of view saturations or anti-aliasing protocol is all done at the host computer. Additionally, the host computer is responsible for image display and processing the data with data manipulation into further data sets and viewing post processed images. The pulse programming computer receives the information from the host computer and converts the parameters into commands understood by the instrumentation. It maintains the safety of the instrument through synchronization of the radio frequency pulses, gradients, and sequence timing. Finally, the array processor carries out the image reconstruction through a fast Fourier transformation on the raw MR matrices where the computer determines the discrete Fourier transformation of the overall sequence.

How the Image is Made

The start of an MRI sequence begins with an initial magnetic pulse which aligns the protons with or against the main magnetic field (B_0) as demonstrated in Figure 1.2. Based on the Boltzmann distribution, predicated by instrument strength, some states are aligned with the main magnetic field, and some against, those against the main magnetic field being of higher energy. Then, the magnetic moment is flipped into the transverse plane through a 90° RF pulse delivered at the Larmor frequency generated in the transmit coil as seen in Figure 1.3.[5] The RF pulse is then switched on and off. A 180 ° pulse can be delivered by doubling the RF initially given and is often used to bring the protons back into phase coherence. As the protons try to re-align with the main magnetic field, energy is released and is received as signal in the receiver coil as seen in

Figure 1.4. The receiver coil "sees" the oscillating magnetic field with the induced voltage that is varying at the Larmor frequency. Then, the free induction decay is recorded as the amplitude of that signal decays to zero.[5]

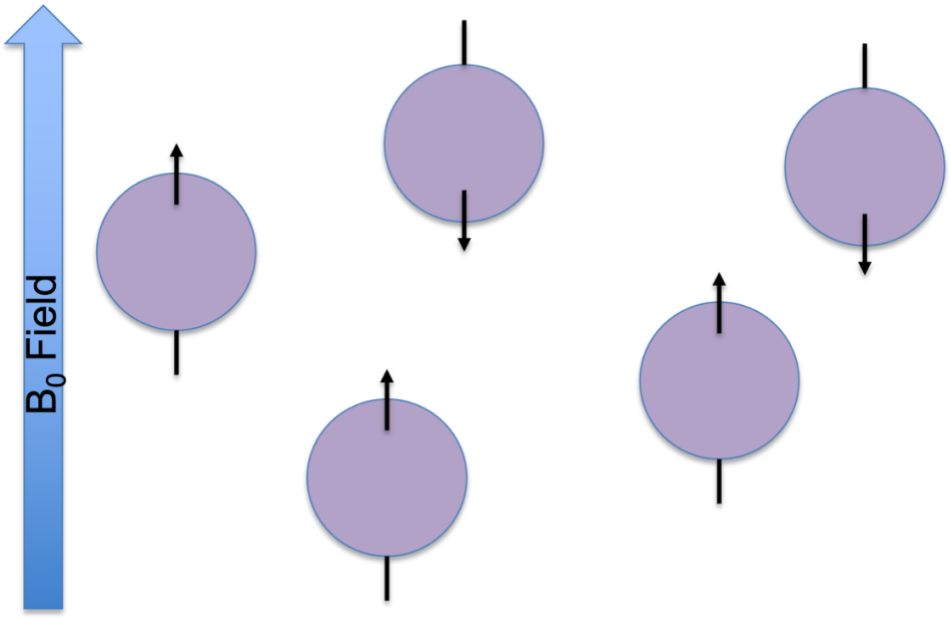

Figure 1.2: Protons are initially aligned with or against the main magnetic field. As the field strength increases, the energy between the protons aligned with or against the main magnetic field increases.

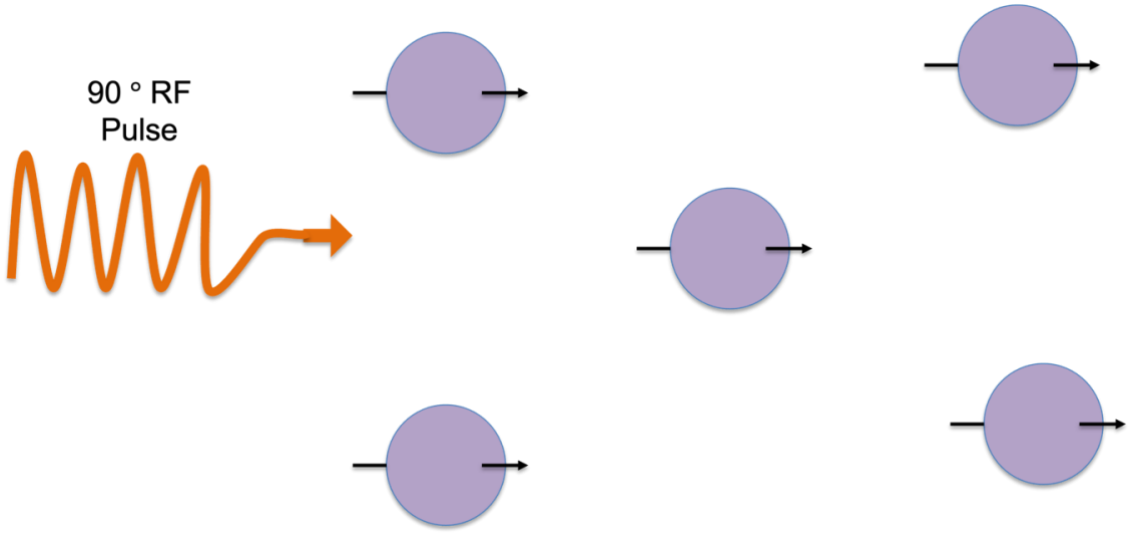

Figure 1.3: 90° RF pulse is used to tip the spins into the transverse plane. Once the RF pulse is switched off, the protons will try to re-align with the main magnetic field.

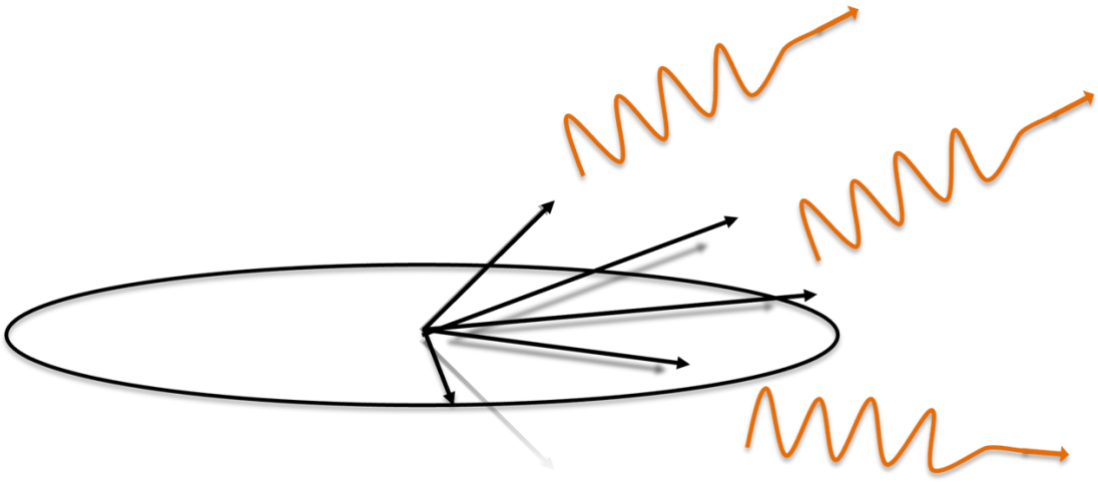

Figure 1.4: Energy is released as protons relax and re-align with the main magnetic field. The induced voltage is detected in the receiver coil and recorded.

Other Components of MRI

Aside from the physical components required to generate an MR image, there are a variety of considerations to keep in mind when determining the ideal parameters and circumstances for generating data. Raw data from imaging is stored in K-space. The size of this raw storage is determined by the frequency (FE) and phase encoding (PE) matrix and will correlate to the matrix size input in the initial parameter optimization. This data is the raw binary format of the image. Within K-space, the data for all the signal and contrast information is stored in the center, and the information regarding image resolution is allocated to the outer edges.[5] Most imaging techniques focus on filling the maximum amount of K-space in the minimal amount of time.

The aforementioned matrix size is determined by the number of phase encoding and frequency encoding steps required for the total sample size to be contained in the field of view, and the desired image resolution. Traditionally, in order to maintain speed, the matrix size is chosen in units of 2^n due to the computer's operation in bits. Generally, the phase encoding

dimension should be kept to no less than half of the size of the frequency encoding dimension in order to prevent the narrowing of pixels, as seen in Figure 1.5. However, the phase encoding

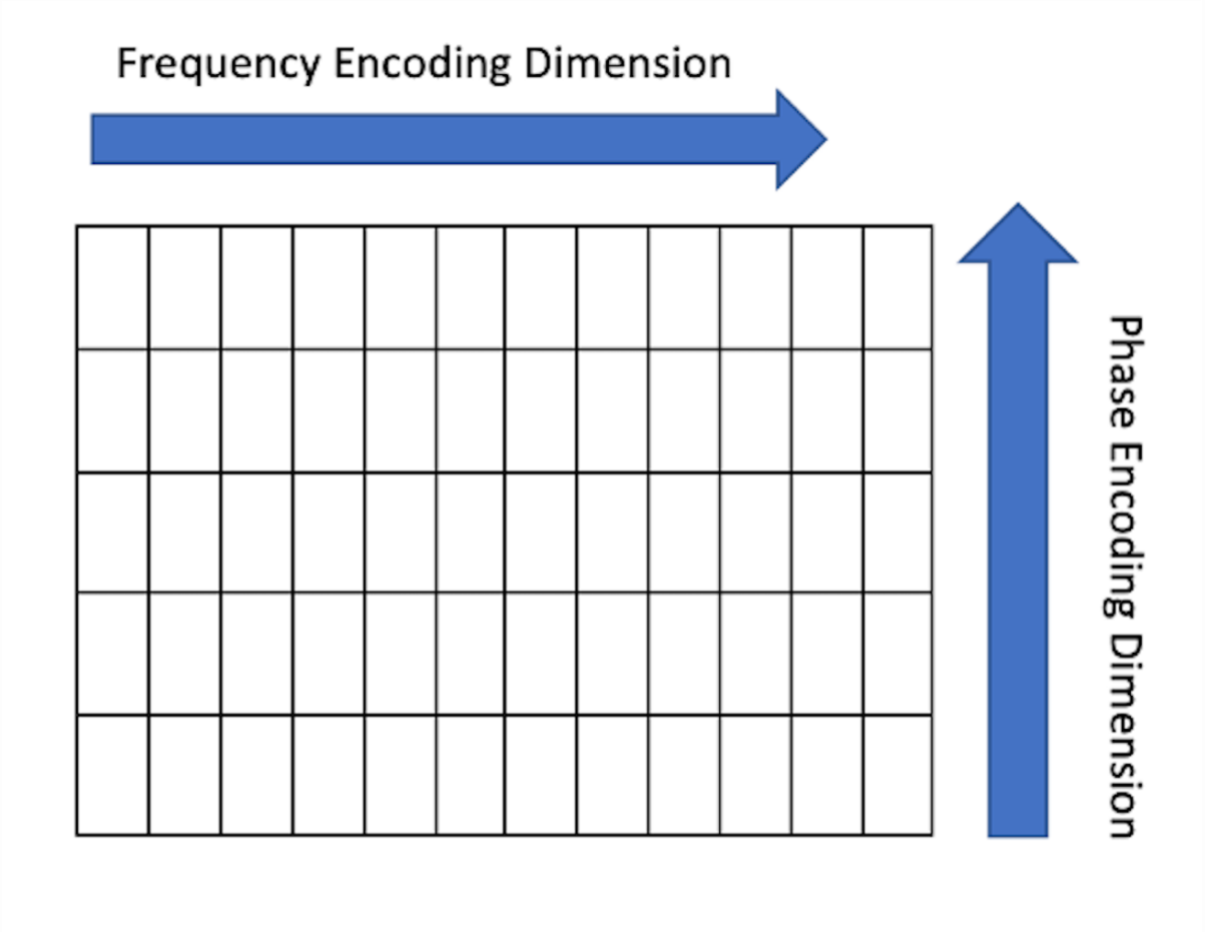

Figure 1.5: Narrowing of the pixels due to the PE direction being less than half the FE direction. Resulting pixels tend to take on a "pencil-like" architecture.

component results in the largest time costs as additional lines of data are acquired for each additional step in the phase encoding dimension. Thus, the phase encoding direction is often chosen as the shortest anatomical or sample direction. A phase encoding gradient is used to specifically acquire the PE components and is applied in one direction and changes the precession of the nuclei according to their relative position along the different encoding axis while maintaining the different phase angles of those nuclei.[5] Figure 1.6 demonstrates the impact

of the phase encoding gradients on the precession of the protons. Alternatively, the frequency encoding direction is often considered a "free" parameter in MRI that does not impact the total scan time, the impact of the frequency encoding gradient is demonstrated in Figure 1.7. It is used to differentiate between the precessing protons at varying frequencies.[10] Finally, the slice selection gradient is applied to determine the region of excitation within the field of view and define the location within and between the slices. All gradients are placed orthogonal to one another and work in combination to generate an image.

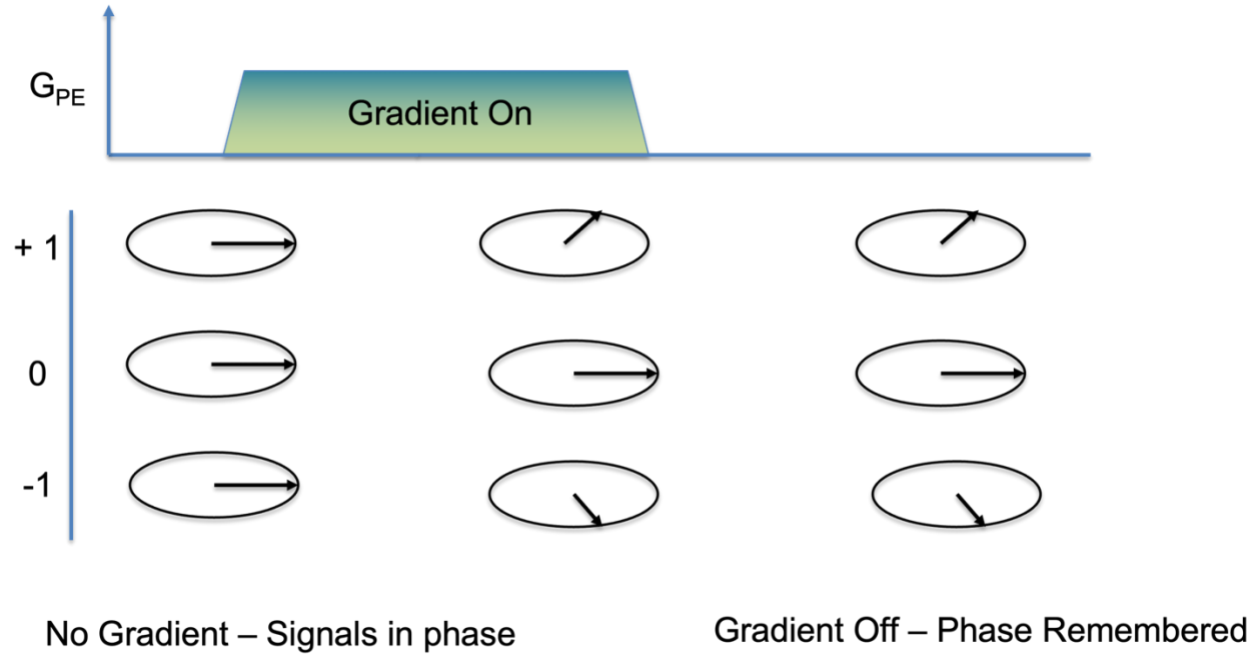

Figure 1.6: The phase encoding gradient in action. With protons initially in-phase in the transverse plane the phase encoding gradient is switched on along one of the axes. The precession of the protons will change speed depending upon their proximity to the gradient resulting in increased dephasing. When the gradient is turned off, the protons will retain the new phase angle achieved during gradient application but have maintained their original frequency. The protons are not considered "phase encoded".[5]

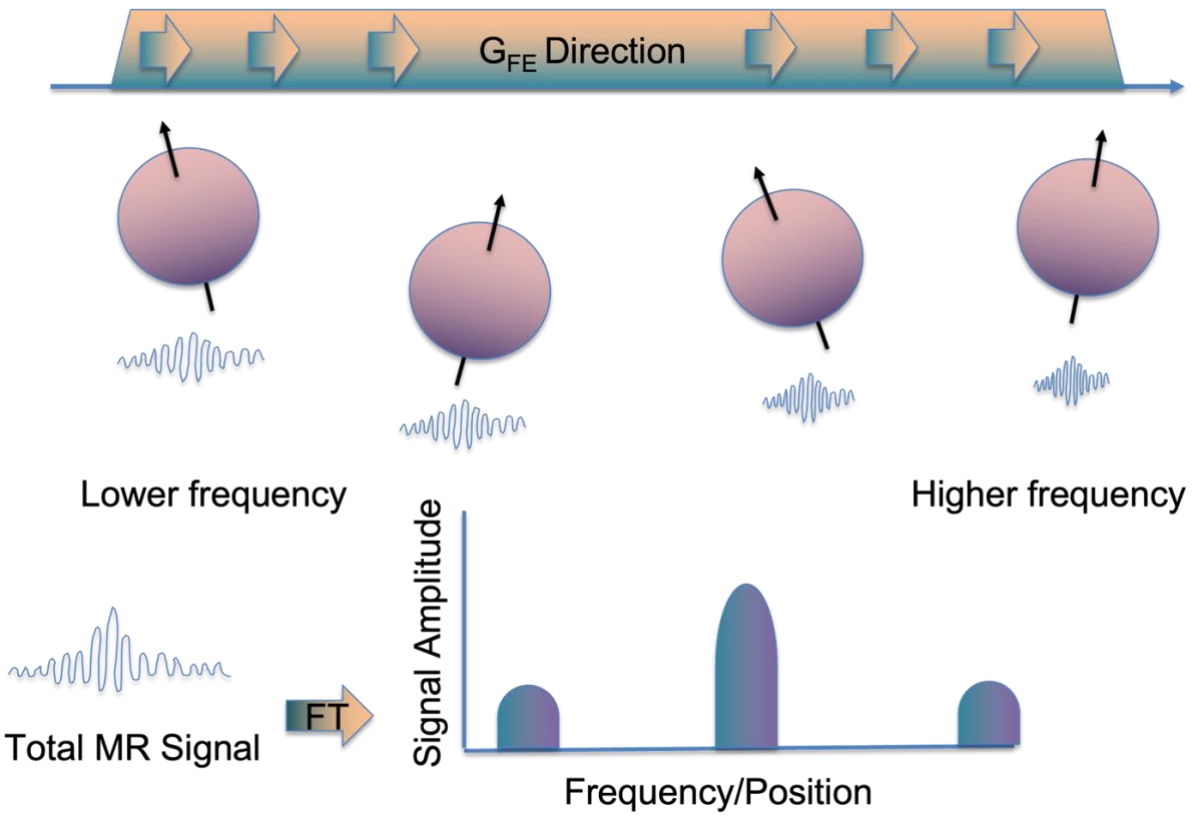

Figure 1.7: The frequency encoding gradient is applied continuously and orthogonal to the main magnetic field and phase encoding gradient. The frequency of the protons will change depending upon their proximity to the gradient. Then, the total MR signal with the phase and frequency information will undergo Fourier Transformation (FT) to produce the reconstructed data.[5]

The variation of intensities for the signals recorded in an MR sequence depends upon the differences in relaxation times of tissues or sample substances. The relaxation times are inherent characteristics to the structure of the tissues and substances and are predicated by the precessional frequencies of the protons as they relax towards the lattice and from each other. The effects and interactions of the protons as they relax towards the main magnetic field will determine the relative longitudinal, T_1, and transverse, T_2, relaxation times.[5]

Pulse Sequences

While there are a large variety of pulse sequences to choose from, they generally can be separated into two broad categories, spin echo (SE) and gradient echo (GE). These are considered the parent pulse sequences which are used to derive many of their alternatives such as rapid acquisition, echo planar, and others. Gradient echo pulse sequences apply a negative gradient lobe after the initial excitation pulse. The negative gradient lobe results in rapid dephasing and transverse magnetization. Following this gradient, an additional positive gradient lobe is applied to reverse the induced magnetic field, but only acts upon the protons which had been dephasing from the negative lobe.[5,11] Now, spins that were precessing at a low frequency due to their relative position within the gradient begin to precess at a higher frequency and spins that were dephasing begin to rephase and form an echo.[5]

The spin echo pulse sequences allow spins to dephase naturally after the initial 90° RF pulse. As the spins dephase, a 180° pulse is applied to flip the spins and reverse the phase angles but does not change the precessional frequency of the protons. The resulting spins now appear to have a higher magnetic field, and dephase in the opposite direction to form an echo.[5,12] Generally, due to the gradient echo pulse sequence requiring the application of only one full RF pulse, and relying on gradients for additional alterations, the echo can be generated more quickly. Therefore, the echo times are generally much shorter for gradient echo pulse sequences than their spin-echo counterparts. This allows for shorter repetition times (TR) for GE sequences and decreased imaging time. However, since only the portion of spins that were dephasing by the negative lobe will experience the reversal, magnetic field inhomogeneity and tissue susceptibility play a large role in the rate of transverse relaxation. Consequently, when acquiring T_2-weighted images with a GE pulse sequence, a T_2*-weighted image will actually be acquired, which is the

result of faster T_2-relaxation due to the interferences mentioned previously. Thus, if true T_2-relaxation is required, the spin-echo pulse sequence can be more useful.[11,12] Since a T_2-weighted image will be dictated by the T_2-relaxation rates of tissues they generate images which are impacted substantially less from the magnetic field inhomogeneity thus circumventing or decreasing the tissues magnetic susceptibility effects in the final image.

Chapter 2 - Quantitative MRI vs Qualitative MRI

While it is possible to argue that all MR imaging is in some way quantitative, in that the analogue components are converted into a digital value in order to produce grayscale images, which are simply conglomerates of intensity value, there is a distinct difference in the way that qualitative and quantitative MRI are evaluated and interpreted. The traditional T_1-weighted/T_2-weighted/Proton Density scans are generally considered a "qualitative" form of MRI. Images are assessed for overly bright or dark signals (depending upon the scan type) indicating alterations to the tissue structure within those regions or build-up of fluids indicating injury or illness. These images are fairly straightforward in their interpretation by a licensed radiologist and the pulse sequences used to acquire them are widely acknowledged, used, and accessible. While the term "qualitative" may discredit the substance and value of these images, that would be a premature assumption. These "qualitative" images are still constructed based upon the relaxation rates of tissues or proton density of a substance which are indicative of the overall structure and health of the region being analyzed. Additionally, due to the high-resolution images possible, made accessible by increasing magnetic fields, these anatomical images can provide extraordinary detail regarding disease progression, contrast agent effectivity, and treatment efficacy in a fraction of the time many quantitative methods will require.

Quantitative MRI requires the measurement of various relaxation rates and diffusion properties to generate not only anatomical images but also discrete values on a per-voxel basis. The values being investigated can range from simple fat and water fractions[13], spin-lattice (T_1) and spin-spin (T_2) relaxation rates, diffusion constants, and more. Traditionally, quantitative MRI has been plagued by a lack of reproducibility between instrument scanners, magnetic inhomogeneity, and lack of resolution leading to low specificity. However, as instrument

strengths have improved, improving their homogeneity alongside it, and computation methods have developed to account for scan-scan and between scanner variation and correct for artifact impacts, quantitative MRI has rapidly gained traction in both research and clinical use. T_1 and T_2 mapping have been leading the charge of quantitative MRI, alluring due to their low acquisition time costs even at high resolution and direct translation to their origin tissues. However, the use of all forms of quantitative MRI still requires more exhaustive testing and validation before it becomes as readily integrated as qualitative forms of imaging are currently.

Qualitative MRI

Qualitative MRI is what typically comes to mind when discussing MR imaging. They are the grayscale images used to discern the presence or absence of anatomical structures, tumors, or fluids. There are a variety of image types that can be used in generating "qualitative" images that provide different insights into physiological structures. Proton density images have carried their way into current MR imaging sequences due to the historical use of spin-echo pulse sequences to produce scans when early generation gradient-echo sequences were known to have poor imaging quality.[5] They measure the overall density of hydrogen protons within a given voxel and are still used today for certain clinical applications. T_1-weighted images, or "anatomy scans", are known for their high contrast to noise ratios and provide clear differentiation between tissues.[5] T_2-weighted scans, or "pathology scans", are often disparaged due to the mid-gray representation of most water and fat-based tissues,[5] but have high importance when looking for pathological conditions which will cause collections of abnormal fluids. Each of these scan types individually contributes its own assets to determining, defining, and monitoring disease progression in clinical settings. Figure 2.1 demonstrates the changes in tissue and fluid presentation with

modest modifications to TR/TE times, the parameters used to generate T_1-, T_2- , or PD- weighted images.

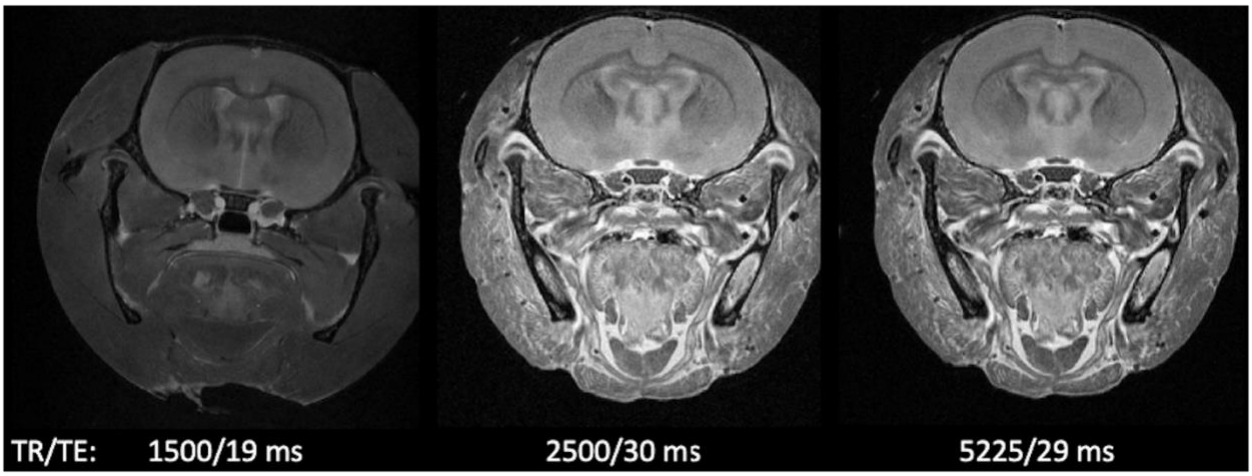

Figure 2.1: Anatomical images of rat brains taken with varying repetition (TR) and echo times (TE) demonstrating the varying impact on contrast from altering TR/TE values.

T_1-Weighted Images

T_1-weighted images are considered the anatomy scans of MRI. They can be acquired in relatively short time frames and produce excellent contrast and boundaries between tissue types. They are regularly used as a metric for overall anatomical structure and reference points in registration techniques for other pulse sequences which suffer from low signal to noise or contrast to noise ratios.[14] T_1-weighted images can be produced from either spin echo or gradient echo pulse sequences and derive their image intensity and contrast ratios from the spin-lattice (T_1) relaxation times of the tissue/sample each voxel encapsulates. The spin-lattice relaxation rate refers to the length of time it takes for a proton to recover its magnetization along the primary z-axis, i.e., with the main magnetic field. They are generated with short repetition and echo times, where tissues with short T_1 relaxation times will provide the brightest signals and those with the slowest will produce the weakest. Ultimately, this causes water-based tissues to

have mid-gray appearances, fat-based tissues with bright signals, and fluids to be dark with the minimal signal in the image.[5]

T_2-Weighted Images

T_2-weighted images are acquired based on the spin-spin (T_2) relaxation time of a given tissue or substance encased within each voxel. These pathology scans take a longer time to acquire than T_1-weighted images but are necessary to investigate regions with abnormal fluid collection. This is due to fluids having the longest T_2-relaxation times which provide the highest (brightest) signal on T_2-weighted images.[5] Scans with T_2-weighting can be acquired with either spin echo or gradient echo pulse sequences, although those acquired with a gradient echo pulse will result in T_2* rather than T_2-weighted images. This is due to the gradient echo pulse sequence being unable to account for the magnetic inhomogeneity present in the main magnetic field, which spin echo pulse sequences nullify with additional RF pulses. Thus, transverse relaxation is impacted by the magnetic field inhomogeneity resulting in increased relaxation rates and producing an image based on the T_2* relaxation rather than T_2.[11] This measurement difference will largely become an issue in the translation from qualitative T_2 imaging to quantitative T_2-mapping, where maps generated from gradient echo pulse sequences will result in recording apparent rather than true T_2-relaxation rates.

Rapid Acquisition with Refocused Echoes (RARE)

Rapid Acquisition with Refocused Echoes is an imaging technique that involves the application of an additional 180° pulse in order to turn back de-phasing protons to make them rephase.[5] This allows for rapid filling of K-space, the location of the raw imaging data, within a single TR pulse and is a frequently employed method to condense imaging time. This does come with the drawback of decreased signal as the number of refocusing pulses are increased, thus a

careful balance must be struck between employing an adequate number of refocusing pulses to decrease overall imaging time while maintaining a high signal to noise ratio. When discussing T_1-RARE or T_2-RARE imaging, the pre-identifier (T_1 or T_2) simply gives perspective on the relative repetition and echo times used throughout the sequences which will indicate the general T_1 or T_2 weighting.

Proton Density

Proton density (PD) images tend to be overlooked in many settings due to the decreased contrast they offer when compared to other modalities. However, they were heavily used in the genesis of MRI when gradient echo pulse sequences were unreliable due to poor image quality from the instrumentation of the era.[5] However, proton density images are often times considered a "free" image capable of being obtained during spin echo pulse sequences when acquiring T_2-weighted images anyway. This is due to the additional echo that can be generated at a shorter echo time during T_2 imaging which can be gathered concurrently with the echoes used for the T_2-weighted images, resulting in a slice-by-slice PD image that will map along the pathology scan. The caveat to PD imaging involves not allowing for full T_1 relaxation, hence they can only be concurrently procured with T_2-weighted images with an additional shorter echo time, but they cannot be obtained similarly with T_1-weighted images. Predominantly scans of this type still hold great importance for lesion detection in conditions like Multiple Sclerosis.[15]

Diffusion-Weighted Imaging

Diffusion-weighted images (DWI), as seen in Figure 2.2, are commonly acquired simultaneously with fractional anisotropy and diffusion tensor images, representing the random motion of water molecules within each voxel.[5] These images can be used qualitatively to investigate the integrity of cell membranes[16] and tissue health due to the relationship of diffusion

and diffusion-weighted images. As the motion of hydrogen protons, and therefore likely water, increase, its motion- whether restricted or free, can be measured in diffusion-weighted images

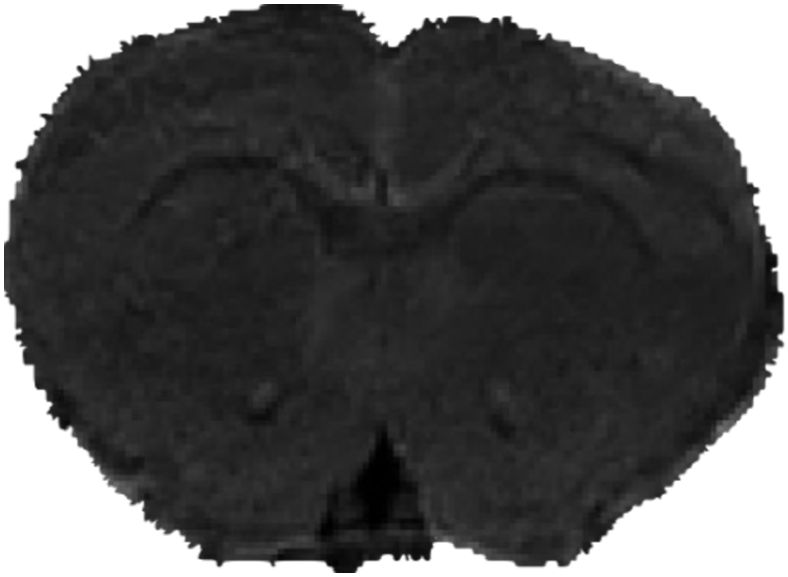

Figure 2.2: Diffusion weighted image of a rat brain with a b value of 1,000 s/mm^2.

which measure the overall times of diffusion within a voxel. Regions of high levels of diffusion and proton movement will present as dark signals and those with low diffusion present as bright signals. This differentiation is due to the application of diffusion gradients canceling out the precession of freely diffusing protons.[17] However, this can be manipulated to enhance different regions and diffusion flow based upon chosen b-values while imaging. The b value, b = $\gamma^2 G^2 \delta^2 \left(\Delta - \frac{\delta}{3}\right)$, provides the degree of diffusion weighting in units of s/mm^2, and is a factor determined by the gradient amplitude (G), the time of the applied gradients during imaging δ, duration between gradients applied Δ, and γ is the proton gyromagnetic ratio.[5] At smaller b-values protons with large diffusion distances will have higher signals, while those with slower movement will be more apparent at larger b-values of over 1,000 s/mm^2.[18] Clinically, diffusion

weighted images are frequently used in diagnosis of strokes and staging of non-small cell lung cancer.[19]

Quantitative MRI

Many of the methods for quantitative MRI arise from initial principles used in generating anatomical images. T_1 and T_2 weighted images which depend upon the longitudinal and transverse relaxation rates of protons can be translated into T_1 and T_2 maps which will map the relaxation time by region of interest or voxel wise. Diffusion weighted imaging can be further expanded to examine diffusion tensors based upon the rates of diffusions and fractional anisotropy which determines the level of constriction within the tissues. Diffusion tensor imaging can be further color coded to denote the overall orientation of diffusion and be used to later perform tractography and connectivity assessments for further study.

Diffusion Tensor Imaging Datasets

Diffusion tensor imaging (DTI) is only one of the potential applications for diffusion imaging but is perhaps the most well-known. Previous investigation into the anisotropic diffusion of samples was limited to fixed samples prior to the implementation of DTI.[17] Diffusion is measured with respect to the restriction based on the surrounding tissues and describes this diffusion through a Gaussian model. The tensor is a positive-definite matrix that is used to determine the eigenvectors and eigenvalues for each voxel. These values can be used to determine the principle direction of diffusion based upon the eigenvector. The eigenvalues can conversely be used to measure the level of diffusivity within that direction. These values can be further manipulated and used in conjunction with one another to determine the fractional anisotropy (FA), mean diffusivity (MD), radial diffusivity (RD), axial diffusivity (AD), and apparent diffusion coefficient maps (ADC).[14]

Diffusion imaging is performed through initial application of the main magnetic field followed by administration of additional gradient pulses which cancel out the signal produced by stationary protons, i.e. those that are isotopically diffusing and therefore look stagnant. This generates a measurable phase shift which can be measured by signal intensity that corresponds to the overall direction of diffusion which coincides with the gradient combination applied. The signal from the application of the diffusion gradient and the signal generated without the application of the diffusion gradients are then applied to solve the Stejskal-Tanner equation[17]; $S_k = S_0 e^{-b\hat{g}_k^T D \hat{g}_k}$, where S_k is the signal acquired with the diffusion gradient, S_0 is the signal with no diffusion gradient applied, b is the b-value, g_k is the gradient in the k^{th} orientation, and D is the diffusion. A system of equations based upon the number of diffusion directions applied are used to solve for the diffusion tensor at each voxel, generally using a least squares method.[20] In order to determine the diffusion tensor, a minimum of six gradient directions, and thus six diffusion images must be obtained in addition to one non-diffusion image with no diffusion gradient application.

Based upon the eigenvalues from the diffusion matrices, it is then possible to extract further values for analysis. The apparent diffusion coefficient map (ADC) can be used to determine the relative diffusion within a region through a grayscale image map. This mapping technique utilizes the signal from the images taken without the application of the diffusion gradients and takes the logarithmic ratio with the signal from the diffusion weighted samples scaled by their corresponding b-values.[21] These images will denote the relative amount of diffusion for each voxel based upon the coefficient's numeric value which are used to scale the image intensity. ADC maps follow an intuitive reasoning where areas of bright contrast indicate higher levels of diffusion, while those of decreased intensity indicate limited diffusion.

Fractional anisotropy (FA) images are an alternative representation of the data. Analysis of FA images is done through determination of the ratios of the eigenvalues and is a method frequently employed to discern tissue organization within a sample.[22] It is determined by the fraction of diffusion which is anisotropic, or constrained in a particular orientation. Thus, regions with high intensity indicate highly ordered diffusion while regions of dark values indicate less organized diffusion patterns.[23]

Traditional color-coded diffusion tensor images are representations of the first eigenvector color coded depending upon orientation overlaid upon the anatomical surface. The colors depend upon which processing software is used but are generally restricted to red, blue, and green with combinations of directions providing some variance within those color options. Intensity, based upon the eigenvalues, is used to indicate the relative level or amount of diffusion with bright areas indicating high levels of diffusion and dark regions those without. This is accomplished in a variety of ways with most software platforms utilizing the primary eigenvalue to regulate intensity and others modulating the first eigenvector by the fractional anisotropy values.

Fractional Anisotropy

While some measurements derived from diffusion tensor imaging provide both magnitude and vector components indicating not only how much, but also in which orientation the diffusion occurs, fractional anisotropy utilizes a 0 to 1 scale to provide insight into the diffusion and anisotropy of a given region as seen in Figure 2.3. This 0 to 1 index indicates how isotropic, a measurement of 0, or anisotropic, a value of 1, the diffusion is. Generally, values closer to 0 indicate loss in neuronal fiber tracts, while higher values closer to 1 indicated high levels of fiber coherence.[14] The degree of anisotropy is based upon the eigenvalues previously

determined from the diffusion matrix by FA = $\sqrt{\frac{1}{2}}\left(\frac{\sqrt{(\lambda_1-\lambda_2)^2+(\lambda_2-\lambda_3)^2+(\lambda_3-\lambda_1)^2}}{\sqrt{\lambda_1^2+\lambda_2^2+\lambda_3^2}}\right)$.[22] Fractional anisotropy is commonly used to measure and monitor axonal and myelin degeneration and injury based upon the degree of anisotropy present during imaging. It is through the use of both anisotropy and eigenvectors that tractography can be performed in voxel-to-voxel fiber tracking.[14]

Figure 2.3: Fractional anisotropy image of rat brain. Grayscale values are scaled between 0 and 1 indicating the degree of anisotropy for each voxel.

Diffusion Tensor Imaging

Diffusion tensor imaging is often the umbrella term that encompasses diffusion weighted images, fractional anisotropy images, apparent diffusion coefficient maps, diffusion tensor images, axial diffusion images, mean diffusion images, and radial diffusion images. However, for the purpose of this section, diffusion tensor imaging will be referring solely to the color

coordinated images modulated in intensity by the anisotropy of the particular tissue where the color indicates the diffusion orientation/direction as seen in Figure 2.4. Solving for the diffusion tensor is a matter of high debate as each analysis pipeline is free to use whichever methodology they are most comfortable with and are generally selected with respect to overall time cost for analysis, although they often operate within a black box paradigm. However, typically analysis strategies to solve for the tensor fall into one of four categories: weighted least squares, non-weighted least squares, nonlinear, or direct.[20]

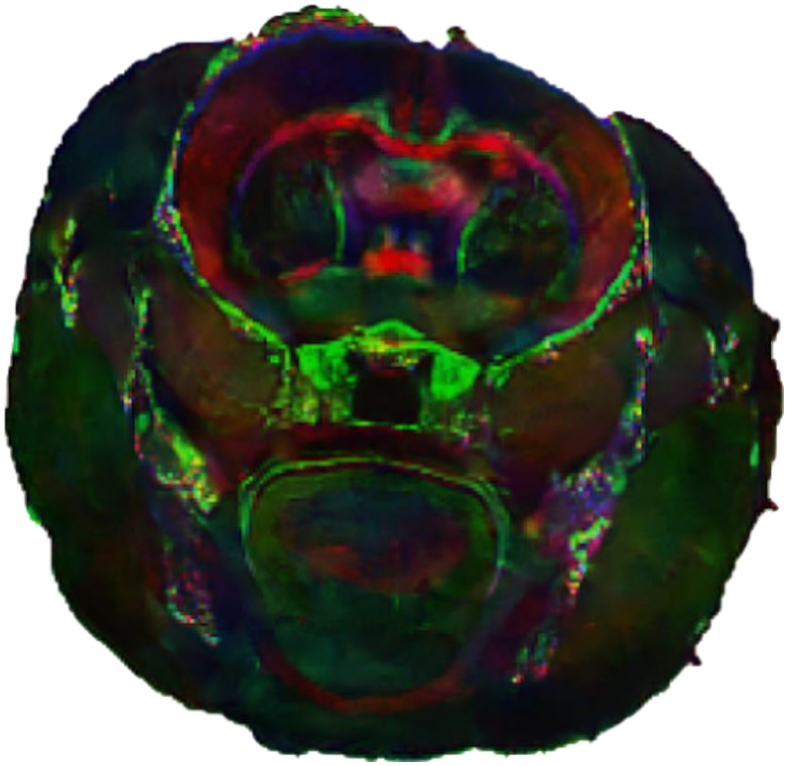

Figure 2.4 Diffusion tensor image of a rat brain taken with 64 diffusion directions.

Should the data set contain only six directions an exact solution is possible, however in most instances more than six directions are used in which case fitting the tensor will require the use of either an unweighted-linear least square fit, weighted-linear least squares fit, or nonlinear fit.[20] Each methodology requires a slightly different manipulation of the variables obtained

during imaging, but should ideally converge with similar solutions. Either way, the tensor imaging is based upon the principal eigenvector describing the diffusion within a given voxel based upon the diffusion data generated from a series of gradient applications. The eigenvector can be color-coded in a multitude of ways, but traditionally, the superior inferior direction is denoted in blue, left-right in red, and anterior-posterior in green.[17] These eigenvectors are then modulated in intensity based upon the primary eigenvalue, which denotes the level of diffusion for a particular voxel.[14,22] Through this combination of direction and relative diffusion information it is possible to assess white matter integrity, disease progression, and with the addition of tractography the connectome of various structures.[17]

Apparent Diffusion Coefficient

Figure 2.5: Apparent diffusion coefficient map of a rat brain imaged with 64 diffusion gradients and a b- value of 1,000 s/mm^2.

The apparent diffusion coefficient is one of the most useful measurements possible through diffusion imaging. It has a broad range of applications from monitoring tumor progression and treatment efficacy to grading degrees of meniscus and vertebral damage.[18] It has

the added benefit of being able to be acquired with fewer directions/gradients (as few as two different b values and diffusion data sets) than other forms of diffusion imaging such as the diffusion tensor and fractional anisotropy image.[17,18] Through taking the logarithm of the ratio between the signal acquired with and without the diffusion gradient applied, the apparent diffusion coefficient map can be calculated to provide voxel-wise monitoring of diffusion within a system.[20] ADC maps tend to present as the "inverse" of DW images, where in a DW image regions of dark signal indicate high levels of diffusion, in an ADC map these same regions would appear bright. An example of an ADC image is seen in Figure 2.5. This allows the ADC map and DW image to act as a checkpoint blockade to verify against very particular but troublesome artifacts. If a region that is overly bright on a DW image is due to T_2 shine through effect, rather than having that same region appear dark on an ADC map it will still appear as a bright signal. The T_2 shine through effect is a bright signal that originates from tissues with long relaxation times that retain their signal after diffusion gradients are applied and are most common when acquiring diffusion images at high b-values, which are frequent in brain diffusion scans.[19] One of the hidden benefits of ADC maps is that they operate as a checkpoint blockade for relaxation rate dependent artifacts but are also independent of the magnetic field strengths as well, making them less sensitive to magnetic field inhomogeneity.[16] Overall, the ADC maps allow for deep investigation into tumor progression as tumors in early stages with high cellularity or stroma will present with restricted diffusion, and therefore present darker on an ADC map. Alternatively post-treatment and in the event of necrosis, the ADC map will show increased diffusion and brighter signal. The ADC map is even sensitive enough to detect inflammation due to the proteins restriction of diffusion as well.[18] With a wide variety of applications possible, the

apparent diffusion coefficient map is a critical component to diffusion imaging and a strong focus of many research and clinical endeavors.

T_1-Mapping

T_1-mapping is a quantitative MRI method that rapidly gained traction in cardiac imaging. It has become such an integral part into understanding cardiac health and disease progression that it is now a routine component of cardiac magnetic resonance imaging.[24] Similar to T_1-weighted imaging, T_1-mapping is controlled by the longitudinal relaxation times of the tissues. However, rather than reporting back a signal based upon this relaxation time, the T_1-relaxation time is calculated and reported back as a T_1 map which can be grayscale or color-coded in a pixel wise manner or on a selected region of interest.[25] While the recorded T_1 relaxation time may simply be referred to as "T_1" it is important to distinguish the three different types of values that can be achieved with T_1 mapping protocol. "Native" T_1 mapping or "native" T_1 times, refer to T_1 maps and relaxation times acquired without the application of contrast agents, which would alter the recorded longitudinal relaxation.[24] This differs substantially from the "apparent" T_1 maps (T_1*) or apparent T_1 relaxation times, which will report back shorter T_1 relaxation times due to increased rate of relaxation caused by the magnetic field which will impact the rate of relaxation as what is similarly seen with GE sequences in T_2-mapping.[24,25] An additional component which can be acquired through T_1-mapping pulse sequences, but is not a measurement of the relaxation times, is the extracellular volume (ECV). This parameter can be defined by T_1 maps taken prior to and post contrast agent administration and is frequently used to understand cardiac disease progression.[24] There are a variety of methodologies in which to acquire T_1 maps as well as a series of hurdles to overcome in order to increase their accuracy and precision. However, this is a promising field which has readily become a great asset clinically for cardiac applications and

shows promise in other disease models with similar instances of fibrosis and tissue malformations.

T_2-mapping

Through the use of T_2 mapping it is possible to measure the differences in T_2-relaxation times of various tissues throughout each voxel of an image. Currently the use of T_2 mapping is heavily focused on applications for joint diseases such as rheumatoid and osteoarthritis, as well as assisting in the grading and assessment of the aggressiveness of prostate cancer.[26-28] However, with more development into the methods for acquisition and standardization, and the growing movement towards ultra-high-field MRI, it will be possible to apply this method to other areas.

The T_2 relaxation value is a time constant determined by the rate at which an excited proton will reach equilibrium and fall out of phase with its cohorts, measured by the time it takes for the precessing protons to lose phase coherence from protons spinning alongside the main magnetic field.[5] Generally, T_2 maps are acquired with the initial application of an RF pulse and collecting a series of images at a variety of echo times. The signals of these images are then used to fit $S(t) = S_0 * e^{-\frac{t}{T_2}}$ for the T_2 relaxation time where S(t) is the signal intensity for one of the series of echo time images, S_0 is the signal at the corresponding voxel at equilibrium, and t is the echo time collection point.[29] If a gradient echo pulse sequence is used to acquire the T_2 map, the true T_2 relaxation value will not be acquired, and instead the apparent T_2 (T_2*) relaxation will be measured and will need to be corrected if transverse relaxation values are required. Unfortunately, T_2 values vary according to instrument strength and pulse sequences that are often used to measure them and tend to present with stimulated echoes which can distort the measured relaxation value. These issues will need to be addressed and circumvented in order to produce

pragmatic results and bring T_2 mapping into mainstream applications for MRI. More information on T_2-mapping can be found in Appendix A.3.

Chapter 3 - Relaxation Time Map Application- T_1 Investigation

While T_2-weighted imaging is often considered "pathology scans", a study performed by Wang et. al. illuminated that tissue characterization of pancreatic ductal adenocarcinoma (PDAC) with T_1-mapping, ADC mapping, and T_2-mapping demonstrated T_1-mapping as a superior characterization tool compared to T_2-mapping. This potentially enhanced differentiation in T_1-mapping to discern between malignant and benign tissues could be due to the larger T_1-relaxation time differences between regions which are fluid based compared to regions of fibrosis and scarring.[30] Therefore, for the close monitoring of PDAC and other multi-system maladies with high rates of metastasis and comorbidities, T_1-mapping may be a potential leading method at monitoring disease progression for complex disease types which cause more subtle changes to surrounding tissue or present with more complex pathology. T_1-mapping offers the ability to monitor disease progression and impacts through non-invasive, non-radiating and repeatable imaging without the use of contrast agents. It has been previously been shown sensitive enough at clinical strengths to detect and quantify fibrosis,[25] and cardiac imaging has already demonstrated its specificity at discriminating between different types of heart disease.[24] Therefore, at higher field strengths which enable higher signal to noise ratios and higher attainable resolution,[5,7] it is believed that T_1-maps acquired at ultra-high fields will be able to detect and monitor the progression of previously difficult to monitor and diagnosis malignancies such as pancreatic ductal adenocarcinoma and adrenocortical carcinoma through repeated imaging at various time points. It is anticipated that a discernable difference between T_1-relaxation values will be readily apparent on color-coded T_1 maps between tumor tissues, surrounding organs, and healthy tissue. While T_1-mapping has already been implemented in cardiac imaging techniques, it is predicted that the higher resolutions possible through ultra-high-

field instrumentation translation into its application for more complex diseases with smaller variations between healthy and malignant tissues, such as what is often seen in early stages of pancreatic ductal adenocarcinomas and adrenocortical carcinomas, will be possible.

Initial Development

Investigating, diagnosing, and treating PDAC is an arduous and often frustrating task. The tumor presence is compounded by the buildup of thick stromal barriers which limit the ability to adequately apply treatments making it one of the deadliest forms of cancer, with a 5-year survival rate under 10%. [31] Additionally, the lack of vasculature in the stromal barrier and highly complex immune involvement make treating PDAC nearly impossible. Furthermore, the propensity for PDAC to spread into microscopic tumors early in the disease progression limits the application and viability of currently available treatment modalities even in the early stage, in addition to causing a rapid spread of cancer to various organs, typically prior to initial diagnosis.[31]

The intricacy of diagnosing PDAC, the rapid tumor development and metastasis, and challenging if not impossible application of potential treatments, has led various investigators to assess better diagnostic and therapeutic tools in order to improve patient outcomes. Utilizing the advancements in MRI it is now possible to more closely and continuously monitor PDAC development and treatments, as well as identifying micro-metastases which can occur early in the cancer's progression. Since MRI is a non-ionizing, radiation-free imaging method, it is possible to sustain repeated monitoring over the course of PDAC patient's disease progression at regular intervals throughout the course of treatment, with expense often being the main limiting factor rather than risk to the patient themselves. Additionally, it is possible at higher field

strengths to achieve ultra-high resolution without the use of additional contrast agents which are known to have potential toxicity and side effects.[32]

T_1 and T_2 mapping have shown promise as a potential tool to assist in diagnosing and monitoring PDAC even at clinical strengths due to their ability to provide voxel-wise tissue tracking regarding changes in structure and composition without the need for contrast agents. This becomes pertinent when working on lower strength instrumentation (such as 1.5 and 3 Tesla), where achieving high resolution anatomical T_1 or T_2 weighted images and diffusion images would be cost prohibitive and time consuming as T_1 and T_2 maps can be acquired in under 10 minutes (pending the number of slices collected).

It is hypothesized that through T_1 mapping it will be possible to monitor the minute level changes in tumor progression and treatment efficacy effectively and with short analysis times, allowing for rapid translation from imaging to result assessment. An experimental model was developed to investigate the effectivity of T_1 mapping in monitoring the growth and development of pancreatic tumors in a murine model. Based upon the progress made in identifying pancreatic tumors through T_1 mapping, there presents an additional opportunity to monitor the effectivity of therapeutic hyperthermia treatments in addition to overall tumor monitoring as an expansion of the previous work done by other researchers.

While various chemo-therapeutic strategies have shown a lack of promise as an effective sole treatment for PDAC their application in conjugation with hyperthermia have shown to generate more fruitful results.[33] Hyperthermia has been shown to increase the overall penetration of different therapeutics making treatment more effective by weakening the stroma and generating cell death if ablation temperatures are reached. Due to T_1 mapping's dependence on temperature, as higher temperatures can lead to increased T_1 relaxation rates[34,35], it may be

employed as a method to monitor the success of hyperthermia applications both in monitoring the effectivity of heating as well as monitoring the alterations to tumor tissues post-hyperthermia as well.

The initial study proposed the orthotopic injection of 1 million KPC cells suspended in Matrigel into six C57BL/6 mice. After a period of 12 days, the mice began preliminary screening through anatomical imaging in a 14.1 T Bruker Avance III MRI to determine tumor growth and positioning. Three mice were imaged each day, offset by one day to allow for adequate recovery time between anesthesia/imaging intervals. Once the tumor was detectable, a T_1 map was acquired in addition to the traditional anatomical images as well as the application of mild hyperthermia for 30 minutes delivered via a custom antenna developed by the Prakash Group (KSU) twice over the course of four to five days. T_1 maps were then acquired before and after the application of hyperthermia, and then continued throughout the disease progression until animals were euthanized due to disease progression resulting in substantial deterioration in comfort or 23 days post tumor implantation, whichever came first.

T_1 mapping can be accomplished in a variety of methods. Traditionally, in cardiac type imaging, the leading method is the Modified Look-Locker (MOLLI) pulse sequences. This is due to its single-shot acquisition, which yields highly reproducible results.[24] However, despite its maturity, inversion recovery (IR) based methods suffer from the initial acquisition of the apparent T_1 (T_1*) relaxation time and require additional corrections in order to obtain the true longitudinal relaxation rates.[24,25] Additionally, IR methods struggle with consistent underestimation of the T_1 time constant when compared to other methods, and are more sensitive to field inhomogeneity, which tends to increase as field strengths increase.[34] Due to these limitations, the alternative Saturation Recovery (SR) method was employed, which uses

saturation RF pulses to allow the longitudinal magnetization to relax independent of its state prior to the previous saturation pulse, allowing for multiple, independent measurements for each pulse application/sequence.[24] SR methods have the additional benefit of increased accuracy and decreased time cost per acquisition due to the ability to start a successive pulse without waiting for full T_1 recovery between them. The increased accuracy is owed to the decreased sensitivity to magnetization transfer effects and true T_1 recovery measurements since the readout is not impacted by inversion pulses.[25]

Starting on day 12, mice were initially anesthetized in an induction chamber using 3-4% isoflurane. Once sufficiently anesthetized, monitored via decreasing respiration, mice are transferred to a cradle with their front teeth hooked onto a tooth bar which is held in place through a nose cone that continues to deliver anesthesia during preparation and imaging. Once the mice are secured to the tooth bar and placed in the nose cone anesthesia is reduced to approximately 2% isoflurane. A respiration pad is placed below the rodent's sternum/abdomen to monitor and track respiration as a means of monitoring welfare as well as to control gating procedures to time TR pulses between respirations using a Small Animal Monitoring System. Finally, a rectal thermometer probe is used during hyperthermia treatments to monitor core body temperature which is synced with the Small Animal Monitoring System alongside the respirations. Suspended via their front teeth on the tooth bar, mice are loaded into the probe and suspended vertically in the magnetic resonance spectrometer. Initial localizer sequences are used to identify rodent positioning, map the field inhomogeneity (B_0 field), and shim the gradient coils to account for additional magnetic field inhomogeneity induced by individual mouse positioning and size. After the initial localization sequence, anatomical imaging is acquired using T_2- Rapid Acquisition with Refocused Echoes (RARE) pulse sequences. Images are obtained using 8

refocusing pulses (a RARE factor of 8), with a repetition time (TR) of 2500 ms, an echo time (TE) of 30.23 ms, and one average (NEX = 1). Image size and resolution were maintained at approximately 256 x 256 matrix size and a resolution of 0.113 x 0.113 mm with a slice thickness of 0.500 mm with the number of slices varying in accordance to image orientation, i.e. more slices being required for the axial view than the sagittal or coronal view due to the relative "length" of the mouse being larger than its relative width. Initial anatomical images from early-stage imaging (day twelve post tumor implantation) can be seen in Figure 3.1. Within this figure the respective location of the kidneys and spleen are readily discernable as well as visible early tumor growth in the axial, sagittal, and coronal images.

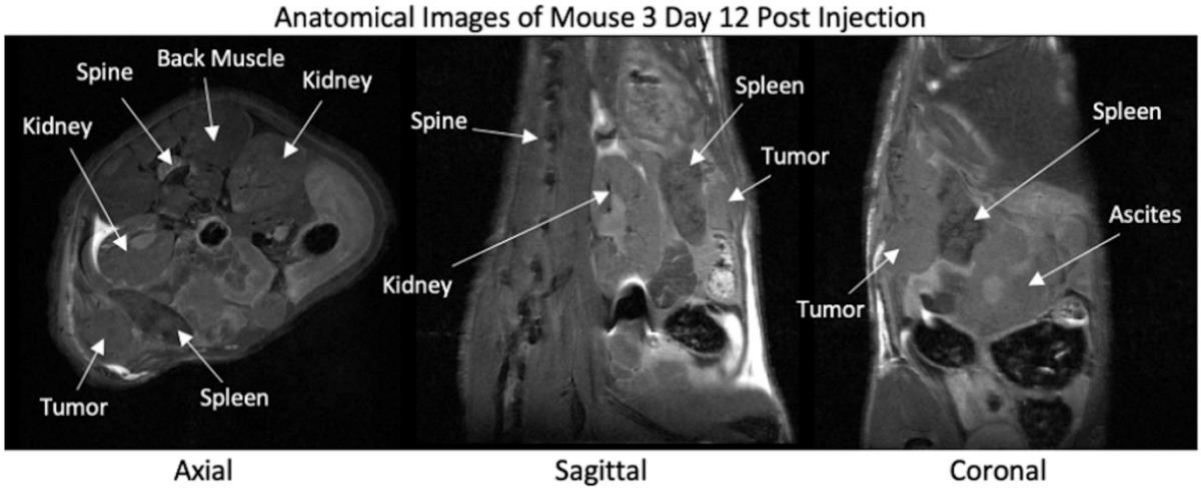

Figure 3.1: T_2-RARE images of the axial, sagittal, and coronal slice for Mouse 3 taken 12 days post tumor implantation with relevant organs, bones, and tissues indicated.

Within 16 days post tumor implantation, pancreatic tumors were easily discernable through anatomical imaging. Subsequently, T_1 maps were acquired through Saturation Recovery methods over a series of three slices which accurately encapsulated the full tumor volume. A slight modification to traditional SR methods involved the use of additional refocusing echoes to condense overall imaging time, as time under anesthesia becomes an increasing concern with the

application of a 30-minute hyperthermia treatment. This allowed a completed T_1-map to be acquired in approximately 5 minutes. Throughout the SR pulse sequences, repetition times of 500, 941, 1510, 2306, 3650, and 10,000 ms were applied, and their corresponding signals at an echo time of 30 ms were recorded and stored as individual images. Figure 3.2 shows the initial images acquired after the T_1 mapping protocol for one slice, while Figure 3.3 demonstrates the voxel-wise color coded T_1 map reconstruction of Mouse 3 across all slices.

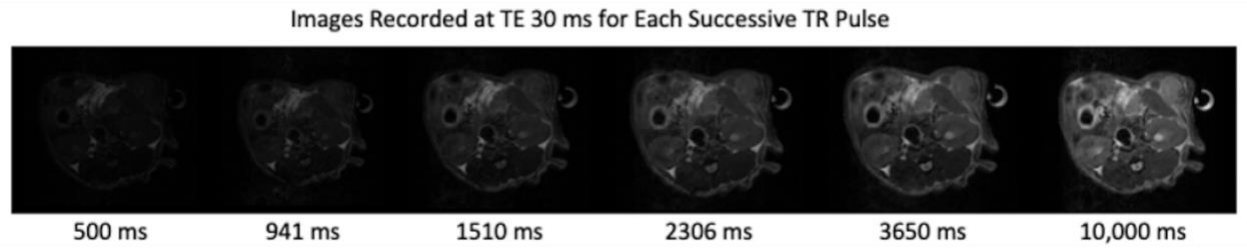

Figure 3.2: Anatomical images of Mouse 3 slice 2 for the T_1-mapping sequence demonstrating the signal change with increasing TR time.

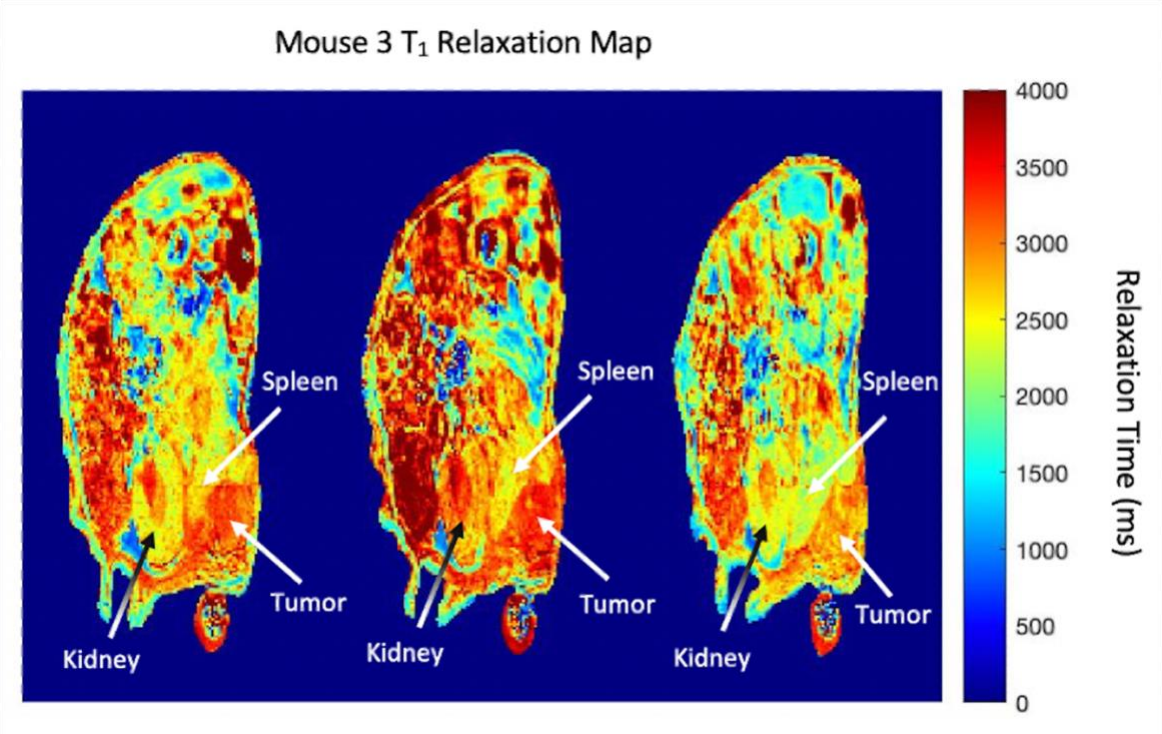

Figure 3.3: Color-coded T_1 Map of Mouse 3 across three subsequent slices prior to hyperthermia demonstrating the voxel wise mapping of T_1 relaxation values. Color denotes the relaxation time

(ms) for the tissues captured within each voxel following the scale to the right. Annotated within each slice is the location of key tissues for comparison: Kidney, Spleen, and KPC Tumor.

Upon day one of imaging (day 12 post tumor implantation), tumors were visible in mice, indicating rapid growth in the KPC model. While analysis of the anatomical images was straightforward through either the host computer or ITKSnap, the generation of pixel-wise T_1 maps requires additional manipulation, as only defined regions of interest (ROI), non-color-coded analysis is available through the Paravision 6.0.1 software. Post-imaging, the files were converted to dicom format. This is the most readily "readable" file format by most commercially available medical image viewing software available. Bruker systems organize the dicom files for T_1 mapping by their repetition time. Thus, in a series of 3 slices, every third image would be the next repetition time image. Prior to conversion to a NIfTI file format, which is the file type needed for analysis due to the indexing of the repetition times, slice, and matrix dimensions, the dicom files were organized according to their slice structure. This was accomplished by creating new files for each respective slice, that consisted of all the dicom files of the varying repetition times. They were then converted to NIfTI through dicm2nii for a total of three independent NIfTI files. At this point, images were segmented to create a binary mask and exclude the background from the T_1 relaxation calculation. Removal of the background allows the T_1 map to be calculated in a cost-effective manner as only the regions of interest are included in the calculation. Then, through a system of equations, the T_1 relaxation time is determined for each voxel, whose values were then used to create a grayscale and color-coded map. Details regarding the T_1-mapping protocol can be found in Appendix A.1 and A.2.

Post-assessment of the T_1 maps indicated that pancreatic tumors from the KPC model had elevated T_1 values as seen in Figure 3.4, which provides the mean relaxation values for various

tissues across various slices and timepoints. However, no considerable change in relaxation values were noticed between hyperthermia applications. This provides promising information that it is possible, especially at higher fields strengths, to identify pancreatic tumors without the application of contrast agents, radiation-based forms of imaging, or invasive biopsy, but does indicate that it may not be an ideal method to monitor hyperthermia treatment efficacy.

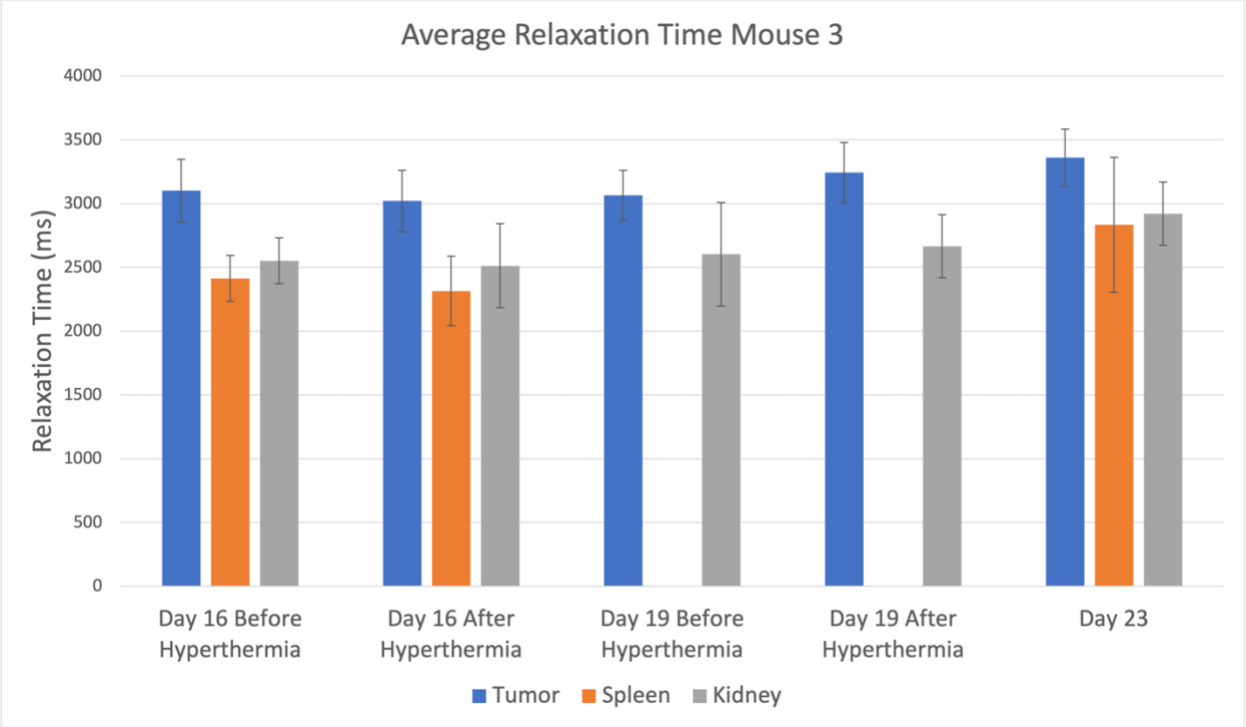

Figure 3.4: Average longitudinal relaxation times for various tissues of Mouse 3 at different time points.

Continued Development – Ablation Trial

Based upon the promising results seen in the previous experimentation, an additional study was proposed which utilized T_1 mapping as a means to identify and monitor tumor growth of adrenocortical carcinomas. Additionally, the use of ablation as the hyperthermia treatment method presented an opportunity to determine if T_1 mapping would be more effective at monitoring the effectivity of a larger thermal dose. Twelve immunodeficient nude (Nu/Nu) mice

were subcutaneously injected with HAC15 cells into the right flank. Once tumors were palpable and measured via calipers to be approximately 5 mm in size, the mice were imaged in a 14.1 T Bruker Avance III MRI. Prior to imaging, mice were anesthetized using an induction chamber at approximately 4% isoflurane, and then placed on a cradle for imaging while suspended from a tooth bar with their two front teeth; once secured anesthesia was reduced to 2%. A microwave antenna was placed alongside the tumor and secured to the skin with adhesive to apply the hyperthermia therapy.

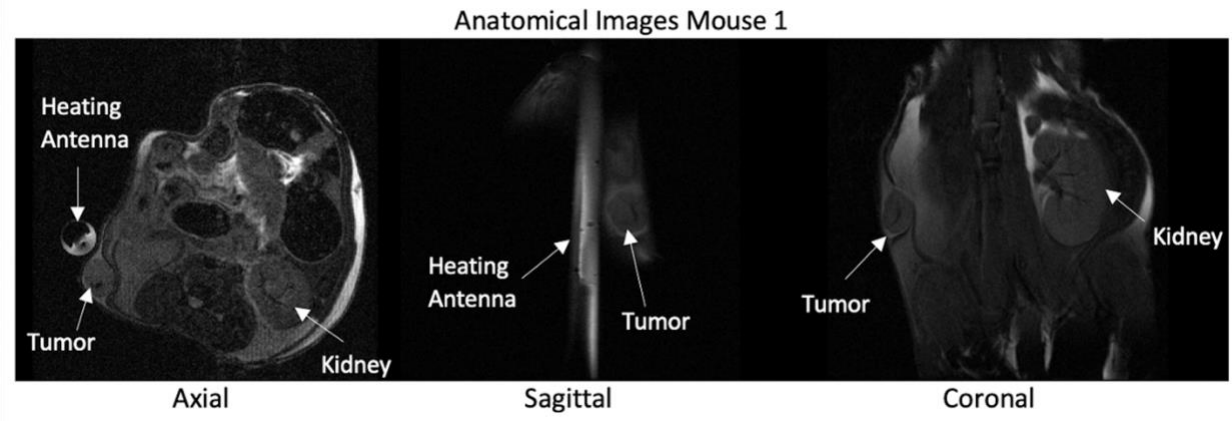

Figure 3.5: Anatomical images of the nude mice with HAC15 tumors grown subcutaneously in the axial, sagittal, and coronal view. Tumor location is denoted with white arrows in addition to the kidney and heating antenna placement.

Once loaded into the spectrometer, a series of anatomical images were used to collect information regarding tumor location and size as well as to assist in the positioning of the water-cooled microwave hyperthermia antenna as seen in Figure 3.5. Once the tumor location and antenna position were deemed sufficient, the tumor was heated to approximately 55°C for three minutes. T_1 maps were taken before and after ablation to investigate if T_1 mapping could provide insight into the level of success of an ablation application. Similar to the previous trial with KPC cells, a saturation recovery T_1 pulse sequence was employed to recover the relaxation maps with

an additional RARE factor to rapidly fill the maximal amount of k-space and conserve time under anesthesia.

After imaging, the data was converted to dicom format for further analysis. The identical protocol outlined for the KPC trial was repeated to determine the color coded T_1 maps. However, the regions of interest selected for comparison against the tumor relaxation values had to be adjusted due to the subcutaneous placement of the tumor model. Since the tumor location was no longer predicated on pancreas location, the additional ROIs for comparison were moved to the erector spinae present in each image and when apparent, the spleen or kidney. Unfortunately, due to the variation in the visibility of the kidney or spleen in the slices captured during T_1 mapping, kidney and spleen ROIs were not consistent between images, thus the ROI of muscle tissue was predominantly used as the reference between subjects. Contrary to what had been seen for the T_1 mapping with the KPC tumor model, the mice with HAC15 implantation presented with a depressed T_1 value for tumor tissues compared to surrounding tissue and organs before ablation. Additionally, HAC15 tumor models presented with more similar T_1 values between cancerous and healthy tissue than the KPC model unless the tumor was extremely well isolated from other tissues as seen in Figure 3.6. The relative amount of T_1 depression was likely due to the

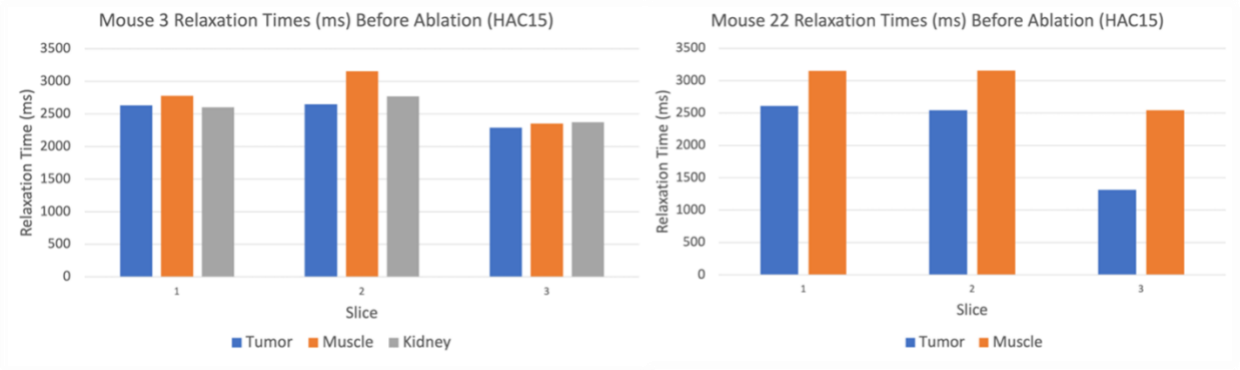

Figure 3.6: Charts denoting average T_1 relaxation times for select ROI across three slices for Mouse 3 and Mouse 22 before ablation. Mouse 3 demonstrates the trend many samples had of

more similar relaxation values between cancerous and healthy tissues while Mouse 22 illustrates the relative depression of T_1 relaxation values compared to healthy ROIs.

individual tumor characteristics based upon its stage of development during imaging. However, the T_1 values did not change with tumor size indicating consistent relaxation value calculations dependent upon tissue type as seen in Figure 3.7.

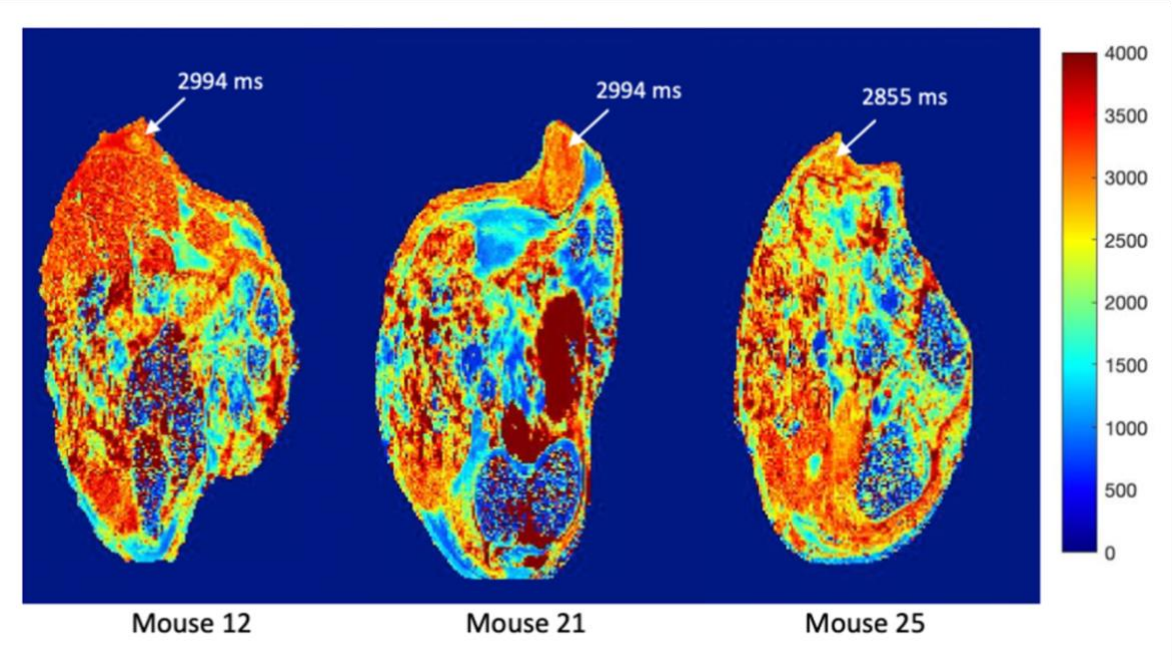

Figure 3.7: Comparison of average T_1 relaxation times for various tumor sizes across three subjects.

Analyzing the T_1 maps can be complicated due to the sensitivity in T_1 maps to variation of different tissue types and the close proximity of the region of interest to surrounding healthy fascia. However, after careful segmentation and comparison of all T_1 values before and after ablation a few key trends became apparent, particularly after comparison with H&E staining of the tumors for histopathological analysis. The T_1 relaxation value difference between tumor and healthy tissue decreased in mice which had a successful ablation based upon findings in the

Hematoxylin and Eosin (H&E) staining. This metric is of value because only one mouse experienced partial ablation, with 80% of the tumor remaining un-ablated and 20% of the tumor periphery being properly ablated. The T_1 relaxation data for Mouse 22 indicated an increase in the difference of the T_1-relaxation between tumor and muscle over all three slices, whereas in mice which had successful ablation identified via H&E demonstrated a decrease in T_1 relaxation time differences. Figure 3.8 demonstrates the differences in T_1 relaxation rates between healthy muscle and tumor for Mouse 12 which was determined to be successfully ablated and Mouse 22, whose tumor was only marginally ablated.

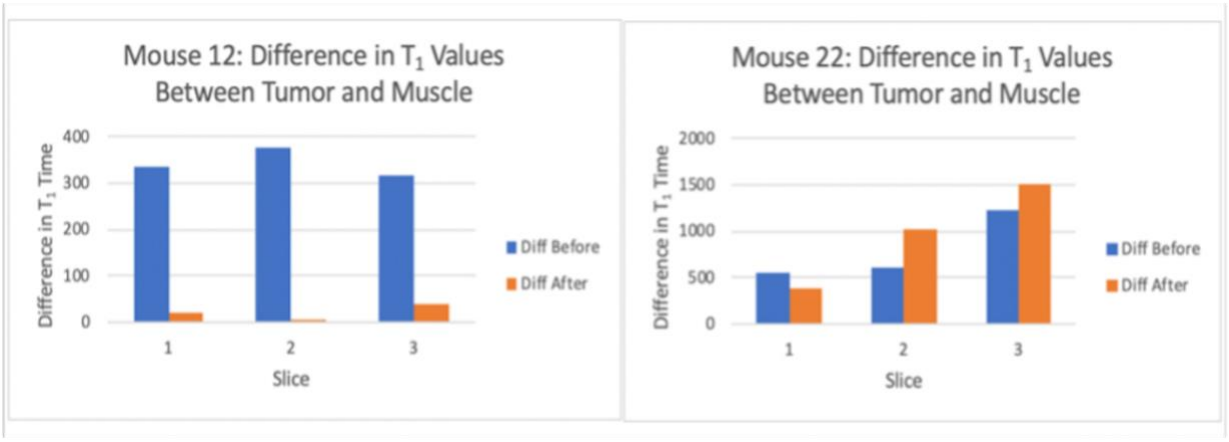

Figure 3.8: T_1 relaxation time differences between muscle and tumor before and after ablation for Mouse 12, which was determined to have a successful ablation based on H&E staining, and Mouse 22, where it was found that only 20% of the tumor periphery had been adequately ablated.

Discussion

The T_1 mapping method of choice was saturation recovery with an additional RARE factor in order to quickly and efficiently fill k-space. The additional refocusing echoes lead to a further decrease in imaging time for a full T_1 map across three slices being acquired in approximately 5 minutes. While RARE imaging can suffer from decreased signal due to loss in

the refocusing pulses, at ultra-high-field-strengths, and with a conservative 8 refocusing pulses, this loss was negligible and did not influence the final images. However, the saturation recovery methodology is known to suffer from decreased precision due to a decreased dynamic range[41] and predisposition to increased noise.[25] This did cause some complications with fairly wide standard deviations between T_1 measurements for each ROI, but it is difficult to discern if the exact cause is due to decreased precision from the SR methodology, or genuine heterogeneity of certain tumor models and organ structures. Alternatively, the accuracy of the saturation recovery methods combined with ultra-high-resolution imaging techniques allows for the discrimination of infinitesimal changes in tissues between voxels, accounting for changes as small as muscle striations, desmoplastic formations, and migrations from the tumor center, and if any of these regions are caught simultaneously within the segmented ROI, i.e. a transition state with a voxel accounting for some percentage of stromal barrier and some percentage of tumor center, could perturb the standard deviation due to less than ideal segmentation strategies. Thus, it is difficult without trialing additional T_1 mapping techniques at varied resolutions to determine the exact cause of the variance between relaxation values within a particular ROI, however it is safe to assume it is likely due to a combination of factors mentioned above. The T_1 values recorded were confirmed with alternative processing software from Bruker, which returned the relaxation rates for each of the respective regions of interest mentioned above, thus analysis software is an unlikely source for the variation.

The initial study which used a murine model with orthotopically implanted KPC cells to generate PDAC tumors resulted in a series of T_1 maps which provided discrimination between the tumor centers and regions of desmoplasia as the slices moved away from the tumor center. Additionally, it provided insight into the formation of secondary tumors and ascites which are

common complications with PDAC.[36] Figure 3.9 demonstrates a T_1 map that captures regions of desmoplasia surrounding the tumor, the tumor center, and an ascites formation across three slices with an anatomical image for reference.

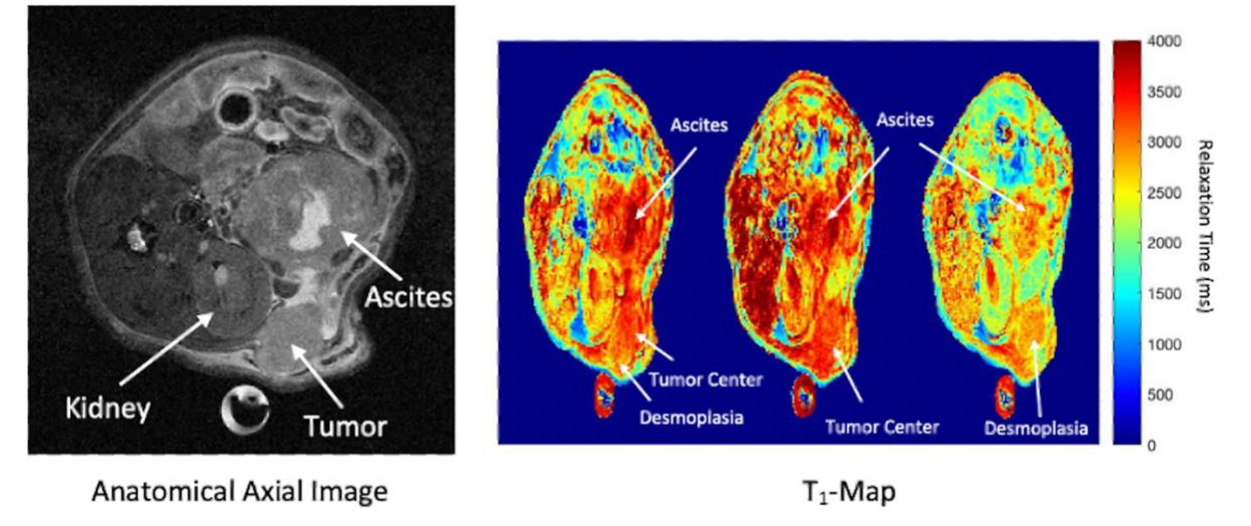

Figure 3.9: Anatomical axial image depicting tumor location, ascites formation, and kidney location in reference to the corresponding T1 map indicating the longitudinal relaxation time for different tissue types and malignancies for the KPC murine model.

The second application probed with T_1-mapping involved determining its potential as a modality to ascertain the efficacy of ablation of adrenocortical carcinomas. Initial hypothesis involved the expectation of different T_1 relaxation values in treated areas due to the relationship of T_1 relaxation times and temperature, where the T_1 relaxation time is expected to vary proportionally with the temperature of the region.[34] There was no consequential difference in T_1 values of the tumor tissue before and after ablation that indicated a change of temperature to the region. This is likely due to the length of time between ablation application and acquisition of the T_1 map, where several minutes had passed between each pulse sequence. However, this lack of variation also indicates that cooling methods used to keep the animals comfortable and prevent

skin scalding were effective, as instrument core temperature and water cooling of the microwave hyperthermia applicator were used in conjunction to maintain the core temperature of the mouse around 36.8 – 37 °C. Additionally, mice were not re-imaged in the days following ablation, so regions of necrosis and cell death were not examined. However, should future experiment involve repeated imaging, alterations of T_1 relaxation times would be expected due to necrosis leading to an increase in relaxation time if ablation was successful. Alternatively, there was a metric that was not anticipated that may provide an alternative data point to examine when monitoring for the success of ablation therapy. Of the mice that were imaged with T_1 mapping before and after ablation, and then subsequently had their tumors excised and histology performed, those which were deemed successfully ablated tumors had T_1 relaxation values which more closely resembled the relaxation values of surrounding "healthy" tissue after ablation compared to the differences between longitudinal relaxation values of tumor and healthy tissue before ablation. For Mouse 22, which was found to have incomplete ablation from H&E staining, with only 20% of the periphery of the tumor adequately ablated, there was no decrease in the difference between the T_1 relaxation values for tumor and healthy tissue and instead showed an overall increase in the difference between longitudinal relaxation values. Thus, the T_1 differences between tumor and healthy tissue before and after ablation may provide insight into ablation success, but further study with intentional partial ablation and complete ablation trials will be necessary to verify this finding. Therefore, there is potential for the similarity of T_1-relaxation values between tumor and surrounding healthy tissue to be a metric for success in therapeutic efficacies rather than mapping the relaxation value of the tumor tissue alone. Overall, the findings did not support the hypothesis that T_1 mapping could be used as a metric to determine if an ablation had been successful. The findings indicated no significant difference in

the longitudinal relaxation times for ablated tumor tissue and non-ablated tumor tissue. However, the findings did help support the proposal that T_1 mapping is sensitive enough to distinguish between tissues in close coordination with one another with statistically significant differences of measured longitudinal relaxation times between the spleen, kidney, and tumor (P<0.5) as seen in Table 3.1. Perhaps most enlightening was that T_1 mapping was sensitive enough to also distinguish between different cell lines, with significantly different longitudinal relaxation values

Table 3.1: P-values for the differences between relaxation times in various tissues for each tumor model.

Tissue Comparison	P-value
KPC Tumor vs. Kidney	1.342×10^{-8}
KPC Tumor vs. Spleen	1.019×10^{-5}
Spleen vs. Kidney	0.880
KPC Tumor Before Hyperthermia vs. KPC Tumor After Hyperthermia	0.426
HAC15 Tumor vs. Muscle	0.010
HAC15 Before Ablation vs. HAC15 After Ablation	0.980
HAC15 vs. KPC	4.135×10^{-6}

for tumors due to HAC15 and KPC cells (P<0.001). In the initial trial which involved a KPC model, the T_1 maps indicated that tumor tissues had an elevated relaxation time (average of 3227 ms) compared to surrounding healthy tissues whereas in the HAC15 the inverse was true, where the T_1 maps indicated a depressed relaxation value for tumor tissues (average 2790 ms) compared to surrounding tissue types, a full scope of the relaxation times for various tissues can be seen in Table 3.2.

Table 3.2: Average T_1 relaxation times (ms) for various tissues.

Tissue Type	Average T_1 Relaxation Time (ms)
KPC Tumor	3227.16
HAC15 Tumor	2790.90
Kidney	2777.94
Spleen	2793.36
Muscle	3022.63

This follows the different presentations of the tumor structures which would lend themselves to elevated or depressed relaxation times. The KPC tumors which model PDAC in humans presents with more heterogeneity, desmoplasia, fibrosis, and inflammation, where regions of high fibrosis and collagen have elevated rates of longitudinal relaxation due to the cellular matrix. Conversely, in the HAC15 tumor model, which replicates adrenocortical carcinoma in humans, present with more homogenous tumors with decreased incidences of desmoplasia and lend themselves to different physiological complications such as hypertension and have a more solid tumor construction with less changes between tumor centers and exteriors leading to depressed relaxation rates which tend to be more consistent across slices. These findings lend themselves to exciting future variations of this experimentation with alternative disease models which have high levels of comorbidities with different effects on the subjects. It also leaves room for further investigation into the ability of T_1 mapping as a technique to discern cancer type based upon relaxation value presentation at early stages of disease in comparison to surrounding tissues, as PDAC often presents as various alternative diseases in early stages making early diagnosis difficult. This would require a library of relaxation times for various tumor models to serve as a benchmark for comparison and repeat studies with early starting points to capture tumor growth at earlier stages and monitor the changes to the relaxation rates.

Furthermore, the voxel-wise mapping of scalar T_1 relaxation values makes T_1 mapping an excellent candidate for deep learning techniques which may remove the need for radiologist interpretation at all. Instead, categorization of cancer type or staging based on a neural network would be possible.

Conclusion

Invasive strategies such as biopsies and radiative based scans such as Positron Emission Tomography (PET) and Computerized Tomography (CT) are the leading methods for diagnosis and staging of numerous cancers, but leave patients exposed to potential routes of infection, painful procedures, and radiation. T_1 mapping has demonstrated the sensitivity to discern between cancer types based upon T_1 relaxation values and discriminate between healthy and malignant tissues. The saturation recovery method has been shown to provide highly detailed T_1 maps without the need for alternative correction commonly seen with the inversion methods frequently employed clinically, leading to rapid and simple relaxation rate determination. Due to the high level of discrimination between tissue types, T_1 mapping does not require the application of contrast agents, further avoiding any potential toxicities and side effects from contrast agent injection common with other imaging techniques. Relaxation mapping can be repeated without exposing the patient to harmful radiation and is acquired in relatively short time frames which allow for quick integration into currently applied imaging protocol. Overall, the methodology is already adapted and integrated into cardiac imaging techniques making translation to other disease types and anatomical regions feasible within a short time frame.

Chapter 4 - Optimizing Diffusion Tensor Imaging

Diffusion tensor imaging's (DTI) origins trace back to 1994 with Peter Basser when he proposed its potential application towards MR imaging based upon diffusion work done with nuclear magnetic resonance spectroscopy.[5,17] From there it has quickly gained popularity in various disciplines due to the ability to discern microstructures, diffusive properties, and cellularity without the need to perform invasive procedures in both clinical and research atmospheres. Unfortunately, due to the prolonged imaging time required to acquire DT images, its application is often impractical for clinical usage or animal-based studies. If diffusion imaging is utilized in these settings, MR technologists are often resigned to using lower resolution parameters with few diffusion directions.

Condensing Imaging Time

From a practical standpoint it is easy to rationalize that an animal should not be subjected to many hours of sedation for the safety of their own health, and that human subjects can only sustain strict positioning for a limited period of time. Thus, throughout this study, our goal was to focus on determining which components dramatically impacted overall scan time, define methodology to condense imaging into more reasonable time frames, and then measure the effect these alterations had on image quality. In order to accrue enough data to make reasonable comparisons and test parameter variation across a broad range of potential subjects, imaging sets were acquired over a two-year period with various samples ranging from rat skulls and live mice to fish brains. Concurrently with optimizing image parameters to condense imaging time, additional modification to analysis methods were pursued to minimize artifact influence and streamline the post processing pipeline.

Diffusion imaging is constrained to the same limitations as anatomical imaging such as T_1 and T_2 weighted imaging, in that the optimal repetition time (TR) and echo time (TE) must be selected depending upon the tissues or samples being imaged to maximize the contrast between tissues of interest. However, diffusion imaging is often complicated by decreased signal due to the impact of diffusion gradients, which nullify freely diffusing protons and instead result on recognition of protons which are only aligned with or against the applied gradient.[22] Yet, within a defined range of potential TR and TE targets, it is possible to condense imaging time based upon manipulation within a narrow range of these values, but it often comes at the expense of contrast to noise ratios, as suboptimal TR pulses will result in less signal separation between the individual tissues and acquiring signal at a less than ideal echo time may miss the full signal when recorded.

While altering the TR/TE times would seem like a quick and efficient way to condense imaging time, it is a difficult parameter to alter without drastic consequences to image quality. This is due to the aforementioned requirement that TR be selected in order to provide the highest differentiation between tissue types and TE being optimally selected to collect at the maximal signal output. Although most statements up to this point have indicated there is little room for variation to maintain optimal image contrast, small adjustments may provide the necessary room to accommodate some variation with respiration gating when imaging live animals. Gating is method in imaging where a respiration pad is used to monitor the breaths of the subject and trigger the RF pulse between respirations to record signal between moments of inhalation and exhalation to minimize motion artifacts. With small modifications in TR and TE time it is possible to more closely match these gating parameters to minimize “dead” time during image while simultaneously decreasing the incidence of motion artifacts. Furthermore, if a shorter TR

time is determined to be sufficient enough to maintain image contrast it is possible to adjust anesthesia (so long as vitals indicate they are neither insufficiently sedated as to be in danger or rousing during imaging nor respirations too depressed to be in danger of expiration) to more closely match these shorter repetition times to acquire more lines of k-space in a shorter period of time. Conversely, more cooperative patients of the humankind can be asked to maintain a more rapid breath hold interval to accomplish the same feat. This is perhaps one of the more direct ways to minorly condense imaging time without dramatic alterations to image parameters.

Perhaps one of the most influential factors which influence scan time is the matrix and image size. While higher resolution images are most sought after, they often require large matrices which increase scan time proportionally with increased phase encoding steps. Though the matrix in the frequency encoding dimension is considered a "free parameter" in MR imaging, which only increases the instances of sampling along the MR signal, it is not possible to continually increase the matrix in the frequency encoding dimension without making accommodation in the phase encoding direction. This is due to the narrowing of pixels that occurs if the PE dimension is less than half of the FE dimension, resulting in pencil like pixels in the final image distorting both diffusion and anatomical information. Additionally, as the PE matrix is condensed, if resolution is maintained with a corresponding shortening of the field of view, the likelihood of aliasing artifacts increases in which the signal "falls back" or is misregistered to the opposite side of the image due to the inability to discern high frequency signals from low frequency signals at low sampling rates.[37] Ideally, the FE dimension is selected to correspond with the largest anatomical dimension of the subject. This is done so the sampling frequency can be increased accordingly and minimize the likelihood of Gibbs artifacts - where bright signals that correspond to rapid contrast changes between voxels, and wrap-around

artifacts - in which the signal is mis-registered to the other side of the imaging due to the high frequency signals at the edges of the field of view being indiscernible from the low frequency signal. However, these artifacts can still occur in the PE direction as well even with a high degree of sampling in the FE direction if all aspects of image size are not accounted for, such as a PE dimension that does not account for the full scope of sample in the alternative dimension. The risk of aliasing in the PE direction is why the PE dimension is frequently selected as the shortest anatomical dimension as each additional PE step requires additional applications of the PE gradient in order to fill the additional lines of k-space. Unfortunately, matrix size (and thus image resolution) can have a dramatic impact on imaging time as seen in Figure 4.1, where the larger the matrix and the higher the resolution the longer the scan time. This is further compounded in diffusion imaging where individual images at this matrix size must be acquired for each diffusion direction in addition to at least one image with no diffusion gradient applied for a minimum of seven repetitions at the set matrix size. Obtained resolutions are the direct result of chosen matrix and field of view sizes and as seen in Figure 4.2, the resolution plays a critical role in the SNR of the final images. Thus, an easy way to conserve imaging time and improve signal to noise ratios is to use conservative resolutions and matrix sizes. So long as there is not a high level of variation between the anatomical structures with vastly different relaxation times then it is reasonable to expect high quality images even at lower resolutions. Unfortunately, if there are fine tissue structures with high levels of variation between relaxation values this method will result in a loss of detail due to partial volume effects where multiple tissues and signal intensities will be averaged out within a single voxel. Additionally, decreased matrix sizes may suffer from aliasing such as wrap around artifacts due to misrepresentation of high frequency signals.

Figure 4.1: Scan time in seconds versus resolution increases in the z dimension. Resolution in x and y dimension was kept consistent at 100 microns for all samples. Images with six diffusion directions and comparable slice packages for each resolution were chosen for comparison.

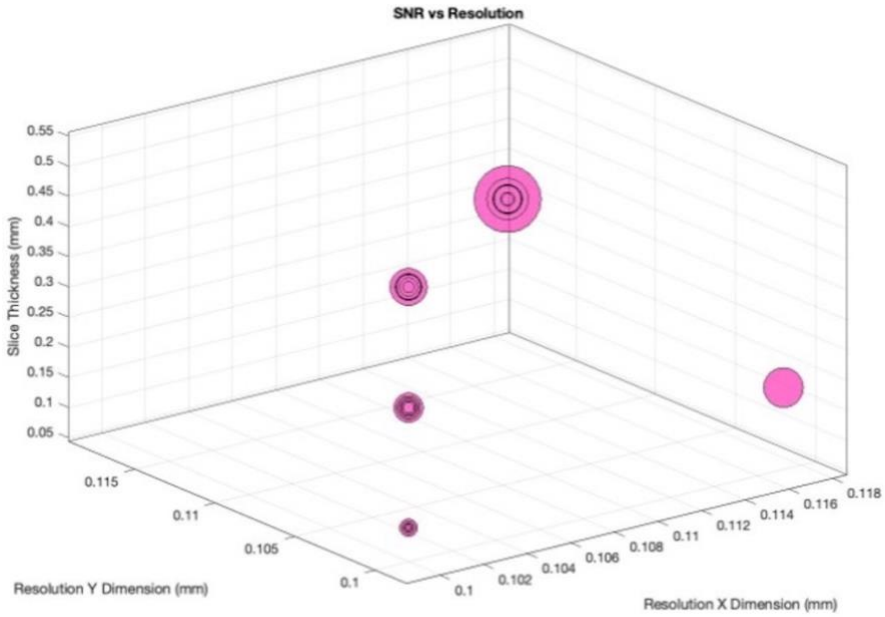

Figure 4.2: A bubble graph of the signal to noise ratio (SNR) with respect to resolution in each dimension indicating higher SNR values by larger bubbles sizes with greater instances denoted with black rings within the bubbles.

An easy method to combat potential aliasing from decreased matrix sizes which do not encompass the entire subject and instead focus on a region of interest is field of view saturation. Field of view saturation uses additional RF pulses to kill the signal outside of the region of interest but can be slightly difficult to apply. The difficulty in application occurs if placed inaccurately along the region of interest, generally with a small gap between the field of view and saturation band boundaries that is not sufficient to prevent misregistration of signal, and affects the relaxation of protons within the field of view resulting in a wraparound artifact due to the signals at the edge of the field of view being affected by the saturation pulse resulting in bright and dark bands across the image as seen in Figure 4.3. However, sufficiently placed

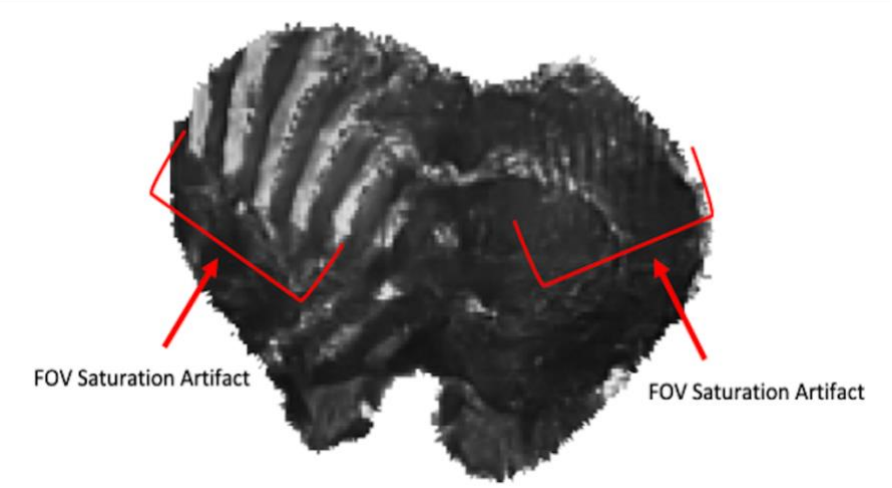

Figure 4.3: Fractional anisotropy image of a rat brain with artifacts due to saturation bands impacting the signal within the field of view due to inappropriate placement.

saturation bands will allow for smaller field of views to be obtained without wrap around artifacts and smaller matrix sizes allowing for condensed imaging time. Figure 4.4 demonstrates an image of a rat brain acquired with an appropriately placed saturation band. This allowed a smaller matrix and field of view (FOV) which only accommodated the brain and immediate

vicinity surrounding it eliminating the need to acquire an image which encompasses the entire skull. Overall, choosing a sensible phase encoding matrix which corresponds to the smallest anatomical direction but is more than half the size of the frequency encoding matrix and employing saturation bands when the region of interest is substantially smaller than the total sample occupying the probe can dramatically decrease the length of scan time needed to generate high resolution images when lower resolution imaging methods are not sufficient.

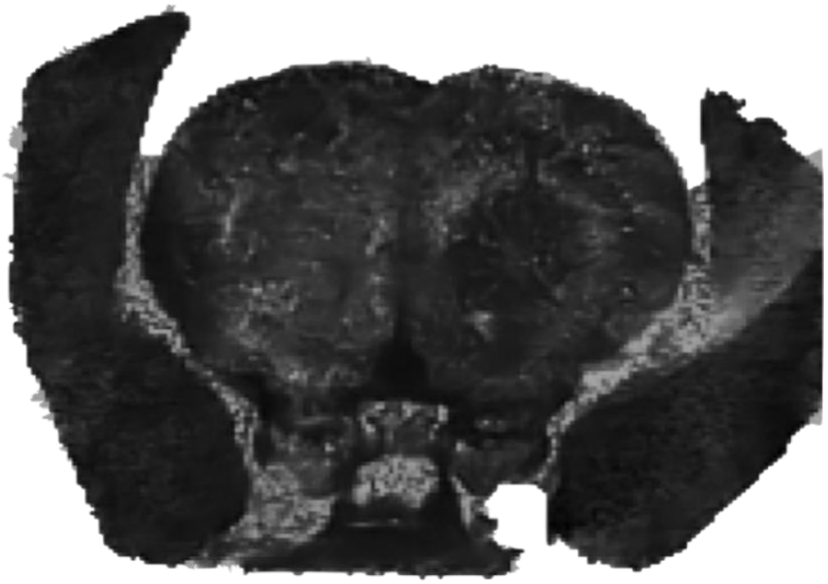

Figure 4.4: Fractional anisotropy image of a rat brain acquired with the help of FOV saturation, demonstrating the difference when saturation bands are placed appropriately allowing for a smaller field of view during imaging which can be used to condense imaging time.

Diffusion tensor imaging's minimal requirements include the acquisition of six diffusion images and an additional non-diffusion weighted image to estimate the diffusion tensor and determine various diffusion metrics.[14,17,38] However, it is not uncommon to add additional gradient orientations and generate additional diffusion images to increase the accuracy and precision of the estimated tensor, but this comes at an increasing time cost with every additional or repeated direction.[39,40] A series of experiments involved the systematic increase in diffusion

directions from 6 to 64 at various resolutions in 3 direction increments to determine the impact on time cost and image quality as diffusion directions increase. Figure 4.5 demonstrates the relative increase in acquisition time corresponding to the increased directionality for each tensor image at various resolutions. The scan time increases with increasing diffusion directions due to

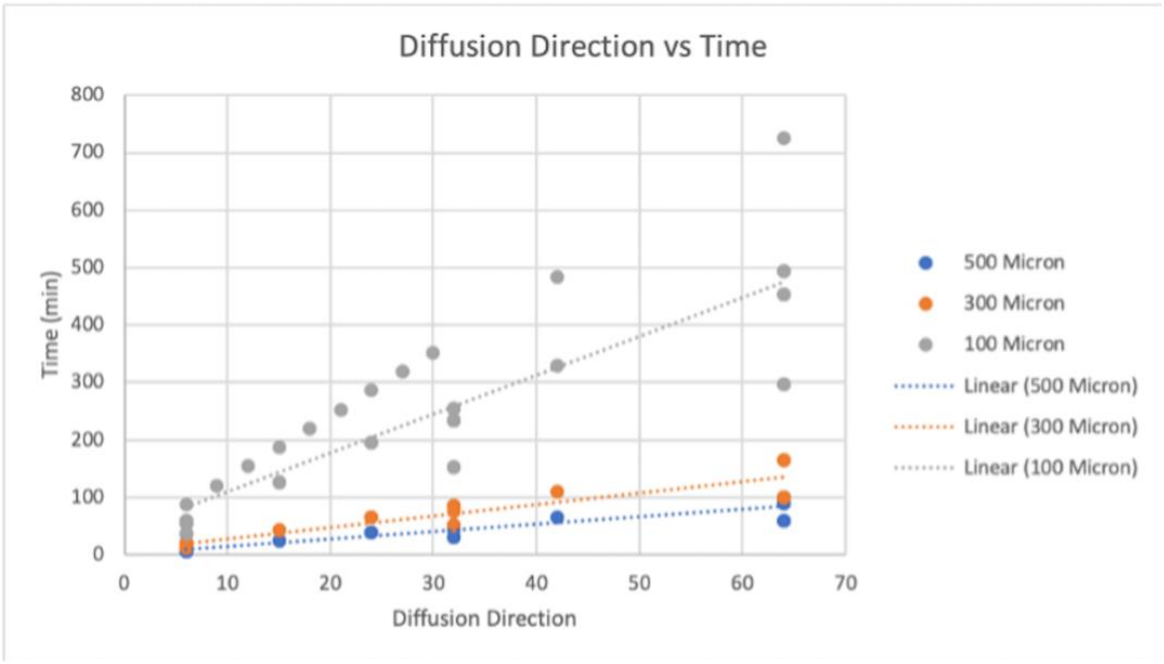

Figure 4.5: Plots correlated to the scan time in minutes with increased diffusion directions at 500-, 300-, and 100-micron slice thickness and 100 by 100 micron in plane resolution.

the need to acquire additional diffusion images for each additional diffusion gradient applied. For studies which are more interested in examining fractional anisotropy and apparent diffusion coefficient images it may not be necessary to produce high direction data sets, as these higher direction images are more important for tractography and higher order tensor studies.[20] Increased diffusion directions are required for tractography based studies in order to improve resolution of fiber trajectory. For studies which focus on utilizing the fractional anisotropy component of diffusion imaging, there has been little observed difference from values derived from six direction scans and higher gradient applications,[39] thus the selection of optimal diffusion

directions to apply will hinge largely on the target of the individual study but condensing the number of diffusion gradients applied will dramatically trim time costs for diffusion imaging.

Single-shot echo planar imaging (EPI) is frequently used to rapidly acquire diffusion data but is predisposed to magnetic susceptibility artifacts and tends to be limited to acquisitions at low resolutions.[5] Segmentation is a parameter which defines the number of segments in k-space which are filled independently in multi-shot EPI sequence and is a method to reduce susceptibility effects. In segmented multi-shot EPI, the phase encoding bandwidth is higher but uses less gradient power thus decreasing the susceptibility effects seen in the final image.[5,41] Additionally, as the number of segments used increases, the duty cycle, or percentage of the time the gradients are applied at full power, decreases, which can help maintain a safe specific absorption rate limit and increase instrument longevity.[42] However, as the number of segments increase so does the required imaging time as seen in Figure 4.6, and lower segmentations can result in mis-registration of voxels in the final image. This mis-registration can be corrected with

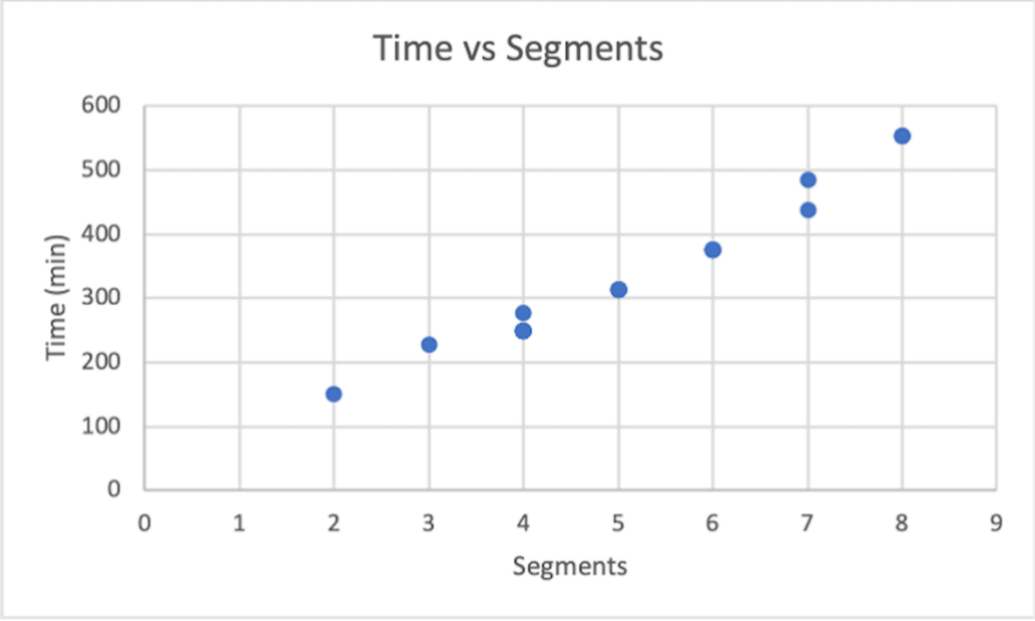

Figure 4.6: Chart demonstrating the time increase with increasing segments acquired through multi-shot EPI. Data sets used for time and segment parameter were 64 diffusion direction with 2

A0 images (images acquired with no diffusion gradients applied) and 1 average with identical matrix sizes, resolutions, and TR/TE times.

motion correction on the finalized images offering the benefit of multi-shot epi without the time costs of acquiring images with more than a few segments.[43] If adequate post-processing techniques are in place, minimizing the segments in multi-shot EPI offers substantial reduction in imaging time while reducing susceptibility effects and offers access to higher resolution imaging. Should lower resolution images be sufficient, and susceptibility affects corrected for in rigorous post-processing steps, signal shot EPI will still offer the most cost-effective acquisition technique.

Finally, an additional method to condense imaging time is to decrease the total number of averages acquired during imaging. Traditionally, additional image sets at each slice and diffusion gradient application are acquired to decrease noise and increase the SNR in the final images.[5] However, with the increased images obtained the scan time increases proportionally, so a scan taken with four averages will take four times as long to complete as one taken with only one average. While increasing image averages is an accessible way to generate high resolution scans with higher SNR than their single average counterparts, with the use of better post-processing pipelines or alternative methods to minimize noise and artifacts, MR spectroscopists have found alternative ways to increase SNR without dramatically increasing scan time. Ultimately, final decisions regarding what time costs required to generate the desired image quality or the minimum required for the metrics needed to be extracted for identification, analysis, and monitoring of various diseases, structure alterations, and treatment progress must be made on a case-by-case basis. The steps outlined above have been identified as major sources of time costs, and their impacts on image quality has been identified to provide a template for consequences to

parameter alterations and potential methods to minimize any drawbacks to the application of these alterations.

Diffusion Imaging Parameter Optimization

When time costs are not a factor, the same techniques which can be used to minimize imaging time can be exploited to yield high resolution, high SNR diffusion data sets. Through investigation of parameter optimization in order to condense imaging time, some components produced drastic effects on image resolution and dramatically increased SNR and are important to mention for those in pursuit of high-quality data. While decreased resolution increases the signal to noise ratio due to the increased proton density for excitation within a voxel, doing so negates the small differentiation between voxels which may be necessary for understanding underlying tissue structure and organization. Figure 4.7 demonstrates the same location for one

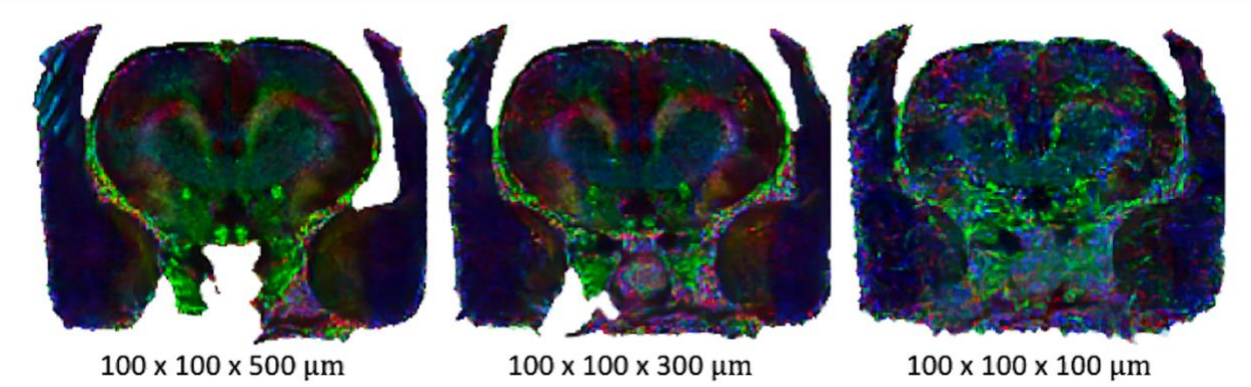

Figure 4.7: Equivalent slices of a diffusion tensor image for a rat brain taken with 64 diffusion gradients applied. Various resolutions demonstrate the relative increase in image noise with increasing resolution.

sample with in-plane resolutions of 100 by 100-micron and slice thicknesses of 500-, 300-, and 100- microns. As seen in Figure 4.8 the signal for 100-micron through plane resolution becomes too weak to discern tissue organization due to the decreased proton density within this region

decreasing signal strength which is further muted due to the application of diffusion gradients. Figure 4.9 demonstrates the lack of information between slices with the lower resolution data sets which can leave large volumes of information regarding tissue organization and integrity

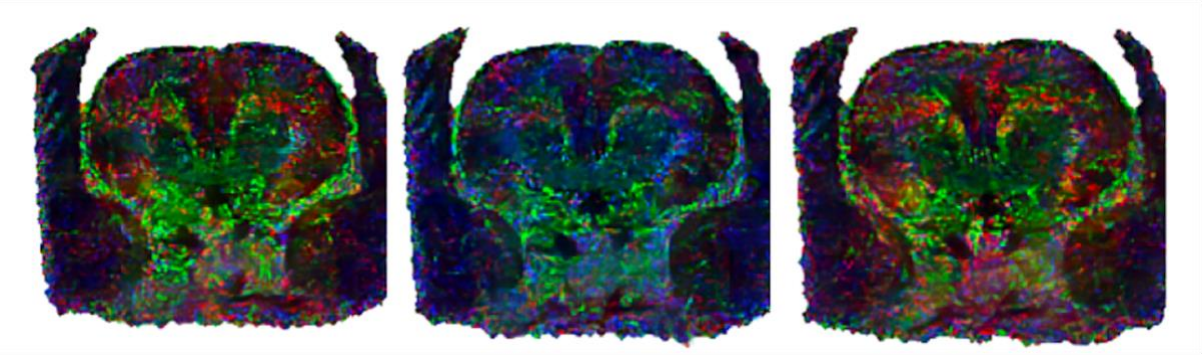

Figure 4.8: Diffusion tensor images taken with 64 directions and 100-micron isotropic resolution in successive slices. While the image demonstrates minimal changes in tissue structure between slices with the ability to closely monitor tissue organization and integrity, the signal is lower causing a decreased SNR which can compromise diffusion measurements as seen in the middle slice (slice 2). The low signal results in unreliable diffusion metrics, such as diffusion orientation (represented by different colors) due to the high levels of noise during analysis.

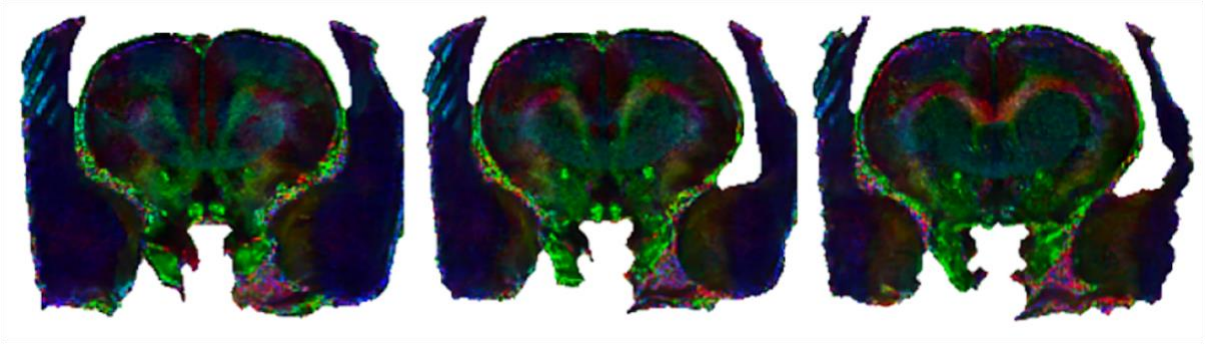

Figure 4.9: Diffusion images taken with 100 x 100 x 500-micron resolution with 64 diffusion gradients applied in three successive slices. Images have increased SNR with more reliable diffusion metrics such as diffusion orientation, but large gaps between slices which result in a lack of structural information for the tissues between each successive slice.

unknown but have the highest signal to noise ratios due to the relative amount of signal in each voxel. Figure 4.10 demonstrates the possible balance with a moderately high-resolution scan that allows for more information to be gained through successive slices while minimizing noise which will perturb diffusion measurement. Thus, careful selection must be made to optimize the number of steps between successive slices without compromising signal intensity. Fortunately,

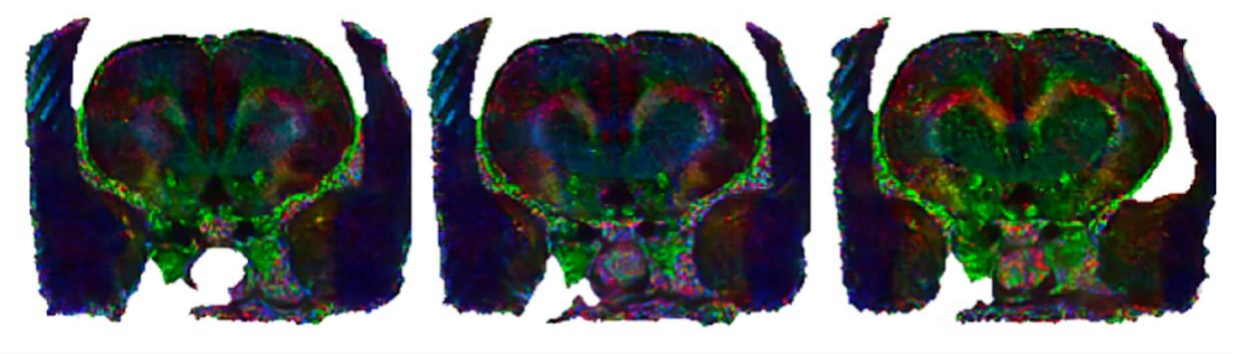

Figure 4.10: Three successive slices of diffusion tensor images at 100 x 100 x 300-micron resolution and 64 diffusion directions. Images demonstrate a balance between details regarding tissue structures and SNR compared to lower and higher resolution options.

an emerging area of super resolution reconstruction may circumvent the decreased signal seen at ultra-high resolution by using a series of low-resolution images to reconstruct the desired high-resolution output. Alternatively, increasing the number of diffusion directions not only clarifies fiber tract orientation but also decreases noise as seen in Figure 4.11. This increased SNR can also be seen with repeated diffusion gradients as well where the gradient is repeated, and the average diffusion signal is computed during processing. Figure 4.12 demonstrates the increased fiber resolution with increasing diffusion gradient applications. Finally, increasing the number of averages acquired during imaging can have a substantial impact on SNR with approximately 120% SNR increase with the addition of only one additional average. Combining these

parameter optimizations with increased resolution, increased averages, and increased diffusion gradient application can yield highly informative diffusion data but comes with an increased time

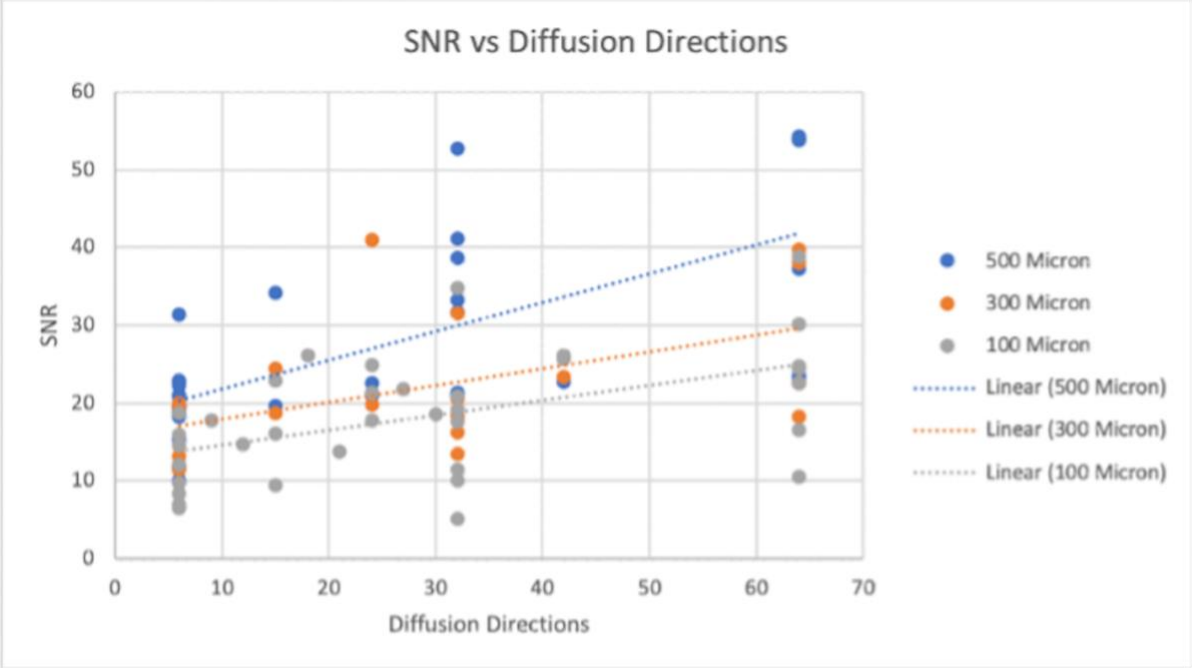

Figure 4.11: SNR vs number of diffusion gradient application at various resolutions.

cost that must be managed with experimental expectations and the output metrics needed. For investigations relying on fractional anisotropy and apparent diffusion coefficient values, parameters optimized for conservative scan times will be more than sufficient to achieve usable results, whereas for studies relying on tractography, increased time costs may be unavoidable to generate high resolution, high direction scans.

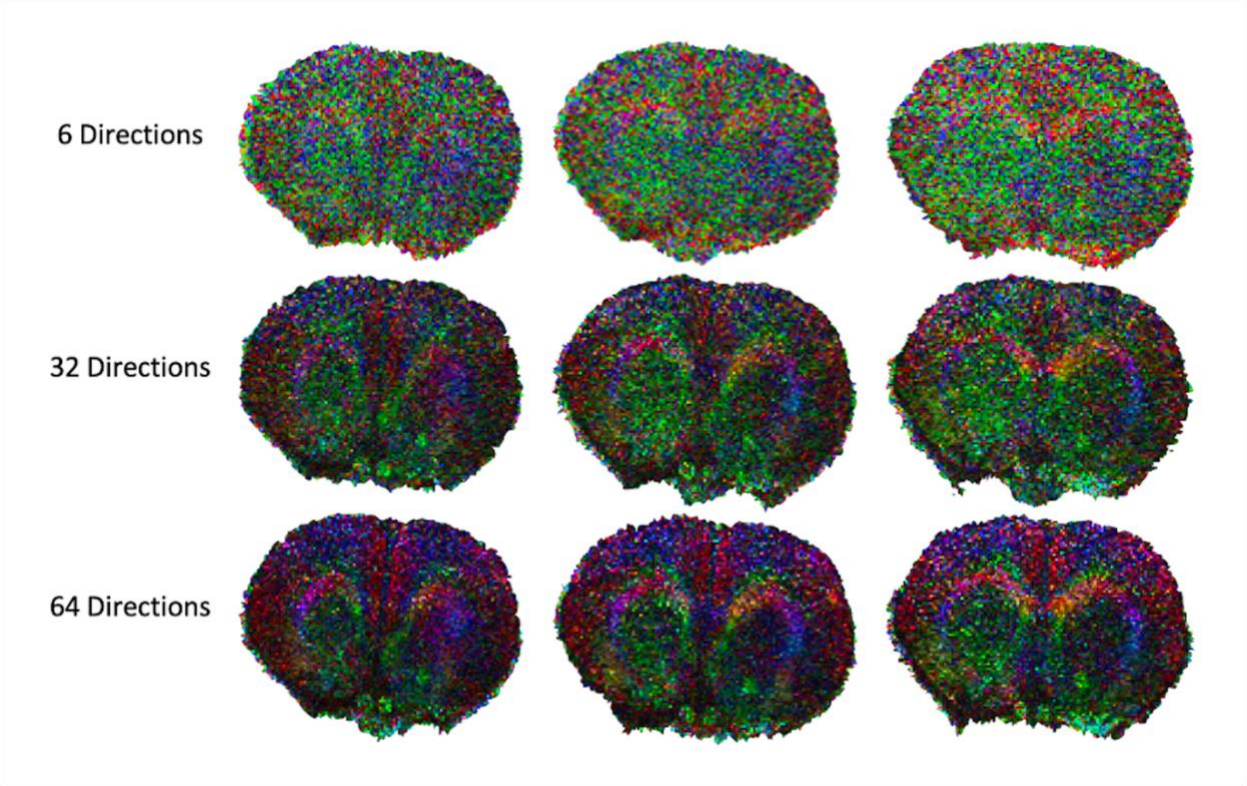

Figure 4.12: Diffusion tensor images of a rat brain across three slices taken with 6, 32, and 64 diffusion directions. As the number of gradients increase, fiber resolution increases and noise decreases. Images are taken prior to noise reduction strategies to illustrate the inherent decrease in noise and fiber resolution from increasing diffusion directions alone.

Post Processing

There are numerous post processing and analysis methods available to scientists as of 2022. Each has their own particular assets and drawbacks, but few will carry imaging processing from the initial collation of raw data to final image analysis and format. While many that are used in a clinical setting will operate with user-friendliness in mind, they tend to have limited applications, and those that are geared more towards research settings tend to be difficult to understand, operate, and modify.[44] Analysis of diffusion data requires a series of steps: the initial image acquisition, correction for eddy currents and B_0 inhomogeneity, re-orientation of gradient

direction based upon eddy current correction, skull-stripping (if applicable), estimation of the tensor and calculation of different image data sets, co-registration of different samples if group analysis are desired, and final artifact/noise removal and suppression. Currently, few software platforms allow for all aspects of analysis to be performed in one location/through one application and are primarily designed to work with clinical instrumentation (GE, Siemens, Philips). When applied to other manufacturers, based upon data presentation in the method data, there is often a compatibility issue with file conversion, gradient table, and b-value extraction, and thus the system is considered underdetermined in most applications.

In order to discern a more user-friendly method, while still providing access to the variety of analysis possible with various diffusion data sets, an experiment over the course of three years involved designing a modified processing technique tested against a variety of sample types, sizes, and desired metric return. Samples ranging from as small as zebra fish to as large as adult sized rat brains were imaged in a 14.1 T (600 MHz), 51 mm bore Bruker Avance III spectrometer with probes ranging from 5 to 30 mm in diameter. Probe sizes were selected depending upon sample size to maximize the signal to noise ratio in final images (in order to maximize signal to noise ratios free space around the sample should be minimized, thus the smallest applicable probe size is often chosen). Imaging software to control pulse sequences, parameter optimization, and initial image viewing was Paravision 6.0.1, and samples were imaged with parameters designed to meet the goal of smaller sub-projects that operated within the broader experiment of optimizing analysis pipelines. The post-processing pipeline was built to be adaptable to a variety of b-values, matrix sizes, sample orientations, slice configurations, 2D and 3D acquisition techniques, and various diffusion gradient directions, both in number of directions applied and orientations.

When the data is initially compiled on the host computer, it is stored in 2dseq file format. The post-processing pipeline can be executed in 2dseq format, but the file will need to be reconstructed into a 4-D matrix based upon information found in the method and acqp files which contain information regarding the pulse sequences and meta data for the diffusion images. Alternatively, the 2dseq file can be converted to dicom files at the host computer, which are traditionally used due to the wide variety of software designed to read medical images accepting this format. Conversely, from either the dicom or 2dseq file, further conversion to NIfTI, NRRD, or Analyze formats are possible if needed for alternative pipelines. However, due to the common regard of dicom files as the "standard" for medical imaging,[45] it is often the primary image extracted and was used as the basis for the initial components of the analysis pipeline. Therefore, after image acquisition, the data was converted to dicom format within Paravision 6.0.1 and stored for later use.

The dicom files are organized by diffusion gradient application, in that if ten slices are acquired during imaging with one A_0, the first ten images in the dicom volume will be the A_0 image for each slice. Then the next set of ten images will be the images based on the signal from the first diffusion gradient applied for each slice, and this will continue on for all diffusion gradients and slices. As a brief aside, the A_0 is the image acquired without the application of any diffusion gradients and will have a contrast similar to that of a T_2 weighted image, although exact weighting will be dependent upon selected repetition and echo times.

The first step in the post-processing pipeline involved corrections for motion artifacts and susceptibility induced artifacts. Susceptibility artifacts, which are common in echo planar imaging and tend to be more common at higher instrument strengths, were largely circumvented by using more segments in the multi-segment EPI sequence used to generate the diffusion data

sets. However, susceptibility artifacts were predominantly corrected with the eddy currents and gross motion in one step. Eddy currents are caused during diffusion imaging when diffusion gradients are rapidly switched on and off and are responsible for shearing and scaling causing image distortion.[14,46] Subject motion, primarily seen in live imaging, can arise from respiration (if gating is not used or insufficient with poor trigger timing), or involuntary movements (head turning in people, involuntary jerks under anesthesia, etc.). Susceptibility artifacts, eddy current correction, and motion artifacts can be corrected through an affine registration to either a T_1 weighted anatomical image, or to the A_0 images. Due to the necessity to conserve time in most small animal imaging studies, this pipeline is focused on extraction of A_0 images for the registration with diffusion images to avoid the need for additional sequences which can become costly for high resolution scans with thick slice packs. If more than one A_0 image was acquired for each data set, they were averaged and collated as a separate image data set. These averaged A_0 are then used as the fixed images for the diffusion sets to be registered to using the affine transformation to account for the shearing and scaling caused by the eddy currents, mis-appropriate of voxels due to susceptibility artifacts (geometric distortions), and rotation and translation caused by possible subject motion as seen in Appendix B.3. Based upon these alterations, the gradient tables must be adjusted accordingly, this can be done with the transformation matrix used in the registration step. The post-registration data sets are then stored as a new data volume and converted to NIfTI files. At this step, the orientation of the subject in the scanner must be considered in order to permute the NIfTI data set into the correct orientation to align with the gradient table and b matrix. It is not uncommon during the conversion for the x and y axis to be flipped, this can be corrected through the permutation or rotating the gradient

table accordingly. The orientation of the sample within the scanner can be found within the method data.

Once the initial NIfTI file is produced, the data can be read into the largest component of the pipeline which estimates the diffusion tensor through an altered script of the work done by Chia-Hao.[47] From the NIfTI file the matrix dimension, number of slices, and diffusion directions are extracted. The gradient table is normalized and adjusted based upon the transformation used in the registration steps. If the b-matrix is accessible depending upon scanner models this can be applied in place of the gradient directions. A mask is determined based using thresholding to create a binary mask which will condense computation time to sample region of interest and exclude background voxels. The tensor is then estimated through a weighted linear approach due to computation speed based on the work done by Basser et.al as seen in Appendix B.4.[38,48] Eigenvalue decomposition is then used to gain the respective primary, secondary, and tertiary eigenvectors which describe the diffusion ellipsoid for each voxel. Using the now determined eigenvalues, the corresponding fractional anisotropy images (FA), axial diffusivity (AD), radial diffusivity (RD), mean diffusivity (MD), and diffusion tensor images (DTI) are calculated. At this point, images are saved in NIfTI format for further analysis. Figure 4.13 demonstrates the images accessible at this stage of the pipeline.

Figure 4.13: The multitude of images accessible after initial analysis through the processing pipeline. Images of the same sample at the same slice demonstrating the DWI, AD, MD, RD, FA, and DTI images.

Once the calculated images have been saved, they undergo a bias correction, which corrects for a low frequency signal due to the magnetic field inhomogeneity that distorts the edges of images and produces regions of hypo and hyper intense signal due to the inhomogeneity as seen in Appendix B.7.[49,50] These artifacts often impact image segmentation and can make further analysis with refined regions of interest more difficult.[49] Images which consist of only two dimensions can have the bias correction done in one pass, where the bias field is approximated and removed from the main image as seen in Figure 4.14. The bias correction

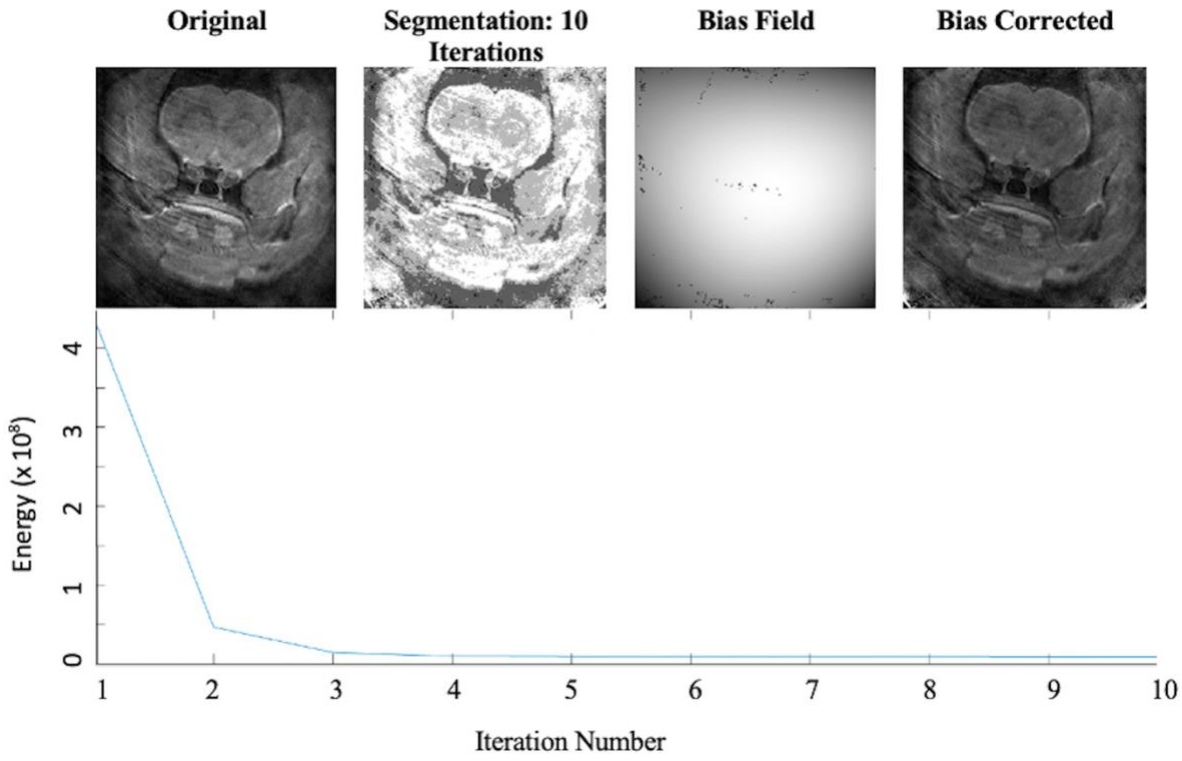

Figure 4.14: Bias field correction on an MR image of a rat brain detailing the original image, the segmentation after 10 iterations, the estimated bias field, and then the corrected image with the energy vs. iteration graph below.

estimation is performed through a multiplicative intrinsic component optimization approach established by Li et. Al, more information regarding the approach mentioned can be found in Appendix B.6.[49,51] For multi-dimensional images (i.e. diffusion tensor images which consist of a series of channels to control the color coding of the tensor), the bias correction is applied to each channel independently and then the final image is reconstructed. Once the bias correction is completed, background noise is removed through a smoothing filter based on a convolution neural network pre-trained through MATLAB with 19 layers as seen in Appendix B.7. This gaussian noise removal will account for the random white noise that exists in every MR image and is necessary at higher resolutions to increase the SNR.

If only select regions of interest are going to be examined, or groups of data sets need to be compared, a rigid registration of the data sets to each other can be performed in order to translate and rotate the samples into the same orientation. Then, regions of interest can be manually segmented, segmented through automatic segmentation techniques, or an atlas can be registered to the data set. This step is best accomplished with the averaged A_0 images which will map with the diffusion data sets due to the higher signals seen in these images. The corresponding mask can then be applied to the corresponding diffusion data sets. Average values can be plotted and compared for selected regions of interest or voxel comparisons can be performed for the entire brain so long as adequate registration has been applied to account for various positioning possible during initial imaging.

Through a series of easy-to-follow steps, as seen in Figure 4.15, we have been able to streamline the data analysis components for diffusion data which yield multiple metrics such as axial diffusivity, apparent diffusion coefficient maps, diffusion tensors, fractional anisotropy images, radial diffusivity, and mean diffusivity images as well as perform registrations and

remove noise in one pipeline without the need for user input at each step. The only dependency upon the user is to set file paths and gradient orientations and the individual components will be

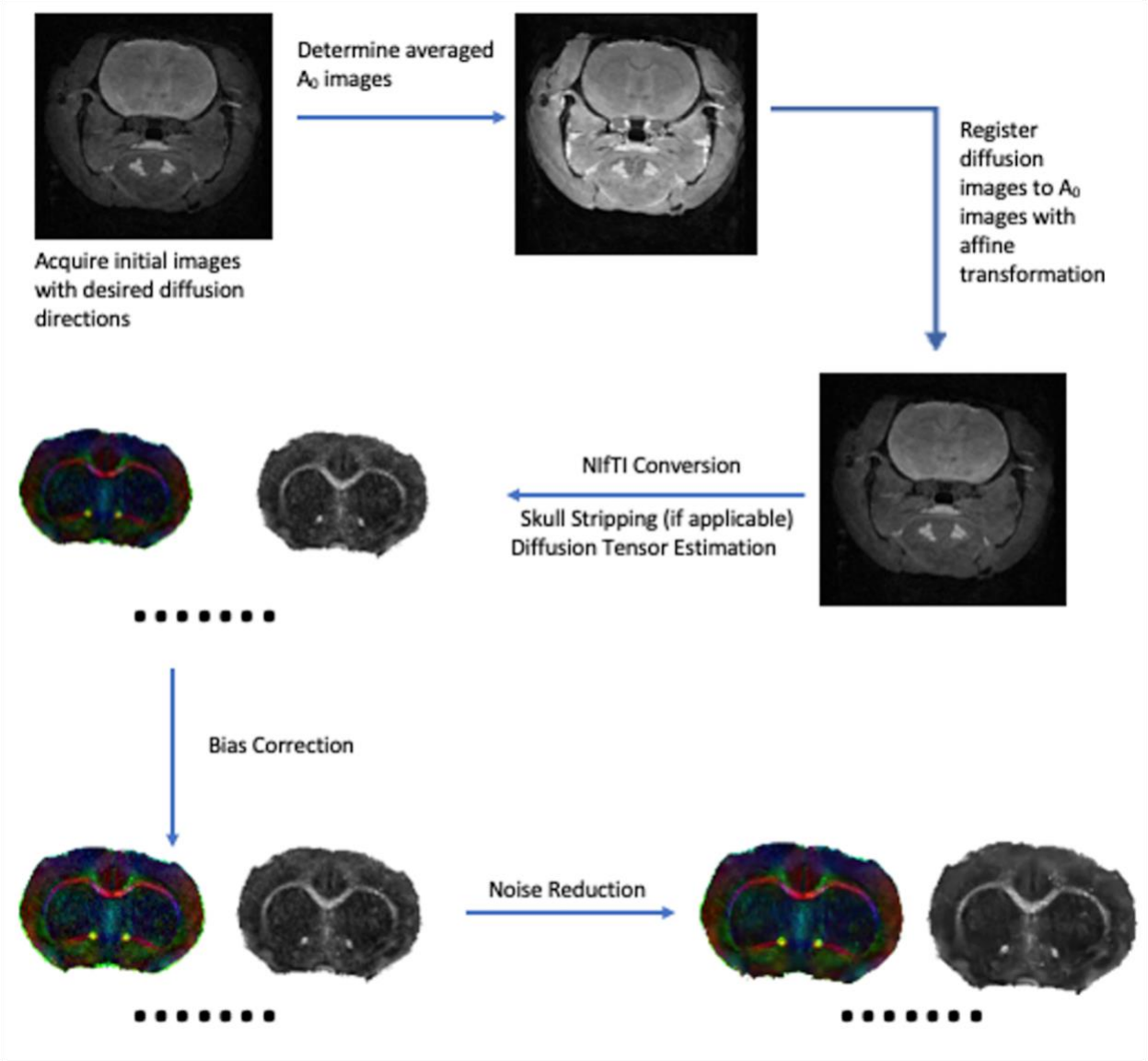

Figure 4.15: processing pipeline from initial image acquisition to final image generation.

computed and stored independently for visualization and analysis. This removes the need to monitor analysis progress and "step in" at each step leading to a more streamlined and efficient method and the full process is outlined in Appendix B. Additionally, this pipeline should be compatible with all manufacturers and removes the complexity from the lack of dicom headers

during NIfTI conversion common with Bruker datasets. These dicom headers traditionally store the b-value and gradient information for other manufacturers but are not carried over with Bruker instrumentation. Additionally, this pipeline is easily tunable if additional alterations are needed.

Conclusion

Diffusion imaging is a key strategy employed by MR spectroscopist to characterize tissue, monitor disease progression, and assess treatment efficacies. However, diffusion imaging is known for its prolonged scan time often making it impractical in many clinical applications. Additionally, as MR instrumentation moves towards ultra-high fields, newly innovative manufacturers become more prevalent in both research and clinical settings. These alternative manufacturers do not always conform to the common data outputs seen in the majority of clinical venues, rendering their data under-determined for analysis in the currently available post-processing pipelines.

Through optimization of diffusion imaging parameters, time costs have been condensed to provide the highest quality images in more reasonable time frames. Applications of FOV saturation bands have demonstrated the ability to acquire more defined regions of interest without the aliasing artifacts commonly seen when excluding various components of anatomy during imaging. Additionally, diffusion gradients have been investigated for their contribution towards signal to noise ratio improvements and impact on primary diffusion estimation. While increasing the number of gradients applied increased the accuracy of diffusion orientation determined, as few as 32 directions was shown to provide optimal signal to noise ratios and may be a sufficient baseline to preserve shorter scan times. Furthermore, resolution was shown to have a profound impact on diffusion metric accuracy. Higher resolution data preserved structural information between slices but had greater variability of diffusion metrics between slices as the

result of increased noise due to depressed signal to noise ratios. Conversely, if resolution was too low, structural information between slices was lost and voxels were prone to partial volume effects making it difficult to apply connectivity studies if needed. Ultimately, resolution and gradient applications must be balanced with time costs and will be dependent upon study goals to maximize the usability of final data. Whether the interest lies more in structural components and general measurements of diffusion or if rigorous connectome assessments are desired as a final endpoint will determine if fewer directions, and higher resolution are of greater value, or higher directions and more moderate resolution targets are optimal.

An alternative post-processing pipeline was introduced to address the difficulty in applying traditional pipelines toward Bruker data, where information regarding imaging parameters found in the headers of most manufacturers is not available. Traditional artifact corrections to account for eddy currents and motion artifacts were addressed through affine registration with data sets not impacted by the gradients applied for diffusion images. The eddy current and motion corrected dataset was then used for diffusion tensor estimation where a full scope of diffusion metrics was determined ranging from axial diffusivity to mode of anisotropy. With the diffusion metrics determined, bias correction can be applied to increase the accuracy of registration between samples and ensure adequate comparisons within experiments between common tissue types. Finally, a white noise reduction can be applied to maintain increased signal to noise ratios and increase readability of final images. Ultimately, a streamlined and tunable post-processing pipeline has been introduced which can be reconfigured for increased speed depending upon final metrics desired by the user. Future development of the pipeline can incorporate a variety of additional analysis strategies, such as inclusion of tractography, and

analysis of multi-shell data which incorporate multiple b-values as well additional artifact corrections.

Chapter 5 - Super-Resolution Reconstructions

Ultra-high field MRI has slowly gained more investigation into its overall safety as interest in utilizing ultra-high field in clinical settings increases. Instrumentation up to 8 Tesla is considered reasonably safe for human subjects older than infancy, and prolonged exposure has been shown to have nominal, if any, impacts.[1] The side effects that were experienced were short lived and homeostasis was achieved within minutes post imaging.[2] Additionally, most complaints regarding imaging from human subjects did not pertain to "true" side-effects in the sense that they were caused by the magnetic field,[3] and instead were general discomfort caused by the noise during imaging and length of imaging time requiring subjects to remain motionless throughout the duration. Some discomfort can be circumvented through headphones and attempts to condense imaging time through parameter optimization and assessing the key factors most important for diagnosis. While moving towards ultra-high-field MRI is a great step towards increasing the depth, accuracy, and precision of information capable of being derived from a multitude of acquisition series, there are some inherent drawbacks as well. Cost is a large factor of procuring new high-resolution instrumentation, and it will be many years before these ultra-high fields are common in a clinical setting, although they have slowly been integrated in some locations. Thus, while obtaining ultra-high-resolution images with high signal to noise ratios are the gold standard for MR imaging, alternative methods must be investigated to obtain access to higher resolution imaging even at clinical field strengths of 1.5-3 Tesla.

Resolution is a key component to imaging; it denotes the size of the voxels with smaller voxels being less prone to partial volume effects but often suffering from lower signal to noise ratios. This can make the final images noisy and difficult to read due to low signal per voxel. Increasing the resolution based on protocol parameter and matrix size selection alone is limited

in a few different aspects. The FOV is limited by probe size and gradient strength and increasing the matrix in the phase encoding dimension comes at the cost of dramatic scan time increases as well.[52] Consequently, acquiring ultra-high-resolution images can be difficult and costly. In order to circumvent the high cost with ultra-high-resolution imaging, an emerging area of super resolution reconstruction offers a potential method to increase the accessibility to high resolution images. Super resolution reconstruction (SRR) is a method which involves acquiring multiple low-resolution images at a series of offsets within the matrix in the phase encoding, frequency encoding, or slice direction.[53] It decreases the time costs associated with high resolution data sets and offers the potential for higher signal to noise ratios as well due to the inherent "averaging" of images used in the reconstruction. Conversely, zero-padding and zero-filling are alternative methods that can be used to increase the resolution artificially and will generate a higher resolution final image while reducing the overall time cost but do not generate true signal information at these zero-filled or zero-padded voxels.

Of the methods for super resolution, zero-padding and zero-filling are often the only ones equipped within the instrumentation. Essentially, the matrix is padded with zeros along the border which would capture background components of the image to prevent the "thinning" of voxels in the phase encoding dimension. This reduces the overall pixelation of the image resulting in an artificially increased resolution with no new information added. Alternatively, zero-filling can be used to fill regions of k-space with zeros. Then, information between regions of data acquired with zeros are interpolated to fill the unknown signal based upon the signals of the neighboring voxels.[5,54,55] Overall, zero-padding and zero-filling do not generate any new information about the regions where signal was not originally acquired and instead are methods

to reduce pixelation and improve acquired resolution based upon the additional clarity in final images.

Alternatively, super resolution reconstruction based upon modeling the image process is capable of producing new images at increased resolution based upon a series of data sets acquired at low resolutions. The basic idea is that low resolution images acquired at an offset contribute new, unknown information to the previous data set acquired that can be estimated and used to generate higher resolution data sets. They operate based upon known geometric information to estimate the high-resolution images which are then used to predict low-resolution images. The error between the estimated low-resolution data set and the acquired low-resolution data set is minimized to improve the estimate for the final high-resolution image.[56,57]

Imaging small samples can make it difficult to achieve high spatial resolution with reasonable signal to noise ratios. This is due to the relatively small amount of space the subject takes within the imaging probe leading to a depressed signal to noise ratio. Additionally, in order to obtain high resolutions, matrix sizes must increase correspondingly due to the high level of tissue variation within a small region which decreases the signal per voxel further exacerbating the SNR dilemma. Therefore, an archerfish brain was used as our initial subject of investigation for the use of super resolution reconstruction. A perfused and fixed archerfish brain was placed within a 20 mm glass tube and filled with phosphate buffer solution and imaged in a 30 mm probe. Anatomical imaging was performed four times to acquire axial slices using a T_2 RARE imaging sequence with a TR/TE of 2500/30.22 ms, a RARE factor of 16, 2 averages, and a matrix size of 256 x 256. One image acquisition series focused on acquiring the high-resolution data set directly with a resolution of 100 x 100 x 100 μm for a total scan time of 5 hours and 41 minutes. The remaining low-resolution images were acquired with a resolution of 100 x 100 x

300 μm and a through-plane offset of 100μm, for a total imaging time of 1 hour and 53 minutes per data set. Images were all acquired in 3 dimensions to keep duty cycles low and acquire the full sample length from olfactory bulb to the end of the cerebellum.

The image was modeled based upon $Y_k = D_k B_k G_k X + E_k,\ k = \{1 \ldots . N\}$ where $\{Y_k\}_N^{k-1}$ are the sets of low-resolution images.[52] D_k is the decimation operator, set based upon the decrease in resolution from a high resolution estimate (X) to the low resolution Y_k. B_k is the blur operator, which is based on the point spread functions of the sensors within the instrumentation.[53,56,57,58] G_k is the geometric transformation which brings all the input images to the same point, and if unknown can be determined based upon image registration.[56] X is the estimated high-resolution images, and E_k is additive noise with a zero-centered mean.[57]

The values for G_k were known based upon the 100-micron shift used in the slice selection direction for the low-resolution series. D_k was set based upon the translation from 100 x 100 x 300 μm data sets to the 100 μm isotropic data set. For super-resolution reconstruction the blur operator, B_k, is considered spatially invariant for simplicity, and the slice selection profile is often used in its place.[57] However, a gaussian function can be used to approximate the blur operator as well. For the reconstruction the pulse profile for slice selection was modeled and a gaussian function was estimated that closely mimicked the excitation profile modeled. Finally, an iterative back-projection method was used to estimate the high-resolution image and generate the super resolution reconstruction. The back-projection approach uses a back-projection kernel to update the current estimations of the high resolution reconstructions such that the low resolution images generated based upon the high resolution estimation have minimal errors between themselves and the ground truth low resolution images acquired.[52,56,57,58] Essentially, the back projection kernel weights contributions from the low resolution pixels to update the current

estimate until the error, $\sqrt{\sum_{k=1}^{N}\left(Y_k - \hat{Y}_k^n\right)^2)}$, is minimized.[52,56,57] The initial guess has been shown to have no impact on the convergence of the final solution, thus an average of the three low resolution images were used to start the reconstruction.[56] Further information regarding the process for high resolution image estimation and super resolution reconstruction can be found in Appendix C.

Figure 5.1 shows the initial images across three slices for the high-resolution image obtained directly and one of the low-resolution images which correlates with the starting image for the high-resolution data set and the following three slices afterwards. The lower resolution

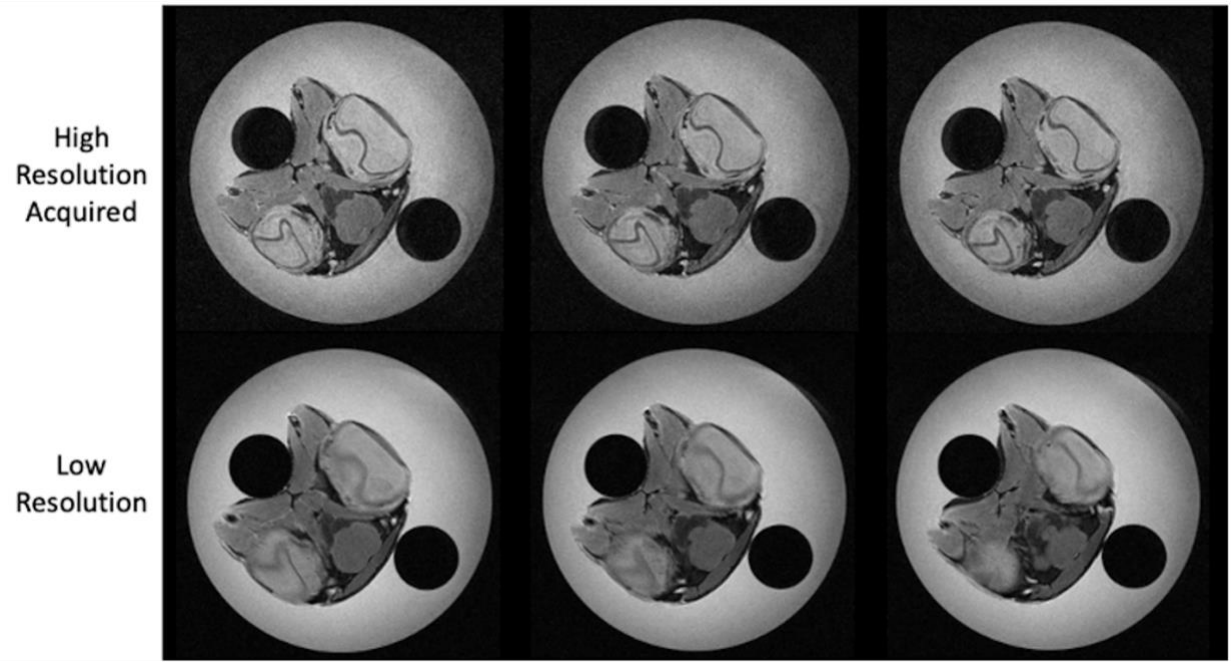

Figure 5.1: Comparison of directly acquired high resolution data set with 100-micron isotropic voxel sizes of an Archerfish with a T_2 RARE image sequence. Below is the comparison with the directly acquired low resolution data set with 100 x 100 x 300-micron voxel sizes which demonstrates the impact of the partial volume effects, more pronounced in the eyes, and the greater jumps between slices.

images have a higher signal to noise ratio due to the greater level of signal encapsulated within each voxel but suffer from partial volume effects and have greater variation between slices which make it difficult to get accurate assessments of the brain architecture for small samples. Figure 5.2 is a series of demonstrative slices of the reconstructed image compared to the matching slice for the directly acquired high resolution data set. It was possible to reconstruct the high-

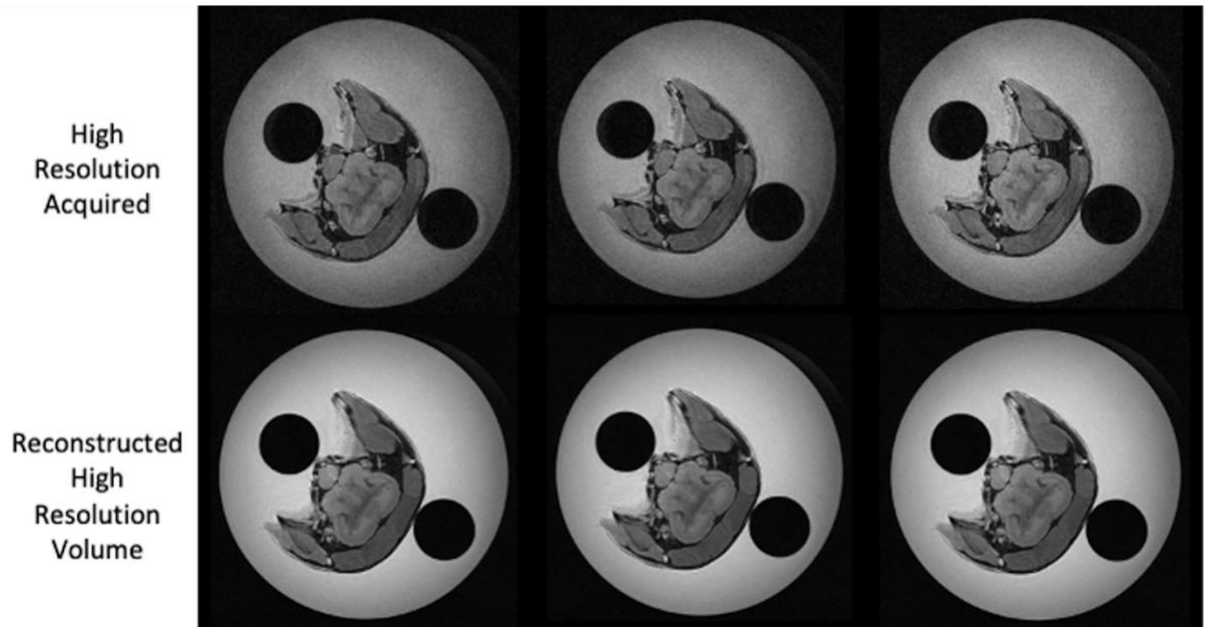

Figure 5.2: Comparison of the directly acquired high resolution data set versus the reconstructed high-resolution volume using iterative back projection with three low -resolution data sets.

resolution image with reasonable agreement to the directly acquired high resolution data with a minimal number of low-resolution data sets to work from (three). The signal to noise ratio and contrast to noise ratio for the directly acquired images were 6.171 and 5.000, while the signal to noise ratio and contrast to noise ratio for the reconstructed images were 8.487 and 7.428. This is predicted to be a smaller improvement for the SNR compared to reconstruction with a greater volume of low-resolution data sets. However, as an initial introduction to SRR, the results demonstrated promising results of closely matching ground truth data through iterative back

projection methods with reasonable increases in not only the signal to noise but also contrast to noise ratios for the final images. Overall, the SNR for the high-resolution images acquired directly was lower and the images took longer to acquire than the reconstructed high resolution data set, indicating SRR as an advantageous method for clinical and research applications using MR technology.

Based upon the success of the pilot study an additional series involved acquiring low resolution data sets in greater volumes in order to examine the effect of the number of low-resolution data sets acquired and the impact on the SNR for the final reconstructed image. Ex-vivo rat brains were imaged in a 28 mm flat bottom glass MRI tube immersed in Fomblin and imaged in a 30 mm probe. Imaging parameters were as follows: TR/TE = 2500/22 ms, RARE factor of 10, 1 average, a matrix size of 289 x 289 and a slice thickness of 0.300 mm for the low-resolution data sets, with 12 data sets acquired with a 25 μm offsets in the slice selection direction with a scan time of 1 hour and 29 minutes and 50 seconds per low resolution data set, or 17 hours and 58 minutes for all twelve low resolution data sets. The high-resolution data set was acquired with 12 averages and a slice thickness of 0.100 mm for a resolution of 100 x 100 x 100 μm and would have taken 53 hours and 54 minutes to acquire. In order to maintain a reasonable time frame fewer slices were acquired (from over 200 to 10) to condense imaging time to 25 hours.

The use of 12 images acquired with a 25 μm slice offset allowed for the reconstruction with 3 images at a 100-micron offset, 4 images at 75-micron, 6 images at 50-micron, and 12 images with a 25-micron offset in order to determine how greatly the number of starting low-resolution images influences the high-resolution reconstruction. The reconstructions were performed using the same method outlined previously with iterative back projection. Figure 5.3

shows the reconstructed images for the varying number of low-resolution sets used in the reconstruction and the corresponding effect on SNR and CNR. As the number of low-resolution

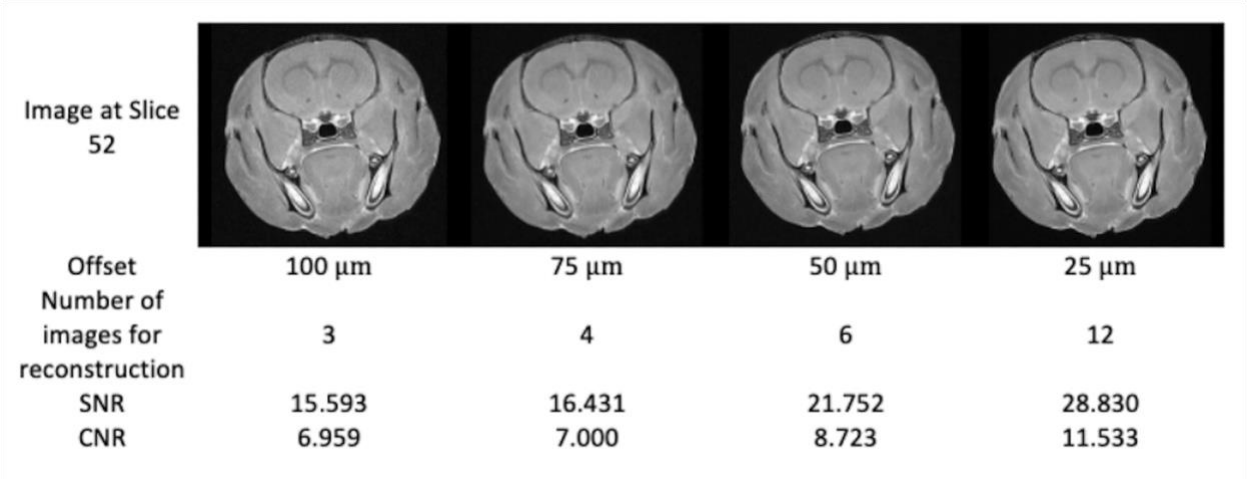

Figure 5.3: Four images from the same slice (slice 52) of the same rat brain reconstructed to a resolution of 100 x 100 x 100-micron from various low resolution data sets with different offsets. The corresponding SNR and CNR for the reconstructed images show increasing SNR and CNR as the number of low-resolution data sets increases.

inputs increased, so did the SNR and CNR, although there is decreased benefit in going from 6 to 12 low resolution inputs. However, the original high-resolution data set directly acquired had an SNR of 78, indicating that for larger sample sizes, which will be adequately sized for the probe used during imaging, SRR may not give comparable SNR measurements compared to direct acquisition although the CNR of 1.08 indicates SRR gives better intensity separation between tissues. Ultimately, for the experimental parameters listed above, a high-resolution data set acquired with 12 averages (akin to using 12 low resolution images in the reconstruction) will take a total of 3,234 minutes to acquire compared to 1078 minutes for the total acquisition of the low-resolution data sets; indicating that in terms of sheer imaging time it may be worth the SNR trade-off to utilize SRR compared to direct acquisition due to imaging time constraints. It is important to mention these metrics are given based upon 3D imaging techniques which take

longer due to the additional spatial encoding for each slice; with 2D image acquisition for the low resolution sets this time difference becomes even more pronounced.

An additional iteration involved using two-dimensional low-resolution data sets in order to reconstruct the three-dimensional high resolution acquired image. In order to maintain a reasonable duty cycle level of under 10%, slices were condensed to 10 slices for a slice thickness of 0.300 mm for the low-resolution data set, and the RARE factor was decreased to 6. Repetition and echo times were 2500 and 22.45 ms for both low- and high-resolution data sets, and average of 2 was used in the low-resolution acquisition. Slice offsets of 50 micron were used to generate two possible reconstructions. Imaging time for the low-resolution data sets was 4 minutes and 5 seconds, for a total of 24 minutes and 30 seconds for the full low resolution data set acquisition of six separate image series. The two-dimensional images do suffer from a lower signal to noise ratio as seen in Figure 5.4 due to the loss of the additional spatial encoding gradient but are much

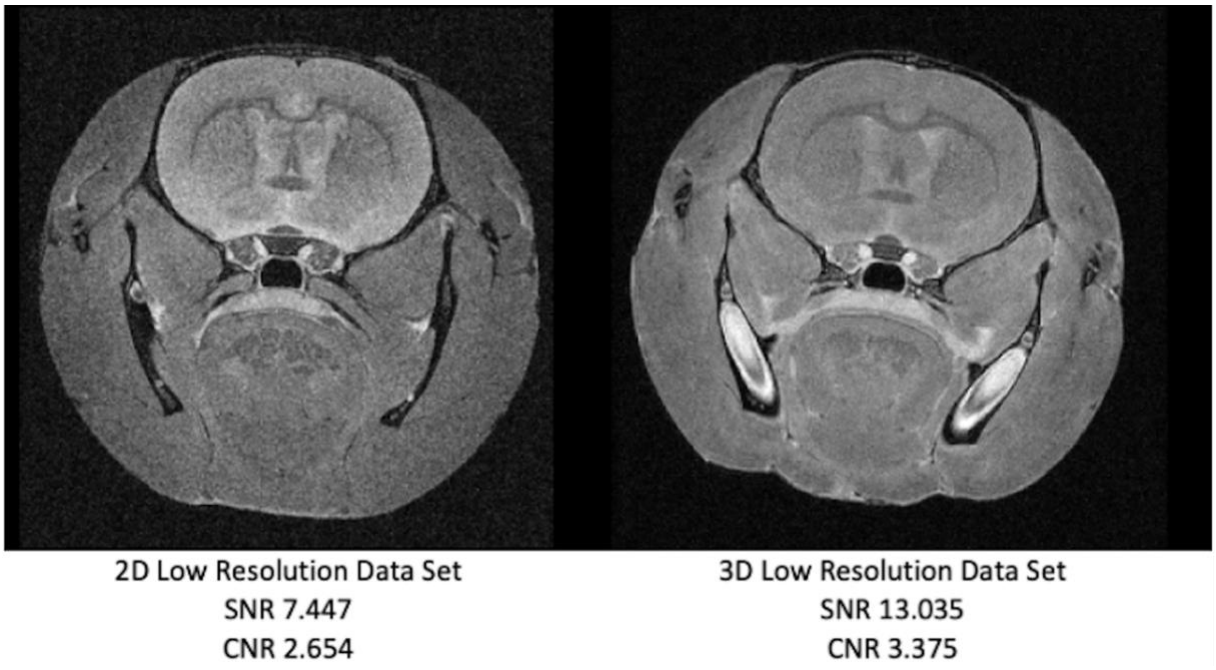

Figure 5.4: Images from a two dimensional and three-dimensional data set with 100 x 100 x 300-micron resolution demonstrating the decreased SNR for two-dimensional image acquisition.

faster to acquire. A comparable three-dimensional high-resolution data set was acquired directly with 100-micron isotropic resolution across 30 slices with 12 averages for a total imaging time of 12 hours.

Figure 5.5 demonstrates four successive slices for the high resolution and low-resolution data sets acquired directly, demonstrating the loss of information between successive slices for the lower resolution images. The same iterative back-projection technique was used to

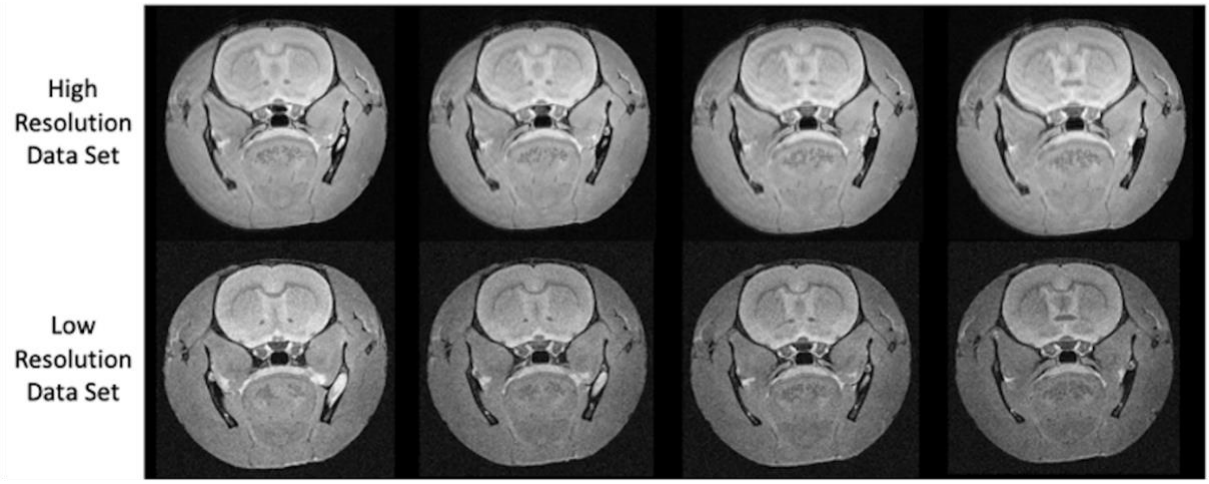

Figure 5.5: Comparison of a high-resolution data set with 100-micron isotropic resolution versus that of a low-resolution data set at 100 x 100 x 300-micron resolution where the initial low resolution data set starts before the first slice of the high-resolution set and the last slice stops after the last slice of the high-resolution data set.

reconstruct the high-resolution images. Figure 5.6 shows the corresponding SNR and CNR for the varying reconstructions against the directly acquired high resolution data sets. While reconstructions with two-dimensional data sets do not demonstrate the same increase in signal to noise ratios compared to the three-dimensional low-resolution data sets, they still produce greater contrast to noise ratios than the directly acquired high resolution set. Contrast to noise ratios are a key parameter in MR imaging and imaging parameters are often focused on acquiring the

highest contrast to noise ratios possible, thus the increase in CNR still indicates the asset to SRR even with noisier low resolution basis sets. Additionally, the SNR efficiency for the SRR for the 100-micron offset reconstruction was 0.449 compared to 0.354 for the high resolution directly acquired, indicating a greater efficiency for the SRR reconstruction with total imaging time of 20 minutes and 30 seconds for the low-resolution data sets to perform SRR with 50-micron offsets while the high-resolution data set acquired directly with 12 averages took over 25 hours.

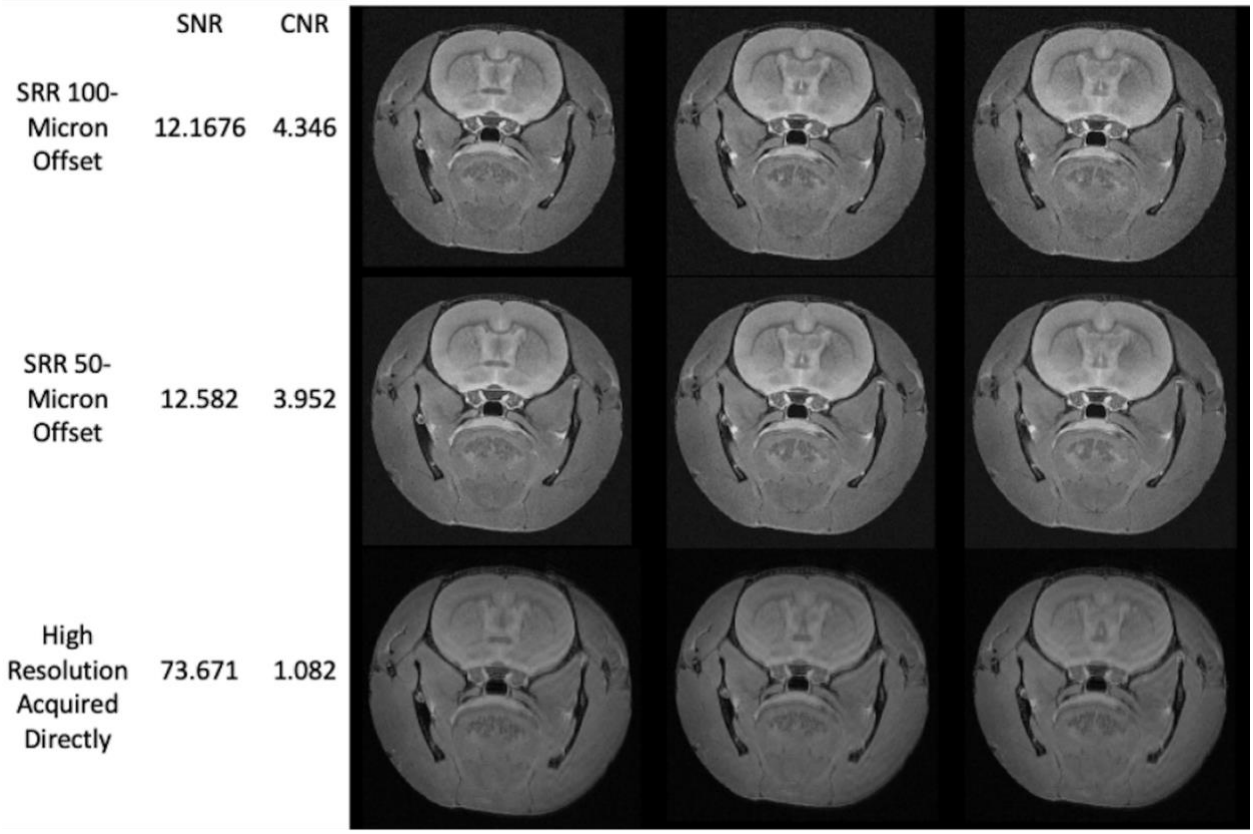

Figure 5.6: Three slices from the reconstruction images using low-resolution images with 100 x 100 x 300-micron resolution compared to corresponding directly acquired high-resolution data set. The corresponding SNR and CNR values for each data sets demonstrate the lower resolution from a 2D reconstruction series but still show improved CNR from reconstruction compared to direct acquisition.

In order to determine at which point the SRR technique no longer captures the information between successive steps, or the point at which the information gained is no longer sufficient to reconstruct the higher resolution goal, the two dimensional low resolution images were again acquired with 0.400, 0.500, and 0.600 mm slice thickness. The imaging parameters changed slightly depending upon the slice thickness. The 0.400- and 0.500-mm data sets were acquired with a TR/TE of 2500/11 ms, with a RARE factor of 4 for a total imaging time of 3 minutes. The 0.600 mm data set was acquired with a TR/TE of 2500/22.45 ms and a RARE factor of 6. An additional high resolution data set was acquired TR/TE of 2500/16 ms and RARE factor of 6 for the 0.400- and 0.500-mm data sets with 100-micron isotropic resolution in order to obtain the echo at the shortest available time point as was done for the 0.400 and 0.500 data sets while maintaining a reasonable imaging time. Each image series was performed with a 100-micron and 50- micron offset to perform two reconstructions with varying levels of low-resolution inputs. Then, the SRR iterative back projection was used to reconstruct 100-micron isotropic resolution data sets, and the SNR and CNR was determined.

Figure 5.7 demonstrates the low-resolution images across three slices and the directly acquired high resolution image. While the images start at comparable slices, the low resolution with 600-micron slice thickness clearly starts slightly before the directly acquired high resolution data set and ends after the high-resolution data set. For the low-resolution sets with 400- and 500-micron slice thickness this difference is less noticeable but can be seen in the last slice where the corpus callosum starts to reach farther down and the anterior commissure is closer together compared to the directly acquired high resolution data sets. The initial signal to noise ratios for the low-resolution data sets show an increase as slice thickness increases (which is to be expected). For comparison the 300-micron low resolution data set had an SNR and CNR of

7.447 and 2.654. With a one hundred micron jump to a slice thickness of 400 micron the SNR and CNR increases to 16.829 and 7.345, at 500 micron these measurements are 19.490 and 6.581. The SNR of the 600-micron slice thickness was less impressive 17.093 but maintained a high CRN of 7.599. The decreased SNR could be due to trying to keep the TR/TE comparable to the matching high resolution data set that was used to compare against it in a separate trial.

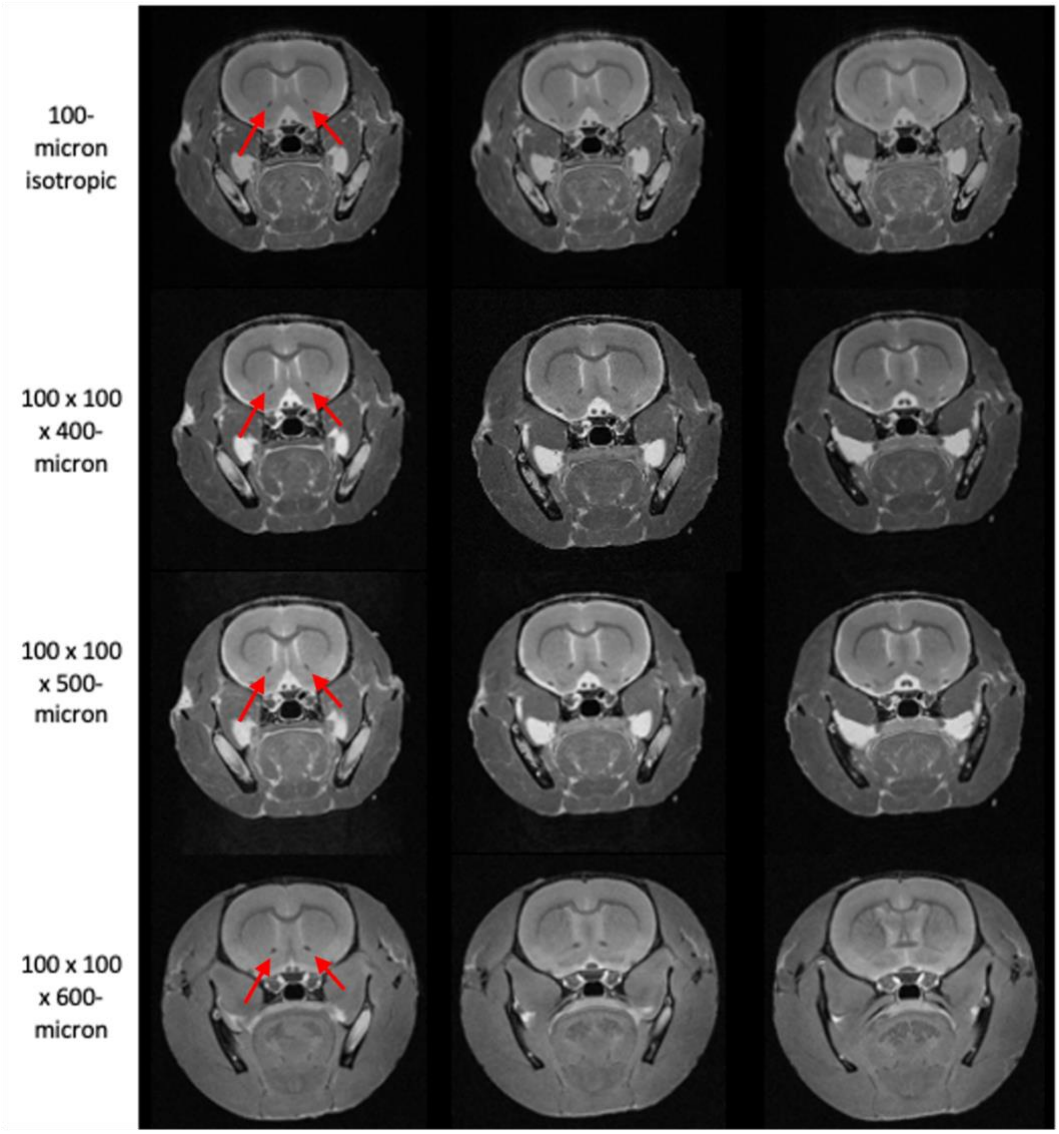

Figure 5.7: Directly acquired high resolution 100-micron isotropic image compared low resolution data sets across three slices with varying slice thickness. The anterior commissure is denoted by red arrows in the first slice.

Reconstructions with each low-resolution data set were performed with 50-micron and 100-micron offset. Figure 5.8 demonstrates the SRR with four low resolution images with a slice thickness of 400-micron and a 100-micron offset, and eight low resolution images with a 50-

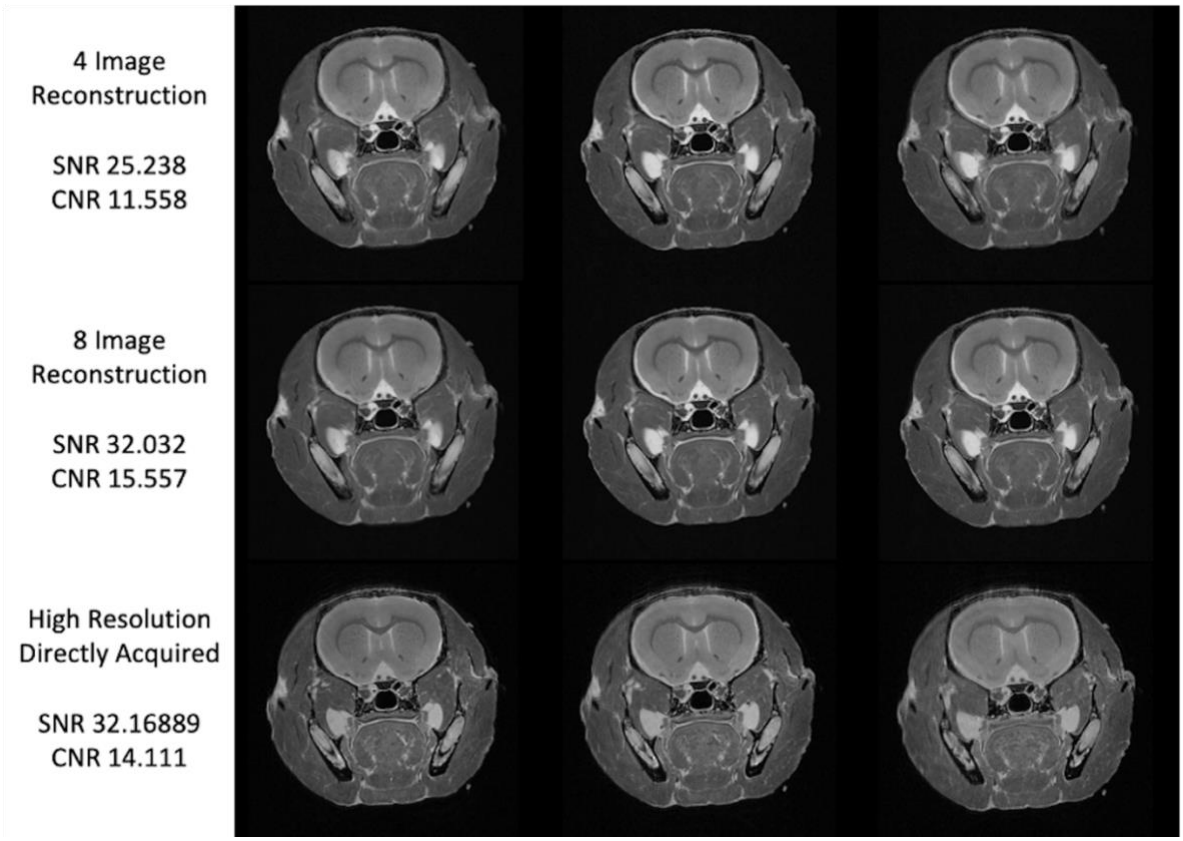

Figure 5.8: SRR with 100 x 100 x 400-micron low-resolution images to reconstruct the high resolution 100 x 100 x 100-micron images. Reconstructions with 4 images with a 100-micron offset are shown first followed by an 8-image reconstruction with a 50-micron offset. Signal to noise ratios and contrast to noise ratios are also denoted in the left column.

micron offset compared to the high-resolution image directly acquired. While the first two images show good coherence with the directly acquired high resolution image, at the third slice, the low-resolution reconstructions show a lack of intensity and more spread of the signal as the conformation of the corpus callosum changes. This is less severe in the 50-micron offset

reconstruction but still shows deviation from the directly acquired high resolution image. Additionally, there is no increase in the signal to noise ratio or contrast to noise ratio for a four-image reconstruction with 100-micron offsets in the through-plane dimension. However, the eight image, 50 micron offset reconstruction does offers comparable signal to noise ratios and subtle increases in contrast to noise ratios compared to that of the directly acquired high resolution data set. While the "blurring" at the change of the corpus callosum is present in the third slice, it is important to mention that within two slices it once again closely mirrors the directly acquired images as seen in Figure 5.9.

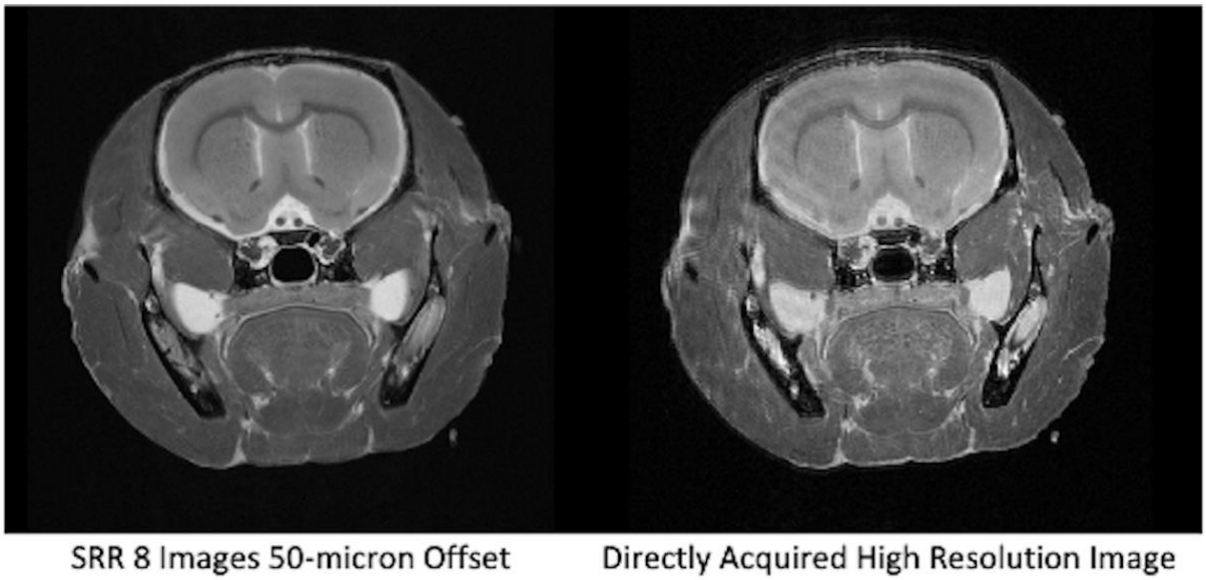

Figure 5.9: Two slices forward from the previous figure demonstrating the re-coherence of the 50-micron offset low resolution reconstruction to the high resolution directly acquired.

Figure 5.10 demonstrates the reconstruction with the 500-micron slice thickness low resolution data set. Unfortunately, with a 5:1 reconstruction between the high to low resolution data sets, the reconstructions start to diverge from the directly acquired reference point. Reconstructions with low resolution data sets show divergence along the corpus callosum within the first slice demonstrated in Figure 5.10 where the new information contained within each

successive offset is not substantial enough/does not contribute enough to the next offset image for robust reconstruction. Additionally, at this reconstruction ratio the signal to noise and

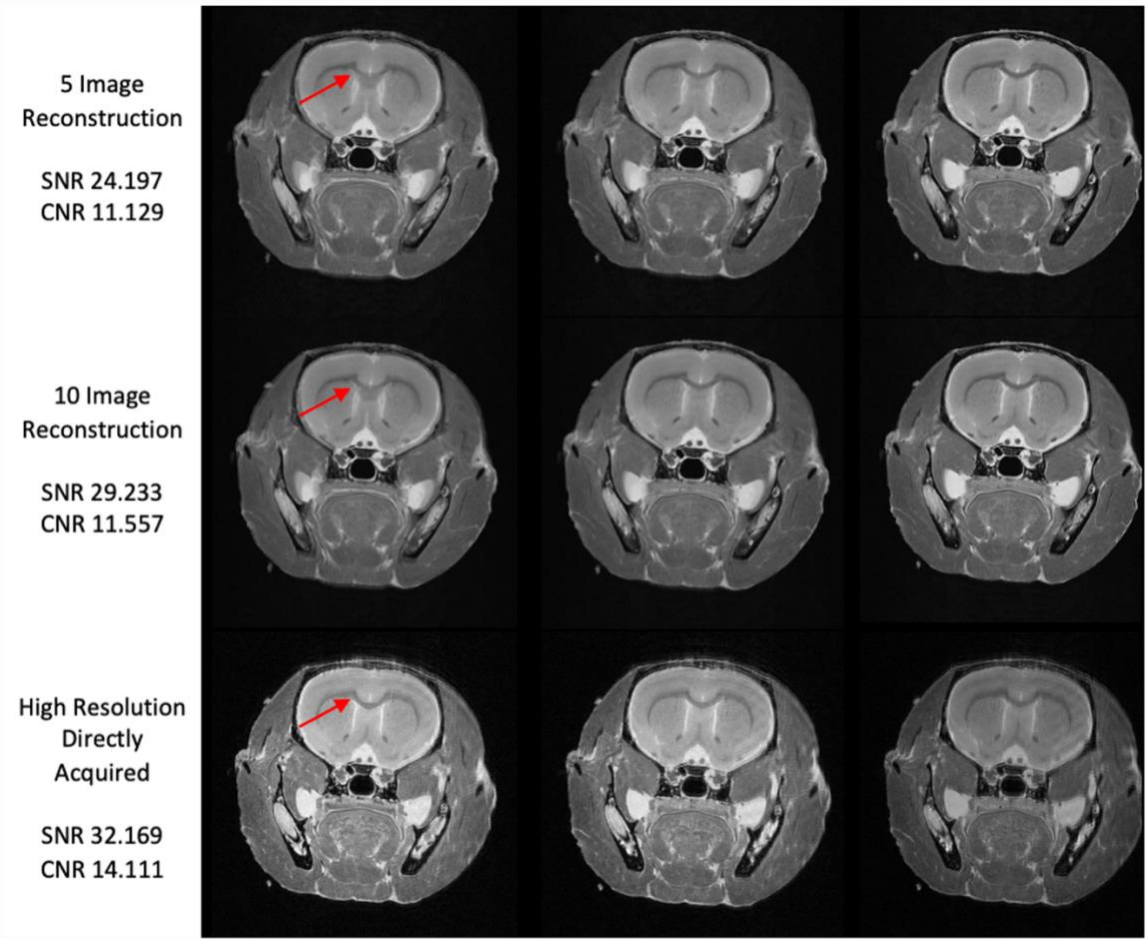

Figure 5.10: Reconstruction with 100 x 100 x 500-micron low resolution data sets using 5 images with a 100-micron offset, and 10 images with a 50-micron offset compared to the directly acquired high resolution image with 100-micron isotropic resolution. SNR and CNR values found in the left column for each respective image series. The corpus callosum is denoted by red arrows in the first slice.

contrast to noise ratio benefits are no longer consistent as what had been seen with previous reconstructions with 3:1 and 4:1 reconstruction ratios between low and high resolution data sets. This divergence from the directly acquired image is exacerbated with the 6:1 ratio between low- and high-resolution data sets as seen in Figure 5.11. While the SNR in the reconstructions is

quite promising, the solution for the super resolution reconstruction fails to identify a solid boundary for the skull causing diffusive signal around the brain. This continues with a failure to adequately follow the changes in the corpus callosum across slices, most evident in the first slice in Figure 5.11. However, much like in the other reconstructions, more images used in the

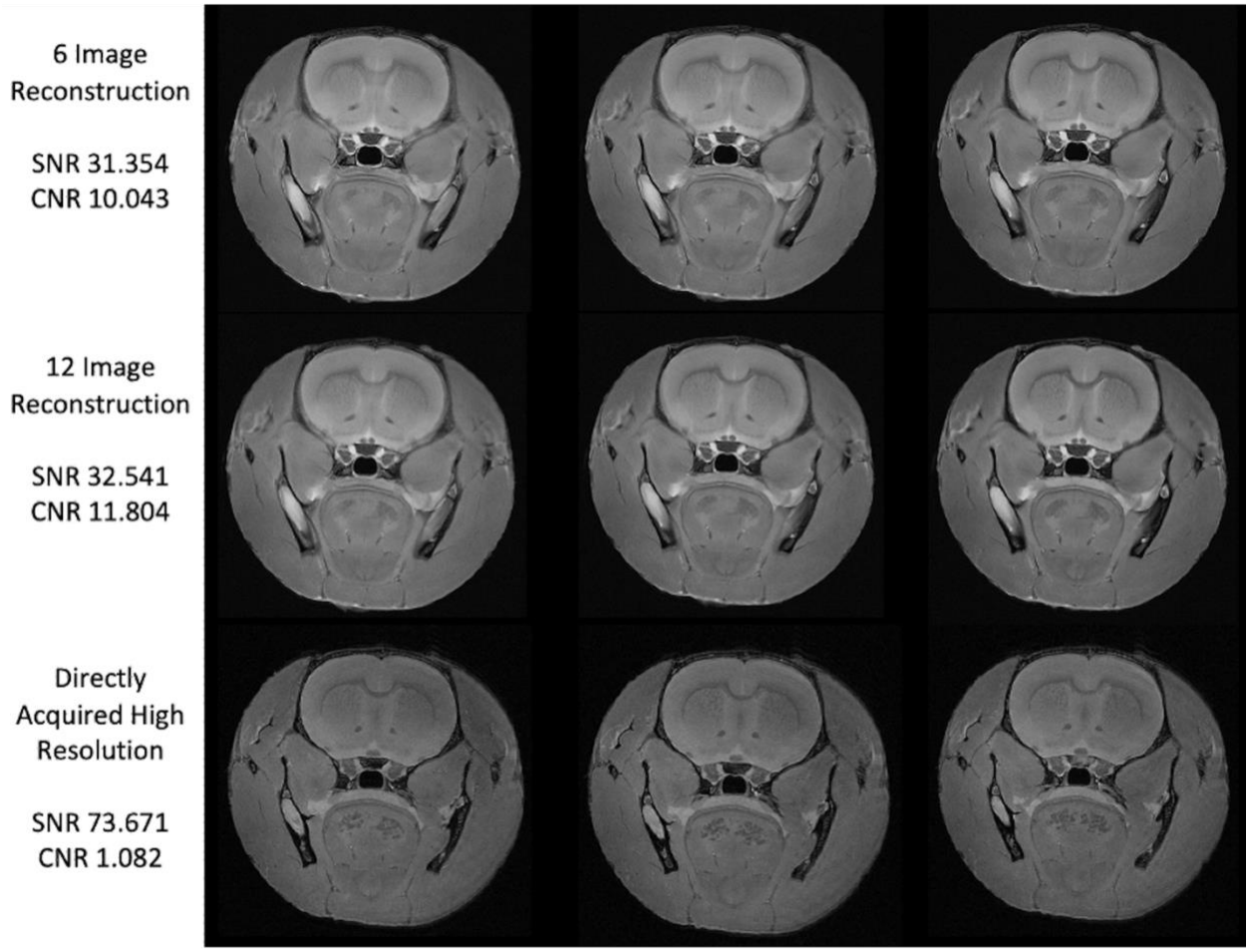

Figure 5.11: Comparison of a directly acquired high-resolution data set to reconstructions with low resolution data sets of 100 x 100 x 600-micron with 6 or 12 images with either a 100- or 50-micron offset.

reconstruction minimizes the decoherence between the directly acquired and reconstructed images as can be seen in the 12 versus 6 image reconstruction, yet the practicality decreases as the ratio between the low- and high-resolution data sets increases, and thus the total number of

low-resolution data sets which would need to be acquired become burdensome. Ultimately, it seems that for optimal reconstructions the high-resolution images to be reconstructed should be no more than four times the resolution of the low-resolution data sets used to reconstruct them, with three to one offering the best results.

Super-resolution reconstruction has been shown to produce images which closely mirror high-resolution images acquired directly and have demonstrated a greater signal to noise ratio efficiency enabling high resolution data sets to be achieved in highly condensed time frames. For instances in which the sample does not occupy the maximal amount of space in the imaging probe, such as the archerfish, the signal to noise ratio and constant to noise ratios are greater than those of the directly acquired high resolution images. This improvement is expected to be seen at greater levels as the number of low-resolution data sets used in the reconstruction increases. This is likely due to the increased noise when samples do not occupy the maximal volume of the imaging probe being eradicated more efficiently during SRR than could be obtained in a reasonable time frame if acquired directly. While three- and two-dimensional reconstructions of samples which occupied the maximal amount of space in the probe did not see the same improvement over the directly acquired high resolution images in terms of SNR, they did demonstrate an increased CNR, an important characteristic in MRI for defining tissue boundaries. Additionally, two-dimensional low-resolution data sets can be generated rapidly in minutes and reconstructed to high-resolution data sets that may be inaccessible due to instrument strengths granting researchers access to high-quality data that would otherwise be too time consuming or press instrument capacity if generated directly. The reconstruction algorithm is a quick and efficient and can provide the high-resolution output in minutes which closely mirror what could be obtained directly (just with a much higher time cost). Based upon these findings,

super-resolution reconstructions could present a unique opportunity in clinical settings to achieve ultra-high resolution even at lower field strengths, providing radiologists with more detailed information during patient assessments.

Conclusion

Super-resolution reconstruction is an innovative strategy to increase the resolution of MR images without suffering from decreased signal which is a frequent side-effect of acquiring ultra-high-resolution images directly. Through a series of experiments, reconstructions of high-resolution images were obtained from both two- and three-dimensional low resolution data sets. While three-dimensional data sets were shown to have greater signal to noise ratios, two-dimensional acquisition provided rapid image acquisition with rapid reconstructions and maintained higher contrast to noise ratios when compared to high resolution images acquired directly. Additionally, the impact on the number of low-resolution data sets used in the reconstructions was shown to have a profound impact on the reconstructed image signal to noise ratio with signal to noise, and contrast to noise ratios increasing as the number of low-resolution inputs increased. Ultimately, the time cost of data acquisition must be maintained for SRR to remain a cost-effective option, with the most substantial signal to noise improvements apparent at 6 low-resolution inputs for reconstruction. This indicates that further data acquisition may not be necessary, although additional SNR improvements are seen with the addition of more low-resolution inputs. The upper limits for through-plane resolution for the low-resolution inputs was investigated and indicated that a 1 to 3 ratio between high- and low-resolution data is optimal for reconstruction and yields the greatest agreement between reconstructed and directly acquired high resolution data. However, 1 to 4 ratios between the high- and low- resolution images may be sufficient so long as the regions of interest have larger boundaries between tissues and easily

defined contrast changes between regions of interest. Reconstructed images were carefully examined and compared against directly acquired high-resolution data acquired with multiple averages (to maintain high signal to noise ratios) to ascertain the accuracy of reconstructions against a variety of landmarks such as the corpus callosum and anterior commissure. Ultimately, SRR has demonstrated itself as an effective strategy to increase image resolution while condensing imaging time and may prove a useful in clinical and research settings to maintain patient comfort, assess greater regions of interest or additional metrics (such as diffusion imaging) in live animal imaging, improve signal to noise and contrast to noise ratios at lower field strengths, and obtain more useful images for samples which do not adequately fill the RF coil and thus are predisposed to increased noise.

Chapter 6 - Conclusion and Future Directions

Magnetic resonance imaging is a powerful tool to noninvasively investigate a variety of physiological processes and maladies, understand neurological disease modes of activation, and monitor the application and efficacy of treatments. There are a variety of applications for magnetic resonance imaging and depending upon the type of analysis and metric required both qualitative and quantitative MRI have their place in clinical and research studies. Qualitative imaging has proven its value in tumor identification,[59] volume alterations due to neurological disability,[60] stroke intervention,[61] and more. Quantitative imaging has now taken center stage in the MR field as a method to remove radiologist bias in diagnosing disease and assessing treatment efficacy. It can take many forms from relaxation mapping, tensor estimation, and analysis of various diffusion metrics. It has proven successful in comparison of various brain pathologies[62,63] and is slowly paving the way for neural networks to classify and separate healthy from diseased tissue as well as grading various levels of disease progression.

An initial foray into quantitative imaging involved a series of experiments to monitor small animals over the course of pancreatic cancer. Qualitative imaging was used to identify tumor locations and monitor tumor growth as well as for adequate probe positioning for hyperthermia treatments. T_1 mapping was then used to monitor disease progression, discriminate between tumor and healthy tissue, and was determined sensitive enough to discriminate between different cell lines. While the ultra-high field strength of 14 T allowed for high signal to noise ratios and high-resolution anatomical imaging, T_1 mapping allowed for in-depth investigation into the cellular level changes to stromal barriers around the tumor and inflammation as the pancreatic cancer progressed.

The burden of long acquisition times and difficult computation and analysis of diffusion data sets acquired on research instrumentation was tackled with a simple way to homogenize both artifact removal, tensor estimation, and metric decomposition into various diffusion measurements in one platform simultaneously. Image time condensation was initially addressed through the key impacts alterations on pertinent parameters have on overall image quality and their potential drawbacks with alterations. These were then targeted as methods in which to acquire the highest resolution diffusion data sets if time costs are not of concern. The analysis pipeline presented provides an alternative way to quickly and effectively gain access to various identifiers to monitor disease progression or alterations to physiological and neurological structures in response to age, illness, or injury.

Furthermore, a method in which to address the ever-present issue of balancing signal to noise ratio and image resolution to circumvent the limitation of clinically available instrumentation was investigated with super resolution reconstruction using iterative back projection. These reconstructed high-resolution images were then compared with directly acquired high resolution data sets made possible through ultra-high field imaging. Low resolution data sets acquired in both two and three dimensions were examined for their ability to reconstruct high resolution data sets and their impact on signal to noise and contrast to noise ratios. The dramatic decrease in imaging time was presented using an SRR method to acquire high-resolution data sets, the limitation for low resolution to high resolution reconstruction was investigated and increasing SNR by increasing the number of basis sets used was illuminated.

Throughout this work a series of key complications and applications of quantitative MRI were addressed in order to provide a launching point for future work. Future applications of quantitative MRI hinge upon generating replicable data sets with slim margins of error in order

to train neural networks that can adequately detect malignancies and anomalies in various samples. For this to reach fruition, it is necessary to standardize the method of analysis for quantitative data sets and thus the analysis tools must be accessible regardless of instrumentation type. Moreover, higher resolution must be obtained in order to minimize the variation between slices, thus SRR algorithms may provide access to higher resolution data sets for various forms of quantitative imaging from relaxation mapping to diffusion tensor imaging which would enable closer examination within tumor structures and tissues. Additionally, the use of reconstruction could also provide access to more accurate tractography studies in the brain resolving regions of high fiber crossover.

This work can be carried forward with more advanced artifact correction of the analysis pipeline which would enable access to even higher quality images. Additionally, alterations to allow for investigation into multi-shell data sets which have more than one b-value could be introduced to examine intra and extra neurite diffusion. The work with super resolution reconstruction can be carried over to investigate its application in T_1 and T_2 mapping. This would potentially enable closer investigation of the desmoplasia layer for pancreatic tumors and help decipher a better method of entry for treatment or help monitor the efficacy of currently available treatment. Furthermore, the use of SRR should be applied to diffusion data sets in order to obtain higher resolution data sets without the signal loss seen previously. This can then be used to determine if more accurate connectivity measurements can be made at ultra-high resolutions, circumventing the primary issue of tractography, the difficulty of resolving regions of high fiber crossover. While quantitative MRI is still in its early stages, the promising applications to understanding complex diseases, physiological, and neurological processes through various metrics, and the potential to remove user bias in analysis and assessment indicate its future

application will become paramount to alternative imaging modalities such as PET and CT in the future.

References

1. Van Osch, Matthias J., and Andrew G. Webb. “Safety of Ultra-High Field MRI: What Are the Specific Risks?” *Current Radiology Reports*, vol. 2, no. 8, 2014, doi:10.1007/s40134-014-0061-0.

2. Kangarlu, A., et al. “Cognitive, Cardiac, and Physiological Safety Studies in Ultra High Field Magnetic Resonance Imaging.” *Magnetic Resonance Imaging*, vol. 17, no. 10, 1999, pp. 1407–1416., doi:10.1016/s0730-725x(99)00086-7.

3. Sadeghi-Tarakameh, Alireza, et al. “In Vivo Human Head MRI at 10.5T: A Radiofrequency Safety Study and Preliminary Imaging Results.” *Magnetic Resonance in Medicine*, vol. 84, no. 1, 2019, pp. 484–496., doi:10.1002/mrm.28093.

4. Tretkoff E. July 1977: MRI Uses Fundamental Physics for Clinical Diagnosis. APS New: This Month in Physics History.

5. McRobbie, Donald W., et al. *MRI from Picture to Proton*. Cambridge University Press, 2017.

6. Budinger, Thomas F., et al. “Toward 20 T Magnetic Resonance for Human Brain Studies: Opportunities for Discovery and Neuroscience Rationale.” *Magnetic Resonance Materials in Physics, Biology and Medicine*, vol. 29, no. 3, 2016, pp. 617–639., doi:10.1007/s10334-016-0561-4.

7. Rutt, Brian K., and Donald H. Lee. “The Impact of Field Strength on Image Quality in MRI.” *Journal of Magnetic Resonance Imaging*, vol. 6, no. 1, 1996, pp. 57–62., doi:10.1002/jmri.1880060111.

8. Tian, Xiaofei, et al. “Effects of 3.5–23.0 T Static Magnetic Fields on Mice: A Safety Study.” *NeuroImage*, vol. 199, 2019, pp. 273–280., doi:10.1016/j.neuroimage.2019.05.070.

9. Bruker. Innovation with Integrity NMR AVANCE Micro Imaging for AVANCE III Systems User Guide.

10. Bangerter, Neal K., et al. “Magnetic Resonance Imaging.” *Bioengineering Innovative Solutions for Cancer*, 2020, pp. 163–194., doi:10.1016/b978-0-12-813886-1.00008-5.

11. Murphy, Andrew, and Usman Bashir. “Gradient Echo Sequences.” *Radiopaedia.org*, 2012, doi:10.53347/rid-18339.

12. Murphy, Andrew, and J. Ballinger. “Spin Echo Sequences.” *Radiopaedia.org*, 2013, doi:10.53347/rid-22136.

13. Gulani, Vikas, and Nicole Seiberlich. “Quantitative MRI: Rationale and Challenges.” *Advances in Magnetic Resonance Technology and Applications*, 2020, pp. xxxvii-li., doi:10.1016/b978-0-12-817057-1.00001-9.

14. Rajagopalan, Venkateswaran, et al. “A Basic Introduction to Diffusion Tensor Imaging Mathematics and Image Processing Steps.” *Brain Disorders & Therapy*, vol. 06, no. 02, 2017, doi:10.4172/2168-975x.1000229.

15. Chong, A.L., et al. “Proton Density MRI Increases Detection of Cervical Spinal Cord Multiple Sclerosis Lesions Compared with T2-Weighted Fast Spin-Echo.” *American Journal of Neuroradiology*, vol. 37, no. 1, 2015, pp. 180–184., doi:10.3174/ajnr.a4476.

16. Koh, Dow-Mu, and David J. Collins. “Diffusion-Weighted MRI in the Body: Applications and Challenges in Oncology.” *American Journal of Roentgenology*, vol. 188, no. 6, 2007, pp. 1622–1635., doi:10.2214/ajr.06.1403.

17. O’Donnell, Lauren J., and Carl-Fredrik Westin. “An Introduction to Diffusion Tensor Image Analysis.” *Neurosurgery Clinics of North America*, vol. 22, no. 2, 2011, pp. 185–196., doi:10.1016/j.nec.2010.12.004.

18. Bhojwani, Nicholas et al. “Diffusion-weighted imaging in musculoskeletal radiology-clinical applications and future directions.” *Quantitative imaging in medicine and surgery*vol. 5,5 (2015): 740-53. doi:10.3978/j.issn.2223-4292.2015.07.07

19. Baliyan, Vinit, et al. “Diffusion Weighted Imaging: Technique and Applications.” *World Journal of Radiology*, vol. 8, no. 9, 2016, p. 785., doi:10.4329/wjr.v8.i9.785.

20. Kingsley, Peter B. “Introduction to Diffusion Tensor Imaging Mathematics: Part III. Tensor Calculation, Noise, Simulations, and Optimization.” *Concepts in Magnetic Resonance Part A*, vol. 28A, no. 2, 2006, pp. 155–179., doi:10.1002/cmr.a.20050.

21. “Mathematics of Diffusion Tensor Imaging.” *Introduction to Diffusion Tensor Imaging*, by Susumu Mori, Elsevier, 2014, pp. 33–37.

22. Mori, Susumu, and Jiangyang Zhang. “Principles of Diffusion Tensor Imaging and Its Applications to Basic Neuroscience Research.” *Neuron*, vol. 51, no. 5, 2006, pp. 527–539., doi:10.1016/j.neuron.2006.08.012.

23. Alba-Ferrara, L. M., and Gabriel A. De Erausquin. “What Does Anisotropy Measure? Insights from Increased and Decreased Anisotropy in Selective Fiber Tracts in Schizophrenia.” *Frontiers in Integrative Neuroscience*, vol. 7, 2013, doi:10.3389/fnint.2013.00009.

24. Taylor, Andrew J et al. “T1 Mapping: Basic Techniques and Clinical Applications.” *JACC. Cardiovascular imaging* vol. 9,1 (2016): 67-81. doi:10.1016/j.jcmg.2015.11.005

25. Kellman, Peter, and Michael S Hansen. “T1-Mapping in the Heart: Accuracy and Precision.” *Journal of Cardiovascular Magnetic Resonance*, vol. 16, no. 1, 2014, doi:10.1186/1532-429x-16-2.

26. Renner, Nina, et al. “T2 Mapping as a New Method for Quantitative Assessment of Cartilage Damage in Rheumatoid Arthritis.” *The Journal of Rheumatology*, vol. 47, no. 6, 2019, pp. 820–825., doi:10.3899/jrheum.180728.

27. Mars, Mokhtar, et al. “MRI T2 Mapping of Knee Articular Cartilage Using Different Acquisition Sequences and Calculation Methods at 1.5 Tesla.” *Medical Principles and Practice*, vol. 27, no. 5, 2018, pp. 443–450., doi:10.1159/000490796.

28. Hoang Dinh, A., et al. “Characterization of Prostate Cancer Using T2 Mapping at 3T: A Multi-Scanner Study.” *Diagnostic and Interventional Imaging*, vol. 96, no. 4, 2015, pp. 365–372., doi:10.1016/j.diii.2014.11.016.

29. Matzat, Stephen J., et al. “T2 Relaxation Time Quantitation Differs between Pulse Sequences in Articular Cartilage.” *Journal of Magnetic Resonance Imaging*, vol. 42, no. 1, 2014, pp. 105–113., doi:10.1002/jmri.24757.

30. Wang, Lixia, et al. “Multiparametric Mapping Magnetic Resonance Imaging of Pancreatic Disease.” *Frontiers in Physiology*, vol. 11, 2020, doi:10.3389/fphys.2020.00008.

31. Sarantis, Panagiotis, et al. “Pancreatic Ductal Adenocarcinoma: Treatment Hurdles, Tumor Microenvironment and Immunotherapy.” *World Journal of Gastrointestinal Oncology*, vol. 12, no. 2, 2020, pp. 173–181., doi:10.4251/wjgo.v12.i2.173.

32. Hasebroock, Kendra M, and Natalie J Serkova. “Toxicity of MRI and CT Contrast Agents.” *Expert Opinion on Drug Metabolism & Toxicology*, vol. 5, no. 4, 2009, pp. 403–416., doi:10.1517/17425250902873796.

33. Van der Horst, Astrid, et al. “The Clinical Benefit of Hyperthermia in Pancreatic Cancer: A Systematic Review.” *International Journal of Hyperthermia*, vol. 34, no. 7, 2017, pp. 969–979., doi:10.1080/02656736.2017.1401126.

34. Stikov, Nikola, et al. “On the Accuracy of t1mapping: Searching for Common Ground.” *Magnetic Resonance in Medicine*, vol. 73, no. 2, 2014, pp. 514–522., doi:10.1002/mrm.25135.

35. Murphy, Andrew, and Zach Drew. “Factors Affecting T1.” *Radiopaedia.org*, 2018, doi:10.53347/rid-60741.

36. Lee, Jae W., et al. “Genetically Engineered Mouse Models of Pancreatic Cancer: The Kpc Model (Lsl-Kras G12D/+ ;LSL-Trp53 R172H/+ ;Pdx-1-Cre), Its Variants, and Their Application in Immuno-Oncology Drug Discovery.” *Current Protocols in Pharmacology*, vol. 73, no. 1, 2016, doi:10.1002/cpph.2.

37. Haouimi, Ammar, and J Yeung. “Aliasing on MRI.” *Radiopaedia.org*, 2011, doi:10.53347/rid-15691.

38. Basser, P.J., et al. “MR Diffusion Tensor Spectroscopy and Imaging.” *Biophysical Journal*, vol. 66, no. 1, 1994, pp. 259–267., doi:10.1016/s0006-3495(94)80775-1.

39. Lebel, Catherine, et al. “Six Is Enough? Comparison of Diffusion Parameters Measured Using Six or More Diffusion-Encoding Gradient Directions with Deterministic Tractography.” *Magnetic Resonance in Medicine*, vol. 68, no. 2, 2011, pp. 474–483., doi:10.1002/mrm.23254.

40. Giannelli, Marco, et al. “Dependence of Brain DTI Maps of Fractional Anisotropy and Mean Diffusivity on the Number of Diffusion Weighting Directions.” *Journal of Applied Clinical Medical Physics*, vol. 11, no. 1, 2009, pp. 176–190., doi:10.1120/jacmp.v11i1.2927.

41. Morelli, John N., et al. “An Image-Based Approach to Understanding the Physics of Mr Artifacts.” *RadioGraphics*, vol. 31, no. 3, 2011, pp. 849–866., doi:10.1148/rg.313105115.

42. Ivanov, Evgeniy N., et al. “Optimization of Duty Cycles for MRI Scanners.” *Concepts in Magnetic Resonance Part B: Magnetic Resonance Engineering*, vol. 37B, no. 3, 2010, pp. 180–192., doi:10.1002/cmr.b.20167.

43. Herbst, Michael, et al. “Prospective Motion Correction of Segmented Diffusion Weighted Epi.” *Magnetic Resonance in Medicine*, vol. 74, no. 6, 2014, pp. 1675–1681., doi:10.1002/mrm.25547.

44. Soares, José M., et al. “A Hitchhiker's Guide to Diffusion Tensor Imaging.” *Frontiers in Neuroscience*, vol. 7, 2013, doi:10.3389/fnins.2013.00031.

45. Larobina, Michele, and Loredana Murino. “Medical Image File Formats.” *Journal of Digital Imaging*, vol. 27, no. 2, 2013, pp. 200–206., doi:10.1007/s10278-013-9657-9.

46. Jones, Jeremy, and Sarah Li. “Eddy Currents.” *Radiopaedia.org*, 2017, doi:10.53347/rid-55242.

47. Chia-Hao C. DTI Fiber Tractography (Streamline Tracking Technique). *MATLAB Central File Exchange* *https://www.mathworks.com/matlabcentral/fileexchange/34008-dti-fiber-tractography-streamline-tracking-technique*. Published online 2012. Retrieved June 20, 2020.

48. Basser, P.J., et al. “Estimation of the Effective Self-Diffusion Tensor from the NMR Spin Echo.” *Journal of Magnetic Resonance, Series B*, vol. 103, no. 3, 1994, pp. 247–254., doi:10.1006/jmrb.1994.1037.

49. Li, Chunming, et al. “Multiplicative Intrinsic Component Optimization (MICO) for MRI Bias Field Estimation and Tissue Segmentation.” *Magnetic Resonance Imaging*, vol. 32, no. 7, 2014, pp. 913–923., doi:10.1016/j.mri.2014.03.010.

50. Juntu, Jaber, et al. “Bias Field Correction for MRI Images.” *Advances in Soft Computing*, 2005, pp. 543–551., doi:10.1007/3-540-32390-2_64.

51. Li C. MRI segmentation and bias field correction. *MATLAB File Exchange https://www.mathworks.com/matlabcentral/fileexchange/59752-mri-segmentation-and-bias-field-correction*. Published online 2016. Retrieved May 8, 2020.

52. Carmi, Eyal, et al. “Resolution Enhancement in MRI.” *Magnetic Resonance Imaging*, vol. 24, no. 2, 2006, pp. 133–154., doi:10.1016/j.mri.2005.09.011.

53. Van Reeth, Eric, et al. “Super-Resolution in Magnetic Resonance Imaging: A Review.” *Concepts in Magnetic Resonance Part A*, vol. 40A, no. 6, 2012, pp. 306–325., doi:10.1002/cmr.a.21249.

54. Zhu, Xiaolu, et al. “A Pixel Is an Artifact: On the Necessity of Zero-Filling in Fourier Imaging.” *Concepts in Magnetic Resonance Part A*, vol. 42A, no. 2, 2013, pp. 32–44., doi:10.1002/cmr.a.21256.

55. Bernstein, Matt A., et al. “Effect of Windowing and Zero-Filled Reconstruction of MRI Data on Spatial Resolution and Acquisition Strategy.” *Journal of Magnetic Resonance Imaging*, vol. 14, no. 3, 2001, pp. 270–280., doi:10.1002/jmri.1183.

56. Irani, Michal, and Shmuel Peleg. “Improving Resolution by Image Registration.” *CVGIP: Graphical Models and Image Processing*, vol. 53, no. 3, 1991, pp. 231–239., doi:10.1016/1049-9652(91)90045-l.

57. Greenspan, H., et al. “MRI Inter-Slice Reconstruction Using Super-Resolution.” *Magnetic Resonance Imaging*, vol. 20, no. 5, 2002, pp. 437–446., doi:10.1016/s0730-725x(02)00511-8.

58. Irani, Michal, and Shmuel Peleg. “Motion Analysis for Image Enhancement: Resolution, Occlusion, and Transparency.” *Journal of Visual Communication and Image Representation*, vol. 4, no. 4, 1993, pp. 324–335., doi:10.1006/jvci.1993.1030.

59. Shukla, Gaurav, et al. “Advanced Magnetic Resonance Imaging in Glioblastoma: A Review.” *Chinese Clinical Oncology*, vol. 6, no. 4, 2017, pp. 40–40., doi:10.21037/cco.2017.06.28.

60. Payne, Macy, et al. “Increased Volumes of Lobule VI in a Valproic Acid Model of Autism Are Associated with Worse Set-Shifting Performance in Male Long-Evan Rats.” *Brain Research*, vol. 1765, 2021, p. 147495., doi:10.1016/j.brainres.2021.147495.

61. Kim, Bum Joon, et al. “Magnetic Resonance Imaging in Acute Ischemic Stroke Treatment.” *Journal of Stroke*, vol. 16, no. 3, 2014, p. 131., doi:10.5853/jos.2014.16.3.131.

62. Mishra, Arabinda, et al. “Longitudinal Changes in DTI Parameters of Specific Spinal White Matter Tracts Correlate with Behavior Following Spinal Cord Injury in Monkeys.” *Scientific Reports*, vol. 10, no. 1, 2020, doi:10.1038/s41598-020-74234-2.

63. Flores-Alvarez, Eduardo, et al. “Correlations between DTI-Derived Metrics and Mrs Metabolites in Tumour Regions of Glioblastoma: A Pilot Study.” *Radiology and Oncology*, vol. 54, no. 4, 2020, pp. 394–408., doi:10.2478/raon-2020-0055.

64. Birkbeck M. Parametric Mapping Scripts for MRI. *MATLAB File Exchange* https://www.mathworks.com/matlabcentral/fileexchange/64579-parametric-mapping-scripts-for-mri-data. Retrieved July 7, 2020.

65. Xiangrui Li (2022). xiangruili/dicm2nii (https://github.com/xiangruili/dicm2nii), GitHub. Retrieved March 21, 2022.

66. Tristán-Vega, Antonio, et al. “Least Squares for Diffusion Tensor Estimation Revisited: Propagation of Uncertainty with Rician and Non-Rician Signals.” *NeuroImage*, vol. 59, no. 4, 2012, pp. 4032–4043., doi:10.1016/j.neuroimage.2011.09.074.

67. Alexander, Andrew L., et al. “Diffusion Tensor Imaging of the Brain.” *Neurotherapeutics*, vol. 4, no. 3, 2007, pp. 316–329., doi:10.1016/j.nurt.2007.05.011.

68. Song, Shuang, et al. “A Review of Methods for Bias Correction in Medical Images.” *Biomedical Engineering Review*, vol. 3, no. 1, 2017, doi:10.18103/bme.v3i1.1550.

70. Yushkevich, Paul A., et al. “User-Guided 3D Active Contour Segmentation of Anatomical Structures: Significantly Improved Efficiency and Reliability.” *NeuroImage*, vol. 31, no. 3, 2006, pp. 1116–1128., doi:10.1016/j.neuroimage.2006.01.015.

71. Bammer, Roland. “Basic Principles of Diffusion-Weighted Imaging.” *European Journal of Radiology*, vol. 45, no. 3, 2003, pp. 169–184., doi:10.1016/s0720-048x(02)00303-0.

72. Basser, P. J., Mattiello, J., & LeBihan, D. (1992). “Diagonal and off-diagonal components of the self-diffusion tensor: their relation to and estimation from the NMR spin-echo signal.”

73. Basser, P. J., & LeBihan, D. (1992). “Fiber orientation mapping in an anisotropic medium with NMR diffusion spectroscopy.”

74. Collins, Christopher M. “Electromagnetics in Magnetic Resonance Imaging Physical Principles, Related Applications, and Ongoing Developments.” 2016, doi:10.1088/978-1-6817-4083-6.

75. Farsiu, S., et al. “Fast and Robust Multiframe Super Resolution.” *IEEE Transactions on Image Processing*, vol. 13, no. 10, 2004, pp. 1327–1344., doi:10.1109/tip.2004.834669.

76. Fridman, Rafael, et al. “Increased Initiation and Growth of Tumor Cell Lines, Cancer Stem Cells and Biopsy Material in Mice Using Basement Membrane Matrix Protein (Cultrex or Matrigel) Co-Injection.” *Nature Protocols*, vol. 7, no. 6, 2012, pp. 1138–1144., doi:10.1038/nprot.2012.053.

77. Ganzetti, Marco, et al. “Whole Brain Myelin Mapping Using T1- and T2-Weighted MR Imaging Data.” *Frontiers in Human Neuroscience*, vol. 8, 2014, doi:10.3389/fnhum.2014.00671.

78. Gholipour, A, et al. “Robust Super-Resolution Volume Reconstruction from Slice Acquisitions: Application to Fetal Brain Mri.” *IEEE Transactions on Medical Imaging*, vol. 29, no. 10, 2010, pp. 1739–1758., doi:10.1109/tmi.2010.2051680.

79. Jones, Derek K., and Peter J. Basser. “Squashing Peanuts and Smashing Pumpkins?: How Noise Distorts Diffusion-Weighted Mr Data.” *Magnetic Resonance in Medicine*, vol. 52, no. 5, 2004, pp. 979–993., doi:10.1002/mrm.20283.

80. Kingsley, Peter B. “Introduction to Diffusion Tensor Imaging Mathematics: Part I. Tensors, Rotations, and Eigenvectors.” *Concepts in Magnetic Resonance Part A*, vol. 28A, no. 2, 2006, pp. 101–122., doi:10.1002/cmr.a.20048.

81. Kingsley, Peter B. “Introduction to Diffusion Tensor Imaging Mathematics: Part II. Anisotropy, Diffusion-Weighting Factors, and Gradient Encoding Schemes.” *Concepts in Magnetic Resonance Part A*, vol. 28A, no. 2, 2006, pp. 123–154., doi:10.1002/cmr.a.20049.

82. Kjos, BO, et al. “Reproducibility of T1 and T2 Relaxation Times Calculated from Routine MR Imaging Sequences: Phantom Study.” *American Journal of Roentgenology*, vol. 144, no. 6, 1985, pp. 1157–1163., doi:10.2214/ajr.144.6.1157.

83. Krueger, E.C., et al. “Changes in T2 Relaxation Times Associated with Maturation of the Human Intervertebral Disk.” *American Journal of Neuroradiology*, vol. 28, no. 7, 2007, pp. 1237–1241., doi:10.3174/ajnr.a0546.

84. Liu, Li, et al. “Changes of T2 Relaxation Time from Neoadjuvant Chemotherapy in Breast Cancer Lesions.” *Iranian Journal of Radiology*, vol. 13, no. 3, 2016, doi:10.5812/iranjradiol.24014.

85. Neil, Jeffrey J. “Measurement of Water Motion (Apparent Diffusion) in Biological Systems.” *Concepts in Magnetic Resonance*, vol. 9, no. 6, 1997, pp. 385–401., doi:10.1002/(sici)1099-0534(1997)9:6<385::aid-cmr2>3.0.co;2-v.

86. Sung Cheol Park, et al. “Super-Resolution Image Reconstruction: A Technical Overview.” *IEEE Signal Processing Magazine*, vol. 20, no. 3, 2003, pp. 21–36., doi:10.1109/msp.2003.1203207.

87. Peeters, Ronald R., et al. “The Use of Super-Resolution Techniques to Reduce Slice Thickness in Functional MRI.” *International Journal of Imaging Systems and Technology*, vol. 14, no. 3, 2004, pp. 131–138., doi:10.1002/ima.20016.

88. Veraart, Jelle, et al. “Weighted Linear Least Squares Estimation of Diffusion MRI Parameters: Strengths, Limitations, and Pitfalls.” *NeuroImage*, vol. 81, 2013, pp. 335–346., doi:10.1016/j.neuroimage.2013.05.028.

89. Webb, Bridgette, et al. “Temperature Dependence of Viscosity, Relaxation Times (T1, T2) and Simulated Contrast for Potential Perfusates in Post-Mortem Mr Angiography (PMMRA).” *International Journal of Legal Medicine*, vol. 131, no. 3, 2016, pp. 739–749., doi:10.1007/s00414-016-1482-5.

90. Weigand, W., Egbert, A., & Sadler, H. (2015). Determining the relaxation times, T1, T2, and T2*, in glycerin using pulsed magnetic resonance. *University of San Diego*.

91. Xiangrui Li (2022). xiangruili/dicm2nii (https://github.com/xiangruili/dicm2nii), GitHub. Retrieved January, 10, 2022.

92. Stephen (2022). Natural-Order Filename Sort. (https://www.mathworks.com/matlabcentral/fileexchange/47434-natural-order-filename-sort), MATLAB Central File Exchange. Retrieved April 27, 2020.

Appendix A.1 – T_1-Mapping Calculation

After acquisition of the T_1-mapping data, the dicom files are organized by repetition time, then by slices. The first step involved separation of each slice into its own file; thus with six repetition times, every sixth dicom file is used to create a new NIfTI file. The final number of NIfTI files corresponds to the total number of slices with an individual image for each TR value applied. The NIfTI file is organized by matrix-x dimension: matrix-y dimension: slice: corresponding TR image (six for the experiments discussed). These files are then registered to the first TR image from each series in order to account for any motion not corrected through gating which times image acquisition between breaths. Then, the NIfTI files are independently loaded into MATLAB, and one selected image from the TR sequence is separated as a new variable to determine the binary mask through segmentation. Finally, analysis of the T_1-relaxation rates was done based on work initially developed by Birkbeck in 2017 with alterations to account for the manually created binary mask for more accurate segmentation and to work with the ordering of the data in the NIfTI files.[65] The generic equation used to solve for the T_1-relaxation time from a saturation recovery sequence is $S(t) = A\left(1 - e^{-\frac{t}{T_1}}\right)$ where *t* is the TR time and *A* is a fitting parameter related to the equilibrium magnetization.[24,25] The "modelfun" function in MATLAB is used to fit the equation to solve for the T_1-relaxation rate with an additional normalization factor added to the end for a final equation of $A\left(1 - e^{\frac{t}{T_1}}\right) - B$, where *B* is the normalization factor. The relaxation rates are then determined for each pixel and reconstructed to generate the color-coded T_1 maps.

Appendix A.2 – T_1-Mapping

```
%% T1 Mapping acquired with Saturation Recovery methods
% Dicom images should be separated into new files according to slice

% T1 mapping equation from MATLAB File Exchange Mathew Birkeck adapted to
% work with NIfTI files and incorporate registration tactics and manual
% segmentation
% Matt Birkbeck (2022). Parametric Mapping Scripts for MRI data
% (https://www.mathworks.com/matlabcentral/fileexchange/64579-parametric-mapping-scripts-for-mri-data),
% MATLAB Central File Exchange.

% Dicom to NIfti conversion based on the work of Xiangrui Li (2022).
% xiangruili/dicm2nii (https://github.com/xiangruili/dicm2nii), GitHub.

% Natsortfiles: Stephen (2022). Natural-Order Filename Sort.
% (https://www.mathworks.com/matlabcentral/fileexchange/47434-natural-order-filename-sort),
% MATLAB Central File Exchange.



%% Read in Dicom Files to Segment and convert to NII

% Where all main files will be stored
Main = '% File for main files for use';

% Slice One - repeat for as many slices as you have
Main_1   = '% Folder for slice 1 dicom files';

% If registration is needed continue on, if not comment out and change
% filepath below
% Main file path for registered images
Main_R1 = '% Folder for slice 1 registered images';

% Must add as many slices as you have for T1 map
S_one = dir('% Folder for slice 1 dicom files"/*.dcm');
S_one = natsortfiles({S_one.name}); % sort files by name so files are ordred by TR when segmenting and registering
[~, a] = size(S_one);
```

```
% Register Images to first image captured in case of motion
for i = 1:a
    filename = strcat(Main_1, S_one(i));
    filename = string({filename});                  % convert filename to string
    info     = dicominfo(filename);                 % you MUST preserve dicom info
    image    = dicomread(info);

    Ffilename = strcat(Main_1, S_one(1));           % fixed image will be the first TR
    Ffilename = string({Ffilename});
    Finfo     = dicominfo(Ffilename);
    FImage    = dicomread(Finfo);
    R         = registerImages_affine_monomodal(image, FImage); % moving image, fixed image
    RegIm     = R(1);                               % denote where in the array the registered image is
    Reg       = RegIm.RegisteredImage;              % pull out registered image
    dicomwrite(Reg, sprintf([Main_1 'Registered/Image_%d.dcm'],i),info); % write new registered image file
end

% Load new Registered Images
clear S_one

S_one = dir('% Folder for slice 1 dicom files"/*.dcm');
S_one = natsortfiles({S_one.name}); % sort files by name so files are ordred by TR when segmenting and registering
[~, a] = size(S_one);

filename = strcat(Main_R1, S_one(1));
filename = string({filename});
info     = dicominfo(filename);
image    = dicomread(info);

Temp_size = size(image);

% Convert dicom to nii

dicm2nii(Main_R1, Main_R1, 0);
D = dir([Main_R1 '*.nii']);
D = natsortfiles({D.name});
```

```
D = string(D);
D = strcat(Main_R1, D);
D = niftiread(D);
niftiwrite(D, [Main '1_slice.nii']);

% Create Binary Mask for each Slice
% Open Image segmenter
% Once segmented "Export Images"
% BW is binary mask

imageSegmenter(image)
pause;                    % Exit pause by hitting enter in main window once
                        % BW image has been exported
BW = permute(BW, [2 1]);    % BW needs to be adjusted to account for inversion
BW = fliplr(BW);            % with nii conversion of image file

imwrite(BW, [Main '1_binaryimage.png']);
clear BW

% Slice Two - repeat for as many slices as you have
Main_2   = '% Folder for slice 2 dicom files';

% If registration is needed continue on, if not comment out and change
% filepath below
% Main file path for registered images
Main_R2 = '% Folder for registered slice 2 dicom files';

% Must add as many slices as you have for T1 map
S_two = dir('% Folder for slice 2 dicom files"/*.dcm');
S_two = natsortfiles({S_two.name}); % sort files by name so files are ordred by TR when segmenting and registering
[~, a] = size(S_two);

% Register Images to first image captured in case of motion
for i = 1:a
    filename = strcat(Main_2, S_two(i));
    filename = string({filename});                % convert filename to string
    info     = dicominfo(filename);               % you MUST preserve dicom info
```

```
    image    = dicomread(info);

    Ffilename = strcat(Main_2, S_two(1));          % fixed image will be the first TR
    Ffilename = string({Ffilename});
    Finfo     = dicominfo(Ffilename);
    FImage    = dicomread(Finfo);
    R         = registerImages_affine_monomodal(image, FImage); % moving image, fixed image
    RegIm     = R(1);                              % denote where in the array the registered image is
    Reg       = RegIm.RegisteredImage;             % pull out registered image
    dicomwrite(Reg, sprintf([Main_2 'Registered/Image_%d.dcm'],i),info); % write new registered image file
end

% Load new Registered Images
clear S_two

S_two = dir('% Folder for registered slice 2 dicom files"/*.dcm');
S_two = natsortfiles({S_two.name}); % sort files by name so files are ordred by TR when segmenting and registering
[~, a] = size(S_two);

filename = strcat(Main_R2, S_two(1));
filename = string({filename});
info     = dicominfo(filename);
image    = dicomread(info);

% Convert dicom to nii

dicm2nii(Main_R2, Main_R2, 0);
D = dir([Main_R2 '*.nii']);
D = natsortfiles({D.name});
D = string(D);
D = strcat(Main_R2, D);
D = niftiread(D);
niftiwrite(D, [Main '2_slice.nii']);

% Create Binary Mask for each Slice
% Open Image segmenter
% Once segmented "Export Images"
```

```
% BW is binary mask
imageSegmenter(image)
pause;

BW = permute(BW, [2 1]);
BW = fliplr(BW);

imwrite(BW, [Main '2_binaryimage.png']);
clear BW

% Slice Three - repeat for as many slices as you have
Main_3   = '% Folder for slice 3 dicom files';

% If registration is needed continue on, if not comment out and change
% filepath below
% Main file path for registered images
Main_R3 = '% Folder for registered slice 3 dicom files';

% Must add as many slices as you have for T1 map
S_three = dir('% Folder for slice 3 dicom files''/*.dcm');
S_three = natsortfiles({S_three.name}); % sort files by name so files are ordred by TR when segmenting and
registering
[~, a] = size(S_three);

% Register Images to first image captured in case of motion
for i = 1:a
   filename = strcat(Main_3, S_three(i));
   filename = string({filename});              % convert filename to string
   info     = dicominfo(filename);             % you MUST preserve dicom info
   image    = dicomread(info);

   Ffilename = strcat(Main_3, S_three(1));      % fixed image will be the first TR
   Ffilename = string({Ffilename});
   Finfo     = dicominfo(Ffilename);
   FImage    = dicomread(Finfo);
   R         = registerImages_affine_monomodal(image, FImage); % moving image, fixed image
   RegIm     = R(1);                           % denote where in the array the registered image is
```

```
    Reg       = RegIm.RegisteredImage;              % pull out registered image
    dicomwrite(Reg, sprintf([Main_3 'Registered/Image_%d.dcm'],i),info); % write new registered image file
end

% Load new Registered Images
clear S_three

S_three = dir('% Folder for registered slice 3 dicom files"/*.dcm');
S_three = natsortfiles({S_three.name}); % sort files by name so files are ordred by TR when segmenting and
registering
[~, a] = size(S_three);

filename = strcat(Main_R3, S_three(1));
filename = string({filename});
info     = dicominfo(filename);
image    = dicomread(info);

dicm2nii(Main_R3, Main_R3, 0);
D = dir([Main_R3 '*.nii']);
D = natsortfiles({D.name});
D = string(D);
D = strcat(Main_R3, D);
D = niftiread(D);
niftiwrite(D, [Main '3_slice.nii']);

% Create Binary Mask for each Slice
% Open Image segmenter
% Once segmented "Export Images"
% BW is binary mask
imageSegmenter(image)
pause;

BW = permute(BW, [2 1]);
BW = fliplr(BW);

imwrite(BW, [Main '3_binaryimage.png']);
clear BW
```

```
%% T1 map

%% Matthew Birkbeck v1 28/9/17

% % This script allows you to calculate a T1 map from a series of
% Saturation Recovery Images at different TR values (ms). The images should
% be in an Analyze7.5 format avaliable through MRIconvert, you will need to
% change the current directory and path for scripts. Ensure you have the
% nasortfiles script and the threshold scripts to run this script. You can
% change the starting, upper and lower bounds of the fit in the loop.

%% Alterations made to work with Bruker file formats and previously segmented data

clear all; close all; clc

% Change directory
cd '% Folder with main files';



% load your data
A     = dir('*.nii');               % changed to .nii
N     = natsortfiles({A.name});
Image1 = niftiread(N{1});            % Changed to niftiread
[x,y]  = size(Image1(:,:,1));        % or [x,y,~] = size(Image1)

% Have to squeeze out singleton dimension in NIfTI data for "slice one"
Image1 = squeeze(Image1);            % so you can see the six TR

% Read in NIfTI files and concatenate them into a matrix gives a
% (M x N x nRepetition x nSlice) Matrix
tr_image = zeros(x,y,size(Image1,3),length(N));
for n=1:length(N)
  tr_image(:,:,:,n) = niftiread(N{n});
  if size(tr_image(:,:,2,n)) ~= size(tr_image(:,:,1,n)) % changed this to work with our file types
     disp('error: image matrix input sizes not equal');
  else
```

```
        % do nothing
    end
end

% tr_image is 256 x 128 x 6 x 3 so (M X N X Rep X Slice
% tr_image = im2double(niftiread(N{1}));

% Create a matrix that has all the masks for each slice of tr_image in
B = dir('*.png');
P = natsortfiles({B.name})';
outputmasks4d = zeros(x,y,6,length(P));
for i = 1:6
    for j = 1:length(P)
    BW    = imread(B(j).name);            % binarized image
    BW2   = imfill(BW,'holes');           % fill in holes
    BW3   = imdilate(BW2,strel('disk',1)); % dilate and erode image
    BW4   = imerode(BW3,strel('disk',1));  %  to smooth edges
    outputmasks4d(:,:,i,j) = BW4;
    end
end

% mask the images
finalimagematrix = (outputmasks4d.*tr_image)./max(outputmasks4d(:));



% input the repetition times
% Information found from method file in BRUKER or
% annotate frome experiment. Can use prompt from Birkbeck below
% or load in from text files with TR times organized as column

% prompt  = {'Repetition Times'};
% def     = {''};
% answer  = inputdlg(prompt,'',1,def);
% tvalues = str2num(answer{1});

% To load in from table
fileID = fopen('T1Times.txt', 'r');
```

```
tvalues = fscanf(fileID, '%f');
tvalues = tvalues';

% predefine map matricies for later
map  = zeros(x,y,length(B));
amap = zeros(x,y,length(B));

% View first few slices of masked images
Image1 = finalimagematrix(:,:,:,1);
Image2 = finalimagematrix(:,:,:,2);
Image3 = finalimagematrix(:,:,:,3);



montage([Image1(:,:,1),Image2(:,:,1),Image3(:,:,1)],...
'DisplayRange',[0 max(Image1(:))])
title('Montage showing Slices')



%% Acceleration
% To save the headache of working in 4D (M x N x nRepetition x nSlice) Matrix
% I have chosen to selectivly process  each image slice independently.
% k = nSlice.

for k = 1:size(tr_image(:,:,:,:),4)

    % This is the image slice which is remapped into a 1D array so it can
    % be processed more efficicently.
    Img             = finalimagematrix(:,:,:,k);
    [x,y,nInversion] = size(Img); % dimension of the Image

    % to get a 1D array representing our Img matrix the Img matrix is
    % re-written to a 1D vector Ydata - this represents our (x*y)
    % seperate problems with n data points in each problem.
    ymat  = reshape(Img,[],nInversion);
    Ydata = reshape(ymat',[],1);
```

```
% xdata is a vector which contains the time for each pixel element
% in Y
xmat  = kron(tvalues,ones(x*y,1));
Xdata = reshape(xmat',[],1);

% It is necessary to index our problem. i.e. in our 1D array which
% element corrisponds to which problem.
batchindex = kron(1:x*y,ones(1,nInversion));

% Set the function parameters. Note its a bit confusing here as we are
% solving for modelfun = 0; so if we want y = m*x+c then we need to write
% modelfun = @(variables) m*x+c-y.
modelfun = @(a,Xk,Yk) a(1)*(1-exp(-Xk/a(2))) - Yk;

estcoefs     = zeros(x*y,2); % estimated coefficients a(1) a(2) a(3)
param0       = [1 500];      % starting values
lb           = [0 0];        % lower bounds
ub           = [1 5000];     % upper bounds
opts         = optimset('lsqnonlin'); % method
opts.Display = 'off';        % verbose off

% we don't want to go all the way through matrix. It is sufficient to
% say that if a pixel is blank at each time point then it will not have a
% relaxation time value.

msk  = sum(Img,3);
mask = reshape(msk,[],1)'>0; % 1 for non zero pixels.

parfor i = 1:x*y
   if  mask(i) == 1 % checks the times have non-zero pixel values
       k  = (batchindex == i);
       Xk = Xdata(k); % this is the time data for problem i
       Yk = Ydata(k); % this is the pixel value data for problem i

       Yk = Yk(:)/max(Yk(:)) % normalise the Y-data for each series

       % estimate coefficients using our earlier model function
```

```
            estcoefs(i,:) = lsqnonlin(@(C) modelfun(C,Xk,Yk),param0,lb,ub,opts);
        end
    end


    % get our values out, reshape them back into an image. Each image
    % is a matrix of [x,y,nSlice].
    T1 = estcoefs(:,2);
    T1img(:,:,k) = reshape(T1,x,y);
    a = estcoefs(:,1);
    aimg(:,:,k) = reshape(a,x,y);
end

% Store T1 relaxation values for each voxel
 T1new = reshape(T1, [256 128]);

% Display results
figure;
montage(reshape(T1img,[size(T1img,1), size(T1img,2), 1, size(T1img,3)]),'DisplayRange',[0 4000]);
colormap(jet) % added to produce color in T1 mapping
colorbar


dcm = datacursormode;
dcm.Enable = 'on';
dcm.DisplayStyle = 'window';




%% Make each file a nifti image format and save to the directory specified above
nio = make_nii(T1img);
save_nii(nio,'T1img');

save T1Mapping
```

Appendix A.3 – T_2-Mapping

The emerging field of quantitative MRI (qMRI) utilizes the ability to measure different constants, such as T_1 and T_2 relaxation times, as a method of analyzing illness and disease difficult to reach through traditional clinical methods. Through the use of T_2- mapping it is possible to measure differences in the T_2- relaxation times of various tissues in order to gain more quantitative data regarding disease progression, prognosis, and effectivity of treatments. Currently the use of T_2 mapping is heavily focused on applications for joint diseases such as rheumatoid and osteoarthritis,[27] as well as assisting in the grading and assessment of aggressiveness of prostate cancer.[28] However, with more development into the methods for acquisition and the current growing movement towards ultra-high field MRI it will be possible to apply this method to other areas such as monitoring pancreatic cancer, pancreatic cancer treatments, psychological disorders, and other maladies.

The T_2 value is a time constant determined by the rate at which once excited protons will reach equilibrium or fall out of phase with its cohorts.[5] This constant is measured by the time taken for precessing protons to lose phase coherence among nuclei spinning alongside the main magnetic field generally defined by $SI = S_0 * e^{-\frac{TE}{T_2}}$ (or some close variation of this equation); where SI is the signal intensity of the pixel at the acquired echo time, TE is the echo time for the corresponding image, and S_0 is the signal at equilibrium magnetization.[5, 27, 28, 64] Unfortunately, the T_2 values change depending upon the strength of the MRI instrumentation. Additionally, the appearance of simulate echoes with certain pulse sequences can make the determination of apparent versus true transverse relaxation times difficult. These issues will need to be addressed and circumvented in order to produce pragmatic results.

Appendix A.4 – Application of T_2 Mapping

While DTI can be insightful and lead to grand revelations regarding the health and integrity of tissue structures, it can be enhanced with an alternative method such as T_2 mapping which records the relaxation times in either single slice, or multi slice approaches. T_2 mapping can also be limited to select regions of interest or spread across an entire subject in order to offer preliminary starting points, determine the impact on surrounding tissues for a variety of maladies, or monitor the improvements made through treatment modalities. The T_2 time constant is determined by the rate at which the once excited protons will reach equilibrium or fall out of phase with its cohorts.[5] Upon the application of a multi-echo spin echo pulse sequence it is possible to generate a T_2 map based upon the T_2 relaxation times of the tissues encapsulated within each voxel to better identify locations of micro-metastases and monitor the progression of pancreatic cancer. Images will be acquired at a variety of echo times as seen in Figure 6.1, and

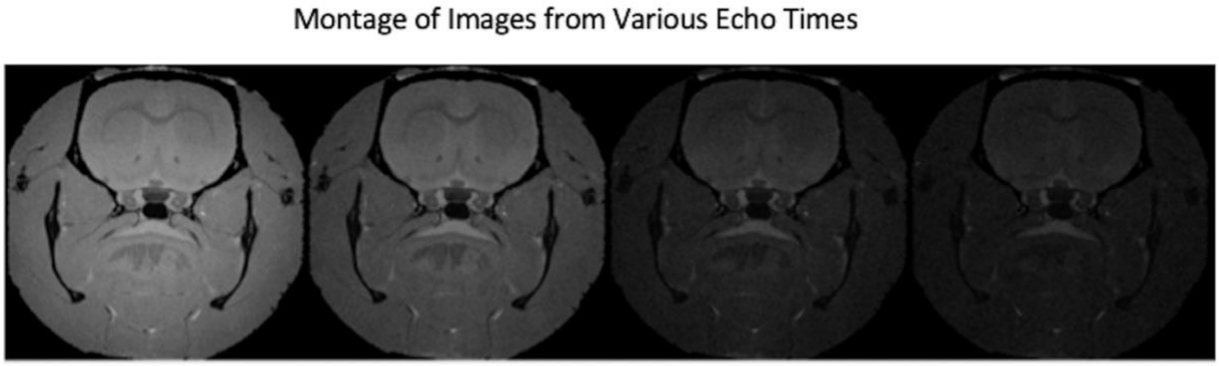

Figure 6.1: Demonstrative images gained from acquisition at a variety of echo times in T_2-mapping protocol.

compiled into a series of slices, where each slice will have a range of 10+ images recorded at various echo time points. The signal intensity of each voxel for the corresponding echo time will then be used to solve a mono-exponential decay equation, $S(t) = S_0 e^{-\frac{t}{T_2}}$, using a non-linear

approach to determine the T_2 measurements for each voxel. In the above equation, S(t) is the signal of the voxel at the acquired echo time, S_0 is the signal at equilibrium magnetization, t is the echo time used during acquisition, and T_2 is the relaxation time to be solved for.[5, 27, 28, 64] In order to mitigate the impact of stimulated echoes, the first echo time will be omitted from the T_2-mapping calculations, and the corresponding T_2 maps similar to what is shown as Figure 6.2 will be generated from the remaining echo time images. The images are compiled into various NIfTI files separated by slices and dependent upon acquired echo time. Images are then segmented and then masked to limit the calculation of relaxation times to the primary regions of interest and condense calculation time by omitting background pixels. The calculation is performed through similar scripts to the T_1 mapping calculation with alterations of the equation used to solve for the transverse relaxation times, and echo times as an input rather than repetition times utilizing optimized scripts based upon the work done by Birkbeck.[64]

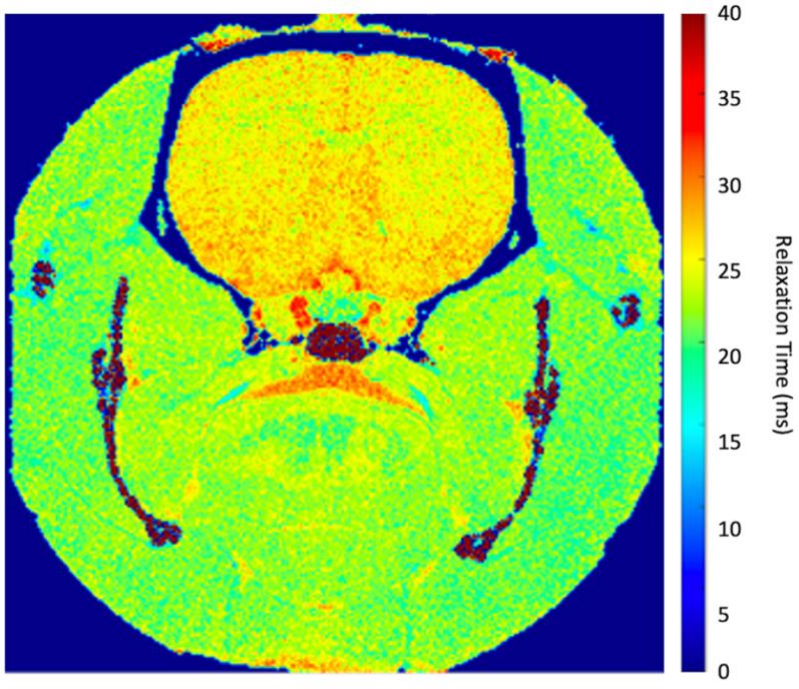

Figure 6.2: T_2 map of a rat brain where various colors indicate the T_2 relaxation time. The color bar on the left gives the corresponding echo time for each color.

Appendix B.1 – Diffusion Tensor Calculations and Other Metrics

Acquiring diffusion data sets requires the use of a minimum of six non-colinear gradients applied to generate six diffusion sensitized images with an additional image with no diffusion gradients applied, called the A_0 image.[14] However, in order to minimize the difficulty in resolving fiber tract crossing and minimize the uncertainty in the tensor, more than six directions are often acquired in order to obtain an overdetermined system.[14,17,20,39] Using the signal from these diffusion images and A_0 images, the diffusion tensor can be estimated through linearization of the Stejskal-Tanner equation.[17,48,67] As diffusion gradients are applied, the corresponding diffusion images are acquired (S_k) and have a decreased signal compared to the initial image(s) acquired without the application of diffusion gradients (S_0). The comparison of these signals is used to determine the diffusion tensor through $S_k = S_o e^{-b\hat{g}_k^T \boldsymbol{D} \hat{g}_k}$ where b is LeBihan's factor describing the pulse sequence and gradient strength, and $\hat{g}_k^T \boldsymbol{D} \hat{g}_k$ is the diffusivity in the direction of the g_k-th gradient orientation.[14,17] For overdetermined systems this equation can be further simplified as $\frac{S_k}{S_0} = e^{-bD}$.[14,17,38,48] Using the full scope of diffusion images, a system of equations is used to solve for the diffusion tensor **D**, the methods of which depends upon the software used to analyze the data. The pipeline presented in chapter 4 utilized a weighted linear fit in order to determine the diffusion tensor due to the faster computation time compared to other methods while balancing the overall accuracy and precision of the measurements compared to alternative approaches. The most common approaches to solving the aforementioned system of equations is weighted least squares, non-weighted least squares, and nonlinear fitting. The weighted least squares utilize an inverse log transformed signal variance to determine the overall weighting, and preferentially weights higher signal values over lower signal values and allows for the utilization of either a **B** matrix or **H** matrix approach. This is important due to the ability to use more than

one b-value in the **B** matrix as well as allowing for multiple A_0 image signals.[20] Additionally, $\mathbf{B^T E^{-1} B}$ where **E** is a diagonal matrix of the squared signal intensity at *i* over the squared variance, and can be used for propagation of error calculations which can in turn be used to improve analysis techniques or imaging parameters.[20] If a non-weighted least squares approach is used, or the **H** matrix is used instead, the images which were taken without a diffusion gradient applied must be averaged for one vector of the $\ln(S_0)$. Due to the different components included in the **B** matrix, and the increased accuracy and precision of the weighted approach, a template of this analysis structure was used in designing the pipeline.

Once the diffusion tensor was estimated, eigenvalue decomposition yielded the necessary eigenvalues and eigenvectors to determine other image types for quantitative and qualitative analysis. The trace was defined as the sum of the eigenvalues and used to normalize most measurements such as FA, C_l, C_p, and C_s. The apparent diffusion coefficient maps (ADC) were determined concurrently with the diffusion estimation based upon $\frac{\ln(S_k)-\ln(S_0)}{b_k}$, where S_k is the signal of the corresponding diffusion gradient image, S_0 is the signal of the averaged A_0 images, and b_k is the b value for the image with the applied diffusion gradient.[20,21,22,67] For the alternative metrics, the mean diffusivity (MD) was determined by $\frac{\lambda_1+\lambda_2+\lambda_3}{3}$ (and is often times referred to as/in place of the ADC), the trace is the sum of the eigenvalues ($\lambda_1 + \lambda_2 + \lambda_3$), fractional anisotropy (FA) is determined by $\sqrt{\frac{(\lambda_1-MD)^2+(\lambda_2-MD)^2+(\lambda_3-MD)^2}{2(\lambda_1^2+\lambda_2^2+\lambda_3^2)}}$, axial diffusivity (AD) by λ_1 (i.e. the primary diffusion direction's relative intensity), radial diffusivity (RD) by $\frac{\lambda_2+\lambda_3}{2}$, and then finally decomposition of the relative anisotropy to describe if the diffusion ellipse is more linear, planar, or spherical in shape via C_l, $\frac{\lambda_1-\lambda_2}{Trace}$, C_p, $\frac{2(\lambda_2-\lambda_3)}{Trace}$, and C_s, $\frac{3\lambda_3}{Trace}$.[14,67] Each of these metrics has been shown to present different reflections of neuronal and axonal health, and are

commonly acquired to monitor different brain disease progressions and grades; however they are starting to become more apparent in alternative applications such as characterization of tumors outside of the brain as well.[30]

The pipeline involved altering a series of existing platforms available through the MATLAB File Exchange and altering a few key components into new functions that could be called upon to analyze each data set. Alterations were made to work with a larger variety of starting file types and setting a variety of values which needed to be input manually for diffusion tensor estimation to handles that could be extracted from the image itself, omitting the need for the user to input identical pieces of information in various components of the pipeline or change them between different sample types manually. Additionally, currently existing scripts determined only the initial fractional anisotropy and diffusion tensor images, so additional components to yield the variety of image types mentioned above were added in order to provide AD, RD, DWI, and MD images as well. The addition of these components was done based upon the equations above.

Appendix B.2– Linked EPI

```
%% Complete EPI Analysis from Pre-Processing to Post-Processing
% Kansas State University
% The Bossmann Group
% Macy Payne

% Diffusion tensor estimation adapted from Chia-Hao C.
% DTI Fiber Tractography (Streamline Tracking Technique).
% MATLAB Central File Exchange https://www.mathworks.com/matlabcentral
% /fileexchange/34008-dti-fiber-tractography-streamline-tracking-technique.
% Published online 2012

% Dicom to Nifti Conversion: . Xiangrui Li (2022). xiangruili/dicm2nii
% (https://github.com/xiangruili/dicm2nii), GitHub.

% Natsortfiles: Stephen (2022). Natural-Order Filename Sort.
% (https://www.mathworks.com/matlabcentral/fileexchange/47434-natural-order-filename-sort),
% MATLAB Central File Exchange.

clear all; close all; clc

%% Change directory to data set being worked with
% All post processing folders should already be located in this filepath
% If not make folders for AD,FA,DWI,DWI for BC and NR
% Make sure no existing NIfTI files present

cd '% directory of files for diffusion imaging';

% Set main filepath name
Main   = '% main filepath';

% Input key image characteristics
FOV    = [];          % FOV of diffusion images
Matrix = [];          % Matrix of diffusion images
Slices = Matrix(3);
A0_num = % Number of A0 images;
```

```
bval   = % b value used in imaging;

fnam = 'KeyVariables.mat';
save(fnam, 'Main');
save(fnam, 'FOV', '-append');
save(fnam, 'Matrix', '-append');
save(fnam, 'Slices', '-append');
save(fnam, 'A0_num', '-append');
save(fnam, 'bval', '-append');

% Input gradient directions applied
% Can also call forward from text files if needed
% Recommended to store various gradient options as text files
% Alternatively, input B-Matrix if available
gradvec = readtable('% name of text file with gradient information');
gradvec = gradvec{:,:};

save(fnam, 'gradvec', '-append');

% Start image registrations (Affine), be sure to save transformation
% matrix for gradient table rotations
% Eddy Current correction
% Motion Correction
PreProcessing

%% To Recover GradVec Rotations

% %Input a Middle Slice
i = 9;
j = i:Slices:b;
j = j(4);
filename = strcat(Folder_Dicom, All_Dicom(j));
filename = string({filename}); % convert filename to string
info     = dicominfo(filename); % you MUST preserve dicom info
image    = mat2gray(dicomread(info));

Ffilename = strcat(Folder_A0, A0_Dicom(i));
```

```
Ffilename = string({Ffilename});
Finfo     = dicominfo(Ffilename);
FImage    = mat2gray(dicomread(Finfo));

% Select Control Points
% Save a minimum of 3 control points to use to retrieve angle information
cpselect(image, FImage)
pause;

tform          = fitgeotrans(movingPoints,fixedPoints,'affine');
tformInv       = invert(tform);
Tinv           = tformInv.T;
ss             = Tinv(2,1);
sc             = Tinv(1,1);
scale_recovered = sqrt(ss*ss + sc*sc);

% Save Theta
theta          = atan2(ss,sc)*180/pi;
fnam = 'KeyVariables.mat';
save(fnam, 'theta', '-append');



% Convert registered dicom to NIfTI

dicm2nii([Main 'RegisteredDicom/'], Main, 0);
D = dir([Main '*.nii']);
D = natsortfiles({D.name});
D = niftiread(string(D));

% Rename NIfTI to whatever you would like
% Sample name works best
% But be sure to put some # in front of it so you can call it forward easily

niftiwrite(D, '% sample name');
Z = dir([Main '*.nii']);
Z = natsortfiles({Z.name});
Z = niftiread(string(Z(1)));
```

```
save(fnam, 'Z', '-append');

%% If permutation needed do so below
% XX = permute(Z, [1 3 2 4]);
% Re-write file again, with a number in front to call forward easily
% niftiwrite(XX, 'Rat_366_11.nii');

% Determine diffusion data
EPI_Analysis_S

% Change file types of noise reduction
Convert_To_png

% Noise Removal
PostProcessing_DTIDWIFA
```

Appendix B.3 – Pre-Processing

```
%% New step in Analysis
% Register to A0
% Motion Correction
% Mild eddy current correction

% Natsortfiles: Stephen (2022). Natural-Order Filename Sort.
% (https://www.mathworks.com/matlabcentral/fileexchange/47434-natural-order-filename-sort),
% MATLAB Central File Exchange.



All_Dicom = dir([Main '1/dicom/*.dcm']);    % load in directory
All_Dicom = natsortfiles({All_Dicom.name}); % filenames must be sorted by name (otherwise they are all out of order
[~, b]    = size(All_Dicom);

Folder_Dicom = [Main '1/dicom/'];
Folder_A0    = [Main 'A0Averages/'];

% Average A0 Images
% Change depending upon number of A0 images taken to acquire average
% of A0 images
for i = 1:Slices                            % to take averages of images every slice
   filename_A = strcat(Folder_Dicom, All_Dicom(i)); % load in individual file
   filename_A = string({filename_A});             % convert filename to string instead of cell array
   info_A     = dicominfo(filename_A);            % you MUST preserve dicom info
   image_A    = dicomread(info_A);
   A          = image_A;

   filename_B = strcat(Folder_Dicom, All_Dicom(i+Slices));
   filename_B = string({filename_B});
   info_B     = dicominfo(filename_B);           % you MUST preserve dicom info
   image_B    = dicomread(info_B);
   B          = image_B;
```

```
    C = (A + B)/2;

    % Save Averaged A0 in new location

    dicomwrite(C,sprintf([Folder_A0 'A0_%d.dcm'],i), info_A);

end


clearvars -except Main Slices


% Load new A0 Average images

Folder_Dicom = [Main '1/dicom/'];
Folder_A0    = [Main 'A0Averages/'];

A0_Dicom = dir([Main 'A0Averages/*.dcm']);  % load in directory
A0_Dicom = natsortfiles({A0_Dicom.name});   % filenames must be sorted by name (otherwise they are all out of
order
[~, a]   = size(A0_Dicom);

% ReLoad the rest of the images
All_Dicom = dir([Main '1/dicom/*.dcm']);    % load in directory
All_Dicom = natsortfiles({All_Dicom.name}); % filenames must be sorted by name (otherwise they are all out of order
[~, b]    = size(All_Dicom);

% Register Images

RegisteredDicom = [Main 'RegisteredDicom/'];


for i = 1:Slices
    for j = i:Slices:b
        filename = strcat(Folder_Dicom, All_Dicom(j));
        filename = string({filename});              % convert filename to string
        info     = dicominfo(filename);             % you MUST preserve dicom info
```

```
        image    = dicomread(info);

        Ffilename = strcat(Folder_A0, A0_Dicom(i));
        Ffilename = string({Ffilename});
        Finfo     = dicominfo(Ffilename);
        FImage    = dicomread(Finfo);
        % If images are of different sizes perform below step
        %FImage = imresize(FImage, size(image));       % fixed image matix must be of same size as moving image
        R     = registerImages_affine_monomodal(image, FImage); % moving image, fixed image
        RegIm = R(1);                                % denote where in the array the registered image is
        Reg   = RegIm.RegisteredImage;               % pull out registered image
        dicomwrite(Reg, sprintf([RegisteredDicom 'Image_%d.dcm'],j),info); % write new registered image file
    end
end


beep on
beep
```

Appendix B.4 – Diffusion Tensor Estimation

```
function EPI_Analysis_S

%% Code initially constructed by Chang Chia-Hao
%% Alterations made by Macy Payne to accommodate Bruker data set,
% include gradvec rotation for eddy current correction
% yield MD, RD, AD metrics as well and import different variables directly
% such as FOV, Matrix dimension, gradient directions and diffusion numbers
% New Package Requirements: Phase Array Package

% This code produces a DTI 'mat'-file for "SearchDTI" if you have a Matlab
% However: "you *must* adapt this code in order for it to work...": your
% input (coded somewhere in this script) should be the DWI matrix and b-matrix (or b-value and gradients)
% If all of this is just to much for you to deal with - you may always send me an example data
% set (with all relevant specs, such as the b-value, gradient directions, or b-matrix, voxel-size, etc...)

%-----------------------------------------------------------------
% At the end, data is being saved...it is a fixed nominal structure:
% variables that should be incorporated in the mat-file in order to be able to be loaded into ExploreDTI are:
%
% 1.      DT (diffusion tensor)                          :        4D single matrix (row, column, slice, 9)
% 2.      DWI (diffusion weighted images)      :        4D uint16 matrix (row, column, slice, number of non-DWIs +
DWIs)
% 3.      FA (fractional anisotropy)               :        3D single matrix (row, column, slice)
% 4.      FE (first eigenvector)                          :        4D single matrix (row, column, slice, 3)
% 5.      SE (second eigenvector)                 :        4D single matrix (row, column, slice, 3)
% 6.      FEFA (FA*abs(FE))                               :        4D single matrix (row, column, slice, 3)
% 7.      MDims (matrix dimensions)                :        1D double matrix (e.g. [128 128 60] for [coronal
sagittal axial])
% 8.      VDims (voxel dimensions)              :        1D double matrix (e.g. [2 2 2] in mm)
% 9.      eigval (eigenvalues)                            :        4D single matrix (row, column, slice, 3)
% 10.     b (b-matrix)                                    :        2D single matrix (number of non-DWIs + DWIs, 6)
% 11.   bval (b-value)                : 0D double number in s/mm^2 (e.g., 1500)
% 12.   g (gradient matrix)           : 2D double matrix (number of gradient directions,3)
% 13.   NrB0 (no of non-DWIs)         : 0D double number (e.g., 7)
```

```
% 14.   chi_sq (chi-square)                              :        3D single matrix (row, column, slice)
%-----------------------------------------------------------------

% Technical note: for historical reasons, the FA is within the range [0 sqrt(3)]...so for further analysis,
% divide it by sqrt(3) when using it outside the ExploreDTI environment.
% I try to follow the radiological convention in orientation...so my matrices A(i,j,k) are as follows:
% front 2 back <--> low 2 high 'i' index
% left 2 right <--> low 2 high 'j' index
% bottom 2 top <--> low 2 high 'k' index

% You need the following toolboxes to run this file: Image processing,
% Statistics, and Optimization.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%
%                        Parameters Setting                        % START
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%
% Load key variables referenced and saved from previous steps
load('KeyVariables.mat');

% Load NIfTI file
t = Z;

% Rotate to accomodate rotation during NIfTI conversion
t = imrotate(t,90);

% Obtain matrix size
YY = size(t);

% Obtain number of slices
Slices = YY(3);

DWI = zeros(YY(1), YY(2), Slices, YY(4)); % matrix size, #slices, #directions
for a = 1:Slices                          % No. of Slice
   for b = 1:YY(4)                        % No. of non-DW images + No. of DW images(= no. of gradients);
       DWI(:,:,a,b) = t(:,:,a,b);
```

```
    end
end


VDims = FOV;                          % define voxel size (in mm)
MDims = [YY(1) YY(2) Slices];         % matrix size (in voxels)
NrB0 = A0_num;                        % No. of non-DW images
bval = bval;                          % b-value (in s/mm^2)
NrOfDirections = (YY(4)-NrB0);        % No. of DW images (= no. of gradients)

% gradient directions (should be of size 'NrOfDirections' x 3)
g = gradvec;


fnam = 'Data.mat';                    % Save mat-file

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%
%                    Parameters Setting                    % END
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%

% Extract A0 images
A0 = DWI(:,:,:,1:NrB0);
A0 = ((sum(A0, 4))./2);

% Extract Diffusion signals
D_sig = DWI(:,:,:,3:YY(4));
for i = 1:Slices
    D_sig_sum(:,:,i) = sum(D_sig(:,:,i), 4);
end

% ADC
ADC = zeros(MDims(1), MDims(2), Slices);
for k = 1:Slices
    ADC(:,:,k) = -(log(D_sig(:,:,k)./A0(:,:,k))./bval);
end
```

```
% Scale ADC for viewing
ADC_image = (ADC) * 1000;
niftiwrite(ADC_image, 'ADC.nii');

% Normalize signal
D_sig = mat2gray(D_sig);

% Write image
niftiwrite(D_sig, 'DWI.nii');



% % normalize gradients
gg = sqrt(sum(g.*g,2));
g = g./repmat(gg,[1 3]);


% %%% IMPORTANT %%%
% % Change, if needed, the x,y,z diffusion directions (i.e. permute) /
% % orientations (i.e. sign)...Coordinate system stuff!
g(:,[1 2 3]) = g(:,[1 3 2]); % change x and y (for example)

% Rotate gradient table
rot = rotz(theta);
g_test = g*rot;

% Sign
%  g(:,1)= - g(:,1); % change x-direction (for example)
%  g(:,2)= - g(:,2);
%  g(:,3)=  g(:,3);

% b-matrix (calculated from gradients and b-value)
% *or* you could load your b-matrix directly of course (check coordinate system!!!)
b = bval*[g(:,1).^2 2*g(:,1).*g(:,2) 2*g(:,1).*g(:,3)...
   g(:,2).^2 2*g(:,2).*g(:,3) g(:,3).^2];
b = [zeros(NrB0, 6); b];
```

```
% bval = mean(sum(b(NrB0+1:end,[1 4 6]),2)); % if only a b-matrix was provided


% b should be of size (NrB0 + NrOfDirections) x 6;
% b =
% ###### load DWI data here for example ######


% ##################################

% Create mask ____________________________________________________________

mask_tuning = .8; % fine-tuning range to calculate the mask: [0.5 1.5]
cluster_s = 2;    % omit pixel clusters smaller than 'cluster_s' during masking

BZ = mean(DWI(:,:,:,NrB0+1:end),4);
BZ = mat2gray(BZ);
level = graythresh(BZ(:));
mask_glass = im2bw(BZ(:),mask_tuning*level);
mask = reshape(mask_glass,size(BZ));

[L,NUM] = bwlabeln(mask,6);
SL = L(L~=0);
sd = hist(SL,1:NUM);
[M,I] = max(sd);
mask(L~=I)=0;
mask=double(mask);
mask = imfill(mask,6,'holes');

for i=1:size(mask,3)

   mask(:,:,i) = imfill(mask(:,:,i),'holes');
   mask(:,:,i) = bwareaopen(mask(:,:,i),cluster_s);

end
mask=permute(mask,[1 3 2]);
```

```
for i=1:size(mask,3)

    mask(:,:,i) = imfill(mask(:,:,i),'holes');
    mask(:,:,i) = bwareaopen(mask(:,:,i),cluster_s);
end
mask=permute(mask,[1 3 2]);
mask=permute(mask,[3 2 1]);
for i=1:size(mask,3)
    mask(:,:,i) = imfill(mask(:,:,i),'holes');
    mask(:,:,i) = bwareaopen(mask(:,:,i),cluster_s);
end
mask=permute(mask,[3 2 1]);

se = strel('disk',2,0);
for i=1:size(mask,3)
    mask(:,:,i) = imdilate(mask(:,:,i),se);
    mask(:,:,i) = imfill(mask(:,:,i),'holes');
    mask(:,:,i) = imerode(mask(:,:,i),se);
end

[L,NUM] = bwlabeln(mask,6);
SL = L(L~=0);
sd = hist(SL,1:NUM);
[M,I] = max(sd);
mask(L~=I)=0;
mask=double(mask);
mask = imfill(mask,6,'holes');

mask = logical(mask);

% Done creating mask _____________________________________________

% Estimate diffusion tensor (simple weighted linear approach)
DT = repmat(single(nan), [MDims(1) MDims(2) MDims(3) 9]);
chi_sq = repmat(single(nan), [MDims(1) MDims(2) MDims(3)]);
chi_sq_iqr = repmat(single(nan), [MDims(1) MDims(2) MDims(3)]);
DWIB0 = repmat(single(nan), [MDims(1) MDims(2) MDims(3)]);
```

```
outlier = repmat(logical(0),size(DWI));
order = [1 2 3 5 6 9];
b2 = [ones(size(DWI,4),1) -b];

for k=1:MDims(3);
   for i=1:MDims(1);
      for j=1:MDims(2);

         if mask(i,j,k)

            dwi = double(squeeze(DWI(i,j,k,:)));

            dwi(dwi==0)=1;

            covar = diag(dwi.^2);
            X = inv(b2'*covar*b2)*(b2'*covar)*log(dwi);

            fit = exp(b2*X);
            resid = abs(fit - dwi);
            DWIB0(i,j,k) = fit(1);

            y = prctile(resid,[25 50 75]);
            chi_sq(i,j,k) = y(2);
            chi_sq_iqr(i,j,k) = y(3)-y(1);
            outlier(i,j,k,:) = abs(resid(:)-y(2))>1.5*(y(3)-y(1));
            DT(i,j,k,order) = single(X(2:end));

         end

      end
   end

end
DT(:,:,:,4) = DT(:,:,:,2);
DT(:,:,:,7) = DT(:,:,:,3);
DT(:,:,:,8) = DT(:,:,:,6);
```

```
% Done estimating tensor ____________________________________________________


% Calculate eigenvalue decomposition ____________________________________

order = [1 2 3 5 6 9];
Dummy = repmat(single(nan),[size(DT,1) size(DT,2) size(DT,3)]);

X = repmat(single(0),[6 sum(mask(:))]);

for k=1:size(X,1);
   Dummy = DT(:,:,:,order(k));
   X(k,:)=single(Dummy(mask(:)));
end

A = double(X);
clear X;

a1 = A(1,:);
a2 = A(2,:);
a3 = A(3,:);
a4 = A(4,:);
a5 = A(5,:);
a6 = A(6,:);

L=length(a1);
clear A;

I1 = a1 + a4 + a6;
I2 = a1.*a4 + a1.*a6 + a4.*a6;
I2 = I2 - (a2.*a2 + a3.*a3 + a5.*a5);
I3 = a1.*a4.*a6 + 2*a2.*a3.*a5;
I3 = I3 - (a6.*a2.*a2 + a4.*a3.*a3 + a1.*a5.*a5);

v = (I1.*I1)/9 - I2/3;
s = (I1.*I1.*I1)/27 - (I1.*I2)/6 + I3/2;
```

```
temp = ((s./v).*(1./sqrt(v)));
temp(temp>1)=1;
temp(temp<-1)=-1;

phi = acos(temp)/3;
clear s;

D = repmat(double(0),[3 L]);

D(1,:) = I1/3 + 2*sqrt(v).*cos(phi);
D(2,:) = I1/3 - 2*sqrt(v).*cos(pi/3 + phi);
D(3,:) = I1/3 - 2*sqrt(v).*cos(pi/3 - phi);



PD = repmat(logical(1),[4 L]);
PD(1,:) = I3>0;
clear I1 I2 I3 v phi;

V = repmat(double(0),[3 3 L]);

for i=1:3

    A = a1-D(i,:);
    B = a4-D(i,:);
    C = a6-D(i,:);

    V(1,i,:) = (a2.*a5 - B.*a3).*(a3.*a5 - C.*a2);
    V(2,i,:) = (a3.*a5 - C.*a2).*(a3.*a2 - A.*a5);
    V(3,i,:) = (a2.*a5 - B.*a3).*(a3.*a2 - A.*a5);

end

qv1 = repmat(sqrt(sum(V(:,1,:).*V(:,1,:),1)),[3 1 1]);
qv2 = repmat(sqrt(sum(V(:,2,:).*V(:,2,:),1)),[3 1 1]);
qv3 = repmat(sqrt(sum(V(:,3,:).*V(:,3,:),1)),[3 1 1]);

qv1(qv1==0)=nan;
```

```
qv2(qv2==0)=nan;
qv3(qv3==0)=nan;

V(:,1,:) = V(:,1,:)./qv1;
V(:,2,:) = V(:,2,:)./qv2;
V(:,3,:) = V(:,3,:)./qv3;

PD(2,:) = (a1.*a4 - a2.*a2)>=0;
PD(3,:) = (a1.*a6 - a3.*a3)>=0;
PD(4,:) = (a4.*a6 - a5.*a5)>=0;

PD = all(PD,1);

PosDef = repmat(logical(0), [size(DT,1) size(DT,2) size(DT,3)]);
PosDef(mask)=~PD;
%imagescn(PosDef,[0 1])

FE = repmat(single(nan), [size(DT,1) size(DT,2) size(DT,3) 3]);
SE = repmat(single(nan), [size(DT,1) size(DT,2) size(DT,3) 3]);
eigval = repmat(single(nan), [size(DT,1) size(DT,2) size(DT,3) 3]);

for i=1:3
   Dummy(mask) = D(i,:);
   eigval(:,:,:,i) = Dummy;
   Dummy(mask) = V(i,1,:);
   FE(:,:,:,i) = Dummy;
   Dummy(mask) = V(i,2,:);
   SE(:,:,:,i) = Dummy;
end

clear V;



FA = repmat(single(nan), [size(DT,1) size(DT,2) size(DT,3)]);
FA(mask) = FrAn(D(1,:),D(2,:),D(3,:));
```

```
FEFA = repmat(single(nan), [size(DT,1) size(DT,2) size(DT,3) 3]);

for i=1:3
    Dummy = FE(:,:,:,i);
    Dummy(mask) = FA(mask).*abs(Dummy(mask));
    FEFA(:,:,:,i) = Dummy;
end
FA(FA>1) = 1;
FEFA(FEFA>1) = 1;
FA(mask) = FA(mask)*sqrt(3);

%% To determine/view color coded images
% Can reference against atlas for accuracy
figure(1)
colorcode = real(FEFA(:,:,5,:));
colorcode = squeeze(colorcode);
imshow(colorcode)

%% To view/determine FA and FA values
%FA_val = FA(:,:,);
figure(2)
imshow(FA(:,:,5));

% Re-Order FEFA in order to save color coded NIfTI file and open in
% alternative software
FEFA2 = permute(FEFA, [1,2,4,3]);

% View DTI data
montage(FEFA2)

% Extract eigenvalues
Lambda1 = zeros(MDims(1), MDims(2), Slices);
Lambda2 = zeros(MDims(1), MDims(2), Slices);
Lambda3 = zeros(MDims(1), MDims(2), Slices);

for i = 1:MDims(1)
    for j = 1:MDims(2)
```

```
        for k = 1:Slices
            Lambda1(i,j,k) = max(abs(eigval(i,j,k,:)));
            Lambda2(i,j,k) = median(abs(eigval(i,j,k,:)));
            Lambda3(i,j,k) = min(abs(eigval(i,j,k,:)));
        end
    end
end

% Determine AD image save values as AD image must be scaled
AD = (Lambda1);
AD_image = mat2gray(AD);

% Determine MD
MD = (((Lambda1 + Lambda2 + Lambda3)/3));
MD_image = mat2gray(MD);
% Determine RD
RD = (((Lambda2 + Lambda3)/2));
RD_image = mat2gray(RD);

% Determine MA image
% Mode = I1*I2*I3/(sqrt[(I1-<D>)^2 + (I2-<D>)^2 + (I3-<D>)^2])^3
% <D> = (I1+I2+I3)/3
Lambda_Av = (Lambda1 + Lambda2 + Lambda3)./3;
MA = (Lambda1 .* Lambda2 .* Lambda3)./(sqrt((Lambda1-Lambda_Av).^2 + (Lambda2-Lambda_Av).^2 ...
    + (Lambda3-Lambda_Av).^2)).^3;
MA = normalize(MA, 'range', [-1 1]);



% Done calculating eigenvalue decomposition _____________________________
% Write NIfTI files
niftiwrite(FEFA2, 'DTI.nii');
niftiwrite(FA, 'FA.nii')
niftiwrite(AD_image, 'AD.nii');
niftiwrite(MD_image, 'MD.nii');
niftiwrite(RD_image, 'RD.nii');

% Save mat file for future use and manipulations if needed
```

```
save(fnam,'DT');
save(fnam,'AD');
save(fnam,'MD');
save(fnam,'RD');
save(fnam,'MA');
save(fnam, 'ADC');
save(fnam,'DWI','-append');
save(fnam, 'FEFA2');
save(fnam,'b','-append');
save(fnam,'FE','-append');
save(fnam,'SE','-append');
save(fnam,'FA','-append');
save(fnam,'FEFA','-append');
save(fnam,'VDims','-append');
save(fnam,'MDims','-append');
save(fnam,'eigval','-append');
save(fnam,'bval','-append');
save(fnam,'g','-append');
save(fnam,'NrB0','-append');
save(fnam,'chi_sq','-append');
save(fnam,'DWIB0','-append');
save(fnam,'outlier','-append');
%time = toc;
%disp(['cost time : ' num2str(time) 'sec.'])

beep on
beep

% calculating the Fractional Anisotropy for large data sets...
   function fa = FrAn(L1,L2,L3)

       mask = ~isnan(L1);
       L1 = L1(mask);
       L2 = L2(mask);
       L3 = L3(mask);

       D1 = L1-L2;
```

```
D1 = D1.^2;

D2 = L2-L3;
D2 = D2.^2;

D1 = D1 + D2;

D2 = L3-L1;
D2 = D2.^2;

D1 = D1 + D2;
D1 = sqrt(D1);

clear D2;

L1 = L1.^2;
L2 = L2.^2;
L3 = L3.^2;

L1 = L1 + L2;
L1 = L1 + L3;

clear L2 L3;

L1 = 2*L1;
L1 = sqrt(L1);
fa = repmat(single(nan),size(mask));
fa(mask) = D1./L1;
```

Appendix B.5 – Convert To PNG

```
%% Convert to PNG

StartFolder = Main;
FA  = im2double(niftiread('FA.nii'));
DWI = niftiread('DWI.nii');
DTI = niftiread('DTI.nii');
MD  = niftiread('MD.nii');
AD  = niftiread('AD.nii');
RD  = niftiread('RD.nii');

slices = size(FA,3);
% FA
for i = 1:slices
    A = FA(:,:,i);
    imwrite(A, sprintf([StartFolder 'FA/FA_%d.png'], i))
end

%DWI

for j = 1:slices
    A = DWI(:,:,j);
    imwrite(A, sprintf([StartFolder 'DWI/DWI_%d.png'], j))
end

%DTI

for k = 1:slices
    A = DTI(:,:,:,k);
    imwrite(A, sprintf([StartFolder 'DTI/DTI_%d.png'], k))
end

%MD

for i = 1:slices
```

```
    A = MD(:,:,i);
    imwrite(A, sprintf([StartFolder 'MD/MD_%d.png'], i))
end

%RD

for i = 1:slices
    A = RD(:,:,i);
    imwrite(A, sprintf([StartFolder 'RD/RD_%d.png'], i))
end

% AD

for i = 1:slices
    A = AD(:,:,i);
    imwrite(A, sprintf([StartFolder 'AD/AD_%d.png'], i))
end
```

Appendix B.6– Bias Correction Background

Bias correction involves the use of energy minimization in order to correct for intensity inhomogeneity due to B_0 and B_1 field inhomogeneity. If left uncorrected it can be difficult to get accurate registration between samples due to variations of intensity amongst common tissues.[57] Bias correction operates based upon the premise that MR image intensities can be separated into the following components $I(x) = b(x)J(x) + n(x)$, where $I(x)$ is the intensity at voxel x of the acquired image, $b(x)$ is the bias field, $J(x)$ denotes the true image, and $n(x)$ is additive noise.[68] The estimation of the bias field predicates itself on being able to separate the true image from the inhomogeneities caused by the bias field, and the correction is often done through two main methodologies. One method, called the prospective method, focuses on using special hardware and pulse sequences to avoid the inhomogeneity in the final images at the point of acquisition. While effective, this approach is limited in that it cannot account for sample dependent inhomogeneities.[68] The alternative method is a retrospective method, which uses information from the acquired image to remove inhomogeneities through a filter. However, it relies on high pass filtering which suppresses low spatial frequency signals that cause the inhomogeneous intensities. Unfortunately, this method has a tendency to filter true components of the image as well since some components will be comprised of low frequency signals.[68]

The approach used in the pipeline described in chapter 4 uses a multiplicative intrinsic approach. The multiplicative intrinsic approach decomposes the MR images into their different components previously mentioned through $I(x) = b(x)J(x) + n(x)$, but assumes knowledge about the true image and noise.[49] The following brief investigation of the Multiplicative Intrinsic Component Optimization approach is an overview of the work done by Li et al. in 2014. The assumptions needed for this optimization are that the b field varies smoothly, the true image (J)

characterizes a physical property of the substance being images, and the noise is zero centered. Due to the assumption that *J(x)* characterizes a physical property, it can be assumed constant for any voxels with the same substance/tissue types. I is then viewed as a function $I: \Omega \rightarrow \Re$, and this scene has similar decomposition as previously mentioned, as such I can be decomposed. Using the piecewise constant property of the true image J, and the smoothly varying property of the bias field, a new set of functions is generated in order to solve for the bias field. The bias field is represented as a linear combination of smooth basis functions (g), and estimation is done by finding the best coefficient *w* through $b(x) = \sum_{k=1}^{M} w_k g_k$, where M is the number of basis functions. The coefficients are represented as a column vector and the basis functions as a row vector so that the bias field is now expressed as $b(x) = \boldsymbol{w}^T G(x)$. Now, the constant property of image J allows for the assumption that for N types of tissues/samples in image domain Ω, the tue image is approximately constant c_i for the *i*-th tissue/sample in the Ω_i region that holds that tissue. This allows each tissue/sample/region to be represented by a membership function u_i, $u_i(x) = 1 \; for \; x \in \Omega_i$ and $u_i(x) = 0 \; for \; x \notin \Omega_i$. However, due to partial volume effects a fuzzy membership function is required where N tissues/samples will take values between 0 and 1 such that $\sum_{i=1}^{N} u_i(x) = 1$, or the percentage of the *i*-th tissue/sample that is present in voxel x. These membership functions can be represented as a column **u**; $u = \triangleq \{\boldsymbol{u} = (u_1, \dots, u_N)^T : 0 \leq u_i(x) \leq 1, i = 1, \dots, N, \text{and } \sum_{i=1}^{N} u_i(x) = 1, \text{ for all } x \in \Omega\}$. The estimated bias field is now $\frac{I}{b}$ and the energy being minimized is $F(b,J) = \int_{\Omega} |I(x) - b(x)J(x)|^2 dx$. Confinement of b and J are done based on the knowledge that J is approximately constant and thus confined to the subspace of $J(x) = \sum_{i=1}^{N} c_i u_i(x)$ and that the bias field is confined to $b(x) = \boldsymbol{w}^T G(x)$. Now, the energy $F(b,J)$ can be represented in terms of **u**, **c**, and **w** such that $F(b,J) = F(\boldsymbol{u}, \boldsymbol{c}, \boldsymbol{w}) = \int_{\Omega} |I(x) - \boldsymbol{w}^T G(x) \sum_{i=1}^{N} c_i u_i(x)|^2 dx$. Optimization is then performed with respect to **u**, **c**, and **w**

in order to minimize the energy by region; $F(\boldsymbol{u}, \boldsymbol{c}, \boldsymbol{w}) = \sum_{i=1}^{N} \int_{\Omega} |I(x) - \boldsymbol{w}^T G(x) C_i|^2 u_i(x) dx$ and rearrangement leads to the final energy function to be minimized $F(\boldsymbol{u}, \boldsymbol{c}, \boldsymbol{w}) = \int_{\Omega} \sum_{i=1}^{N} |I(x) - \boldsymbol{w}^T G(x) c_i|^2 u_i(x) dx$. Once the best coefficient for the bias field has been determined, $\widehat{w}$, the bias field can be found with $b\ (x) = \widehat{w} G(x)$ and removed from the acquired images. For the purpose of our studies, the segmentation was removed, as manual segmentation was preferred to allow for defined ROI between tissue types with similar intensity values yet contributed to different biological processes for the small studies that existed within the broader pipeline development process.

The available script from the MATLAB File Exchange built by Li et al. was altered to work with the medical image filetypes used for the diffusion analysis component of our pipeline (NIfTI), with some additional rearrangement of indexing in order to coordinate with the orientation of the data sets (matrix dimensions vs slices). This has become a key component to our pipeline in order to minimize intensity inhomogeneity artifacts and enhance the accuracy of the quantitative diffusion measurements by minimizing the impact of noise that arises from the inherent inhomogeneity of the B_0 and B_1 fields. This is a key asset due to the increasing field inhomogeneity as magnetic field strengths increase, indicating the use of bias field correction as an important step when analyzing data from ultra-high-field instrumentation.[5] This allowed for more accurate registration for comparison between data sets for a series of diffusion studies. Thus, the Multiplicative Intrinsic Component Optimization (MICO) correction is a valuable tool in harnessing the full advantages of ultra-high field strength MRI to generate quantitative data sets.

Appendix B.7 – Post-Processing

```
%% Post-Processing Diffusion Data

% Based upon work done by Li et.al.
% Li C. MRI segmentation and bias field correction.
% MATLAB File Exchange https://www.mathworks.com/matlabcentral/
% fileexchange/59752-mri-segmentation-and-bias-field-correction.
% Published online 2016.

% Alterations made to work with data types and ordering

% Natsortfiles: Stephen (2022). Natural-Order Filename Sort.
% (https://www.mathworks.com/matlabcentral/fileexchange/47434-natural-order-filename-sort),
% MATLAB Central File Exchange.



StartFolder = Main;

load('KeyVariables.mat');



iterNum = 10;
N_region=3;  q=1;

%% Bias Correction
%% Read Images
epi_images = dir([StartFolder 'FA/*.png']);
epi_images = natsortfiles({epi_images.name});
[~, z]     = size(epi_images); % denote matrix size for all images

for i = 1:Slices
    filename = strcat([StartFolder 'FA/'], string(epi_images(i)));
    image = imread(filename); % now read in the image portion
    image = double(uint8(255 * mat2gray(image))); % 255 is the top end of intensity, mat2gray changes it to intensity
based image
    %load ROI
```

```
A=255;
Img_original = image;
[nrow,ncol] = size(image);n = nrow*ncol;

ROI = (image>5); ROI = double(ROI);

tic

Bas=getBasisOrder3(nrow,ncol);
N_bas=size(Bas,3);
for ii=1:N_bas
    imageG{ii} = image.*Bas(:,:,ii).*ROI;
    for jj=ii:N_bas
        GGT{ii,jj} = Bas(:,:,ii).*Bas(:,:,jj).*ROI;
        GGT{jj,ii} = GGT{ii,jj} ;
    end
end


energy_MICO = zeros(3,iterNum);

b=ones(size(image));
for ini_num = 1:1
    C=rand(3,1);
    C=C*A;
    M=rand(nrow,ncol,3);
    a=sum(M,3);
    for k = 1 : N_region
        M(:,:,k)=M(:,:,k)./a;
    end

    [e_max,N_max] = max(M,[], 3);
    for kk=1:size(M,3)
        M(:,:,kk) = (N_max == kk);
    end

    M_old = M; chg=10000;
```

```
        energy_MICO(ini_num,1) = get_energy(image,b,C,M,ROI,q);


        for n = 2:iterNum
            pause(0.1)

            [M, b, C]=  MICO(image,q,ROI,M,C,b,Bas,GGT,imageG,1, 1);
            energy_MICO(ini_num,n) = get_energy(image,b,C,M,ROI,q);

            %figure(2),
            if(mod(n,1) == 0)
                PC=zeros(size(image));
                for k = 1 : N_region
                    PC=PC+C(k)*M(:,:,k);
                end
                        % subplot(241),imshow(uint8(image)),title('original')
                        % subplot(242),imshow(PC.*ROI,[]); colormap(gray);
                        % iterNums=['segmentation: ',num2str(n), ' iterations'];
                        % title(iterNums);
                        % subplot(243),imshow(b.*ROI,[]),title('bias field')
                img_bc = image./b;  % bias field corrected image
                        %subplot(244),imshow(uint8(img_bc.*ROI),[]),title('bias corrected')
                        %subplot(2,4,[5 6 7 8]),plot(energy_MICO(ini_num,:))
                        %xlabel('iteration number');
                        %ylabel('energy');
                pause(0.1)
            end
        end
    end

    [M,C]=sortMemC(M,C);

    bias_corr = img_bc.* ROI;
    X = uint8(bias_corr);
    imwrite(X,sprintf([StartFolder 'FA/BC/' 'FA_BC_%d.png'],i));
end
```

```
net = denoisingNetwork('DnCNN');
Folder_name = ([StartFolder 'FA/BC/']);
All_im = dir([StartFolder 'FA/BC/*.png']);
All_im = natsortfiles({All_im.name});

for j = 1:Slices

    I = imread(string({strcat(Folder_name, All_im(j))}));
    I = imnoise(I,'gaussian',0,0.0005);
    denoisedI = denoiseImage(I, net);
    imwrite(denoisedI, sprintf([StartFolder 'FA/BC/NoiseReduction/FA_BC_NR_%d.png'], j));
end

%% Post-Processing DWI

%% Read Images
epi_images = dir([StartFolder 'DWI/*.png']);
epi_images = natsortfiles({epi_images.name});
[~, z] = size(epi_images); % denote matrix size for all images

for i = 1:Slices
    filename = strcat([StartFolder 'DWI/'], string(epi_images(i)));
    image = imread(filename); % now read in the image portion
    image = double(uint8(255 * mat2gray(image))); % 255 is the top end of intensity, mat2gray changes it to intensity
based image
    %load ROI
    A=255;
    Img_original = image;
    [nrow,ncol] = size(image);n = nrow*ncol;

    ROI = (image>5); ROI = double(ROI);

    tic

    Bas=getBasisOrder3(nrow,ncol);
    N_bas=size(Bas,3);
```

```
for ii=1:N_bas
    imageG{ii} = image.*Bas(:,:,ii).*ROI;
    for jj=ii:N_bas
        GGT{ii,jj} = Bas(:,:,ii).*Bas(:,:,jj).*ROI;
        GGT{jj,ii} = GGT{ii,jj} ;
    end
end


energy_MICO = zeros(3,iterNum);

b=ones(size(image));
for ini_num = 1:1
    C=rand(3,1);
    C=C*A;
    M=rand(nrow,ncol,3);
    a=sum(M,3);
    for k = 1 : N_region
        M(:,:,k)=M(:,:,k)./a;
    end

    [e_max,N_max] = max(M,[], 3);
    for kk=1:size(M,3)
        M(:,:,kk) = (N_max == kk);
    end

    M_old = M; chg=10000;
    energy_MICO(ini_num,1) = get_energy(image,b,C,M,ROI,q);


    for n = 2:iterNum
        pause(0.1)

        [M, b, C]=  MICO(image,q,ROI,M,C,b,Bas,GGT,imageG,1, 1);
        energy_MICO(ini_num,n) = get_energy(image,b,C,M,ROI,q);

        %figure(2),
```

```
        if(mod(n,1) == 0)
            PC=zeros(size(image));
            for k = 1 : N_region
                PC=PC+C(k)*M(:,:,k);
            end
                    % subplot(241),imshow(uint8(image)),title('original')
                    % subplot(242),imshow(PC.*ROI,[]); colormap(gray);
                    % iterNums=['segmentation: ',num2str(n), ' iterations'];
                    % title(iterNums);
                    % subplot(243),imshow(b.*ROI,[]),title('bias field')
            img_bc = image./b;  % bias field corrected image
                    %subplot(244),imshow(uint8(img_bc.*ROI),[]),title('bias corrected')
                    %subplot(2,4,[5 6 7 8]),plot(energy_MICO(ini_num,:))
                    %xlabel('iteration number');
                    %ylabel('energy');
            pause(0.1)
        end
      end
    end

    [M,C]=sortMemC(M,C);

    bias_corr = img_bc.* ROI;
    X = uint8(bias_corr);
    imwrite(X,sprintf([StartFolder 'DWI/BC/' 'DWI_BC_%d.png'],i));
end

net = denoisingNetwork('DnCNN');
Folder_name = ([StartFolder 'DWI/BC/']);
All_im = dir([StartFolder 'DWI/BC/*.png']);
All_im = natsortfiles({All_im.name});

for j = 1:Slices

    I = imread(string({strcat(Folder_name, All_im(j))}));
    I = imnoise(I,'gaussian',0,0.0005);
    denoisedI = denoiseImage(I, net);
```

```
    imwrite(denoisedI, sprintf([StartFolder 'DWI/BC/NoiseReduction/DWI_BC_NR_%d.png'], j));
end

%% Read Images
epi_images = dir([StartFolder 'DTI/*.png']);
epi_images = natsortfiles({epi_images.name});
[~, z] = size(epi_images); % denote matrix size for all images

for i = 1:Slices
    filename = strcat([StartFolder 'DTI/'], string(epi_images(i)));
    image = imread(filename); % now read in the image portion
    [noisyR, noisyG, noisyB] = imsplit(image);
    imageR  = double(uint8(255 * mat2gray(noisyR))); % 255 is the top end of intensity, mat2gray changes it to
intensity based image
    imageGr = double(uint8(255 * mat2gray(noisyG)));
    imageB  = double(uint8(255 * mat2gray(noisyB)));
    %load ROI
    A=255;
    Img_originalR = imageR;
    [nrowR,ncolR] = size(imageR);
    nR = nrowR*ncolR;

    Img_originalG = imageGr;
    [nrowG,ncolG] = size(imageGr);
    nG = nrowG*ncolG;

    Img_originalB = imageB;
    [nrowB,ncolB] = size(imageB);
    nB = nrowB*ncolB;

    ROIR = (imageR>5);
    ROIR = double(ROIR);


    ROIG = (imageGr>5);
    ROIG = double(ROIG);
```

```
ROIB = (imageB>5);
ROIB = double(ROIB);

tic

%% Red Channel

Bas=getBasisOrder3(nrowR,ncolR);
N_bas=size(Bas,3);
for ii=1:N_bas
    imageG{ii} = imageR.*Bas(:,:,ii).*ROIR;
    for jj=ii:N_bas
        GGT{ii,jj} = Bas(:,:,ii).*Bas(:,:,jj).*ROIR;
        GGT{jj,ii} = GGT{ii,jj} ;
    end
end


energy_MICO = zeros(3,iterNum);

b=ones(size(imageR));
for ini_num = 1:1
    C=rand(3,1);
    C=C*A;
    M=rand(nrowR,ncolR,3);
    a=sum(M,3);
    for k = 1 : N_region
        M(:,:,k)=M(:,:,k)./a;
    end

    [e_max,N_max] = max(M,[], 3);
    for kk=1:size(M,3)
        M(:,:,kk) = (N_max == kk);
    end

    M_old = M; chg=10000;
```

```
    energy_MICO(ini_num,1) = get_energy(imageR,b,C,M,ROIR,q);


    for n = 2:iterNum
        pause(0.1)

        [M, b, C]=  MICO(imageR,q,ROIR,M,C,b,Bas,GGT,imageG,1, 1);
        energy_MICO(ini_num,n) = get_energy(imageR,b,C,M,ROIR,q);

        %figure(2),
        if(mod(n,1) == 0)
            PC=zeros(size(imageR));
            for k = 1 : N_region
                PC=PC+C(k)*M(:,:,k);
            end
                    % subplot(241),imshow(uint8(image)),title('original')
                    % subplot(242),imshow(PC.*ROI,[]); colormap(gray);
                    % iterNums=['segmentation: ',num2str(n), ' iterations'];
                    % title(iterNums);
                    % subplot(243),imshow(b.*ROI,[]),title('bias field')
            img_bcR = imageR./b;  % bias field corrected image
                    %subplot(244),imshow(uint8(img_bc.*ROI),[]),title('bias corrected')
                    %subplot(2,4,[5 6 7 8]),plot(energy_MICO(ini_num,:))
                    %xlabel('iteration number');
                    %ylabel('energy');
            pause(0.1)
        end
    end
end

[M,C]=sortMemC(M,C);

bias_corrR = img_bcR.* ROIR;
XR = uint8(bias_corrR);

%% Channel Blue
```

```
Bas=getBasisOrder3(nrowB,ncolB);
N_bas=size(Bas,3);
for ii=1:N_bas
   imageG{ii} = imageB.*Bas(:,:,ii).*ROIB;
   for jj=ii:N_bas
      GGT{ii,jj} = Bas(:,:,ii).*Bas(:,:,jj).*ROIB;
      GGT{jj,ii} = GGT{ii,jj} ;
   end
end


energy_MICO = zeros(3,iterNum);

b=ones(size(imageB));
for ini_num = 1:1
   C=rand(3,1);
   C=C*A;
   M=rand(nrowB,ncolB,3);
   a=sum(M,3);
   for k = 1 : N_region
      M(:,:,k)=M(:,:,k)./a;
   end

   [e_max,N_max] = max(M,[], 3);
   for kk=1:size(M,3)
      M(:,:,kk) = (N_max == kk);
   end

   M_old = M; chg=10000;
   energy_MICO(ini_num,1) = get_energy(imageB,b,C,M,ROIB,q);


   for n = 2:iterNum
      pause(0.1)

      [M, b, C]=  MICO(imageB,q,ROIB,M,C,b,Bas,GGT,imageG,1, 1);
      energy_MICO(ini_num,n) = get_energy(imageB,b,C,M,ROIB,q);
```

```
        %figure(2),
        if(mod(n,1) == 0)
            PC=zeros(size(imageB));
            for k = 1 : N_region
                PC=PC+C(k)*M(:,:,k);
            end
                    % subplot(241),imshow(uint8(image)),title('original')
                    % subplot(242),imshow(PC.*ROI,[]); colormap(gray);
                    %iterNums=['segmentation: ',num2str(n), ' iterations'];
                    % title(iterNums);
                    % subplot(243),imshow(b.*ROI,[]),title('bias field')
            img_bcB = imageB./b;  % bias field corrected image
                    %subplot(244),imshow(uint8(img_bc.*ROI),[]),title('bias corrected')
                    %subplot(2,4,[5 6 7 8]),plot(energy_MICO(ini_num,:))
                    %xlabel('iteration number');
                    %ylabel('energy');
            pause(0.1)
        end
    end
end

[M,C]=sortMemC(M,C);

bias_corrB = img_bcB.* ROIB;
XB = uint8(bias_corrB);

%% Green Channel

Bas=getBasisOrder3(nrowG,ncolG);
N_bas=size(Bas,3);
for ii=1:N_bas
    imageG{ii} = imageGr.*Bas(:,:,ii).*ROIG;
    for jj=ii:N_bas
        GGT{ii,jj} = Bas(:,:,ii).*Bas(:,:,jj).*ROIG;
        GGT{jj,ii} = GGT{ii,jj} ;
    end
```

```
end


energy_MICO = zeros(3,iterNum);

b=ones(size(imageGr));
for ini_num = 1:1
    C=rand(3,1);
    C=C*A;
    M=rand(nrowG,ncolG,3);
    a=sum(M,3);
    for k = 1 : N_region
        M(:,:,k)=M(:,:,k)./a;
    end

    [e_max,N_max] = max(M,[], 3);
    for kk=1:size(M,3)
        M(:,:,kk) = (N_max == kk);
    end

    M_old = M; chg=10000;
    energy_MICO(ini_num,1) = get_energy(imageGr,b,C,M,ROIG,q);


    for n = 2:iterNum
        pause(0.1)

        [M, b, C]=  MICO(imageGr,q,ROIG,M,C,b,Bas,GGT,imageG,1, 1);
        energy_MICO(ini_num,n) = get_energy(imageGr,b,C,M,ROIG,q);

        %figure(2),
        if(mod(n,1) == 0)
            PC=zeros(size(imageGr));
            for k = 1 : N_region
                PC=PC+C(k)*M(:,:,k);
            end
                    % subplot(241),imshow(uint8(image)),title('original')
```

```
                % subplot(242),imshow(PC.*ROI,[]); colormap(gray);
                % iterNums=['segmentation: ',num2str(n), ' iterations'];
                % title(iterNums);
                % subplot(243),imshow(b.*ROI,[]),title('bias field')
            img_bcGr = imageGr./b;  % bias field corrected image
                %subplot(244),imshow(uint8(img_bc.*ROI),[]),title('bias corrected')
                %subplot(2,4,[5 6 7 8]),plot(energy_MICO(ini_num,:))
                %xlabel('iteration number');
                %ylabel('energy');
            pause(0.1)
          end
        end
    end

    [M,C]=sortMemC(M,C);

    bias_corrG = img_bcGr.* ROIG;
    XG = uint8(bias_corrG);

    X = cat(3,XR,XG,XB);

    imwrite(X,sprintf([StartFolder 'DTI/BC/' 'DTIBC_%d.png'],i));
end



%% For Color Images

net = denoisingNetwork('DnCNN');
Folder_name = ([StartFolder 'DTI/BC/']);
All_im = dir([StartFolder 'DTI/BC/*.png']);
All_im = natsortfiles({All_im.name});

for j = 1:Slices

    I = imread(string({strcat(Folder_name, All_im(j))}));
    I = imnoise(I,'gaussian',0,0.0005);
```

```
    [noisyR, noisyG, noisyB] = imsplit(I);
    denoisedR = denoiseImage(noisyR, net);
    denoisedG = denoiseImage(noisyG, net);
    denoisedB = denoiseImage(noisyB, net);
    denoisedI = cat(3, denoisedR, denoisedG, denoisedB);
    imwrite(denoisedI, sprintf([StartFolder 'DTI/BC/NoiseReduction/DTI_BCNR_%d.png'], j));
end

%% MD

epi_images = dir([StartFolder 'MD/*.png']);
epi_images = natsortfiles({epi_images.name});
[~, z] = size(epi_images); % denote matrix size for all images
base_images = []; % create field for base images

for i = 1:Slices
    filename = strcat([StartFolder 'MD/'], string(epi_images(i)));
    image = imread(filename); % now read in the image portion
    image = double(uint8(255 * mat2gray(image))); % 255 is the top end of intensity, mat2gray changes it to intensity
based image
    %load ROI
    A=255;
    Img_original = image;
    [nrow,ncol] = size(image);n = nrow*ncol;

    ROI = (image>5); ROI = double(ROI);

    tic

    Bas=getBasisOrder3(nrow,ncol);
    N_bas=size(Bas,3);
    for ii=1:N_bas
        imageG{ii} = image.*Bas(:,:,ii).*ROI;
        for jj=ii:N_bas
            GGT{ii,jj} = Bas(:,:,ii).*Bas(:,:,jj).*ROI;
            GGT{jj,ii} = GGT{ii,jj} ;
        end
```

```
end


energy_MICO = zeros(3,iterNum);

b=ones(size(image));
for ini_num = 1:1
    C=rand(3,1);
    C=C*A;
    M=rand(nrow,ncol,3);
    a=sum(M,3);
    for k = 1 : N_region
        M(:,:,k)=M(:,:,k)./a;
    end

    [e_max,N_max] = max(M,[], 3);
    for kk=1:size(M,3)
        M(:,:,kk) = (N_max == kk);
    end

    M_old = M; chg=10000;
    energy_MICO(ini_num,1) = get_energy(image,b,C,M,ROI,q);


    for n = 2:iterNum
        pause(0.1)

        [M, b, C]=  MICO(image,q,ROI,M,C,b,Bas,GGT,imageG,1, 1);
        energy_MICO(ini_num,n) = get_energy(image,b,C,M,ROI,q);

        %figure(2),
        if(mod(n,1) == 0)
            PC=zeros(size(image));
            for k = 1 : N_region
                PC=PC+C(k)*M(:,:,k);
            end
                    % subplot(241),imshow(uint8(image)),title('original')
```

```
                % subplot(242),imshow(PC.*ROI,[]); colormap(gray);
                % iterNums=['segmentation: ',num2str(n), ' iterations'];
                % title(iterNums);
                %  subplot(243),imshow(b.*ROI,[]),title('bias field')
            img_bc = image./b;  % bias field corrected image
                %subplot(244),imshow(uint8(img_bc.*ROI),[]),title('bias corrected')
                %subplot(2,4,[5 6 7 8]),plot(energy_MICO(ini_num,:))
                %xlabel('iteration number');
                %ylabel('energy');
            pause(0.1)
          end
        end
    end

    [M,C]=sortMemC(M,C);

    bias_corr = img_bc.* ROI;
    X = uint8(bias_corr);
    imwrite(X,sprintf([StartFolder 'MD/BC/' 'MD_BC_%d.png'],i));
end

net = denoisingNetwork('DnCNN');
Folder_name = ([StartFolder 'MD/BC/']);
All_im = dir([StartFolder 'MD/BC/*.png']);
All_im = natsortfiles({All_im.name});

for j = 1:Slices

    I = imread(string({strcat(Folder_name, All_im(j))}));
    I = imnoise(I,'gaussian',0,0.0005);
    denoisedI = denoiseImage(I, net);
    imwrite(denoisedI, sprintf([StartFolder 'MD/BC/NoiseReduction/MD_BC_NR_%d.png'], j));
end

%% RD

epi_images = dir([StartFolder 'RD/*.png']);
```

```
epi_images = natsortfiles({epi_images.name});
[~, z] = size(epi_images); % denote matrix size for all images

for i = 1:Slices
    filename = strcat([StartFolder 'RD/'], string(epi_images(i)));
    image = imread(filename); % now read in the image portion
    image = double(uint8(255 * mat2gray(image))); % 255 is the top end of intensity, mat2gray changes it to intensity
based image
    %load ROI
    A=255;
    Img_original = image;
    [nrow,ncol] = size(image);n = nrow*ncol;

    ROI = (image>5); ROI = double(ROI);

    tic

    Bas=getBasisOrder3(nrow,ncol);
    N_bas=size(Bas,3);
    for ii=1:N_bas
        imageG{ii} = image.*Bas(:,:,ii).*ROI;
        for jj=ii:N_bas
            GGT{ii,jj} = Bas(:,:,ii).*Bas(:,:,jj).*ROI;
            GGT{jj,ii} = GGT{ii,jj} ;
        end
    end


    energy_MICO = zeros(3,iterNum);

    b=ones(size(image));
    for ini_num = 1:1
        C=rand(3,1);
        C=C*A;
        M=rand(nrow,ncol,3);
        a=sum(M,3);
        for k = 1 : N_region
```

```
    M(:,:,k)=M(:,:,k)./a;
end

[e_max,N_max] = max(M,[], 3);
for kk=1:size(M,3)
    M(:,:,kk) = (N_max == kk);
end

M_old = M; chg=10000;
energy_MICO(ini_num,1) = get_energy(image,b,C,M,ROI,q);


for n = 2:iterNum
    pause(0.1)

    [M, b, C]=  MICO(image,q,ROI,M,C,b,Bas,GGT,imageG,1, 1);
    energy_MICO(ini_num,n) = get_energy(image,b,C,M,ROI,q);

    %figure(2),
    if(mod(n,1) == 0)
        PC=zeros(size(image));
        for k = 1 : N_region
            PC=PC+C(k)*M(:,:,k);
        end
                % subplot(241),imshow(uint8(image)),title('original')
                % subplot(242),imshow(PC.*ROI,[]); colormap(gray);
                % iterNums=['segmentation: ',num2str(n), ' iterations'];
                % title(iterNums);
                % subplot(243),imshow(b.*ROI,[]),title('bias field')
        img_bc = image./b;  % bias field corrected image
                %subplot(244),imshow(uint8(img_bc.*ROI),[]),title('bias corrected')
                %subplot(2,4,[5 6 7 8]),plot(energy_MICO(ini_num,:))
                %xlabel('iteration number');
                %ylabel('energy');
        pause(0.1)
    end
end
```

```
    end

    [M,C]=sortMemC(M,C);

    bias_corr = img_bc.* ROI;
    X = uint8(bias_corr);
    imwrite(X,sprintf([StartFolder 'RD/BC/' 'RD_BC_%d.png'],i));
end

net = denoisingNetwork('DnCNN');
Folder_name = ([StartFolder 'RD/BC/']);
All_im = dir([StartFolder 'RD/BC/*.png']);
All_im = natsortfiles({All_im.name});

for j = 1:Slices

    I = imread(string({strcat(Folder_name, All_im(j))}));
    I = imnoise(I,'gaussian',0,0.0005);
    denoisedI = denoiseImage(I, net);
    imwrite(denoisedI, sprintf([StartFolder 'RD/BC/NoiseReduction/RD_BC_NR_%d.png'], j));
end

%% AD

epi_images = dir([StartFolder 'AD/*.png']);
epi_images = natsortfiles({epi_images.name});
[~, z] = size(epi_images); % denote matrix size for all images

for i = 1:Slices
    filename = strcat([StartFolder 'AD/'], string(epi_images(i)));
    image = imread(filename); % now read in the image portion
    image = double(uint8(255 * mat2gray(image))); % 255 is the top end of intensity, mat2gray changes it to intensity
based image
    %load ROI
    A=255;
    Img_original = image;
    [nrow,ncol] = size(image);n = nrow*ncol;
```

```
ROI = (image>5); ROI = double(ROI);

tic

Bas=getBasisOrder3(nrow,ncol);
N_bas=size(Bas,3);
for ii=1:N_bas
    imageG{ii} = image.*Bas(:,:,ii).*ROI;
    for jj=ii:N_bas
        GGT{ii,jj} = Bas(:,:,ii).*Bas(:,:,jj).*ROI;
        GGT{jj,ii} = GGT{ii,jj} ;
    end
end


energy_MICO = zeros(3,iterNum);

b=ones(size(image));
for ini_num = 1:1
    C=rand(3,1);
    C=C*A;
    M=rand(nrow,ncol,3);
    a=sum(M,3);
    for k = 1 : N_region
        M(:,:,k)=M(:,:,k)./a;
    end

    [e_max,N_max] = max(M,[], 3);
    for kk=1:size(M,3)
        M(:,:,kk) = (N_max == kk);
    end

    M_old = M; chg=10000;
    energy_MICO(ini_num,1) = get_energy(image,b,C,M,ROI,q);
```

```
        for n = 2:iterNum
            pause(0.1)

            [M, b, C]=  MICO(image,q,ROI,M,C,b,Bas,GGT,imageG,1, 1);
            energy_MICO(ini_num,n) = get_energy(image,b,C,M,ROI,q);

            %figure(2),
            if(mod(n,1) == 0)
                PC=zeros(size(image));
                for k = 1 : N_region
                    PC=PC+C(k)*M(:,:,k);
                end
                        % subplot(241),imshow(uint8(image)),title('original')
                        % subplot(242),imshow(PC.*ROI,[]); colormap(gray);
                        % iterNums=['segmentation: ',num2str(n), ' iterations'];
                        % title(iterNums);
                        % subplot(243),imshow(b.*ROI,[]),title('bias field')
                img_bc = image./b;  % bias field corrected image
                        %subplot(244),imshow(uint8(img_bc.*ROI),[]),title('bias corrected')
                        %subplot(2,4,[5 6 7 8]),plot(energy_MICO(ini_num,:))
                        %xlabel('iteration number');
                        %ylabel('energy');
                pause(0.1)
            end
        end
    end

    [M,C]=sortMemC(M,C);

    bias_corr = img_bc.* ROI;
    X = uint8(bias_corr);
    imwrite(X,sprintf([StartFolder 'AD/BC/' 'AD_BC_%d.png'],i));
end

net = denoisingNetwork('DnCNN');
Folder_name = ([StartFolder 'AD/BC/']);
All_im = dir([StartFolder 'AD/BC/*.png']);
```

```
All_im = natsortfiles({All_im.name});

for j = 1:Slices

    I = imread(string({strcat(Folder_name, All_im(j))}));
    I = imnoise(I,'gaussian',0,0.0005);
    denoisedI = denoiseImage(I, net);
    imwrite(denoisedI, sprintf([StartFolder 'AD/BC/NoiseReduction/AD_BC_NR_%d.png'], j));
end

beep on
beep
```

Appendix C.1 – SRR with Functions (3:1)

```
%% SRR for Low Resolution 3:1 High Resolution

% Natsortfiles: Stephen (2022). Natural-Order Filename Sort.
% (https://www.mathworks.com/matlabcentral/fileexchange/47434-natural-order-filename-sort),
% MATLAB Central File Exchange.


Output = '% Output file';

LR1 = '% Low resolution data set 1';
LR2 = '% Low resolution data set 2';
LR3 = '% Low resolution data set 3';

LR_1 = dir([LR1 '*.png']);
LR_1 = natsortfiles({LR_1.name});
[~, a] = size(LR_1);
Test_Image = imread(string(strcat(LR1, LR_1(1))));
Image_Size = size(Test_Image);

LR_2 = dir([LR2 '*.png']);
LR_2 = natsortfiles({LR_2.name});

LR_3 = dir([LR3 '*.png']);
LR_3 = natsortfiles({LR_3.name});

% Create 3D array of images
LR_one = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
   image = im2double(imread(string(strcat(LR1, LR_1(i)))));
   LR_one(:,:,i) = image;
end

LR_two = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
   image = im2double(imread(string(strcat(LR2, LR_2(i)))));
```

```
    LR_two(:,:,i) = image;
end

LR_three = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR3, LR_3(i)))));
    LR_three(:,:,i) = image;
end

% Create 4D array of all low resolution images to translate through
gk(:,:,:,1) = LR_one;
gk(:,:,:,2) = LR_two;
gk(:,:,:,3) = LR_three;



% Blur function is a blurring filter via MATLAB
psf = fspecial('gaussian', 3, 0.3);

K   = 3; % Number of low resolution Inputs

% Back projection kernel == inverse of psf
% No true inverse to matrix
% Take pseudoinverse or put in kernel directly
bp = pinv(psf);

% Offset
off = 3;

% Initialize some random error amount
err = ones(100, K);

% Offset for low to high resolution count
count = 1;

for slice = 1:a-1

    for offset = 1:off
```

```
        upsampled =  upsample(gk,slice, offset, K);
        sum_us = sum(upsampled, 3);
        f = (1/K) .* sum_us;
        gk_guess = zeros(Image_Size(1), Image_Size(2), K);
        tol(100,K) = 1;
        p = 1;
        while sum(tol(p,:)) > 0.00001
            gk_guess = lowresguess(f,psf, K);
            gkdiff = diffg(gk, gk_guess, K, slice, offset);
            f = f + (1/K) * conv2(gkdiff, bp, 'same');
            gk_error = err_GK(gk, gk_guess, K, p, slice, offset);
            err(p,:) = gk_error(p,:);
            if p == 1
                tol(p,:) = err(p,:);
            else
                tol(p,:) = (err(p-1, :) - err(p, :));
            end
            p = p + 1;
        end
        imwrite(f, sprintf([Output 'Recon_%d.png'], count));
        count = count + 1;



    end
end
```

Appendix C.2 – Upsample Function

```
function upsampled = upsample(gk, slice, offset, K)


if offset == 1
   for i = 1:K
      upsampled(:,:,i) = gk(:,:,slice, i);
   end
elseif offset == 2
   for i = 1:K
      if i == 1
         upsampled(:,:,i) = gk(:,:,slice+1, i);
      else
         upsampled(:,:,i) = gk(:,:,slice, i);
      end
   end
elseif offset == 3
   for i = 1:K
      if (i == 1) || (i == 2)
         upsampled(:,:,i) = gk(:,:,slice+1, i);
      else
         upsampled(:,:,i) = gk(:,:,slice, i);


      end
   end
elseif offset == 4
   for i = 1:K
      if (i == 1) || (i == 2) || (i == 3)
         upsampled(:,:,i) = gk(:,:,slice+1, i);
      else
         upsampled(:,:,i) = gk(:,:,slice, i);


      end
   end
elseif offset == 5
   for i = 1:K
      if (i == 1) || (i == 2)|| (i == 3) || (i == 4)
```

```
            upsampled(:,:,i) = gk(:,:,slice+1, i);
        else
            upsampled(:,:,i) = gk(:,:,slice, i);

        end
    end
elseif offset == 6
    for i = 1:K
        if (i == 1) || (i == 2)|| (i == 3) || (i == 4) || (i == 5)
            upsampled(:,:,i) = gk(:,:,slice+1, i);
        else
            upsampled(:,:,i) = gk(:,:,slice, i);

        end
    end
end
```

Appendix C.3 – Low Resolution Estimation Function

```
function gk_guess = lowresguess(f,psf,K)
    for i = 1:K
        gk_guess(:,:,i) = conv2(f,psf,'same');
    end
end
```

Appendix C.4 – Difference Between Low-Res Estimate and True Function

```
function gkdiff = diffg(gk, gk_guess, K, slice, offset)
if offset == 1
   for j = 1:K
      gkdiff(:,:,j) = (gk(:,:,slice,j) - gk_guess(:,:,j));
   end
elseif offset == 2
   for j = 1:K
      if j ==1
         gkdiff(:,:,j) = (gk(:,:,slice+1,j) - gk_guess(:,:,j));
      else
         gkdiff(:,:,j) = (gk(:,:,slice,j) - gk_guess(:,:,j));
      end
   end
elseif offset == 3
   for j = 1:K
      if (j == 1) || (j == 2)
         gkdiff(:,:,j) = (gk(:,:,slice+1,j) - gk_guess(:,:,j));
      else
         gkdiff(:,:,j) = (gk(:,:,slice,j) - gk_guess(:,:,j));
      end
   end
elseif offset == 4
   for j = 1:K
      if (j == 1) || (j == 2) || (j ==3)
         gkdiff(:,:,j) = (gk(:,:,slice+1,j) - gk_guess(:,:,j));
      else
         gkdiff(:,:,j) = (gk(:,:,slice,j) - gk_guess(:,:,j));
      end
   end
elseif offset == 5
   for j = 1:K
      if (j == 1) || (j == 2)||(j == 3) || (j == 4)
```

```
            gkdiff(:,:,j) = (gk(:,:,slice+1,j) - gk_guess(:,:,j));
        else
            gkdiff(:,:,j) = (gk(:,:,slice,j) - gk_guess(:,:,j));
        end
    end
elseif offset == 6
    for j = 1:K
        if (j == 1) || (j == 2)|| (j == 3) || (j == 4) || (j == 5)
            gkdiff(:,:,j) = (gk(:,:,slice+1,j) - gk_guess(:,:,j));
        else
            gkdiff(:,:,j) = (gk(:,:,slice,j) - gk_guess(:,:,j));
        end
    end
end
gkdiff = sum(gkdiff, 3);
```

Appendix C.5 – Error Between Low-Res Estimate and True

```
function gk_error = err_GK(gk, gk_guess, K, p, slice, offset)
if offset == 1
   for j = 1:K
      gk_error(p, j) = immse(gk(:,:,slice, j), gk_guess(:,:,j));
   end
elseif offset == 2
   for j = 1:K
      if j == 1
         gk_error(p, j) = immse(gk(:,:,slice+1, j), gk_guess(:,:,j));
      else
         gk_error(p, j) = immse(gk(:,:,slice, j), gk_guess(:,:,j));
      end
   end
elseif offset ==3
   for j = 1:K
      if (j ==1) || (j ==2)
         gk_error(p, j) = immse(gk(:,:,slice+1, j), gk_guess(:,:,j));
      else
         gk_error(p, j) = immse(gk(:,:,slice, j), gk_guess(:,:,j));
      end
   end
elseif offset == 4
   for j = 1:K
      if (j ==1) || (j ==2) || (j == 3)
         gk_error(p, j) = immse(gk(:,:,slice+1, j), gk_guess(:,:,j));
      else
         gk_error(p, j) = immse(gk(:,:,slice, j), gk_guess(:,:,j));
      end
   end
elseif offset == 5
   for j = 1:K
      if (j ==1) || (j ==2) || (j == 3) || (j == 4)
         gk_error(p, j) = immse(gk(:,:,slice+1, j), gk_guess(:,:,j));
      else
         gk_error(p, j) = immse(gk(:,:,slice, j), gk_guess(:,:,j));
```

```
        end
    end
elseif offset == 6
    for j = 1:K
        if (j ==1) || (j ==2) || (j == 3) || (j ==4) || (j == 5)
            gk_error(p, j) = immse(gk(:,:,slice+1, j), gk_guess(:,:,j));
        else
            gk_error(p, j) = immse(gk(:,:,slice, j), gk_guess(:,:,j));
        end
    end
end
```

Appendix C.6 – Alternate Approach for SRR (3:1)

```
%% SRR for 3 input images

% Natsortfiles: Stephen (2022). Natural-Order Filename Sort.
% (https://www.mathworks.com/matlabcentral/fileexchange/47434-natural-order-filename-sort),
% MATLAB Central File Exchange.

% Output folder
Output = '% Output Folder';

% Load low resolution images
LR1 = '% Low resolution data set 1';
LR2 = '% Low resolution data set 2';
LR3 = '% Low resolution data set 3';

% Sort files
LR_1 = dir([LR1 '*.png']);
LR_1 = natsortfiles({LR_1.name});
[~, a] = size(LR_1);



LR_2 = dir([LR2 '*.png']);
LR_2 = natsortfiles({LR_2.name});

LR_3 = dir([LR3 '*.png']);
LR_3 = natsortfiles({LR_3.name});

% Input psf estimation
psf = fspecial('gaussian', 3, 0.3);

% Number of low resolution images
K = 3;

% Number of offsets
off = 3;
```

```
% Back projection kernel == inverse of psf
% No true inverse to matrix
% Take pseudoinverse or put in kernel directly
bp = pinv(psf);

% Count to offset low and high resolution images
count = 1;

for i = 1:(a-1)

   for offset = 1:off

      if offset == 1

         % Load low resolution images
         filename_1 = strcat(LR1, LR_1(i+1));
         filename_1 = string({filename_1});
         image_1 = imread(filename_1);
         image_1 = im2double(image_1);

         filename_2 = strcat(LR2, LR_2(i));
         filename_2 = string({filename_2});
         image_2 = imread(filename_2);
         image_2 = im2double(image_2);

         filename_3 = strcat(LR3, LR_3(i));
         filename_3 = string({filename_3});
         image_3 = imread(filename_3);
         image_3 = im2double(image_3);

         % Temporary variables of low resolution images
         g(:,:,1) = image_1;
         g(:,:,2) = image_2;
         g(:,:,3) = image_3;

         % Initial error estimation
         err = ones([1 3]);
```

```
% Initial guess of high resolution image
f = sum(g,3);

% Estimate low resolution images
g_k = conv2(f, psf, 'same');

% Starting variable for while loop
c = 1;
while sum(err(:)) > 0.003
   if c < 500
      for k = 1:3

         % Determine differences between estimated and true
         % low resolution images
         diff = g(:,:,k) - g_k;
         % Update high resolution estimate
         f = f + (1/K) * conv2(diff, bp, 'same');
         % Estimate low resolution images
         g_k = conv2(f,psf, 'same');

         % Determine error between estimated and true low
         % resolution images
         err(k) = immse(g(:,:,k), g_k);
      end

      c = c + 1;
   else
      break
   end
end

% Write high resolution estimate
imwrite(f, sprintf([Output 'Recon_%d.png'], count));
clear err

% Update count
```

```
    count = count + 1;

elseif offset == 2

    filename_1 = strcat(LR1, LR_1(i + 1));
    filename_1 = string({filename_1});
    image_1 = imread(filename_1);
    image_1 = im2double(image_1);

    filename_2 = strcat(LR2, LR_2(i+1));
    filename_2 = string({filename_2});
    image_2 = imread(filename_2);
    image_2 = im2double(image_2);

    filename_3 = strcat(LR3, LR_3(i));
    filename_3 = string({filename_3});
    image_3 = imread(filename_3);
    image_3 = im2double(image_3);

    err = ones([1 3]);

    g(:,:,1) = image_1;
    g(:,:,2) = image_2;
    g(:,:,3) = image_3;

    f = sum(g,3);

    g_k = conv2(f, psf, 'same');

    c = 1;

    while sum(err(:)) > 0.03
        if c < 500;
            for k = 1:3
                diff = g(:,:,k) - g_k;
                f = f + (1/K) * conv2(diff, bp, 'same');
                g_k = conv2(f, psf, 'same');
```

```
            err(k) = immse(g(:,:,k), g_k);
        end
        c = c + 1;
    else
        break
    end
end

imwrite(f, sprintf([Output 'Recon_%d.png'], count));
clear err

count = count + 1;

elseif offset == 3


filename_1 = strcat(LR1, LR_1(i + 1));
filename_1 = string({filename_1});
image_1 = imread(filename_1);
image_1 = im2double(image_1);

filename_2 = strcat(LR2, LR_2(i+1));
filename_2 = string({filename_2});
image_2 = imread(filename_2);
image_2 = im2double(image_2);

filename_3 = strcat(LR3, LR_3(i+1));
filename_3 = string({filename_3});
image_3 = imread(filename_3);
image_3 = im2double(image_3);

err = ones([1 3]);

g(:,:,1) = image_1;
g(:,:,2) = image_2;
g(:,:,3) = image_3;
```

```
            f = sum(g,3);

            g_k = conv2(f, psf, 'same');

            c = 1;
            while sum(err(:)) > 0.03
               if c < 500
                  for k = 1:3
                     diff = g(:,:,k) - g_k;
                     f = f + (1/K) * conv2(diff, bp, 'same');
                     g_k = conv2(f, psf, 'same');
                     err(k) = immse(g(:,:,k), g_k);
                  end
                  c = c + 1;
               else
                  break
               end

            end

            imwrite(f, sprintf([Output 'Recon_%d.png'], count));
            clear err

            count = count + 1;
         end
      end
   end
end
```

Appendix C.7 – SRR 12 Low-Res Images (3:1)

```
% Super Resolution Algorithm using back propagation

% Natsortfiles: Stephen (2022). Natural-Order Filename Sort.
% (https://www.mathworks.com/matlabcentral/fileexchange/47434-natural-order-filename-sort),
% MATLAB Central File Exchange.

% Load low resolution images

LR1  = '% Low resolution data set 1';
LR2  = '% Low resolution data set 2';
LR3  = '% Low resolution data set 3';
LR4  = '% Low resolution data set 4';
LR5  = '% Low resolution data set 5';
LR6  = '% Low resolution data set 6';
LR7  = '% Low resolution data set 7';
LR8  = '% Low resolution data set 8';
LR9  = '% Low resolution data set 9';
LR10 = '% Low resolution data set 10';
LR11 = '% Low resolution data set 11';
LR12 = '% Low resolution data set 12';

% Designate output location
Output = '% Output file location';



LR_1 = dir([LR1 '*.png']);
LR_1 = natsortfiles({LR_1.name});
[~, a] = size(LR_1);

LR_2 = dir([LR2 '*.png']);
LR_2 = natsortfiles({LR_2.name});

LR_3 = dir([LR3 '*.png']);
LR_3 = natsortfiles({LR_3.name});
```

```
LR_4 = dir([LR4 '*.png']);
LR_4 = natsortfiles({LR_4.name});

LR_5 = dir([LR5 '*.png']);
LR_5 = natsortfiles({LR_5.name});

LR_6 = dir([LR6 '*.png']);
LR_6 = natsortfiles({LR_6.name});

LR_7 = dir([LR7 '*.png']);
LR_7 = natsortfiles({LR_7.name});

LR_8 = dir([LR8 '*.png']);
LR_8 = natsortfiles({LR_8.name});

LR_9 = dir([LR9 '*.png']);
LR_9 = natsortfiles({LR_9.name});

LR_10 = dir([LR10 '*.png']);
LR_10 = natsortfiles({LR_10.name});

LR_11 = dir([LR11 '*.png']);
LR_11= natsortfiles({LR_11.name});

LR_12 = dir([LR12 '*.png']);
LR_12 = natsortfiles({LR_12.name});

% Number of low resolution images
K = 12;

% Offset number
off = 3;

% Bluring
psf = fspecial('gaussian', 3, 0.3);

% Back projection kernel == inverse of psf
```

```
% No true inverse to matrix
% Take pseudoinverse or put in kernel directly
bp = pinv(psf);



count = 1;

for i = 1:(a-1)

   for offset = 1:off

      if offset == 1

         filename_1a = strcat(LR1, LR_1(i));
         filename_1a = string({filename_1a});
         image_1a = imread(filename_1a);
         image_1a = mat2gray(image_1a);

         filename_1b = strcat(LR1, LR_1(i+1));
         filename_1b = string({filename_1b});
         image_1b = imread(filename_1b);
         image_1b = mat2gray(image_1b);

         filename_2a = strcat(LR2, LR_2(i));
         filename_2a = string({filename_2a});
         image_2a = imread(filename_2a);
         image_2a = mat2gray(image_2a);

         filename_2b = strcat(LR2, LR_2(i+1));
         filename_2b = string({filename_2b});
         image_2b = imread(filename_2b);
         image_2b = mat2gray(image_2b);

         filename_3a = strcat(LR3, LR_3(i));
         filename_3a = string({filename_3a});
         image_3a = imread(filename_3a);
```

```
image_3a = mat2gray(image_3a);

filename_3b = strcat(LR3, LR_3(i+1));
filename_3b = string({filename_3b});
image_3b = imread(filename_3b);
image_3b = mat2gray(image_3b);

filename_4 = strcat(LR4, LR_4(i));
filename_4 = string({filename_4});
image_4 = imread(filename_4);
image_4 = mat2gray(image_4);

filename_5 = strcat(LR5, LR_5(i));
filename_5 = string({filename_5});
image_5 = imread(filename_5);
image_5 = mat2gray(image_5);

filename_6 = strcat(LR6, LR_6(i));
filename_6 = string({filename_6});
image_6 = imread(filename_6);
image_6 = mat2gray(image_6);

filename_7 = strcat(LR7, LR_7(i));
filename_7 = string({filename_7});
image_7 = imread(filename_7);
image_7 = mat2gray(image_7);

filename_8 = strcat(LR8, LR_8(i));
filename_8 = string({filename_8});
image_8 = imread(filename_8);
image_8 = mat2gray(image_8);

filename_9 = strcat(LR9, LR_9(i));
filename_9 = string({filename_9});
image_9 = imread(filename_9);
image_9 = mat2gray(image_9);
```

```
filename_10 = strcat(LR10, LR_10(i));
filename_10 = string({filename_10});
image_10 = imread(filename_10);
image_10 = mat2gray(image_10);

filename_11 = strcat(LR11, LR_11(i));
filename_11 = string({filename_11});
image_11 = imread(filename_11);
image_11 = mat2gray(image_11);

filename_12 = strcat(LR12, LR_12(i));
filename_12 = string({filename_12});
image_12 = imread(filename_12);
image_12 = mat2gray(image_12);

err = ones([1 12]);
psf = fspecial('gaussian');


g(:,:,1) = (image_1a + image_1b)./2;
g(:,:,2) = (image_2a + image_2b) ./ 2;
g(:,:,3) = (image_3a + image_3b)./ 2;
g(:,:,4) = image_4;
g(:,:,5) = image_5;
g(:,:,6) = image_6;
g(:,:,7) = image_7;
g(:,:,8) = image_8;
g(:,:,9) = image_9;
g(:,:,10) = image_10;
g(:,:,11)  = image_11;
g(:,:,12) = image_12;

f = sum(g, 3);

g_k = conv2(f, psf, 'same');

c = 1;
```

```
    while (sum(err(:)) >= 0.015)
        if c < 500
        for k = 1:K
            diff = g(:,:,k) - g_k;
            f = f + (1/K) * conv2(diff, bp, 'same');

            g_k = conv2(f, psf, 'same');

            err(k) = immse(g(:,:,k), g_k);

        end
        c = c+1;

        else
            break
        end

    end

    imwrite(f, sprintf([Output 'ReconSlice_%d.png'], count));
    clear err

    count = count + 1;

elseif offset == 2

     filename_1= strcat(LR1, LR_1(i+1));
    filename_1 = string({filename_1});
    image_1 = imread(filename_1);
    image_1 = mat2gray(image_1);

    filename_2 = strcat(LR2, LR_2(i+1));
    filename_2 = string({filename_2});
    image_2 = imread(filename_2);
    image_2 = mat2gray(image_2);
```

```
filename_3 = strcat(LR3, LR_3(i+1));
filename_3 = string({filename_3});
image_3 = imread(filename_3);
image_3 = mat2gray(image_3);

filename_4 = strcat(LR4, LR_4(i+1));
filename_4 = string({filename_4});
image_4 = imread(filename_4);
image_4 = mat2gray(image_4);

filename_5a = strcat(LR5, LR_5(i));
filename_5a = string({filename_5a});
image_5a = imread(filename_5a);
image_5a = mat2gray(image_5a);

filename_5b = strcat(LR5, LR_5(i+1));
filename_5b = string({filename_5b});
image_5b = imread(filename_5b);
image_5b = mat2gray(image_5b);

filename_6a = strcat(LR6, LR_6(i));
filename_6a = string({filename_6a});
image_6a = imread(filename_6a);
image_6a = mat2gray(image_6a);

filename_6b = strcat(LR6, LR_6(i+1));
filename_6b = string({filename_6b});
image_6b = imread(filename_6b);
image_6b = mat2gray(image_6b);

filename_7a = strcat(LR7, LR_7(i));
filename_7a = string({filename_7a});
image_7a = imread(filename_7a);
image_7a = mat2gray(image_7a);

filename_7b = strcat(LR7, LR_7(i+1));
```

```
filename_7b = string({filename_7b});
image_7b = imread(filename_7b);
image_7b = mat2gray(image_7b);

filename_8 = strcat(LR8, LR_8(i));
filename_8 = string({filename_8});
image_8 = imread(filename_8);
image_8 = mat2gray(image_8);

filename_9 = strcat(LR9, LR_9(i));
filename_9 = string({filename_9});
image_9 = imread(filename_9);
image_9 = mat2gray(image_9);

filename_10 = strcat(LR10, LR_10(i));
filename_10 = string({filename_10});
image_10 = imread(filename_10);
image_10 = mat2gray(image_10);

filename_11 = strcat(LR11, LR_11(i));
filename_11 = string({filename_11});
image_11 = imread(filename_11);
image_11 = mat2gray(image_11);

filename_12 = strcat(LR12, LR_12(i));
filename_12 = string({filename_12});
image_12 = imread(filename_12);
image_12 = mat2gray(image_12);

err = ones([1 12]);


g(:,:,1) = image_1;
g(:,:,2) = image_2;
g(:,:,3) = image_3;
g(:,:,4) = image_4;
g(:,:,5) = (image_5a + image_5b)./2;
```

```
    g(:,:,6) = (image_6a + image_6b)./ 2;
    g(:,:,7) = (image_7a + image_7b)./ 2;
    g(:,:,8) = image_8;
    g(:,:,9) = image_9;
    g(:,:,10) = image_10;
    g(:,:,11) = image_11;
    g(:,:,12) = image_12;

    f = sum(g,3);
    g_k = conv2(f, psf, 'same');

    c = 1;
    while (sum(err(:)) >= 0.015)

        if c < 500
        for k = 1:K
            diff = g(:,:,k) - g_k;
            f = f + (1/K) * conv2(diff, bp, 'same');

            g_k = conv2(f, psf, 'same');

            err(k) = immse(g(:,:,k), g_k);

        end
        c = c + 1;
        else
            break
        end

    end

    imwrite(f, sprintf([Output 'ReconSlice_%d.png'], count));
    clear err

    count = count + 1;

elseif offset == 3
```

```
filename_1= strcat(LR1, LR_1(i+1));
filename_1 = string({filename_1});
image_1 = imread(filename_1);
image_1 = mat2gray(image_1);

filename_2 = strcat(LR2, LR_2(i+1));
filename_2 = string({filename_2});
image_2 = imread(filename_2);
image_2 = mat2gray(image_2);


filename_3 = strcat(LR3, LR_3(i+1));
filename_3 = string({filename_3});
image_3 = imread(filename_3);
image_3 = mat2gray(image_3);

filename_4 = strcat(LR4, LR_4(i+1));
filename_4 = string({filename_4});
image_4 = imread(filename_4);
image_4 = mat2gray(image_4);

filename_5 = strcat(LR5, LR_5(i+1));
filename_5 = string({filename_5});
image_5 = imread(filename_5);
image_5 = mat2gray(image_5);

filename_6 = strcat(LR6, LR_6(i + 1));
filename_6 = string({filename_6});
image_6 = imread(filename_6);
image_6 = mat2gray(image_6);

filename_7 = strcat(LR7, LR_7(i+1));
filename_7 = string({filename_7});
image_7 = imread(filename_7);
image_7 = mat2gray(image_7);
```

```
filename_8 = strcat(LR8, LR_8(i+1));
filename_8 = string({filename_8});
image_8 = imread(filename_8);
image_8 = mat2gray(image_8);

filename_9a = strcat(LR9, LR_9(i));
filename_9a = string({filename_9a});
image_9a = imread(filename_9a);
image_9a = mat2gray(image_9a);

filename_9b = strcat(LR9, LR_9(i+1));
filename_9b = string({filename_9b});
image_9b = imread(filename_9b);
image_9b = mat2gray(image_9b);

filename_10a = strcat(LR10, LR_10(i));
filename_10a = string({filename_10a});
image_10a = imread(filename_10a);
image_10a = mat2gray(image_10a);

filename_10b = strcat(LR10, LR_10(i+1));
filename_10b = string({filename_10b});
image_10b = imread(filename_10b);
image_10b = mat2gray(image_10b);

filename_11a = strcat(LR11, LR_11(i));
filename_11a = string({filename_11a});
image_11a = imread(filename_11a);
image_11a = mat2gray(image_11a);

filename_11b = strcat(LR11, LR_11(i+1));
filename_11b = string({filename_11b});
image_11b = imread(filename_11b);
image_11b = mat2gray(image_11b);

filename_12 = strcat(LR12, LR_12(i));
filename_12 = string({filename_12});
```

```
image_12 = imread(filename_12);
image_12 = mat2gray(image_12);



err = ones([1 12]);

g(:,:,1) = image_1;
g(:,:,2) = image_2;
g(:,:,3) = image_3;
g(:,:,4) = image_4;
g(:,:,5) = image_5;
g(:,:,6) = image_6;
g(:,:,7) = image_7;
g(:,:,8) = image_8;
g(:,:,9) = (image_9a + image_9b)./2;
g(:,:,10) = (image_10a + image_10b) ./2;
g(:,:,11) = (image_11a + image_11b) ./2 ;
g(:,:,12) = image_12 ;

f = sum(g,3);
g_k = conv2(f, psf, 'same');

c = 1;
while (err(:) >= 0.002)
   if c < 500
   for k = 1:K
      diff = g(:,:,k) - g_k;
      f = f + (1/K) * conv2(diff, bp, 'same');

      g_k = conv2(f, psf, 'same');

      err(k) = immse(g(:,:,k), g_k);

   end
   c + c+1;
   else
```

```
            break
        end

    end

    imwrite(f, sprintf([Output 'ReconSlice_%d.png'], count));
    clear err

    count = count + 1;
  end
 end
end
```

Appendix C.8 – SRR 12 Low-Res Images (3:1) with Functions

```
%% SRR with 12 Low Resolution Inputs
% 3 to 1 Low Res to High Res

% Natsortfiles: Stephen (2022). Natural-Order Filename Sort.
% (https://www.mathworks.com/matlabcentral/fileexchange/47434-natural-order-filename-sort),
% MATLAB Central File Exchange.

Output = '% Ouput Folder';

LR1  = '% Low resolution data set 1';
LR2  = '% Low resolution data set 2';
LR3  = '% Low resolution data set 3';
LR4  = '% Low resolution data set 4';
LR5  = '% Low resolution data set 5';
LR6  = '% Low resolution data set 6';
LR7  = '% Low resolution data set 7';
LR8  = '% Low resolution data set 8';
LR9  = '% Low resolution data set 9';
LR10 = '% Low resolution data set 10';
LR11 = '% Low resolution data set 11';
LR12 = '% Low resolution data set 12';

LR_1 = dir([LR1 '*.png']);
LR_1 = natsortfiles({LR_1.name});
[~, a] = size(LR_1);
Test_Image = imread(string(strcat(LR1, LR_1(1))));
Image_Size = size(Test_Image);

LR_2 = dir([LR2 '*.png']);
LR_2 = natsortfiles({LR_2.name});

LR_3 = dir([LR3 '*.png']);
LR_3 = natsortfiles({LR_3.name});

LR_4 = dir([LR4 '*.png']);
```

```
LR_4 = natsortfiles({LR_4.name});

LR_5 = dir([LR5 '*.png']);
LR_5 = natsortfiles({LR_5.name});

LR_6 = dir([LR6 '*.png']);
LR_6 = natsortfiles({LR_6.name});

LR_7 = dir([LR7 '*.png']);
LR_7 = natsortfiles({LR_7.name});

LR_8 = dir([LR8 '*.png']);
LR_8 = natsortfiles({LR_8.name});

LR_9 = dir([LR9 '*.png']);
LR_9 = natsortfiles({LR_9.name});

LR_10 = dir([LR10 '*.png']);
LR_10 = natsortfiles({LR_10.name});

LR_11 = dir([LR11 '*.png']);
LR_11= natsortfiles({LR_11.name});

LR_12 = dir([LR12 '*.png']);
LR_12 = natsortfiles({LR_12.name});


% Create 3D array of images
LR_one = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR1, LR_1(i)))));
    LR_one(:,:,i) = image;
end

LR_two = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR2, LR_2(i)))));
```

```
    LR_two(:,:,i) = image;
end

LR_three = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR3, LR_3(i)))));
    LR_three(:,:,i) = image;
end

LR_four = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR4, LR_4(i)))));
    LR_four(:,:,i) = image;
end

LR_five = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR5, LR_5(i)))));
    LR_five(:,:,i) = image;
end

LR_six = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR6, LR_6(i)))));
    LR_six(:,:,i) = image;
end

LR_seven = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR7, LR_7(i)))));
    LR_seven(:,:,i) = image;
end

LR_eight = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR8, LR_8(i)))));
    LR_eight(:,:,i) = image;
```

```
end


LR_nine = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR9, LR_9(i)))));
    LR_nine(:,:,i) = image;
end

LR_ten = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR10, LR_10(i)))));
    LR_ten(:,:,i) = image;
end

LR_eleven = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR11, LR_11(i)))));
    LR_eleven(:,:,i) = image;
end

LR_twelve = zeros(Image_Size(1), Image_Size(2), a);
for i = 1:a
    image = im2double(imread(string(strcat(LR12, LR_12(i)))));
    LR_twelve(:,:,i) = image;
end

% Create 4D array of all low resolution images to translate through
gk(:,:,:,1)  = LR_one;
gk(:,:,:,2)  = LR_two;
gk(:,:,:,3)  = LR_three;
gk(:,:,:,4)  = LR_four;
gk(:,:,:,5)  = LR_five;
gk(:,:,:,6)  = LR_six;
gk(:,:,:,7)  = LR_seven;
gk(:,:,:,8)  = LR_eight;
gk(:,:,:,9)  = LR_nine;
```

```
gk(:,:,:,10) = LR_ten;
gk(:,:,:,11) = LR_eleven;
gk(:,:,:,12) = LR_twelve;

% Blur function is a blurring filter via MATLAB
psf = fspecial('gaussian', 3, 0.3);

K   = 12; % Number of low resolution Inputs


off = 3; % LR to HR images

% Back projection kernel == inverse of psf
% No true inverse to matrix
% Take pseudoinverse or put in kernel directly
bp = pinv(psf);

% Initialize some random error amount
err = ones(100, K);

% Offset for low to high resolution count
count = 1;

for slice = 1:a-1

   for offset = 1:off


       upsampled =  upsample_12im(gk, slice, offset, K);
       sum_us = sum(upsampled, 3);
       f = (1/K) .* sum_us;
       gk_guess = zeros(Image_Size(1), Image_Size(2), K);
       tol(100,K) = 1;
       p = 1;
       while sum(tol(p,:)) > 0.00001
          gk_guess = lowresguess(f,psf, K);
          gkdiff = diffg12im(gk, gk_guess, K, slice, offset);
```

```
        f = f + (1/K) * conv2(gkdiff, bp, 'same');
        gk_error = err_GK12im(gk, gk_guess, K, p, slice, offset);
        err(p,:) = gk_error(p,:);
        if p == 1
            tol(p,:) = err(p,:);
        else
            tol(p,:) = (err(p-1, :) - err(p, :))
        end
        p = p + 1;
    end
    imwrite(f, sprintf([Output 'Recon_%d.png'], count));
    count = count + 1;


  end
end
```

Appendix C.9 – Up-sample Function for 12 Images

```
function upsampled = upsample_12im(gk, slice, offset, K)


if offset == 1
   for i = 1:K
      if (i == 1) || (i == 2) || (i == 3)
         upsampled(:,:,i) = (gk(:,:,slice, i) + gk(:,:,slice+1, i))./2;
      else
         upsampled(:,:,i) = gk(:,:,slice, i);
      end
   end
elseif offset == 2
   for i = 1:K
      if (i == 1) || (i == 2) || (i == 3) || (i == 4)
         upsampled(:,:,i) = gk(:,:,slice+1, i);
      elseif (i == 5) || (i == 6) || (i == 7)
         upsampled(:,:,i) = (gk(:,:,slice, i) + gk(:,:,slice+1, i))./2;
      else
         upsampled(:,:,i) = gk(:,:,slice, i);
      end
   end
elseif offset == 3
   for i = 1:K
      if (i == 1) || (i == 2) || (i == 3) || (i == 4) || (i == 5) || (i == 6) || (i ==7) || (i == 8)
         upsampled(:,:,i) = gk(:,:,slice+1, i);
      elseif (i == 9) || (i == 10) || (i == 11)
         upsampled(:,:,i) = (gk(:,:,slice, i) + gk(:,:,slice+1, i))./2;
      else


         upsampled(:,:,i) = gk(:,:,slice, i);
      end
   end
end
```

Appendix C.10 – Difference of Estimated vs. True Low-Resolution Images (12)

```
function gkdiff = diffg12im(gk, gk_guess, K, slice, offset)
if offset == 1
   for j = 1:K
      if (j == 1) || (j == 2) || (j == 3)
         gkdiff(:,:,j) = (((gk(:,:,slice,j) + gk(:,:,slice+1,j))./2) - gk_guess(:,:,j));
      else
         gkdiff(:,:,j) = (gk(:,:,slice,j) - gk_guess(:,:,j));
      end
   end
elseif offset == 2
   for j = 1:K
      if (j ==1) || (j == 2) || (j == 3) || (j == 4)
         gkdiff(:,:,j) = (gk(:,:,slice+1,j) - gk_guess(:,:,j));
      elseif (j == 5) || (j == 6) || (j == 7)
         gkdiff(:,:,j) = (((gk(:,:,slice,j) + gk(:,:,slice+1, j))./2) - gk_guess(:,:,j));
      else
         gkdiff(:,:,j) = (gk(:,:,slice,j) - gk_guess(:,:,j));
      end
   end
elseif offset == 3
   for j = 1:K
      if (j == 1) || (j == 2) || (j == 3) || (j == 4) || (j == 5) || (j == 6) || (j ==7) || (j == 8)
         gkdiff(:,:,j) = (gk(:,:,slice+1,j) - gk_guess(:,:,j));
      elseif (j == 9) || (j == 10) || (j == 11)
         gkdiff(:,:,j) = (((gk(:,:,slice,j) + gk(:,:,slice+1, j))./2) - gk_guess(:,:,j));
      else
         gkdiff(:,:,j) = (gk(:,:,slice,j) - gk_guess(:,:,j));
      end
   end
end
gkdiff = sum(gkdiff, 3);
```

Appendix C.11 – Error Between Estimated vs. True Low-Resolution Images (12)

```
function gk_error = err_GK12im(gk, gk_guess, K, slice, offset)
if offset == 1
   for j = 1:K
      if (j == 1) || (j == 2) || (j == 3)
         gk_error(p,j) = immse(((gk(:,:,slice,j) + gk(:,:,slice+1,j))./2), gk_guess(:,:,j));
      else
         gk_error(p,j) = immse(gk(:,:,slice,j), gk_guess(:,:,j));
      end
   end

elseif offset == 2
   for j = 1:K
      if (j ==1) || (j == 2) || (j == 3) || (j == 4)
         gk_error(p,j) = immse(gk(:,:,slice+1,j), gk_guess(:,:,j));
      elseif (j == 5) || (j == 6) || (j == 7)
         gk_error(p,j) = immse(((gk(:,:,slice,j) + gk(:,:,slice+1, j))./2), gk_guess(:,:,j));
      else
         gk_error(p,j) = immse(gk(:,:,slice,j), gk_guess(:,:,j));
      end
   end
elseif offset == 3
   for j = 1:K
      if (j == 1) || (j == 2) || (j == 3) || (j == 4) || (j == 5) || (j == 6) || (j ==7) || (j == 8)
         gk_error(p,j) = immse(gk(:,:,slice+1,j), gk_guess(:,:,j));
      elseif (j == 9) || (j == 10) || (j == 11)
         gk_error(p,j) = immse(((gk(:,:,slice,j) + gk(:,:,slice+1, j))./2), gk_guess(:,:,j));
      else
         gkdiff(:,:,j) = immse(gk(:,:,slice,j), gk_guess(:,:,j));
      end
   end
end
```

Appendix C.12 – Return to NIfTI File Format

```
%% Return from png to nii format

% Natsortfiles: Stephen (2022). Natural-Order Filename Sort.
% (https://www.mathworks.com/matlabcentral/fileexchange/47434-natural-order-filename-sort),
% MATLAB Central File Exchange.



Main = '% Input main file name';
Out = '% Input where you want new nifti file to go';

% Be sure *.filetype is changed if using dicom to *.dcm
A = dir('% Main file name *.png');
A = natsortfiles({A.name});
[~, b] = size(A);

% Input image size
filename = im2double(imread(string({strcat(Main, A(1))})));
Image_size = size(filename);
Nift = zeros([Image_size(1) Image_size(2) b]);

for i = 1:b
    % If dicom change imread to dicomread
    filename = im2double(imread(string({strcat(Main, A(i))})));
    Nift(:,:,i) = Nift(:,:,i) + filename;
end

niftiwrite(Nift, strcat(Out, 'Name of nifti file.nii'));
```