MRI-Based Susceptibility Mapping for In-Vivo Iron and Blood Oximetry Measurements

by

H. E. Erdevig

Department of Physics

Thesis defense date

April 10th, 2017

Thesis advisor: Dr. Stephen Russek

National Institute of Standards and Technology

Honors Council representative: Prof. John Cumalat

Department of Physics

Third reader: Prof. Corey Neu

Department of Mechanical Engineering

Fourth reader: Prof. Allan Franklin

Department of Physics

Erdevig, H. E.

MRI-Based Susceptibility Mapping for In-Vivo Iron and Blood Oximetry Measurements

Thesis directed by Dr. Stephen Russek

MRI is increasingly used in mapping tissue susceptibility to identify cerebral microbleeds associated with traumatic brain injury and pathological iron deposits associated with neurodegenerative diseases such as Parkinson's and Alzheimer's disease [1, 2]. Accurate measurement is important for determining oxygen and iron content in blood vessels and tissue in the brain, which are in turn used for noninvasive clinical diagnosis and treatment assessments. Magnetic field distortions with a resolution of a few parts per billion can be measured using MRI phase maps. The field distortion map can then be inverted to obtain a quantitative susceptibility map. The primary focus of this thesis project is to determine the accuracy of these MRI-based susceptibility measurements and to demonstrate their ability to reliably measure the concentration of oxygenated hemoglobin in-vitro. The susceptibility of paramagnetic salts in cylindrical containers with varied temperature and orientation relative to the static MRI field were compared with theoretical predictions. The MRI susceptibility measurements were compared with SQUID magnetometry. Limitations of these measurements were investigated with Finite Element Method and Monte Carlo simulations of the macroscopic and microscopic field shifts in our samples, respectively. Measurements of oxygen concentration of bovine hemoglobin samples will be tested against optical absorption techniques to test the potential functionality of MRI oximetry in in-vivo diagnostics.

Dedication

To my dog Turtle, my mom Dana, and my high school physics teacher Dr. Gavin Polhemus. None of the work presented here would have been possible without their inspiration and support.

Acknowledgements

I would like to thank, first and foremost, my thesis advisor Dr. Stephen Russek for his positive encouragement, humor, and support. I feel incredibly privileged to have a brilliant mentor who has guided my research while giving me the freedom to pursue my specific interests, who has always been willing to take the time to thoroughly answer any physics questions I had, and who has provided invaluable opportunities for me to grow as a scientist and researcher.

I am also sincerely grateful to the rest of my honors thesis committee, Prof. John Cumalat, Prof. Corey Neu, and Prof. Allan Franklin for their time and generous feedback as well as the rest of the MRI Standards Group at NIST: Dr. Kathryn Keenan, Dr. Karl Stupic, Dr. Slavka Carnicka, Dr. Bryan Yunker, and Dr. Michael Boss. All of whom have unreservedly provided their time and guidance to me while completing the work presented here.

I would like to acknowledge the National Institute of Standards and Technology Summer Undergraduate Fellowship (NIST SURF) Program and the U.S. Department of Commerce Pathways Internship Program for funding me during the completion of my thesis research.

I am particularly grateful to Dr. Allen D. Elster for publishing "Questions and Answers in Magnetic Resonance Imaging" as a free online resource. His explanations of difficult MRI physics concepts given in accessible terms has been invaluable to me while studying MRI.

Finally, I would like to thank all of the dedicated scientists that came before me whose discoveries in magnetic resonance I was able to build upon with this thesis work.

Contents

Chapter

Appendix

Figures

Figure

Chapter 1

Introduction

1.1 Background and Motivation

Magnetic Resonance Imaging (MRI) is a powerful imaging technique in medicine that utilizes the quantum mechanical spin of protons in the hydrogen atoms of water molecules to create a proton density map that allows clinicians to visualize tissue inside the human body. Since its development in the 1980s, MRI has proven to have a wide range of applications in both clinical diagnoses and image guided therapies. The foundation of the technique having been established, subsequent research has focused on optimizing MRI for a variety of specific applications. MRI scanning protocols and post-processing techniques have been developed that can highlight or suppress different types of tissues or map a particular disease biomarker. One such biomarker is the magnetic susceptibility of tissue, which can be mapped using the phase component of the complex data that MRI collects. Most current MRI research is dedicated to developing MRI techniques for making quantitative measurements that can be used for objective diagnosis in the clinical setting. Specifically, Quantitative Susceptibility Mapping (QSM)[3] using MRI is becoming more prevalent than traditional qualitative techniques, such as susceptibility weighted imaging.[4]

Several promising applications for QSM have provided substantial motivation for producing measurement methods ready for translational medicine. Such applications include mapping neural diseases, traumatic brain injuries,[5, 6, 7] blood oxygen content,[8] and iron overload in the heart and liver.[9] Neurodegenerative diseases, such as Parkinson's and Alzheimer's disease, have been associated with excess iron in the brain.[1, 2] These iron deposits would be quantifiable with QSM, opening the door for objective sans-autopsy diagnoses and the ability to monitor treatment prospects. Another application of QSM, relevant to the work in this thesis, is finding and determining the severity of cerebral microbleeds resulting from traumatic brain injury. For this application, a reproducible and quantitative method is particularly important. Finally, measurements of iron overload in the heart and liver, caused by diseases such as hemochromatosis, are important because iron can catalyze the conversion of hydrogen peroxide into free radicals, causing damage to cell membranes, proteins, and DNA.[10] MRI has the unique ability to measure oxygen concentration of deep-lying vessels in the brain, which is necessary for most of these applications. However, more research is required to validate the accuracy of this technique and of MRI-based susceptibility measurements in general. The research described in this thesis has contributed to achieving this validation and paving the way for measurement standards to be developed for clinical QSM.

Accurate in-vivo measurements of magnetic susceptibility, along with the necessary calibrations and post-processing techniques, are required to use magnetic susceptibility as a quantitative biomarker. Specifically, use of MRI to measure quantitative susceptibility for clinical applications requires creating a standard scanning protocol, a reconstruction and analysis software package,[11] and a calibration phantom for comparing data collected from different scanners and data taken before and after changes to a single system. Creating standard measurement protocols and calibration phantoms would help ensure site-to-site comparability of data and allow QSM to be more widely and reliably used in clinical applications. With these standards created through advancing research, in-vivo MRI susceptibility measurements may become the gold standard for tissue susceptibility quantification in medicine as a whole.

This chapter is designed as an introduction and review of the necessary components to understand content of the rest of the work presented in this thesis. The following section presents an overview of MRI susceptometry, an explanation of the theory behind the measurement technique, and a discussion of MRI's unique ability to make these measurements. The last section includes an in-depth look at the issues and research gaps delaying the use of this important medical tool in radiology. Additionally, the ability of the work presented in this thesis to accelerate QSM's translation to medicine will be examined.

For quantitative, non-invasive, localized measurements of blood-oxygen saturation MRI may be the only qualified tool [12]. MRI can potentially map blood-oxygen concentration more precisely and with higher resolution than current standard techniques because these measurements come from the proton precession frequency within individual atoms. In-vivo MRI susceptibility measurements, if done properly, could become the gold standard for tissue susceptibility quantification. The development of this technique as a standard requires verification of the accuracy of MRI susceptibility measurements relative to the aforementioned traditional methods. The foundational work of the research presented here achieved verification of MRI measurement accuracy through comparison with current techniques. Verified quantitative susceptibility measurements allow for the creation of standard reference materials needed for clinical calibration phantoms. The work presented here establishes that the relative susceptibilities can, in fact, be accurately determined from local magnetic field shifts for simple geometries and agree with primary measurements of susceptibility when compared with existing standards.

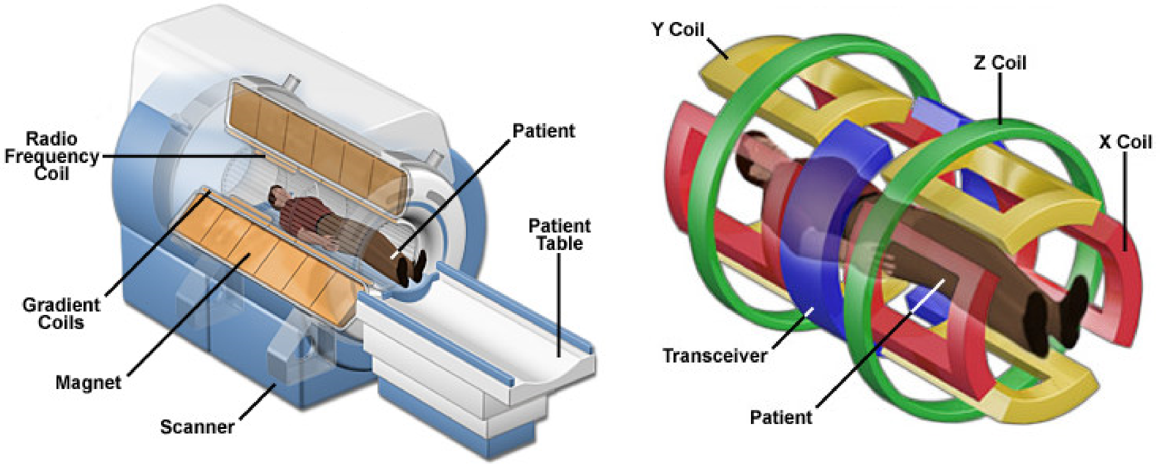
To further develop the technique of MRI susceptometry, several tasks must be completed. More suitable primary standards are required to validate MRI susceptibility measurements in complex geometries. More extensive investigation into how the local field depends on microscopic tissue geometry is required to determine the accuracy of local field models. A standard for MRI susceptibility mapping is needed that also assesses the best algorithms for the field-susceptibility inversion among the changing subjects and scanners. Liu et al. state that "Such an assessment should include sequence parameters, phase unwrapping, background phase removal, and susceptibility inversion algorithms"[11]. Challenges still exist in understanding the field contributions of tissue when more than one susceptibility source that needs quantification is present. The work presented here attempts to fulfill the need for standard reference materials for QSM and developing a deeper understanding of the distortion corrections needed for geometries that mimic tissue structure.

The work of this thesis is to develop a standard model for MRI susceptometry and oximetry required verification of the accuracy of these measurements via comparison with current standards, a determination of the limitations of such measurements, and an assessment of the theoretical models and algorithms on which they are based. The intended product from this work was a calibration phantom designed specifically for quantitative susceptibility mapping, a software package for standardized reconstruction and analysis of the MRI data, and publications speaking to their capabilities.

1.2 MRI and MRI-Based Measurements

An MRI scanner is composed of three different magnet systems shown in Figure 1.1; the main magnet, gradient coils, and a radio frequency coil. The first and largest magnet is cryogenically cooled with helium, so that it can generate a static field, B_0, with a magnetic field strength of up to 7 T, which is the maximum field strength FDA approved for human clinical scanners. A secondary, gradient magnetic field, G, is produced by a cylindrical shell of conductive sheets surrounding the bore of the magnet. This linear gradient field varies the total applied field, $B_a = B_0 + G(x, y, z)$, throughout the measurement volume to allow for localization of the proton spin packets.

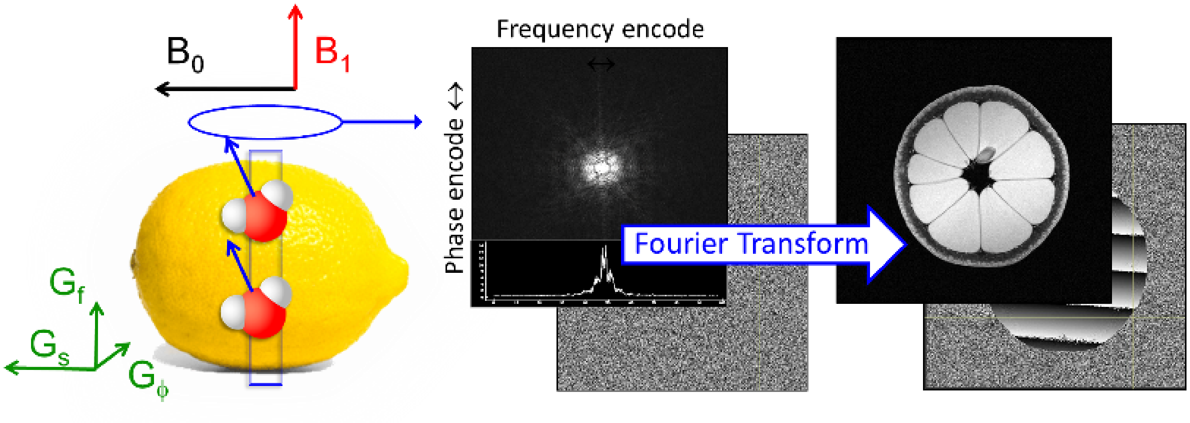
Figure 1.1: Clinical MRI Scanner Cutaway shown with gradient coils[13]



A radio frequency coil then applies an oscillating magnetic field with a frequency tuned to the resonant frequency, ω, of the hydrogen proton in water to excite the proton spins to an excited state. This resonant frequency is called the Larmor frequency and is dependent on the total magnetic field applied to the proton and the proton's gyromagnetic ratio unique to the atom to which the proton belongs $\omega = \gamma B_a$. The gyromagnetic ratio for hydrogen is $\gamma = 42.58\,\mathrm{MHz/T}$.

A pulse sequence manipulates the alignment and dephasing of the spin packets and applies gradients for slice selection and phase/frequency encoding. The resulting precession induces a current in the same radio frequency coil that applied the oscillating magnetic field. The magnitude and phase data is collected from this coil into a complex array called "k-space". This data is reconstructed into a "real space" magnitude and phase image with a 2-D Fourier transform. Figure 1.2 is a schematic representing the data collected from a lemon and reconstructed into a complex array in image space with a Fourier Transform (Equation 1.1).

Figure 1.2: Water molecules within a single slice of a lemon, scanned with NIST's pre-clinical MRI scanner, are shown to have a magnetic moment interacting with the MRI static field, B_0, and oscillating field, B_1, and the gradient field's slice, frequency, and phase encoding components. The complex k-space array shows the signal from a single slice organized according to phase and frequency encoding. A Fourier transform reconstructs the k-space array into an image-space array. The magnitude component of each complex array is shown in front of its phase counterpart.



$$S(x,y) = \frac{4}{\pi^2} \iint e^{2\pi i(k_x x + k_y y)}\, dk_x\, dk_y \tag{1.1}$$

The reconstructed phase image represents the relative phase of the proton magnetic moments in a sample. Appendices B and C show screenshots and the source code, respectively, of a reconstruction and data manipulation tool that was developed to support the work presented in this thesis and is included in the software package called PhantomViewer.

Proton spin precession is modeled by a set of macroscopic differential equations, called Bloch equations, that calculate the nuclear magnetization vector components as a function of time. These Equations 1.2, 1.3, and 1.4 model the precessional motion, spin dephasing and relaxation of proton magnetic moments as they realign themselves with B_0 during an MRI scan.

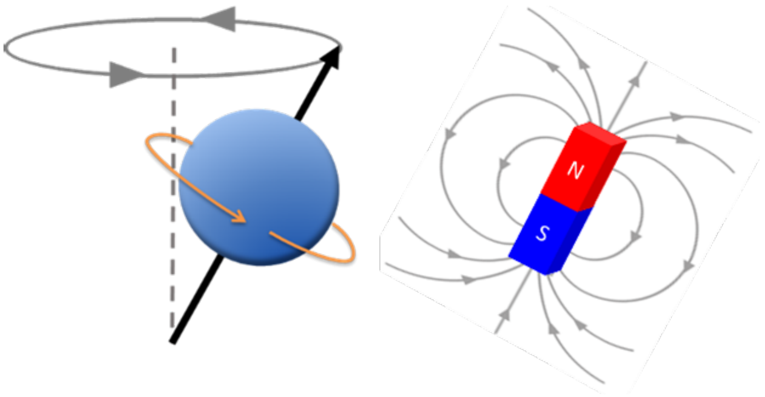
$$\frac{dM_x(t)}{dt} = \gamma(M(t) \times B(t))_x - \frac{M_x(t)}{T_2} \tag{1.2}$$

$$\frac{dM_y(t)}{dt} = \gamma(M(t) \times B(t))_y - \frac{M_y(t)}{T_2} \tag{1.3}$$

$$\frac{dM_z(t)}{dt} = \gamma(M(t) \times B(t))_z - \frac{M_z(t) - M_0}{T_1} \tag{1.4}$$

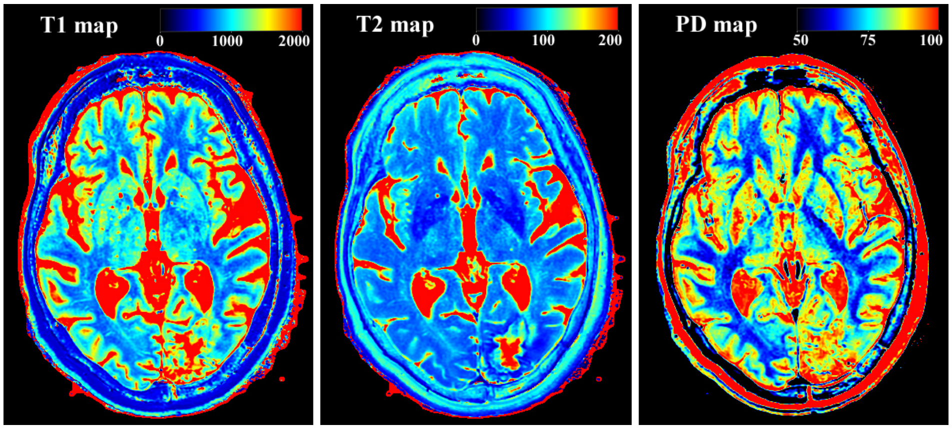
$\vec{M}(t) =< M_x(t), M_y(t), M_z(t) >$ is the nuclear magnetization, γ is the gyromagnetic ratio, and $\vec{B}(t) =< B_x(t), B_y(t), B_0 + \Delta B_z(t) >$ is the total magnetic field experienced by the proton. T_1 is the time constant for the regrowth of the longitudinal magnetization, M_z, during spin-lattice relaxation of the proton magnetic moment. T_2 is the time constant for the decay of the transverse magnetization, M_{xy}, of the proton generally caused by dephasing of the spin packets by static local field disturbances and spin-spin interactions. Figure 1.3 shows the motion of the proton magnetic moment during precession. Shown with the proton, is a bar magnet representing the field produced by the proton magnetic moment.

Figure 1.3: Diagram showing the precessional motion of a proton magnetic moment and its magnetic field.



In an MRI system, the total field, $\vec{B}$, is comprised of the applied magnetic field as well as the local magnetic field created by neighboring proton magnetic moments as well as any nearby magnetically susceptible material. This total magnetic field, $B_z = B_0 + B_{local}$, is what determines the proton motion in the above Bloch equations, and consequently the resulting signal collected by the MRI. The Bloch equations' dependency on B_{Local} and the relaxation constants allows MRI to directly measure parameters such as T_1, T_2, and local field values. Figure 1.4 shows 3 processed images from the same axial MRI brain scan. The first image shows a map of T_1 values and the second shows a map of T_2 values, measured in milliseconds. The third image is the proton density map represented with percentage values.

Figure 1.4: Quantitative T_1, T_2, and proton density maps of an axial brain scan.(syntheticmr.com)



In this example, it can be seen that the different relaxation constants, that depend on different mechanics and interactions of the proton spin packets, can highlight different fine structure in the brain. Using pulse sequences designed to exploit this difference, a T_1 map can be generated that highlights white matter in the brain and a T_2 map can be generated that highlights the gray matter and cerebrospinal fluid. Some T_1 or T_2 sequences with use of contrast agents can also be used to distinguish healthy tissue from tumors. Measurement of tissue susceptibility is possible with MRI because of the sensitivity of proton precession frequency to local field disturbances. Susceptibility weighted images and quantitative susceptibility maps can then be generated from measurements of the local field shifts for mapping disease associated with changes in magnetic susceptibility.

Chapter 2

Physics of Magnetic Susceptibility and Traditional Measurement Methods

2.1 Physics of Magnetic Susceptibility

The reconstructed phase image of an MRI scan represents the relative phase of the proton magnetic moments in the scanned object. Local magnetic fields that are created by paramagnetic materials exposed to B_0, cause a predictable shift in phase of the protons in that material relative to its surroundings by an amount proportional to its susceptibility. Magnetic susceptibility is a dimensionless quantity and proportionality constant that indicates the degree of magnetization, M, of a material in response to an applied magnetic field, H. Magnetically susceptible materials create these local fields by adopting a magnetization that either contributes to or opposes the applied field depending on the type of magnetic susceptibility.

Human tissue is predominately water with a diamagnetic susceptibility of $\chi = -9.04 \times 10^{-6}$ at 20 °C. Among the diamagnetic tissue are small paramagnetic, super-paramagnetic and antiferromagnetic components. The work presented in this thesis is only concerned with measuring the paramagnetic signature of biomimics against a diamagnetic reference. This section will discuss the physics of the interactions of these types of materials with magnetic fields using only SI units, as is the case with the rest of the work presented in this thesis.

In a diamagnetic material, the electrons circulate in closed orbital shells, allowing the collection of electron spins in an atomic orbital to act as a current loop. According to Lenz's Law and Faraday's Law, an applied magnetic field will induce a current in an existing current loop in order to oppose the change in the magnetic field and keep the magnetic flux through the loop constant.

At the atomic level, this causes the electrons in a diamagnet to reconfigure themselves in response to an external magnetic field such that the current that arises from the collection of the electrons creates an opposing magnetic field.

In a paramagnetic material, the magnetic moments of electrons do not completely cancel out each other because of the presence of unpaired electrons in the valence shell of the atoms within the material. These unpaired electron spins create a magnetic field that aligns with an applied external magnetic field in order to minimize the torque on these dipoles created by the magnetic field. Paramagnetism is temperature dependent according to Curie's Law.

The deoxygenated hemoglobin present in cerebral microbleeds is paramagnetic having released its O_2 ligand from its original diamagnetic oxyhemoglobin structure, leaving iron in the ferrous (Fe^{2+}) state. Fe^{2+} has one paired set of electrons and four unpaired electrons in its outer 3d shell. The spin magnetic moment resulting from these four unpaired electrons is 4.90 bohr magnetons, which creates a paramagnetic signature measurable with MRI.

The local field created by deoxygenated hemoglobin or any other paramagnetic material depends on the magnetization of that material. The magnetization (Equation 2.1) of a diamagnetic or paramagnetic material is linearly dependent on the magnetic field applied to that material with the material-dependent susceptibility, χ, as a constant.

$$\vec{M}(\vec{r}) = \chi(\vec{r})\vec{H}(\vec{r}) \tag{2.1}$$

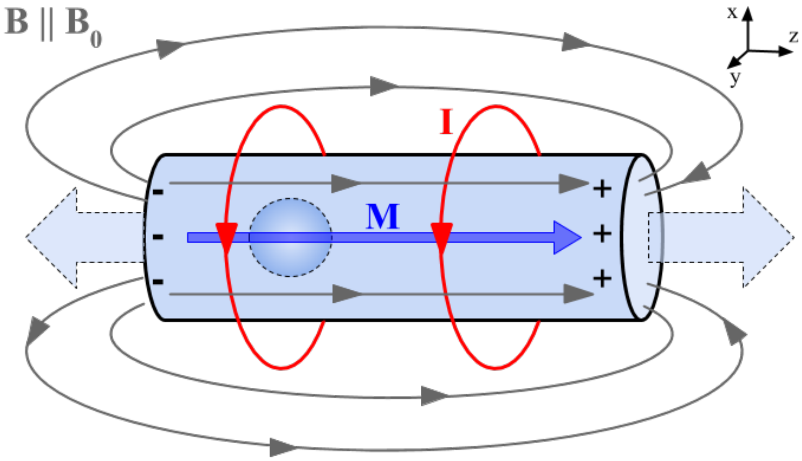
Both the magnetization, M, and the magnetic field strength, H, are measured in amperes/meter (A/m).

For the special case of an infinitely long and uniformly magnetized cylinder aligned with the static magnetic field, B_0, in the z direction (depicted in Figure 2.1), the magnetization, M, can be viewed as a surface current, I, equivalent to that of an infinitely long solenoid with a current per unit length equal to M. The average field, B, inside the cylinder is given by Equation 2.2.

$$\vec{B} = \vec{B_0} + \mu_0 \vec{M} = B_0 + \frac{\chi B_0}{3} \tag{2.2}$$

However B_{Local} is not the same as the average field and a correction needs to be made determined by Lorentz field. The Lorentz correction to cancel out the local field of a sphere within the volume of the cylinder is calculated considering a uniformly magnetized sphere with magnetization opposite to that of the cylinder in order to cancel out the existing field due to a spherical portion of the cylinder and replace it with its actual microscopic field contribution, which is assumed to be zero in Equation 2.3.

Figure 2.1: Analytical model for paramagnetic tissue in magnetic field



2.2 Magnetic Properties of Human Tissue

To understand the magnetic field shifts of a uniform paramagnetic object, we need only know Maxwell's equations' description of the macroscopic magnetic field in matter. An important assumption made in modeling the local magnetic field of a paramagnet is that the position of diamagnetic water molecules relative to each other and to the paramagnetic ions. The local field differs from the macroscopic field and is given by the macroscopic field minus the Lorentz field. The Lorentz field is a correction to the macroscopic continuum model and attempts to account for the local microscopic distribution of moments. One of the main approximations in MRI-based susceptibility measurements is to assume that the local field, B_{Local}, is given by the average field, B, minus the Lorentz field shown in Equation 2.3.

$$B_{Local} = B - \frac{2}{3}\chi B_0 \tag{2.3}$$

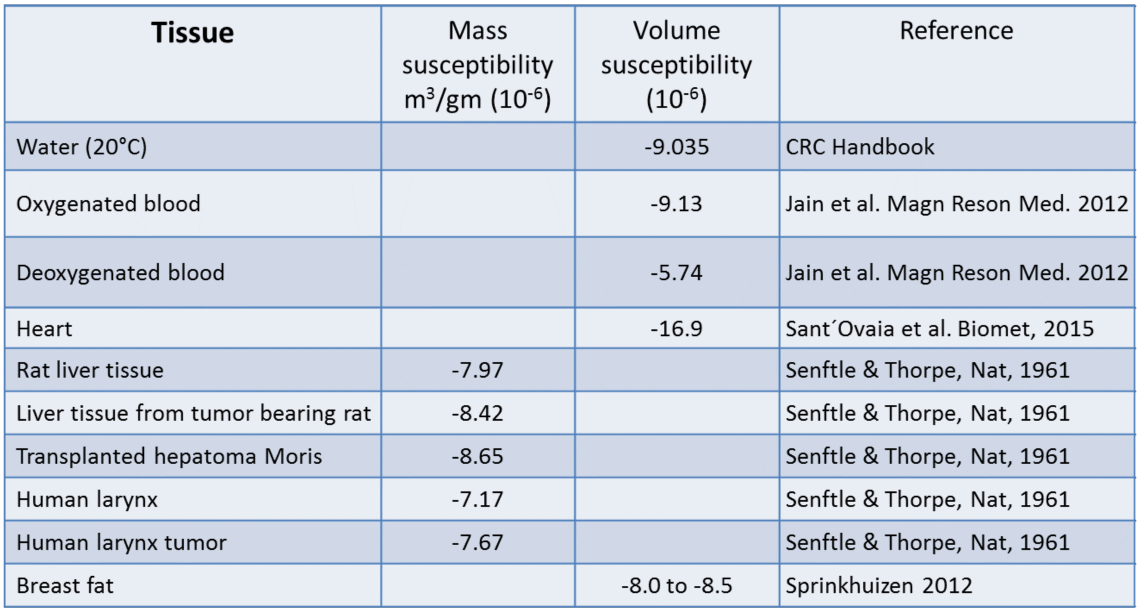
This assumes that the local microscopic fields average to zero. A portion of the work presented in this thesis is dedicated to testing the validity of this assumption.

With Quantitative Susceptibility Mapping, we need to see the response of proton spins to local magnetic fields created by magnetically susceptible tissue. If we can characterize a tissue's magnetic properties, then these field distortions can be used to calculate its susceptibility. When exposed to an external magnetic field, diamagnetic, paramagnetic, and ferromagnetic materials develop a weak anti-parallel, parallel, and strong parallel magnetic dipole moment, respectively.

2.3 SQUID Measurements

Magnetic susceptibility, χ, is a dimensionless proportionality constant that indicates the degree of magnetization, M, of a material in response to an external magnetic field, B: $M = \chi \cdot \frac{B}{\mu_0}$, where B is measured in Tesla as opposed to M and H which are measured in amperes per meter. Being dimensionless, magnetic susceptibility does not have an SI unit standard, but SI standards do exist for measurements of the magnetic dipole moment of materials. The magnetization of a

Figure 2.2: Literature values of susceptibility of different types of human tissue. Measured ex-vivo.



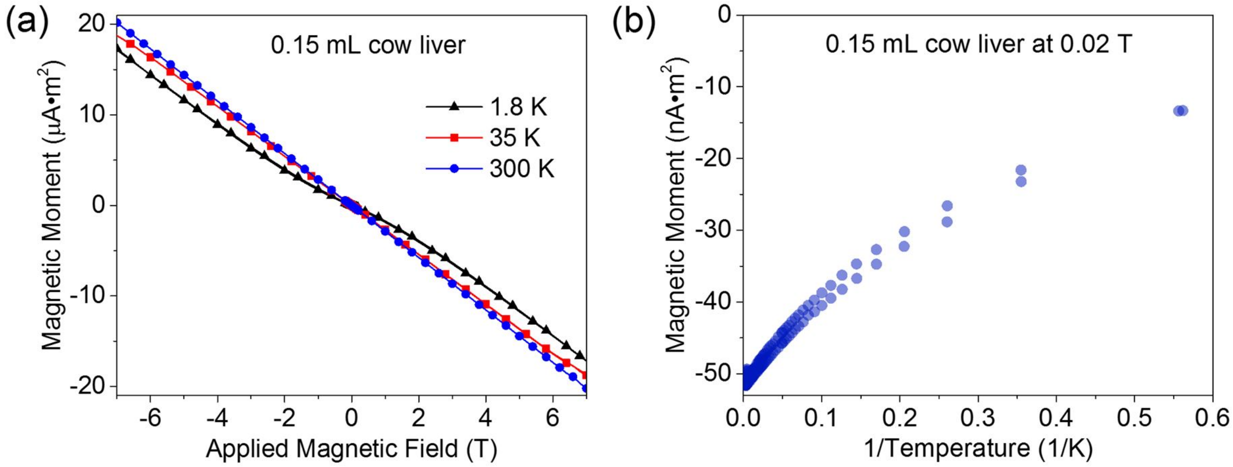
Tissue	Mass susceptibility m^3/gm (10^{-6})	Volume susceptibility (10^{-6})	Reference
Water (20°C)		-9.035	CRC Handbook
Oxygenated blood		-9.13	Jain et al. Magn Reson Med. 2012
Deoxygenated blood		-5.74	Jain et al. Magn Reson Med. 2012
Heart		-16.9	Sant´Ovaia et al. Biomet, 2015
Rat liver tissue	-7.97		Senftle & Thorpe, Nat, 1961
Liver tissue from tumor bearing rat	-8.42		Senftle & Thorpe, Nat, 1961
Transplanted hepatoma Moris	-8.65		Senftle & Thorpe, Nat, 1961
Human larynx	-7.17		Senftle & Thorpe, Nat, 1961
Human larynx tumor	-7.67		Senftle & Thorpe, Nat, 1961
Breast fat		-8.0 to -8.5	Sprinkhuizen 2012

material is defined as the magnetic dipole moment per unit volume. Currently, the most sensitive measurements of magnetic dipole moments can only be made with a superconducting quantum interference device (SQUID) magnetometer. SQUID magnetometers can measure extremely small magnetic fields by detecting magnetic flux through a superconducting ring consisting of two parallel Josephson junctions. For a hydrated solution or biological sample, a diamagnetic component to the local magnetic field will be present. SQUID magnetometers can only measure the total susceptibility of a sample and the paramagnetic component's behavior is only observed while it overshadows this diamagnetic moment at extremely low temperatures. Curie's Law ($\chi = C/T$), with C being the material dependent Curie constant, describes the inverse relation between paramagnetic susceptibility and temperature that is responsible for this phenomenon. This temperature requirement restricts the magnetometer's use to ex-vivo measurements as the human body cannot survive at temperatures of only a few Kelvin. These ex-vivo measurements would be conducted on excised tissue. Excision is not only a highly invasive procedure, but the sample itself will have different properties outside of living tissue. Such inevitable characteristic changes would include: dehydration that would affect the total volume of the sample as well as the amount of diamagnetic water that would normally be present while in the body, and blood deoxygenation as the iron in hemoglobin loses oxygen to the surrounding air thus altering its paramagnetic property. Thus the SQUID standard is not usable for accurate, room temperature measurements of the paramagnetic component of tissue.

2.3.1 Tissue Measurements

To test the ability of MRI to accurately measure susceptibility in human tissue, appropriate biomimic materials with verified susceptibilities were used. Tissue is predominantly diamagnetic at body temperature 310 K and room temperature 300 K. This is seen in Figure 2.3, which shows the magnetic moment vs. field for cow liver. The magnetic susceptibility is dominated by the diamagnetic susceptibilities of water (-9.05×10^{-6}) and fat typically (-10.0×10^{-6}) [14]. All susceptibility values in this paper are reported in SI units. The complex magnetic structure of tissue is seen at lower temperatures. Figure 2.3(a) shows a decrease in the diamagnetic (negative) slope as the temperature decreases indicating the presence of a paramagnetic component. At low temperature (1.8 K) there is a deviation in linearity due to paramagnetic and ferrimagnetic components. The presence of a ferrimagnetic component is seen in Figure 2.3(b), which plots the moment vs. inverse temperature. If there were only a paramagnetic component, the data would be linear. For liver, the paramagnetic and ferrimagnetic components are predominantly due to blood iron in deoxygenated hemoglobin and iron oxide deposits (ferritin).

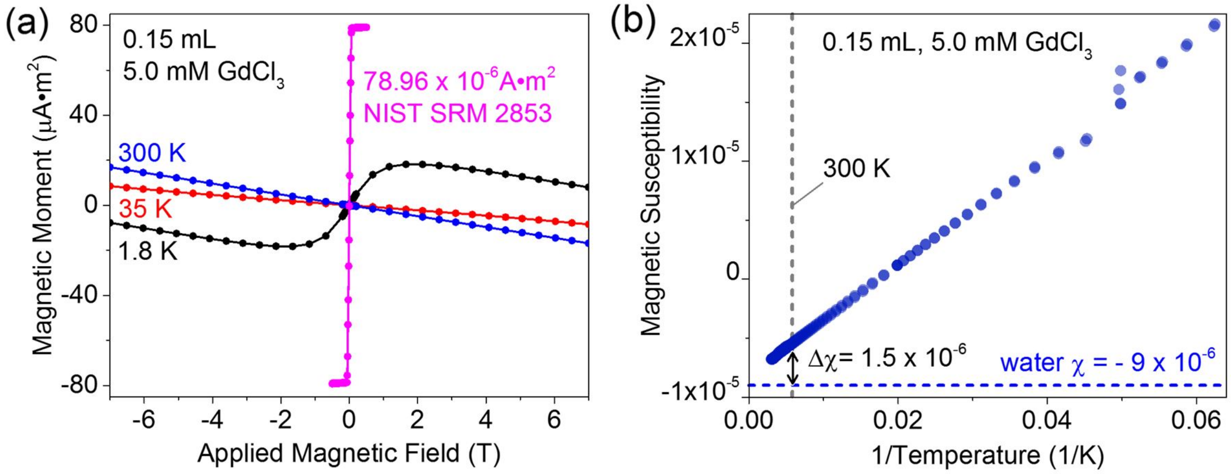
Figure 2.3: (a) SQUID magnetometer measurements of magnetic moment vs. applied field for a sample of cow liver. (b) Magnetic moment vs. inverse temperature, upon heating and cooling, of the same sample.



2.3.2 Tissue Mimics

To mimic the susceptibility properties of tissue, one can use a solution of paramagnetic salts in water. Figure 2.4(b) shows schematically how water, with a diamagnetic susceptibility, with little temperature-dependence, and a paramagnetic component can roughly approximate the magnetic properties of tissue. We present data from $GdCl_3$ solutions, whose magnetic properties are shown in Figure 2.4(a),(b) for a 5.0 mM solution in deionized water. The SQUID magnetometer is calibrated with a NIST YIG (yttrium iron garnet) sphere standard reference material (SRM #2852) whose room temperature moment is $(79.9 \pm 0.3) \times 10^{-6}\,\mathrm{A\,m^2}$. The moment, m, vs. applied field, B_a, data can be fit assuming a diamagnetic component and a paramagnetic component. Equation 2.4 gives the magnetic moment of a gadolinium chloride solution.

Figure 2.4: (a) SQUID magnetometer measurements of the magnetic moment vs. applied field of the 5.0 mM $GdCl_3$ solution. Also shown is the calibration curve obtained from a NIST moment standard reference material. (b) Magnetic susceptibility vs. inverse temperature for the same solution showing paramagnetic behavior. The horizontal dotted line schematically shows the diamagnetic susceptibility of water. The arrow indicates the susceptibility contribution from the Gd^{3+} ions at 300 K.



$$m = N_{Gd} V g \mu_B J \cdot B_J\left(\frac{gJ\mu_B B_a}{k_B T}\right) - \frac{\chi_w V B_a}{\mu_0} \tag{2.4}$$

N_{Gd} is the concentration of Gd^{3+} ions, V is the volume of the sample, g is the Landé g-factor (which is 2.0 for Gd since the angular momentum vanishes), μ_B is the Bohr magneton, J is the ion angular momentum quantum number, B_J the Brillouin function, k_B is Boltzmann's constant, T is the temperature of the sample, and χ_w is the magnitude of the diamagnetic susceptibility of water. The susceptibility due to the Gd^{3+} ions can be calculated from the model (Equation 2.4) using the best fit parameters and the measured volume. The measured Gd susceptibility for a 5.0 mM solution at 300 K, shown in Figure 2.4 is $\chi_{Gd} =(1.58 \pm 0.16) \times 10^{-6}$, comparable to the theoretical value of $\chi_{th} = 1.89 \times 10^{-6}$. Comparing the tissue magnetic properties, shown in Figure

2.3, to those of the standard Gd solutions, shown in Figure 2.4, one can see that the reference solutions are a good starting point to mimic the magnetic properties of tissue, although they lack the full complexity of tissue. The errors in the measured value come from errors in the moment measurement, the volume measurement and from the extraction of the smaller Gd moment from the larger diamagnetic moment of water. For comparison, the difference in susceptibility between deoxygenated and oxygenated blood, as measured by MRI, is $(3.43 \pm 0.08) \times 10^{-6}$ [15].

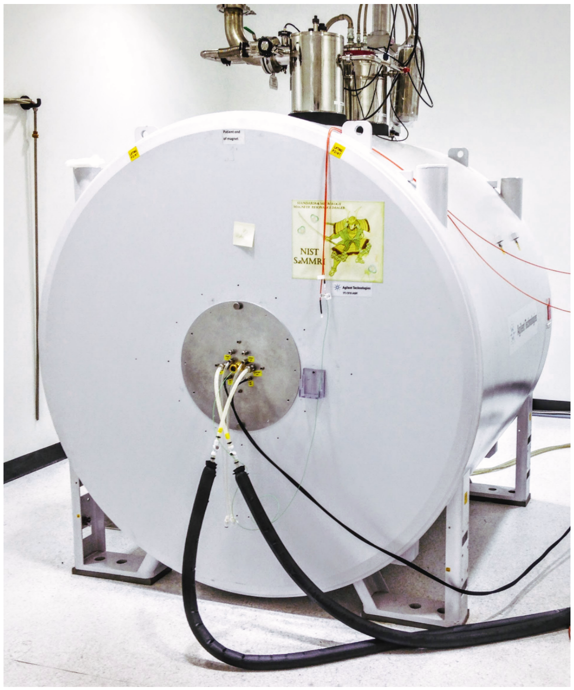
Chapter 3

MRI Measurements

MRI susceptibility measurements are typically achieved by acquiring magnitude and phase data from a gradient echo sequence with multiple echo times. Magnitude and phase images of a phantom are shown in Figure 3.2. The phase image clearly shows distortion of the phase fronts due to the enhanced susceptibility of the paramagnetic salt solution contained within the vial. The imaging was performed in a 30 cm bore preclinical scanner (Figure 3.1) designed to image at 1.5 T, 3.0 T, or 7.0 T. The data in this paper were obtained with a static field of $B_0 = (1.502\,102 \pm 0.000\,006)$ T. The error in the field represents the typical field variation over the active volume with a standard shimming procedure. The phase image must be unwrapped and the low-spatial frequency background phase variations, due to an imperfect shimming of the magnet and to susceptibility discontinuities far from the region of interest, subtracted. These post-processing algorithms are performed on the collected data with the reconstruction and analysis software package, PhantomViewer. Appendix C contains code written as part of this thesis work for the reconstruction and distortion correction tools provided in PhantomViewer.

The phantoms used for the MRI measurements of paramagnetic susceptibility presented were designed to take advantage of the simple analytical model described in Section 2.1. Not only do cylindrically shaped paramagnetic salt solution containers allow us to measure relative susceptibility without having to invert the magnetic field profile, but they can approximate the local field contributions of paramagnetic material in cylindrical blood vessels.

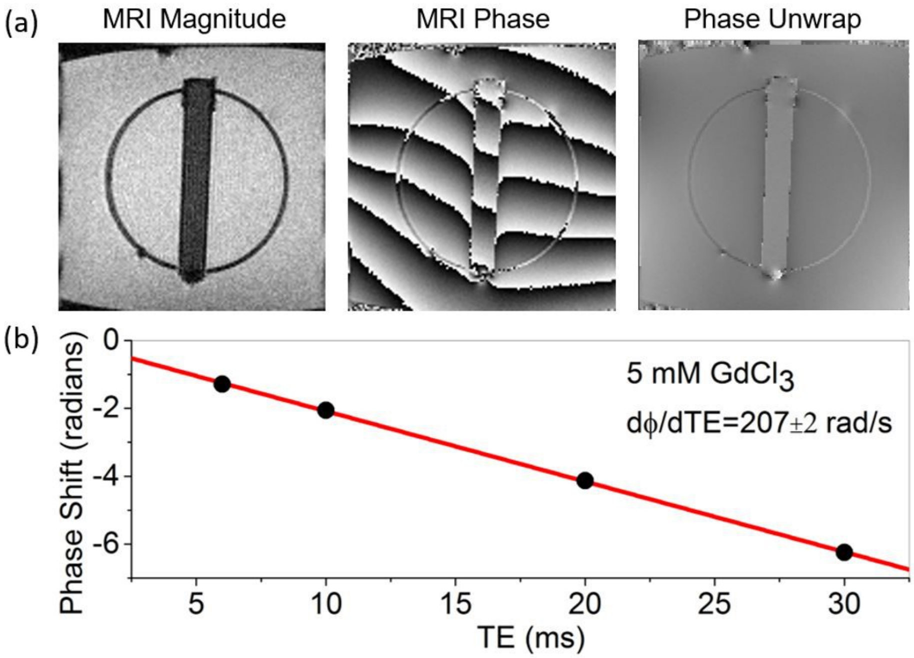
Figure 3.1: NIST pre-clinical variable field MRI scanner



The difference in proton phase (inside relative to outside the cylindrical vial), $\delta\phi$, after an echo time, TE, is proportional to the local induced field, δB_L, along the main field direction: $\delta\phi = \gamma_p \cdot \delta B_L \cdot TE$, where γ_p is the shielded proton gyromagnetic ratio. Figure 3.2(b) shows a plot of the phase shift measured across the vial from scans taken with different echo times (TE). The slope of this line ($\delta\phi/\delta TE$) is proportional to the local field distortion created by the paramagnetic cylinder and therefore also the susceptibility of the paramagnetic salt solution.

The local field differs from the macroscopic field and is given by the macroscopic field minus the Lorentz field. The Lorentz field is a correction to the macroscopic continuum model and attempts to account for the local microscopic distribution of moments. The slope of the measured phase difference vs. echo time, as shown in Figure 3.2(b), will yield δB_L. The magnetic field distortion is a convolution of the magnetic susceptibility distribution, $\chi(r)$, with the magnetic dipole kernel, d($\mathbf{r}$): $\delta B_L(\vec{r}) = d(\vec{r}) \otimes \chi(\vec{r})$.[16] The susceptibility map can be obtained by inverting the field profile, although complex methods are required since this inversion is not unique.[17, 18, 19, 20]

Figure 3.2: (a) Magnitude and phase images of a vial containing 5.0 mM $GdCl_3$. The dark circle in the MRI magnitude image is a 76 mm diameter polycarbonate support for the vials. The third image shows the phase after unwrapping and after the long wavelength background has been subtracted. (b) Phase difference as a function of echo time (TE) taken from phase maps.



This problem is discussed further in Chapter 4. In an effort to avoid this ill-posed inversion problem, the measurements presented here are limited to simple cylindrical geometries. This allowed for use of the simple model described in Section 2.1, where the induced local magnetic fields are simply related to the susceptibility. The analytical formulas in Equations 3.1 and 3.2 below are derived from Maxwell's equations to calculate these fields. For a long cylinder the internal and external field distortion is given by:[20]

$$\text{Internal: } \delta B_L = \frac{\Delta\chi B_0}{6}(3\cos^2\theta - 1) \tag{3.1}$$

$$\text{External: } \delta B_L = \frac{\Delta\chi B_0 a^2}{2r^2}\sin^2\theta\cos 2\phi \tag{3.2}$$

Where $\Delta\chi$ is the susceptibility difference between the inside and outside of the cylinder, θ is the angle of the cylinder axis with respect to the main field, ϕ is the azimuthal angle of the observation point relative to the plane of the main field and cylinder axis, and a is the radius of the cylinder. For the simple case where the cylinder is aligned with the main field ($\theta = 0$), the susceptibility difference is given by $\Delta\chi = 3\delta\phi/(\gamma_p B_0 TE)$. By measuring the slope of $\delta\phi$ vs. TE, as seen in Figure 3.2, the susceptibility can be determined. The susceptibility difference of the 5.0 mM $GdCl_3$ solution at 300 K, was $(1.71 \pm 0.02) \times 10^{-6}$, which agrees with the SQUID magnetometer measurements. The intrinsic errors for the SQUID measurements are larger than the MRI measurements, although the systematic errors for the MRI measurements have not yet been determined. Though, in comparing the accuracy of these susceptibility measuring techniques, we must consider that SQUID measurements of excised tissue susceptibility are also inherently inaccurate due to inevitable water loss, blood oxidation, and volume changes. All three of these changes in excised tissue result in significant shifts in the paramagnetic and diamagnetic properties of the tissue from those that would be measured in-vivo.

3.1 Composition Dependence

To test the efficacy of our technique in measuring susceptibility with MRI and to test the sensitivity of MRI in making very small phase measurements, a phantom, shown in Figure 3.3, was constructed. This phantom holds four cylindrical containers of very low concentration solutions (1 mM, 0.5 mM, 0.2 mM, and 0.1 mM) of aqueous $GdCl_3$. The containers were 12 mm diameter polypropylene straws with a wall thickness of 0.22 mm (sub-voxel size to avoid regions of no signal). The magnetization of the 0.5 mM sample was measured on a commercial Superconducting Quantum Interference Device (SQUID) magnetometer to determine the angular momentum quantum number of the gadolinium ions. All four straws were scanned with a gradient echo sequence in the NIST MRI scanner at 1.5 Tesla while the phantom's temperature control system kept the solutions and surrounding water at 14.80 °C, 25.15 °C, and 33.00 °C. Five scans with echo times of 10, 20, 30, 40, and 50 ms were taken at each temperature in order to obtain a phase shift in relation to a change in TE $\left(\frac{\delta\phi}{\delta TE}\right)$.

Figure 3.3: Phantom designed to hold solutions in a temperature-controlled water bath for measurement in NIST's pre-clinical scanner (Figure 3.1).

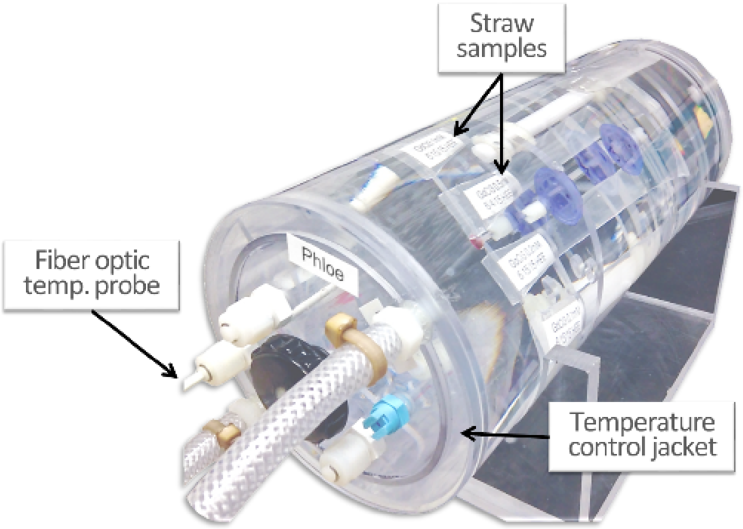
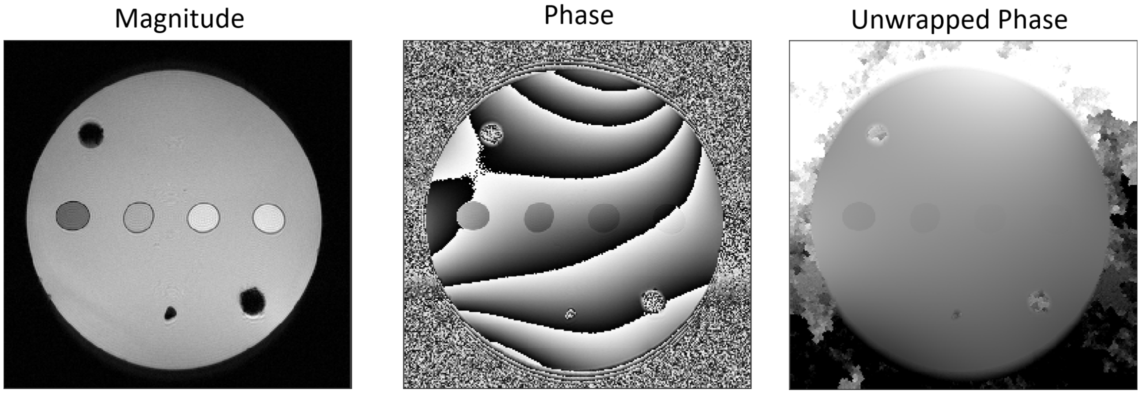


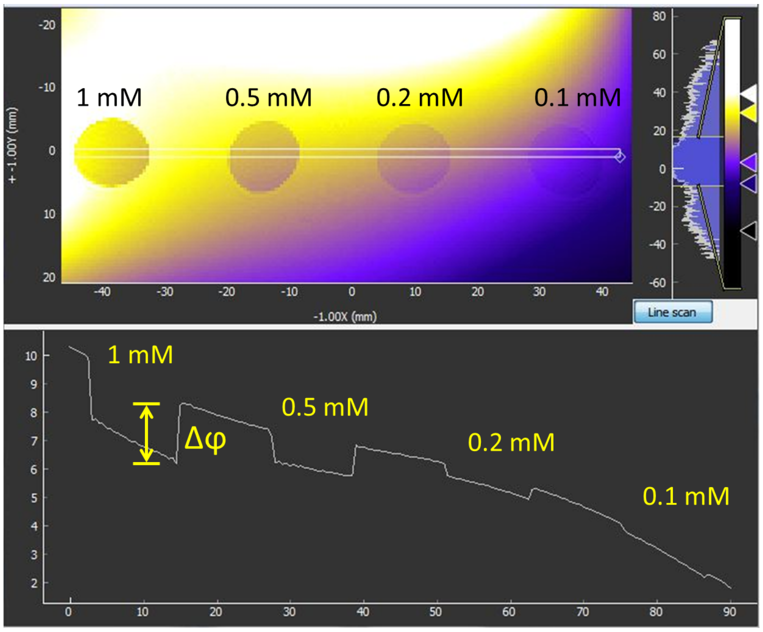
Figure 3.4 shows the reconstructed magnitude and phase images and the unwrapped phase image of an axial scan of the phantom pictured in Figure 3.3. The reconstruction and phase unwrapping was performed using PhantomViewer (see Appendix B for screenshots of the program in use).

Figure 3.4: Axial magnitude, phase, and unwrapped phase images of Phloe phantom produced with PhantomViewer reconstruction package.



With the unwrapped phase image, a measurement of the phase inside of each straw can be measured relative to the surrounding water. Figure 3.5 is a screenshot from PhantomViewer depicting the signal from the unwrapped phase image along a line scan going through each straw. Phase "dips" can be seen that show the expected correlation between phase shift and the concentration of paramagnetic ions in water.

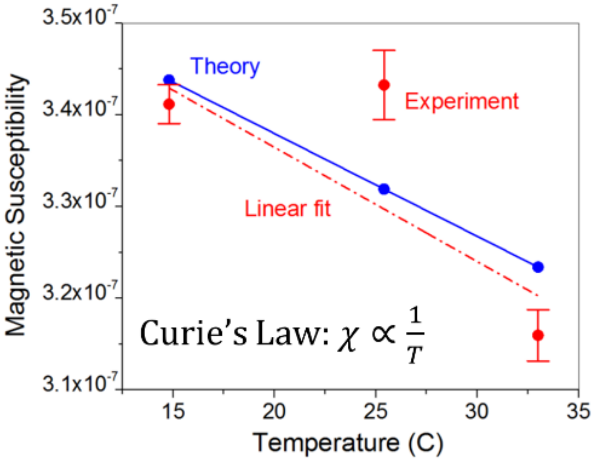
Figure 3.5: Measurement of phase shift within paramagnetic solution containers relative to surrounding diamagnetic water performed with PhantomViewer.



3.2 Temperature Dependence

The temperature dependence of the magnetic susceptibility of $GdCl_3$ was measured with MRI. Measured values fell within a few ppb of the simplified theoretical values and within 40 ppb of the SQUID-determined theoretical values. The 40 ppb discrepancy is likely due to a SQUID system calibration error. The experimental values should exhibit the "$1/T$" dependence dictated by Curie's Law, but the higher temperature data did not lay close to the best fit line despite a close agreement between the best fit line and the theoretical line. These discrepancies could be a result of imperfect shimming of the gradient coils and a lower static magnetic field used by the NIST scanner at the time of measurement.

Figure 3.6: MRI-measured susceptibility of paramagnetic cylinders at three different temperatures. Susceptibility was measured for each temperature as $\Delta\chi = 3\delta\phi/(\gamma_p B_0 T E)$.



3.3 Orientation Dependence

To test the orientational dependence, MRI phase maps were obtained from a phantom with vials (80 mm long, 5.0 mL volume); oriented along and perpendicular to the B_0 field; the vials were filled with 5.0 mM $GdCl_3$. The main compartment of the phantom was filled with deionized water. Line scans through the cylinders are shown in Figure 3.7 along with the predicted phase change and induced fields obtained from Equations 3.1 and 3.2. Good agreement is observed, although there is some deviation at the edges of the vials, in part due to the loss of signal from the plastic vial.

To more precisely verify the orientation dependence, a rotating phantom was constructed in which the 80 mm vials could be continuously rotated while in the MRI scanner. A schematic of the rotating phantom is shown in the inset in Figure 3.9. Four 80 mm vials filled with 1.0 mM and 5.0 mM $GdCl_3$ solutions were placed in the scanner. A rod extended from the outside of the scanner to the internal rotation gears; each revolution corresponded to 19-degree mechanical rotation of the phantom insert. Figure 3.8 shows axial and sagittal magnitude MRI images of the rotating barrel within the phantom that holds the gadolinium chloride solutions.

Figure 3.7: Line scans (opaque lines) of phase and corresponding field distortions taken with the field parallel (blue) and perpendicular (red) to the cylinder axis. When the field was perpendicular to the cylinder axis, the line scan was taken along B_0 ($\phi = 0$). Also shown are the predicted phase shifts (lighter lines) from Equations 3.1 and 3.2.

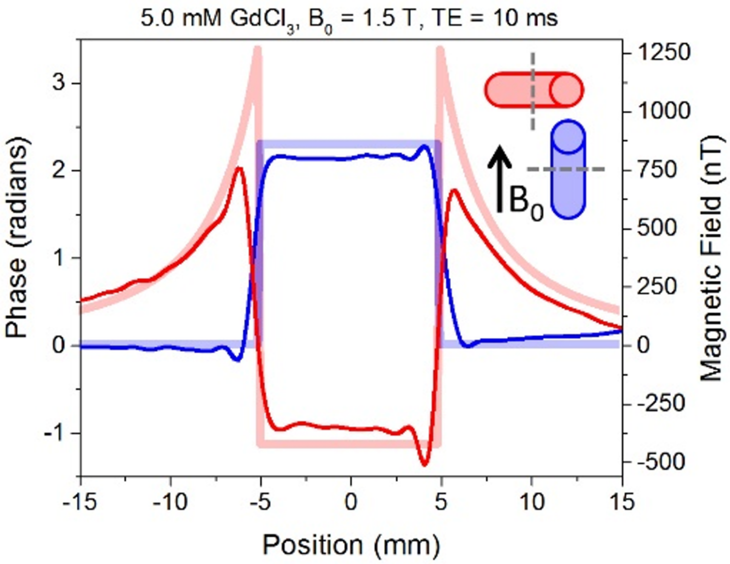
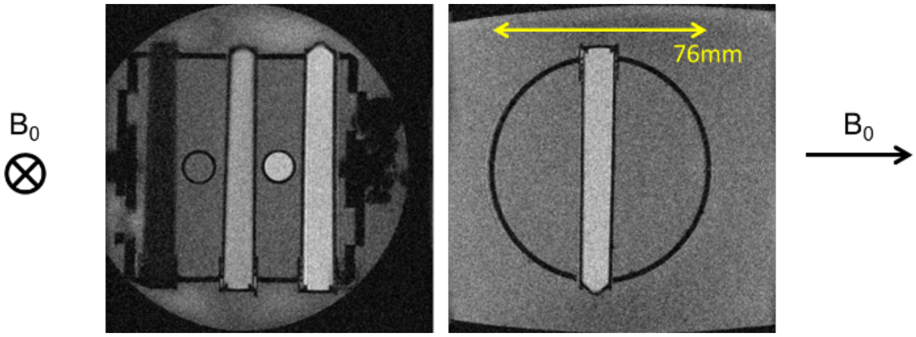
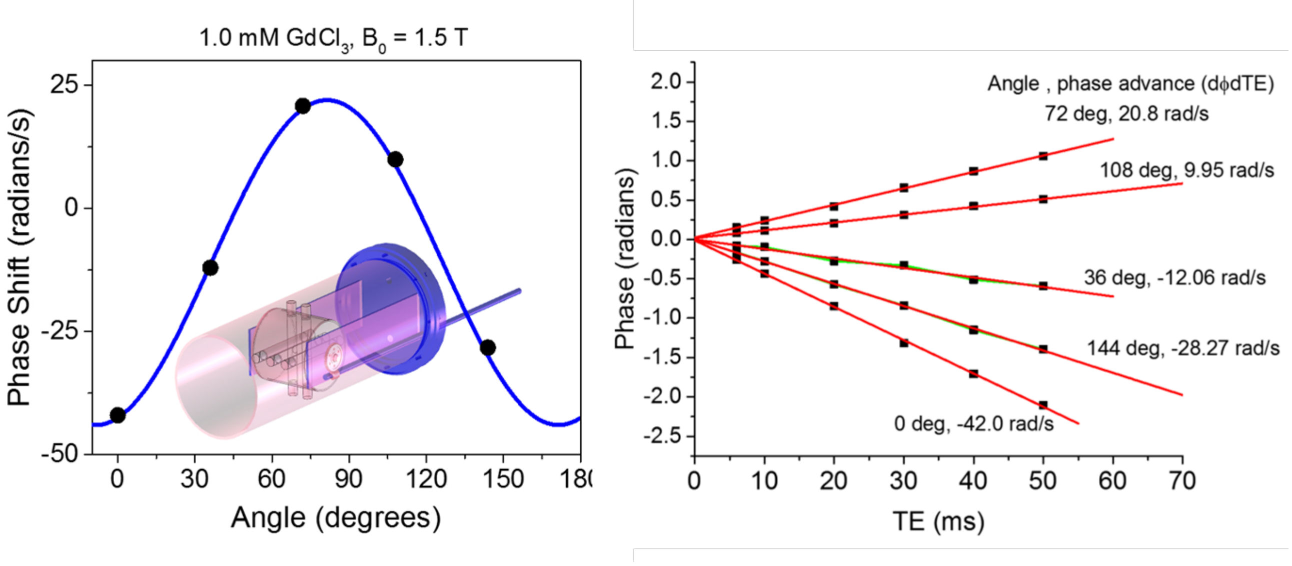


Figure 3.8: Axial and sagittal scans of rotating phantom holding five cylindrical vials of different concentrations of gadolinium chloride solutions.



The change of phase between the center of each vial and the surrounding water was collected as a function of angle, Figure 3.9. The data were fit using Equation 3.1 yielding $\Delta\chi$=$(3.24 \pm 0.05) \times 10^{-7}$ for the 1.0 mM solution.

Figure 3.9: Plot of the change of phase with echo time within a cylinder of 1.0 mM $GdCl_3$ as a function of angle of the cylinder axis relative to the B_0 field. Also plotted is a fit using Equation 3.1 (blue line). The inset a schematic of the rotating phantom used for the experiment.



The angle dependent measurements collected agreed with the values predicted by the model in Equations 3.1 and 3.2. The measured susceptibility values for each of the five different angles scanned all agreed within 5 parts per billion of the theoretical model.

Chapter 4

Numerical Simulation

Numerical calculations of the field distortions produced by the phantom shown in the inset in Figure 3.9, with four vials of paramagnetic salt solution with a susceptibility of 3.0×10^{-6}. The macroscopic field distribution is plotted, not the local field, since the macroscopic field is what is calculated using the macroscopic Maxwell equations. The local field is considered separately to test the assumption that the field contributions from randomly dispersed dipoles will average to zero in a spherically symmetric shape.

4.1 Complex K-space Inversion to Dipole Kernel

Determining the accuracy of MRI susceptibility measurements requires verifying the susceptibility distribution calculated by inverting the full 3D phase map, where the dipole kernel considers neighboring voxels throughout the volume instead of just neighboring pixels in the plane. The interactions of dipoles out-of-plane of the MRI scan could have a significant contribution to the field distortion measured at a particular pixel in-plane, especially when imaging human tissue with complex geometry. The magnetic field variation in each voxel in the imaging volume can be represented as a convolution of the magnetic susceptibility distribution, $\chi(r)$, with the magnetic dipole kernel, d($\mathbf{r}$): $\delta B_L(\vec{r}) = d(\vec{r}) \otimes \chi(\vec{r})$.[16] The susceptibility map $\chi(r)$ can then be obtained by inverting the field profile in Equation 4.1.

$$\chi(x,y,z) = FT^{-1}\left[FT\left(\frac{\phi(x,y,z)}{-\gamma \cdot B_0 \cdot TE}\right) \cdot F\right] \tag{4.1}$$

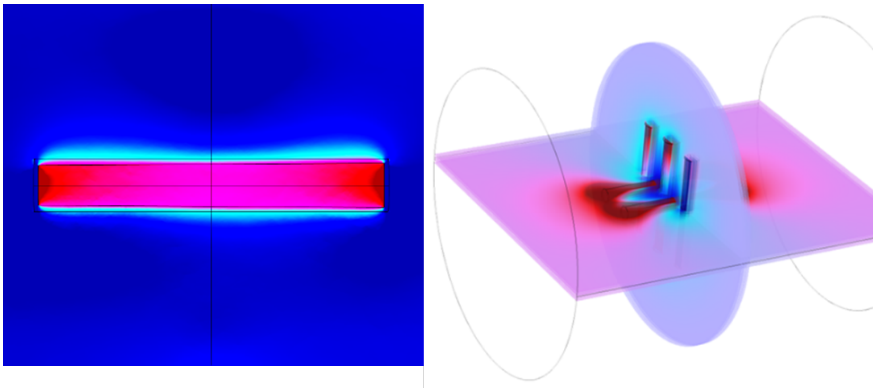
$$\text{Where the dipole kernel is: } F = \left(\frac{1}{3} - \frac{k_z^2}{k_x^2 + k_y^2 + k_z^2} \right)^{-1} \tag{4.2}$$

This deconvolution is ill-conditioned[21] due to the fact that null values in k-space occur when $k_x^2 + k_y^2 + k_z^2 = 3k_z^2$, which allow for multiple non-unique solutions. Complex methods are thus required to ensure the accuracy of the resulting quantitative susceptibility map.

4.2 Finite Element Method Simulation

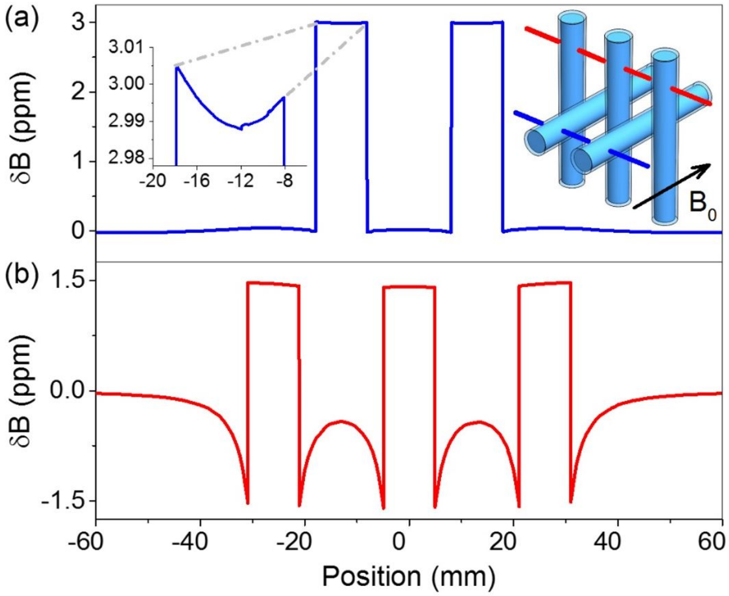
A multiphysics finite element simulation with a package for modeling magnetic fields without currents was used to compute the macroscopic field of the five perpendicular vials, shown in the inset of Figure 4.2(a). The vials were filled with a solution with a magnetic susceptibility of 3.0×10^{-6} relative to the surrounding water to simulate our experiment with 5.0 mM $GdCl_3$. The geometry of the phantom as represented in the simulation is shown in Figure 4.1 along with planes through which a field distortion has been calculated to amplify the local field interactions between the different vials.

Figure 4.1: Magnetic field interactions between the neighboring vials within the rotating phantom are qualitatively shown in a sagittal slice through the center vial (left) and the three-dimensional field distortion maps plotted along axial and coronal planes passing through each of the five vials (right).



The numerical accuracy of the field distortion was estimated to be ±7% by varying degrees of freedom from 2 to 5 million. Finite element calculations of extremely small field perturbations on a very large B_0 field gave significant numerical errors. Figure 4.2 show the field distortions when the B_0 field is parallel and perpendicular to the vial axes, respectively. The field profiles within the vials are not constant, as predicted by the simple models, due to the fields from neighboring vials, the finite length of the vials, and the phantom structure. Determining the local susceptibility from the full inversion of the 3-dimensional phase map should account for these distortions.

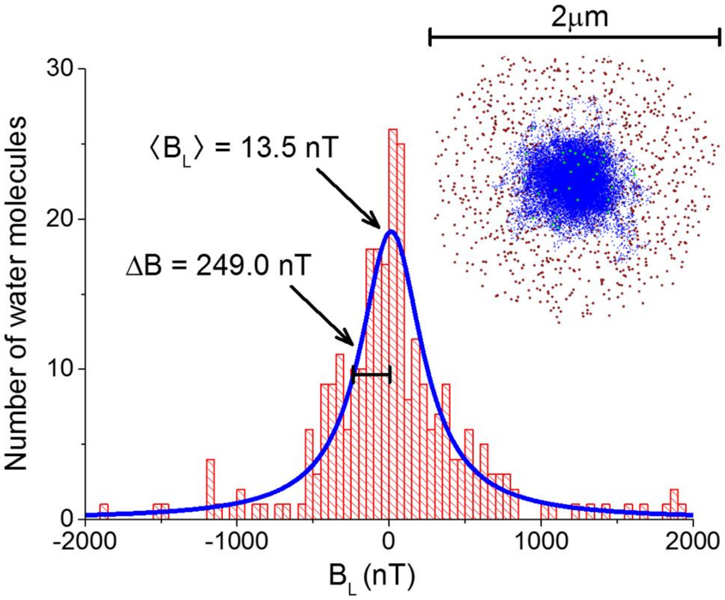
Figure 4.2: (a) The field distortion calculated by a finite element method when the vial axis is parallel to B0 field. The inset graph shows the variation within the vial due to neighboring vials and structures. (b) The field distortion when the B_0 field is perpendicular to the axis of the vial and the line scan is taken perpendicular to both B_0 field and the vial axis.



4.3 Monte Carlo Simulation

One of the main approximations in MRI-based susceptibility measurements is to assume that the local field is given by the macroscopic field minus the Lorentz field: $B_L = B_m - \frac{2}{3}\chi B_0$. This assumes that the local microscopic fields average to zero. To determine the local field, precise microscopic calculations are needed. As a simple test, we performed a Monte Carlo calculation where 2.5×10^6 Gd spins were randomly distributed in $2\,\mu\text{m}$ diameter sphere and 300 water molecules were allowed to randomly diffuse throughout the volume to simulate Brownian motion. The fields sensed by the water molecules after a time of 0.15 ms are plotted in Figure 4.3. The Gd density corresponds to 1 mM concentration and a susceptibility of 3.2×10^{-7}. The microscopic field calculated from the simulation is 13.5 nT, which is much smaller than the Lorentz field B_L= 320 nT. The simulation supports the assumption that the microscopic fields due to neighboring spins average to zero and the local field approximation is valid. For tissues, which may have more complex local geometry, this local field assumption may not be valid. As an example, the complex geometry of tissue could invalidate this microscopic field assumption if spherical symmetry is not present, such would be the case if paramagnetic ions are excluded from cells or trapped in blood vessels amongst predominately diamagnetic tissue.

Figure 4.3: Monte Carlo simulation generated histogram of microscopic fields experienced by an ensemble of water molecules diffusing (with a diffusion constant of $2.0 \times 10^{-3}\,\mathrm{mm}^2\,\mathrm{s}^{-1}$) in a 1.0 mM Gd solution. The geometry is shown in the inset with the red and blue dots representing Gd^{3+} ions and water, respectively.



The Monte Carlo simulation in Figure 4.3 shows a Gaussian distribution in microscopic fields, which had a standard deviation of 249 nT. This field distribution gives rise to a short total dephasing time T_2*. The T_2* value can be measured with the same data set as the susceptibility using the magnitude images and extracting the exponential decrease in the magnitude signal with echo time TE. The T_2* value can be used to obtain measurements of the local iron concentration in tissue.[1] While the decrease in T_2* and the change in phase both arise, in the system studied here, from the Gd spins, T_2* is strongly affected by the local microscopic structure while the phase shift is not.

4.4 Conclusions

The work presented in this thesis has shown that the relative phase shifts and local induced magnetic fields can be measured very precisely with MRI as compared to established techniques where measurement standards exist. The relative susceptibilities can be accurately determined from magnetic field shifts for simple geometries and agree with primary measurements of susceptibility where standards exist. These findings provide an important first step to developing standard reference materials and measurement methods necessary for translation to clinical medicine. More suitable primary standards than that of SQUID magnetometry, however, will be required to validate MRI susceptibility measurements in complex geometries. More extensive investigation into how the local field depends on microscopic tissue geometry is required to determine the accuracy of local field models. Developing standards for QSM will also require investigating the validity of different algorithms that try to invert the 3D field map to get a quantitative susceptibility map to extend our measurements to complex geometries that do not allow simple line scan measurements of relative phase.

Chapter 5

Future Directions

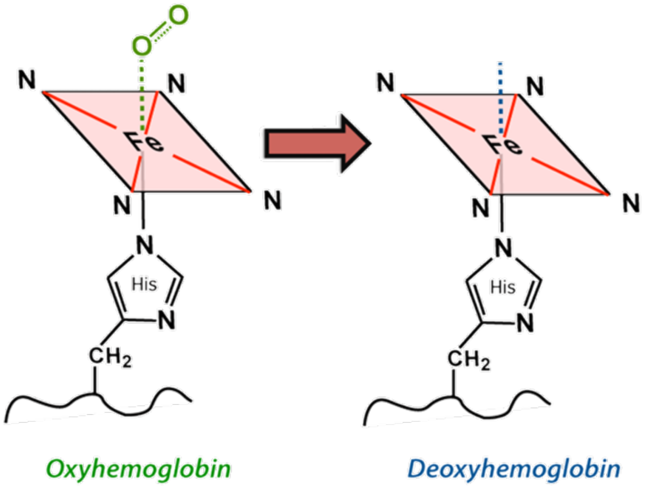
As an extension of the work presented in this thesis, experiments have been planned to test the ability of MRI to measure blood-oxygen content from susceptibility. This will require simultaneous measurement of susceptibility and oxygen-content via current standard techniques.

5.1 Blood-Oxygen Concentration vs. Magnetic Susceptibility

Traditional Methods: Pulse Oximetry The current gold standard method of blood oxygen saturation measurement is optical absorption. The familiar pulse oximeter probe is a common example of this technology's implementation in the clinical setting. Oxyhemoglobin and deoxyhemoglobin absorb red (650 nm) and infrared (950 nm) light in different relative amounts. The pulse oximeter probe transmits light of each wavelength through a patient's index finger, toward a detector. This detector compiles a percent oxygen result from the relative absorbance of each wavelength. These two points on the absorbance vs. wavelength curve of the inhomogeneous hemoglobin is a superposition of the individual absorbance curves of hemoglobin and deoxyhemoglobin. The coefficients of the two terms in this superposition can be interpreted as the relative concentration of each type. Pulse oximeter probes make oxygen concentration measurements non-invasive while also being inexpensive and easy to use. However, their convenience does not mitigate their ineffectiveness when it comes to probing deep tissue. The light used by the pulse oximeter cannot penetrate the human skull to make accurate measurements of blood oxygen saturation. Without this accuracy, optical absorbance is not able to measure the oxygen saturation of cerebral microbleeds or determine the severity of a traumatic brain injury and the measurement technique is inherently non-spatial.

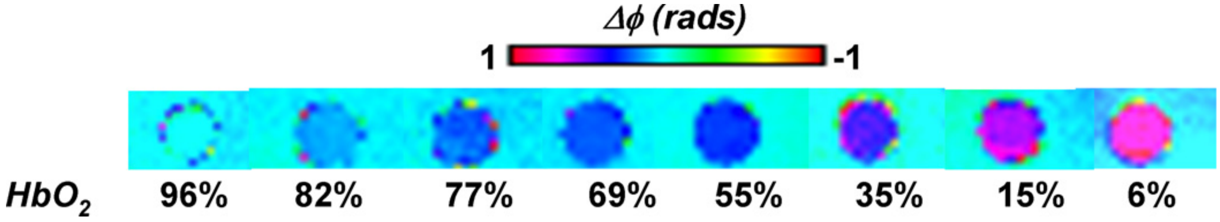
Magnetic Properties of Hemoglobin In the event of a cerebral hemorrhage, diamagnetic oxyhemoglobin in blood releases its O_2 to form deoxyhemoglobin, giving the iron atom four unpaired electrons. These unpaired electrons are responsible for deoxyhemoglobin's strong paramagnetic signature. This change in susceptibility makes it possible to perform oximetry measurements with magnetic resonance imaging techniques for use in clinical diagnostics.

Figure 5.1: Molecular oxygen reversibly binds to a coordination site for iron in each heme unit of hemoglobin, resulting in either oxyhemoglobin or deoxyhemoglobin. (Questions and Answers in MRI - AD Elster, ELSTER LLC)



Some work has already been completed to test the relationship between blood oxygen concentration and MR susceptibility measurements. For example, Jain et. al. measured whole-blood oxygen saturation via the long-cylinder approximation and phase difference method[15]. The future work following this thesis involves extending these tests to simultaneous optical absorption and MR susceptibility measurements for real-time validation with the current blood oxygenation measurement standard, as well as investigating the susceptibility of hemoglobin measured by inverting the full three-dimensional phase map, so that dipole interactions out of plane of the MRI scans are

Figure 5.2: "Representative phase difference images of cylindrical sample tubes filled with blood oxygenated to various HbO_2 levels, oriented parallel to the B_0 field and immersed in distilled water. Note the change in contrast at different oxygenation levels."[15]

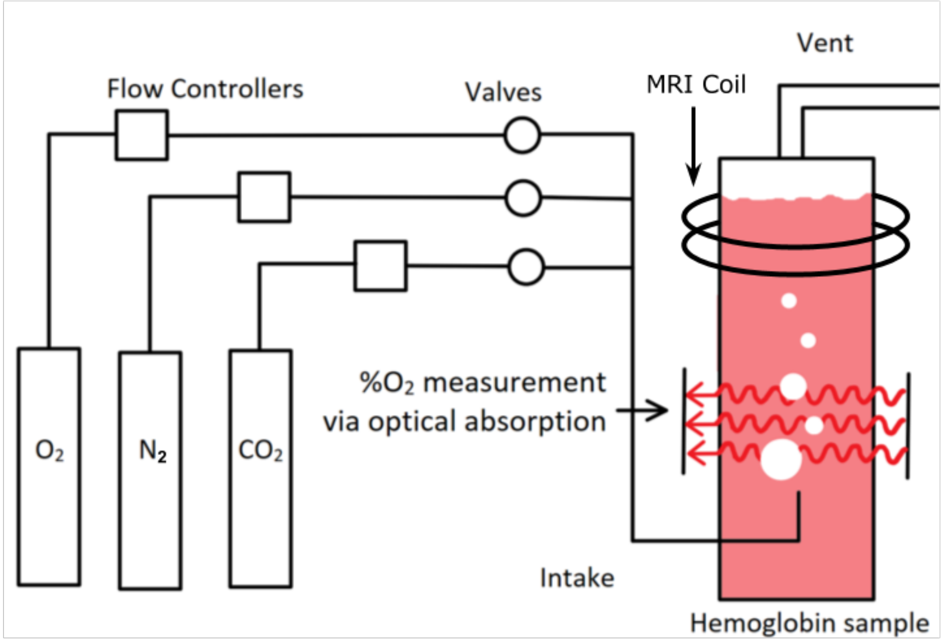


taken into account. Figure 5.2 shows the results from the existing study displayed as the colorized phase maps of cylindrical sample tubes of blood of varying oxygen concentration.

A clear relationship between the MR-measured susceptibility and the oxygenation levels of the blood can be seen. In the work following this thesis, this relationship will be examined more closely to see if the linear relationship remains while considering the other sources of susceptibility arising through the natural deoxygenation of hemoglobin (oxyhemoglobin → deoxyhemoglobin → methemoglobin → ferritin/hemosiderin).

Setup Figure 5.3 is a schematic of the existing experimental setup for oxygenating hemoglobin. This setup allows the hemoglobin sample to be aerated with a combination of oxygen, nitrogen, and carbon dioxide that will create a biologically accurate mimic of the in-vivo composition of oxygenated hemoglobin.

Figure 5.3: A gas flow system is used to control the aeration a sample of hemoglobin. Real-time blood-oxygen concentration is monitored by optical absorption while MRI gradient echo scans are performed for susceptibility map generation.



Bibliography

[1] Fumiyuki Mitsumori, Hidehiro Watanabe, and Nobuhiro Takaya. Estimation of brain iron concentration in vivo using a linear relationship between regional iron and apparent transverse relaxation rate of the tissue water at 4.7T. Magnetic resonance in medicine, 62(5):1326–30, nov 2009.

[2] Satoru Oshiro, Masaki S Morioka, and Masataka Kikuchi. Dysregulation of Iron Metabolism in Alzheimer's Disease, Parkinson's Disease, and Amyotrophic Lateral Sclerosis. Advances in Pharmacological Sciences, 2011:378278, oct 2011.

[3] Yi Wang and Tian Liu. Quantitative susceptibility mapping (QSM): Decoding MRI data for a tissue magnetic biomarker. Magnetic Resonance in Medicine, 73(1):82–101, 2015.

[4] E M Haacke, S Mittal, Z Wu, J Neelavalli, and Y-C N Cheng. Susceptibility-weighted imaging: technical aspects and clinical applications, part 1. AJNR. American journal of neuroradiology, 30(1):19–30, 2009.

[5] Jens H. Jensen, Kamila Szulc, Caixia Hu, Anita Ramani, Hanzhang Lu, Liang Xuan, Maria F. Falangola, Ramesh Chandra, Edmond A. Knopp, John Schenck, Earl A. Zimmerman, and Joseph A. Helpern. Magnetic field correlation as a measure of iron-generated magnetic field inhomogeneities in the brain. Magnetic Resonance in Medicine, 61(2):481–485, 2009.

[6] Karin Shmueli, Jacco A de Zwart, Peter van Gelderen, Tie-Qiang Li, Stephen J Dodd, and Jeff H Duyn. Magnetic susceptibility mapping of brain tissue in vivo using MRI phase data. Magn. Reson. Med., 62(6):1510–1522, 2009.

[7] Huan Tan, Tian Liu, Ying Wu, Jon Thacker, Robert Shenkar, Abdul Ghani Mikati, Changbin Shi, Conner Dykstra, Yi Wang, Pottumarthi V Prasad, Robert R Edelman, and Issam A Awad. Evaluation of iron content in human cerebral cavernous malformation using quantitative susceptibility mapping. Investigative radiology, 49(7):498–504, 2014.

[8] E. Mark Haacke, Karthik Prabhakaran, Ilaya Raja Elangovan, Zhen Wu, and Jaladhar Neelavalli. Susceptibility Weighted Imaging in MRI, 2011.

[9] Samir D Sharma, Diego Hernando, Debra E Horng, and Scott B Reeder. Quantitative susceptibility mapping in the abdomen as an imaging biomarker of hepatic iron overload. Magnetic resonance in medicine, 74(3):673–83, sep 2015.

[10] Nancy C Andrews. Disorders of Iron Metabolism. New England Journal of Medicine, 341(26):1986–1995, dec 1999.

[11] Chunlei Liu, Wei Li, Karen A Tong, Kristen W Yeom, and Samuel Kuzminski. Susceptibility-weighted imaging and quantitative susceptibility mapping in the brain. Journal of magnetic resonance imaging : JMRI, 42(1):23–41, 2015.

[12] Michael C Langham, Jeremy F Magland, Charles L Epstein, Thomas F Floyd, and Felix W Wehrli. Accuracy and precision of MR blood oximetry based on the long paramagnetic cylinder approximation of large vessels. Magnetic resonance in medicine, 62(2):333–40, aug 2009.

[13] Kristen Coyne. MRI: A Guided Tour.

[14] John F. Schenck. The role of magnetic susceptibility in magnetic resonance imaging: MRI magnetic compatibility of the first and second kinds. Medical Physics, 23(6):815, 1996.

[15] V Jain, O Abdulmalik, K.J Propert, and F.W Wehrli. Investigating the magnetic susceptibility properties of fresh human blood for non invasive oxygen saturation quantification. Magnetic Resonance in Medicine, 68(3):863–867, 2013.

[16] Ferdinand Schweser, Andreas Deistung, Berengar W Lehr, and JuÌĹrgen R Reichenbach. Differentiation between diamagnetic and paramagnetic cerebral lesions based on magnetic susceptibility mapping. Medical Physics, 37(10):5165, 2010.

[17] Tian Liu, Jing Liu, Ludovic De Rochefort, Pascal Spincemaille, Ildar Khalidov, James Robert Ledoux, and Yi Wang. Morphology enabled dipole inversion (MEDI) from a single-angle acquisition: Comparison with COSMOS in human brain imaging. Magnetic Resonance in Medicine, 66(3):777–783, 2011.

[18] Ludovic De Rochefort, Ryan Brown, Martin R. Prince, and Yi Wang. Quantitative MR susceptibility mapping using piece-wise constant regularized inversion of the magnetic field. Magnetic Resonance in Medicine, 60(4):1003–1009, 2008.

[19] Tian Liu, Pascal Spincemaille, Ludovic De Rochefort, Bryan Kressler, and Yi Wang. Calculation of susceptibility through multiple orientation sampling (COSMOS): A method for conditioning the inverse problem from measured magnetic field map to susceptibility source image in MRI. Magnetic Resonance in Medicine, 61(1):196–204, 2009.

[20] Yu-Chung N Cheng, Jaladhar Neelavalli, and E Mark Haacke. Limitations of calculating field distributions and magnetic susceptibilities in MRI using a Fourier based method. Phys. Med. Biol. Phys. Med. Biol, 54(54):1169–1189, 2009.

[21] Bryan Kressler, Ludovic De Rochefort, Tian Liu, Pascal Spincemaille, Quan Jiang, and Yi Wang. Nonlinear regularization for per voxel estimation of magnetic susceptibility distributions from MRI field maps. IEEE Transactions on Medical Imaging, 29(2):273–281, 2010.

[22] T Liu, K Surapaneni, M Lou, L Cheng, P Spincemaille, and Y Wang. Cerebral microbleeds: burden assessment by using quantitative susceptibility mapping. Radiology, 262(1):269–278, 2012.

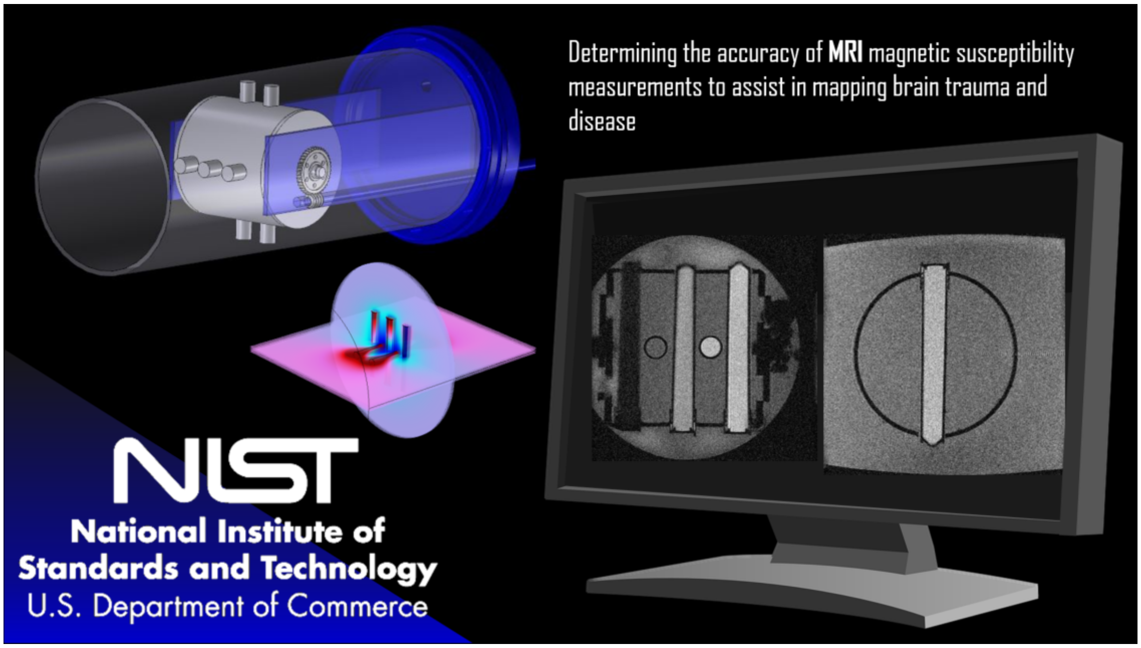
[23] Jaladhar Neelavalli and Yu-Chung Norman Cheng. Magnetic Susceptibility. In Susceptibility Weighted Imaging in MRI, pages 17–31. John Wiley & Sons, Inc., 2011.

Appendix A

Video of rotating phantom presented at APS March Meeting 2016

This short video shows the amplitude and phase scans of a rotating sample in an MRI. These scans are used to evaluate the accuracy of MRI Susceptibility Mapping, a technique increasingly being used to evaluate brain microbleeds, traumatic brain injury, and neurological diseases such as Alzheimer's and Parkinson's. Accuracy is important for determining oxygen and iron content in blood vessels and tissue in the brain, which are in turn used for noninvasive clinical diagnosis and treatment assessments.

Figure A.1: Accuracy of MRI Susceptibility Mapping: `https://youtu.be/JQQKl3puy4U`



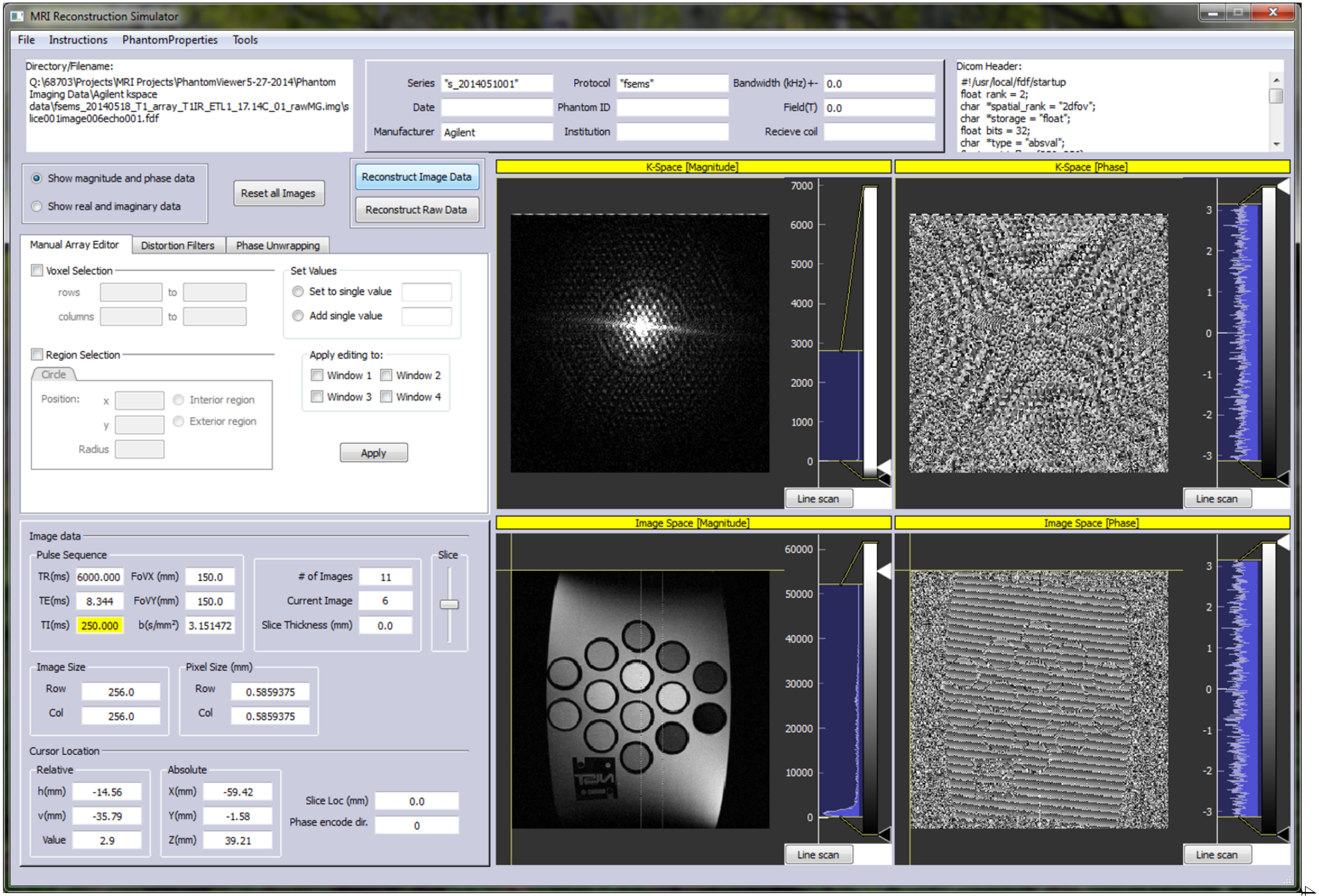
Determining the accuracy of **MRI** magnetic susceptibility measurements to assist in mapping brain trauma and disease

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

Appendix B

PhantomViewer Screenshots

Full software package available at https://github.com/StephenRussek/PhantomViewer



MRI Reconstruction Simulator

Directory/Filename:
Q:\68703\Projects\MRI Projects\PhantomViewer5-27-2014\Phantom Imaging Data\Agilent kspace data\fsems_20140518_T1_array_T1IR_ETL1_17.14C_01_rawMG.img\slice001image006echo001.fdf

Series "s_2014051001"
Date
Manufacturer Agilent
Protocol "fsems"
Phantom ID
Institution
Bandwidth (kHz)+- 0.0
Field(T) 0.0
Recieve coil

Dicom Header:
```
#!/usr/local/fdf/startup
float  rank = 2;
char  *spatial_rank = "2dfov";
char  *storage = "float";
float  bits = 32;
char  *type = "absval";
```

● Show magnitude and phase data
○ Show real and imaginary data

Manual Array Editor

□ Voxel Selection
rows [ ] to [ ]
columns [ ] to [ ]

Set Values
○ Set to single value
○ Add single value

□ Region Selection
Circle
Position: x [ ] y [ ] Radius [ ]
○ Interior region
○ Exterior region

Apply editing to:
□ Window 1 □ Window 2
□ Window 3 □ Window 4

Image data

Pulse Sequence
TR(ms) 6000.000
TE(ms) 8.344
TI(ms) 250.000
FoVX (mm) 150.0
FoVY(mm) 150.0
b(s/mm²) 3.151472

# of Images 11
Current Image 6
Slice Thickness (mm) 0.0

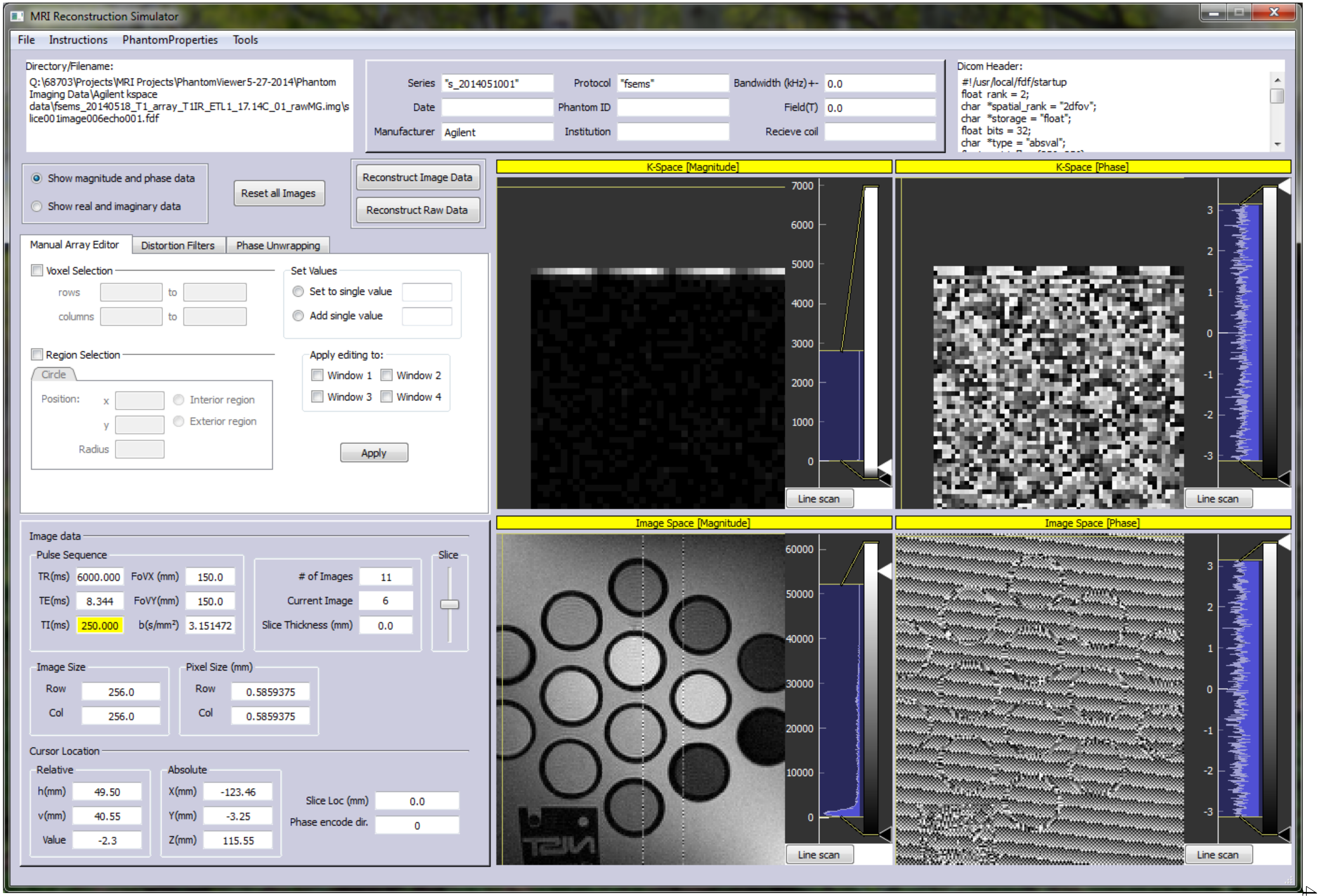
Image Size
Row 256.0
Col 256.0

Pixel Size (mm)
Row 0.5859375
Col 0.5859375

Cursor Location
Relative
h(mm) -14.56
v(mm) -35.79
Value 2.9

Absolute
X(mm) -59.42
Y(mm) -1.58
Z(mm) 39.21

Slice Loc (mm) 0.0
Phase encode dir. 0



Directory/Filename:
Q:\68703\Projects\MRI Projects\PhantomViewer5-27-2014\Phantom Imaging Data\Agilent kspace data\fsems_20140518_T1_array_T1IR_ETL1_17.14C_01_rawMG.img\slice001image006echo001.fdf

Series	"s_2014051001"	Protocol	"fsems"	Bandwidth (kHz)+-	0.0
Date		Phantom ID		Field(T)	0.0
Manufacturer	Agilent	Institution		Recieve coil	

Dicom Header:

```
#!/usr/local/fdf/startup
float  rank = 2;
char  *spatial_rank = "2dfov";
char  *storage = "float";
float  bits = 32;
char  *type = "absval";
```

● Show magnitude and phase data
○ Show real and imaginary data

□ Voxel Selection
rows [ ] to [ ]
columns [ ] to [ ]

Set Values
○ Set to single value [ ]
○ Add single value [ ]

□ Region Selection
Circle
Position: x [ ] ○ Interior region
y [ ] ○ Exterior region
Radius [ ]

Apply editing to:
□ Window 1 □ Window 2
□ Window 3 □ Window 4

Image data

Pulse Sequence

TR(ms)	6000.000	FoVX (mm)	150.0
TE(ms)	8.344	FoVY(mm)	150.0
TI(ms)	250.000	b(s/mm²)	3.151472

# of Images	11
Current Image	6
Slice Thickness (mm)	0.0

Slice

Image Size

Row	256.0
Col	256.0

Pixel Size (mm)

Row	0.5859375
Col	0.5859375

Cursor Location

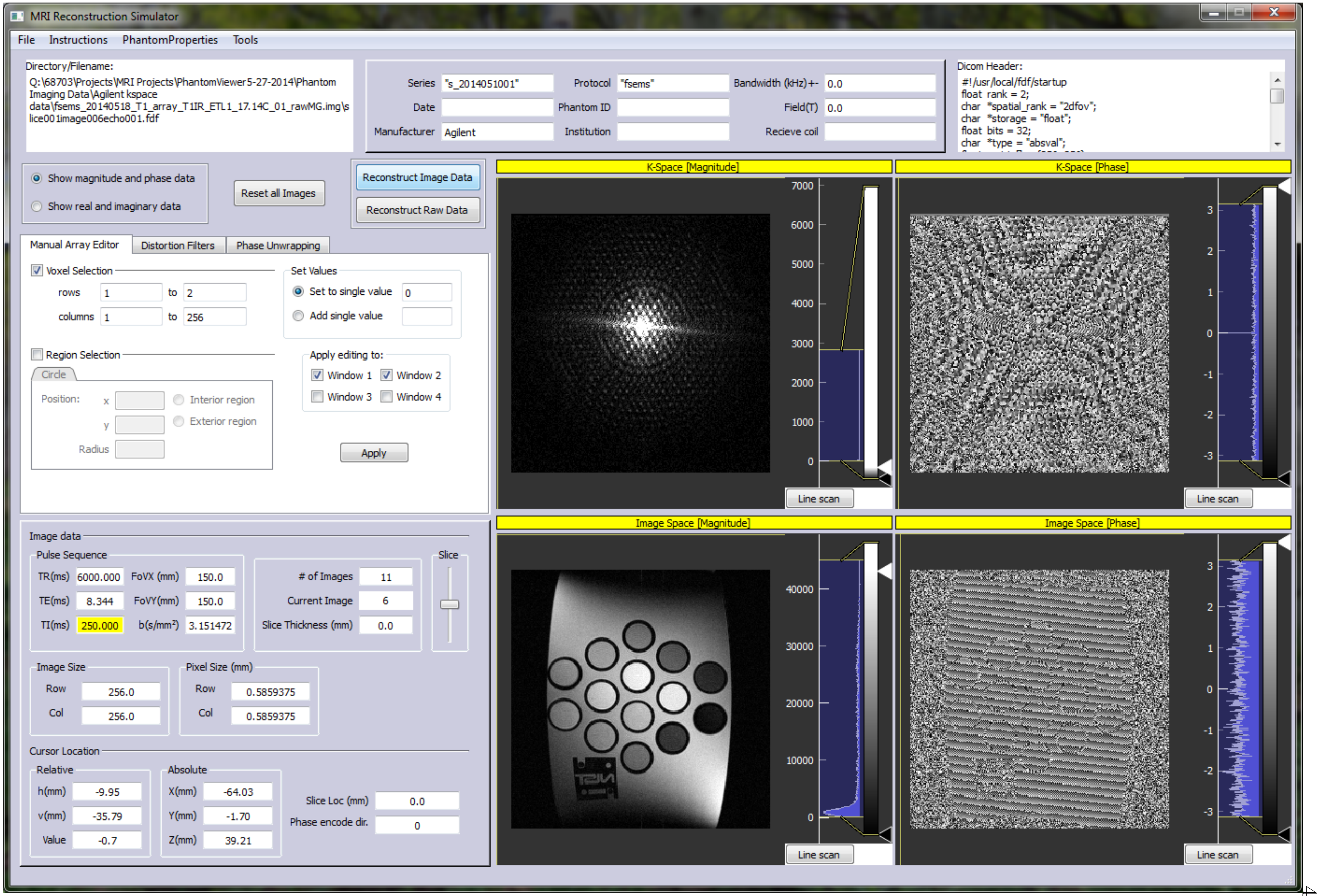
Relative

h(mm)	49.50
v(mm)	40.55
Value	-2.3

Absolute

X(mm)	-123.46
Y(mm)	-3.25
Z(mm)	115.55

Slice Loc (mm)	0.0
Phase encode dir.	0



Directory/Filename:
Q:\68703\Projects\MRI Projects\PhantomViewer5-27-2014\Phantom Imaging Data\Agilent kspace data\fsems_20140518_T1_array_T1IR_ETL1_17.14C_01_rawMG.img\slice001image006echo001.fdf

Series "s_2014051001"
Protocol "fsems"
Bandwidth (kHz)+- 0.0
Date
Phantom ID
Field(T) 0.0
Manufacturer Agilent
Institution
Recieve coil

Dicom Header:

```
#!/usr/local/fdf/startup
float  rank = 2;
char  *spatial_rank = "2dfov";
char  *storage = "float";
float  bits = 32;
char  *type = "absval";
```

● Show magnitude and phase data
○ Show real and imaginary data

☑ Voxel Selection
rows 1 to 2
columns 1 to 256

Set Values
● Set to single value 0
○ Add single value

☐ Region Selection
Circle
Position: x
y
Radius
○ Interior region
○ Exterior region

Apply editing to:
☑ Window 1 ☑ Window 2
☐ Window 3 ☐ Window 4

Image data

Pulse Sequence
TR(ms) 6000.000 FoVX (mm) 150.0
TE(ms) 8.344 FoVY(mm) 150.0
TI(ms) 250.000 b(s/mm²) 3.151472

# of Images 11
Current Image 6
Slice Thickness (mm) 0.0

Image Size
Row 256.0
Col 256.0

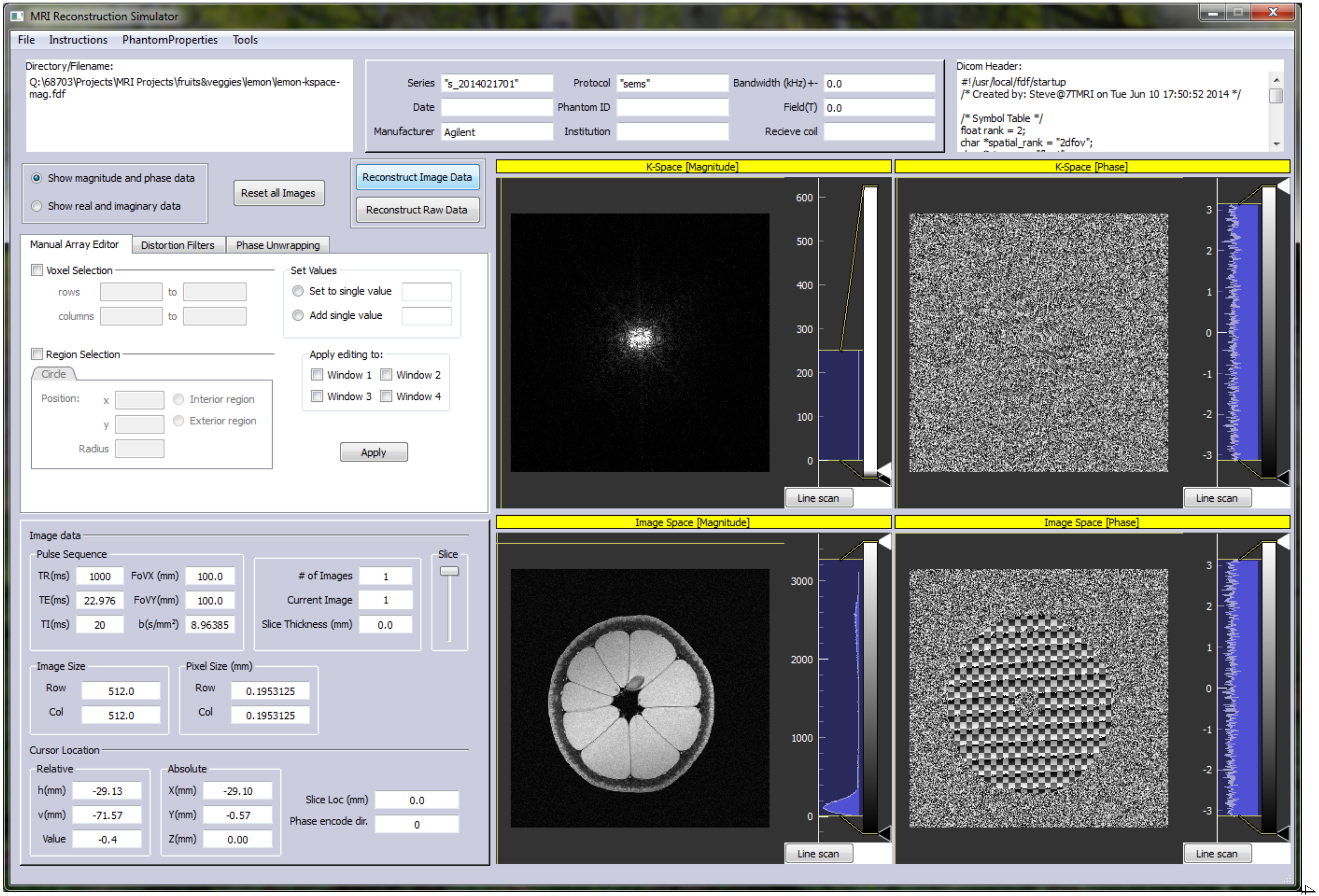
Pixel Size (mm)
Row 0.5859375
Col 0.5859375

Cursor Location

Relative
h(mm) -9.95
v(mm) -35.79
Value -0.7

Absolute
X(mm) -64.03
Y(mm) -1.70
Z(mm) 39.21

Slice Loc (mm) 0.0
Phase encode dir. 0



MRI Reconstruction Simulator

Directory/Filename:
Q:\68703\Projects\MRI Projects\fruits&veggies\lemon\lemon-kspace-mag.fdf

Series "s_2014021701"
Date
Manufacturer Agilent
Protocol "sems"
Phantom ID
Institution
Bandwidth (kHz)+- 0.0
Field(T) 0.0
Recieve coil

Dicom Header:

```
#!/usr/local/fdf/startup
/* Created by: Steve@7TMRI on Tue Jun 10 17:50:52 2014 */

/* Symbol Table */
float rank = 2;
char *spatial_rank = "2dfov";
```

● Show magnitude and phase data
○ Show real and imaginary data

Manual Array Editor | Distortion Filters | Phase Unwrapping

□ Voxel Selection: rows ___ to ___; columns ___ to ___

Set Values: ○ Set to single value ○ Add single value

□ Region Selection: Circle; Position: x ___ y ___; Radius ___; ○ Interior region ○ Exterior region

Apply editing to: □ Window 1 □ Window 2 □ Window 3 □ Window 4

Image data

Pulse Sequence

TR(ms)	1000	FoVX (mm)	100.0
TE(ms)	22.976	FoVY(mm)	100.0
TI(ms)	20	b(s/mm²)	8.96385

# of Images	1
Current Image	1
Slice Thickness (mm)	0.0

Image Size

Row	512.0
Col	512.0

Pixel Size (mm)

Row	0.1953125
Col	0.1953125

Cursor Location

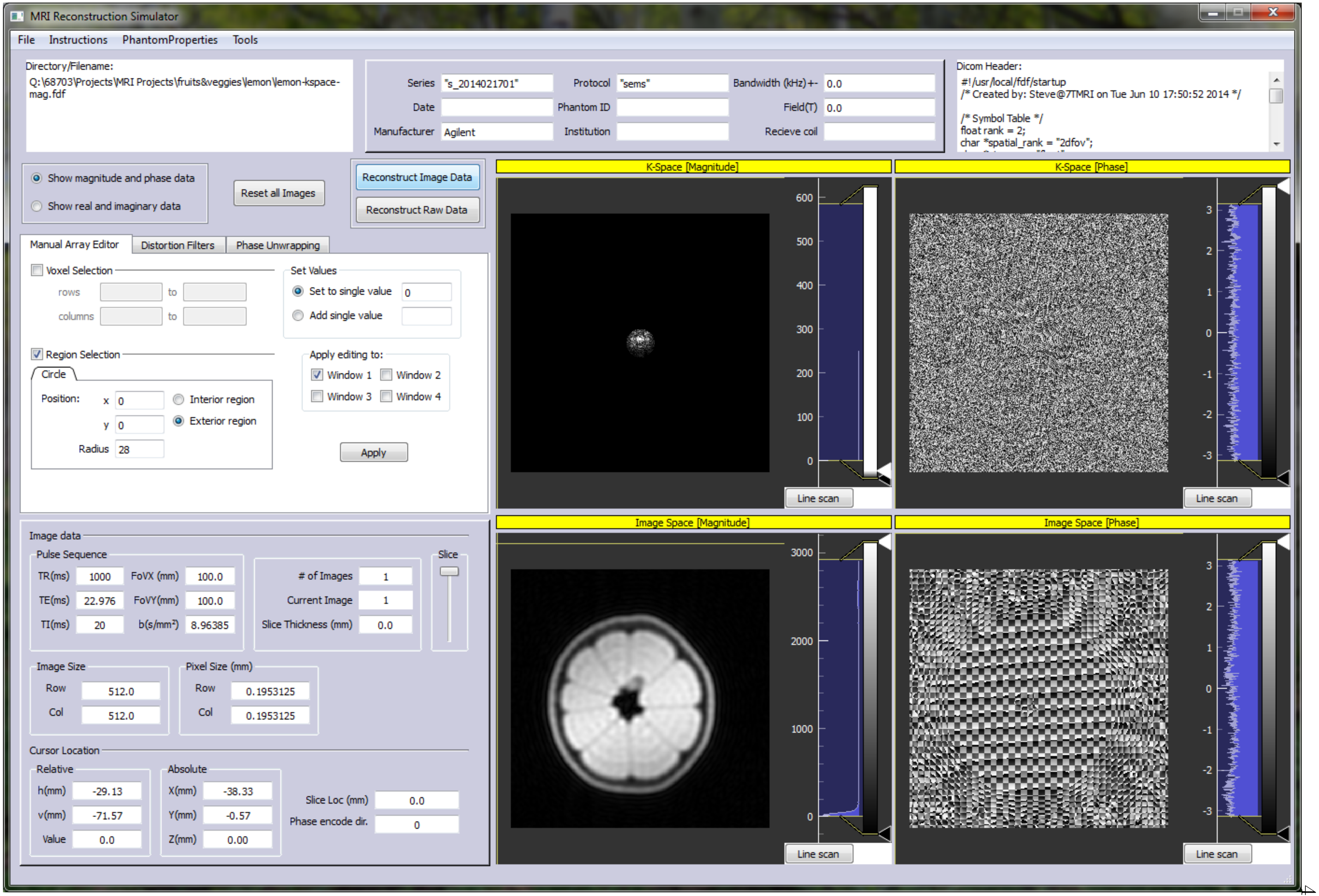
Relative

h(mm)	-29.13
v(mm)	-71.57
Value	-0.4

Absolute

X(mm)	-29.10
Y(mm)	-0.57
Z(mm)	0.00

Slice Loc (mm)	0.0
Phase encode dir.	0



Directory/Filename:
Q:\68703\Projects\MRI Projects\fruits&veggies\lemon\lemon-kspace-mag.fdf

Series "s_2014021701"
Protocol "sems"
Bandwidth (kHz)+- 0.0
Date
Phantom ID
Field(T) 0.0
Manufacturer Agilent
Institution
Recieve coil

Dicom Header:

```
#!/usr/local/fdf/startup
/* Created by: Steve@7TMRI on Tue Jun 10 17:50:52 2014 */

/* Symbol Table */
float rank = 2;
char *spatial_rank = "2dfov";
```

● Show magnitude and phase data
○ Show real and imaginary data

□ Voxel Selection
rows ___ to ___
columns ___ to ___

Set Values
● Set to single value 0
○ Add single value

☑ Region Selection
Circle
Position: x 0
y 0
Radius 28
○ Interior region
● Exterior region

Apply editing to:
☑ Window 1 □ Window 2
□ Window 3 □ Window 4

Image data

Pulse Sequence

TR(ms)	1000	FoVX (mm)	100.0
TE(ms)	22.976	FoVY(mm)	100.0
TI(ms)	20	b(s/mm²)	8.96385

# of Images	1
Current Image	1
Slice Thickness (mm)	0.0

Slice

Image Size

Row	512.0
Col	512.0

Pixel Size (mm)

Row	0.1953125
Col	0.1953125

Cursor Location

Relative

h(mm)	-29.13
v(mm)	-71.57
Value	0.0

Absolute

X(mm)	-38.33
Y(mm)	-0.57
Z(mm)	0.00

Slice Loc (mm)	0.0
Phase encode dir.	0

Appendix C

PhantomViewer Code snippets

```
# -*- coding: utf-8 -*-
"""
Created on Tue Jun 03 10:56:26 2014
Class to reconstruct and manipulate phantom data
@author: Hannah Erdevig
"""
import sys
import os    #operating system file/directory names
from PyQt4 import QtGui, QtCore
from ReconGUI05 import Ui_ReconGUI  # GUI module
from ImageList import ImageList     # file import and export helper module
import numpy as np
import scipy
import pyqtgraph as pg
import pyqtgraph.opengl as gl
import pyqtgraph.functions as fn

class Recon(QtGui.QMainWindow):
    def __init__(self , parent = None):
        super(Recon, self).__init__()
        pg.setConfigOption(’background’, 0.2)   #Background on plots 0 = black, 1 = white
        pg.setConfigOption(’foreground’, ’w’)
        self.ui = Ui_ReconGUI()
        self.ui.setupUi(self)
        self.dataSetIsNew = False
    #window 1
        self.imv1 = self.ui.widget_k1
        #self.imv1.getView().setLabel(’bottom’,"H","mm")    # labels that keep the window from sizing properly
        #self.imv1.getView().setLabel(’left’,"V","mm")
        self.imv1.ui.normBtn.hide()
        self.imv1.ui.roiBtn.setText("Line scan")
        self.imv1.vLine = pg.InfiniteLine(pos=None, angle=90, pen=None, movable=False, bounds=None)   #cross hairs
        self.imv1.hLine = pg.InfiniteLine(pos=None, angle=0, pen=None, movable=False, bounds=None)
        self.imv1.addItem(self.imv1.vLine, ignoreBounds=True)
        self.imv1.addItem(self.imv1.hLine, ignoreBounds=True)
        if self.dataSetIsNew == False:
            self.proxy = pg.SignalProxy(self.imv1.view.scene().sigMouseMoved, rateLimit=60, slot=self.mouseMoved)
    #window 2
        self.imv2 = self.ui.widget_k2
        #self.imv2.getView().setLabel(’bottom’,"H","mm")
        #self.imv2.getView().setLabel(’left’,"V","mm")
        self.imv2.ui.normBtn.hide()
        self.imv2.ui.roiBtn.setText("Line scan")
        self.imv2.vLine = pg.InfiniteLine(pos=None, angle=90, pen=None, movable=False, bounds=None)
        self.imv2.hLine = pg.InfiniteLine(pos=None, angle=0, pen=None, movable=False, bounds=None)
        self.imv2.addItem(self.imv2.vLine, ignoreBounds=True)
        self.imv2.addItem(self.imv2.hLine, ignoreBounds=True)
        if self.dataSetIsNew == False:
            self.proxy2 = pg.SignalProxy(self.imv2.view.scene().sigMouseMoved, rateLimit=60, slot=self.mouseMoved2)
    #window 3
        self.imv3 = self.ui.widget_imag
        #self.imv3.getView().setLabel('bottom',"H","mm")
        #self.imv3.getView().setLabel('left',"V","mm")
        self.imv3.ui.normBtn.hide()
        self.imv3.ui.roiBtn.setText("Line scan")
        self.imv3.vLine = pg.InfiniteLine(pos=None, angle=90, pen=None, movable=False, bounds=None)
        self.imv3.hLine = pg.InfiniteLine(pos=None, angle=0, pen=None, movable=False, bounds=None)
        self.imv3.addItem(self.imv3.vLine, ignoreBounds=True)
        self.imv3.addItem(self.imv3.hLine, ignoreBounds=True)
        if self.dataSetIsNew == False:
            self.proxy3 = pg.SignalProxy(self.imv3.view.scene().sigMouseMoved, rateLimit=60, slot=self.mouseMoved3)
    #window 4
        self.imv4=self.ui.widget_iphase
        #self.imv4.getView().setLabel('bottom',"H","mm")
        #self.imv4.getView().setLabel('left',"V","mm")
        self.imv4.ui.normBtn.hide()
        self.imv4.ui.roiBtn.setText("Line scan")
        self.imv4.vLine = pg.InfiniteLine(pos=None, angle=90, pen=None, movable=False, bounds=None)
        self.imv4.hLine = pg.InfiniteLine(pos=None, angle=0, pen=None, movable=False, bounds=None)
        self.imv4.addItem(self.imv4.vLine, ignoreBounds=True)
        self.imv4.addItem(self.imv4.hLine, ignoreBounds=True)
        if self.dataSetIsNew == False:
            self.proxy4 = pg.SignalProxy(self.imv4.view.scene().sigMouseMoved, rateLimit=60, slot=self.mouseMoved4)

        self.nImages = 0
        self.nCurrentImage = 0
        self.dicomHeader = "DICOM Header"
        self.ui.lineEdit_nimages.setText((str(self.nImages)))
        self.ui.label.setText("none")
        self.dsRe = ImageList()    # Use ImageList.py to create list of image data sets
        self.dsIm = ImageList()
        self.dsMg = ImageList()
        self.dsPh = ImageList()
        self.dsOriginalComplex = ImageList()
        self.dsComplex = ImageList()
        self.dsComplexImage = ImageList()
        self.dsImageMag = ImageList()
        self.dsImagePhase = ImageList()
        self.seriesFileNames = []
        self.windows = [0,0,0,0]
        self.dataSet = [0,0,0,0]
    #signals and slots
#         self.ui.actionNew.triggered.connect(self.NewFile)
        self.ui.actionOpenRI.triggered.connect(self.OpenFileReIm)
        self.ui.actionOpenMP.triggered.connect(self.OpenFileMgPh)
        self.ui.actionSave12.triggered.connect(self.writeDicomFiles12)
        self.ui.actionSave34.triggered.connect(self.writeDicomFiles34)
        self.ui.actionClear.triggered.connect(self.ClearImages)
        self.ui.actionDeleteCurrent.triggered.connect(self.deleteCurrentImage)
        self.ui.verticalSlider_slice.valueChanged.connect(self.ImageSlider)
        self.ui.radioButton_ri.clicked.connect(self.SwitchDisplaytoRI)#(self.dsRe, self.dsIm))
        self.ui.radioButton_mp.clicked.connect(self.SwitchDisplaytoMP)#(self.dsMg, self.dsPh))
        self.ui.pushButton_reconstructI.clicked.connect(self.ReconstructImageData)
        self.ui.pushButton_reconstructK.clicked.connect(self.ReconstructRawData)
        self.ui.pushButton_reset.clicked.connect(self.ResetData)
        self.ui.pushButton_apply.clicked.connect(self.EditData)

#     def NewFile (self):
#         self.dataSetIsNew = True
#         self.dsRe = ImageList()    # Use ImageList.py to create list of image data sets
#         self.dsIm = ImageList()
#         self.dsMg = ImageList()
#         self.dsPh = ImageList()
#         self.dsOriginalComplex = ImageList()
#         self.dsComplex = ImageList()
#         self.dsComplexImage = ImageList()
#         self.dsImageMag = ImageList()
#         self.dsImagePhase = ImageList()
#         self.dsMg.PA.append(np.zeros([256,256]))
#         self.dsPh.PA.append(np.zeros([256,256]))
#         self.dsRe.PA.append(np.zeros([256,256]))
#         self.dsIm.PA.append(np.zeros([256,256]))
#         self.dsImageMag.PA.append(np.zeros([256,256]))
#         self.dsImagePhase.PA.append(np.zeros([256,256]))
#         self.dataSet = self.dsMg, self.dsPh, self.dsImageMag, self.dsImagePhase
#         self.windows = [1,1,1,1]
#         self.nImages = 1
#         self.ui.lineEdit_nimages.setText(str(self.nImages))
#         self.ui.verticalSlider_slice.setMinimum(1)       #set slider to go from 1 to the number of images
#         self.ui.verticalSlider_slice.setMaximum(self.nImages)
#         self.nCurrentImage = 1
#         self.ui.verticalSlider_slice.setValue(self.nCurrentImage)
#         self.DisplayCurrentImage(self.imv1, self.dsMg)
#         self.DisplayCurrentImage(self.imv2, self.dsPh)
#         self.DisplayCurrentImage(self.imv3, self.dsImageMag)
#         self.DisplayCurrentImage(self.imv4, self.dsImagePhase)
#         self.ui.label_k1.setText("K-Space [Magnitude]")
#         self.ui.label_k2.setText("K-Space [Phase]")

    def OpenFileReIm (self):
        self.dataSetIsNew = False
        self.dsRe = ImageList()    # Use ImageList.py to create list of image data sets
        self.dsIm = ImageList()
        self.dsMg = ImageList()
        self.dsPh = ImageList()
        self.dsOriginalComplex = ImageList()
        self.dsComplex = ImageList()
        self.dsComplexImage = ImageList()
        self.dsImageMag = ImageList()
        self.dsImagePhase = ImageList()

        #REAL FILES
        self.fileNamesRe = QtGui.QFileDialog.getOpenFileNames(self,"Open Real Image Files", "/home/file",
 "Image Files (*.dcm *.DCM *.bmp *.tif *.fdf)")
        if not self.fileNamesRe:  #if cancel is pressed return
            return None
        self.seriesFileNames.extend(self.fileNamesRe)     #concatenate new file list with previous file list
        for i in range(len(self.fileNamesRe)):
            fileName = self.fileNamesRe[i]
            self.dsRe.addFile(fileName)
            self.dsOriginalComplex.addFile(fileName)
            self.dsComplex.addFile(fileName)
            self.dsComplexImage.addFile(fileName)
            self.dsImageMag.addFile(fileName)
            self.dsImagePhase.addFile(fileName)
            self.dsMg.addFile(fileName)
            self.dsPh.addFile(fileName)
        self.windows[0] = 1
        self.dataSet[0] = self.dsRe
        self.nImages=self.nImages+len(self.fileNamesRe)
        self.ui.lineEdit_nimages.setText(str(self.nImages))
        self.ui.verticalSlider_slice.setMinimum(1)       #set slider to go from 1 to the number of images
        self.ui.verticalSlider_slice.setMaximum(self.nImages)
        self.nCurrentImage=1
        self.ui.verticalSlider_slice.setValue(self.nCurrentImage)
        self.DisplayCurrentImage(self.imv1, self.dsRe)
        self.ui.label_k1.setText("K-Space [Real]")

        #IMAGINARY FILES
        self.fileNamesIm = QtGui.QFileDialog.getOpenFileNames(self,"Open Imaginary Image Files", "/home/file",
 "Image Files (*.dcm *.DCM *.bmp *.tif *.fdf)")
        if not self.fileNamesIm:  #if cancel is pressed return
            return None
        self.seriesFileNames.extend(self.fileNamesIm)     #concatenate new file list with previous file list
        for i in range(len(self.fileNamesIm)):
            fileName = self.fileNamesIm[i]
            self.dsIm.addFile(fileName)
        self.windows[1] = 1
        self.dataSet[1] = self.dsIm
        self.ui.lineEdit_nimages.setText(str(self.nImages))
        self.ui.verticalSlider_slice.setMinimum(1)       #set slider to go from 1 to the number of images
        self.ui.verticalSlider_slice.setMaximum(self.nImages)
        self.nCurrentImage=1
        self.ui.verticalSlider_slice.setValue(self.nCurrentImage)
        self.DisplayCurrentImage(self.imv2,self.dsIm)
        self.ui.label_k2.setText("K-Space [Imaginary]")

        #CREATE COMPLEX ARRAY
#         self.dsMg = self.dataSet[0]
#         self.dsPh = self.dataSet[0]
#         self.dsImageMag = self.dataSet[0]
#         self.dsImagePhase = self.dataSet[0]
        for i in range(1, len(self.dsMg.PA)):
            self.dsOriginalComplex.PA[i] = self.dsRe.PA[i] + 1j*self.dsIm.PA[i]
            self.dsComplex.PA[i] = self.dsRe.PA[i] + 1j*self.dsIm.PA[i]
            self.dsMg.PA[i] = np.absolute(self.dsComplex.PA[i])
            self.dsPh.PA[i] = np.angle(self.dsComplex.PA[i])
            self.dsComplexImage.PA[i] = np.fft.fft2(self.dsComplex.PA[i])
            self.dsImageMag.PA[i] = np.absolute(self.dsComplexImage.PA[i])
            self.dsImagePhase.PA[i] = np.angle(self.dsComplexImage.PA[i])

    def OpenFileMgPh (self):
        self.dataSetIsNew = False
        self.dsRe = ImageList()    # Use ImageList.py to create list of image data sets
        self.dsIm = ImageList()
        self.dsMg = ImageList()
        self.dsPh = ImageList()
        self.dsOriginalComplex = ImageList()
        self.dsComplex = ImageList()
        self.dsComplexImage = ImageList()
        self.dsImageMag = ImageList()
        self.dsImagePhase = ImageList()

        #MAGNITUDE FILES
        self.fileNamesMg = QtGui.QFileDialog.getOpenFileNames(self,"Open Magnitude Image Files", "/home/file",
 "Image Files (*.dcm *.DCM *.bmp *.tif *.fdf)")
        if not self.fileNamesMg:  #if cancel is pressed return
            return None
        self.seriesFileNames.extend(self.fileNamesMg)     #concatenate new file list with previous file list
        d1Mg= []  #d1 is 3d data stack for 3d images
        for i in range(len(self.fileNamesMg)):
            fileName = self.fileNamesMg[i]
            self.dsMg.addFile(fileName)
            self.dsOriginalComplex.addFile(fileName)
            self.dsComplex.addFile(fileName)
            self.dsComplexImage.addFile(fileName)
            self.dsImageMag.addFile(fileName)
            self.dsImagePhase.addFile(fileName)
            self.dsRe.addFile(fileName)
            self.dsIm.addFile(fileName)
            d1Mg.append(self.dsMg.PA[i+1])
        self.windows[0] = 1
        self.dataSet[0] = self.dsMg
        self.nImages=self.nImages+len(self.fileNamesMg)
        self.ui.lineEdit_nimages.setText(str(self.nImages))
        self.ui.verticalSlider_slice.setMinimum(1)       #set slider to go from 1 to the number of images
        self.ui.verticalSlider_slice.setMaximum(self.nImages)
        self.nCurrentImage=1
        self.ui.verticalSlider_slice.setValue(self.nCurrentImage)
        self.DisplayCurrentImage(self.imv1,self.dsMg)
        self.ui.label_k1.setText("K-Space [Magnitude]")
        self.image3D = np.dstack(d1Mg)
#         self.msgPrint("image size" + str(d1Mg[0].shape) + "; image3D size" + str(self.image3D.shape)+ os.linesep)

        #PHASE FILES
        self.fileNamesPh = QtGui.QFileDialog.getOpenFileNames(self,"Open Phase Image Files", "/home/file",
 "Image Files (*.dcm *.DCM *.bmp *.tif *.fdf)")
        if not self.fileNamesPh:  #if cancel is pressed return
            return None
        self.seriesFileNames.extend(self.fileNamesPh)     #concatenate new file list with previous file list
        for i in range(len(self.fileNamesPh)):
            fileName = self.fileNamesPh[i]
            self.dsPh.addFile(fileName)
        self.windows[1] = 1
        self.dataSet[1] = self.dsPh
        self.ui.lineEdit_nimages.setText(str(self.nImages))
        self.ui.verticalSlider_slice.setMinimum(1)       #set slider to go from 1 to the number of images
        self.ui.verticalSlider_slice.setMaximum(self.nImages)
        self.nCurrentImage=1    # 0th item is dummy image
        self.ui.verticalSlider_slice.setValue(self.nCurrentImage)
        self.DisplayCurrentImage(self.imv2, self.dsPh)
        self.ui.label_k2.setText("K-Space [Phase]")

        #CREATE COMPLEX ARRAY
        for i in range(1, len(self.dsMg.PA)):
            self.dsOriginalComplex.PA[i] = np.multiply(self.dsMg.PA[i],np.exp(1j*self.dsPh.PA[i]))
            self.dsComplex.PA[i] = np.multiply(self.dsMg.PA[i],np.exp(1j*self.dsPh.PA[i]))
            self.dsRe.PA[i] = self.dsComplex.PA[i].real
            self.dsIm.PA[i] = self.dsComplex.PA[i].imag
            self.dsComplexImage.PA[i] = np.fft.fft2(self.dsComplex.PA[i])
            self.dsComplexImage.PA[i] = np.fft.fftshift(self.dsComplexImage.PA[i])
            self.dsImageMag.PA[i] = np.absolute(self.dsComplexImage.PA[i])
            self.dsImagePhase.PA[i] = np.angle(self.dsComplexImage.PA[i])

    def writeDicomFiles12 (self):
        fileName = QtGui.QFileDialog.getSaveFileName(parent=None, caption="Dicom File Name")
        if not fileName:  #if cancel is pressed return
            return None
 #write current image list in DICOM format to filename+ imagenumber + .dcm
        self.dataSet[0].writeDicomFiles(fileName + "kmag")
        self.dataSet[1].writeDicomFiles(fileName + "kphase")

    def writeDicomFiles34 (self):
        fileName = QtGui.QFileDialog.getSaveFileName(parent=None, caption="Dicom File Name")
        if not fileName:  #if cancel is pressed return
            return None
 #write current image list in DICOM format to filename+ imagenumber + .dcm
        self.dataSet[2].writeDicomFiles(fileName + "rmag")
        self.dataSet[3].writeDicomFiles(fileName + "rphase")

    def EditData (self):
        if self.ui.radioButton_setValue.isChecked():
            val = float(self.ui.lineEdit_setValue.text())
            if self.ui.groupBox_voxelSelection.isChecked(): #if voxel selection is selected
                rstart, rend = int(self.ui.lineEdit_voxelRowStart.text()), int(self.ui.lineEdit_voxelRowEnd.text())
                cstart, cend = int(self.ui.lineEdit_voxelColStart.text()), int(self.ui.lineEdit_voxelColEnd.text())
                if self.ui.checkBox_w1.isChecked():         # only apply editing to window if corresponding checkBox is checked
                    for i in range(1, len(self.dsMg.PA)):   # for each image in the stack (same for all types)
                        for c in range(cstart-1, cend):     # user inputs pixels numbered 1:imagesize
                            for r in range(rstart-1, rend):
                                self.dataSet[0].PA[i][c, r] = val
                    self.DisplayCurrentImage(self.imv1,self.dataSet[0])
                if self.ui.checkBox_w2.isChecked():
                    for i in range(1, len(self.dsMg.PA)):               # for each image in the stack (same for all types)
                        for c in range(cstart-1, cend):                 # user inputs pixels numbered 1:imagesize
                            for r in range(rstart-1, rend):
                                self.dataSet[1].PA[i][c, r] = val
                    self.DisplayCurrentImage(self.imv2,self.dataSet[1])

            if self.ui.groupBox_regionSelection.isChecked():            #if region selection is selected
                xCenter, yCenter, radius = int(self.ui.lineEdit_Xpos.text()), int(self.ui.lineEdit_Ypos.text()),
  int(self.ui.lineEdit_radius.text())
                # selects a circular region in coordinate system where x and y range from [-ImageSize/2, ImageSize/2]
                if self.ui.checkBox_w1.isChecked():       # only apply editing to window if corresponding checkBox is checked
                    for i in range(1, len(self.dataSet[0].PA)):            # iterate through arrays
                        if (self.ui.radioButton_interior.isChecked()):     # voxel is inside circle
                            lx, ly = self.dataSet[0].PA[i].shape
                            X, Y = np.ogrid[0:lx, 0:ly]
                            mask = (X - lx/2-xCenter)**2 + (Y - ly/2+yCenter)**2 <= radius**2
                            self.dataSet[0].PA[i][mask] = val
                        elif (self.ui.radioButton_exterior.isChecked()):    # voxel is outside circle
                            lx, ly = self.dataSet[0].PA[i].shape
                            X, Y = np.ogrid[0:lx, 0:ly]
                            mask = (X - lx/2-xCenter)**2 + (Y - ly/2+yCenter)**2 > radius**2
                            self.dataSet[0].PA[i][mask] = val
```

```
                  self.DisplayCurrentImage(self.imv1,self.dataSet[0])

              if self.ui.checkBox_w2.isChecked():        # only apply editing to window if corresponding checkBox is checked
                  for i in range(1, len(self.dataSet[1].PA)):            # iterate through arrays
                      if (self.ui.radioButton_interior.isChecked()):     # voxel is inside circle
                          lx, ly = self.dataSet[1].PA[i].shape
                          X, Y = np.ogrid[0:lx, 0:ly]
                          mask = (X - lx/2-xCenter)**2 + (Y - ly/2+yCenter)**2 <= radius**2
                          self.dataSet[1].PA[i][mask] = val
                      elif (self.ui.radioButton_exterior.isChecked()):    # voxel is outside circle
                          lx, ly = self.dataSet[1].PA[i].shape
                          X, Y = np.ogrid[0:lx, 0:ly]
                          mask = (X - lx/2-xCenter)**2 + (Y - ly/2+yCenter)**2 > radius**2
                          self.dataSet[1].PA[i][mask] = val
                  self.DisplayCurrentImage(self.imv2,self.dataSet[1])

      if self.ui.radioButton_addValue.isChecked():
          val = float(self.ui.lineEdit_addValue.text())
          if self.ui.groupBox_voxelSelection.isChecked():    #if voxel selection is selected
              rstart, rend = int(self.ui.lineEdit_voxelRowStart.text()), int(self.ui.lineEdit_voxelRowEnd.text())
              cstart, cend = int(self.ui.lineEdit_voxelColStart.text()), int(self.ui.lineEdit_voxelColEnd.text())
              if self.ui.checkBox_w1.isChecked():        # only apply editing to window if corresponding checkBox is checked
                  for i in range(1, len(self.dsMg.PA)):  # for each image in the stack (same for all types)
                      for c in range(cstart-1, cend):    # user inputs pixels numbered 1:imagesize
                          for r in range(rstart-1, rend):
                              self.dataSet[0].PA[i][c, r] += val
                  self.DisplayCurrentImage(self.imv1,self.dataSet[0])
              if self.ui.checkBox_w2.isChecked():
                  for i in range(1, len(self.dsMg.PA)):               # for each image in the stack (same for all types)
                      for c in range(cstart-1, cend):                 # user inputs pixels numbered 1:imagesize
                          for r in range(rstart-1, rend):
                              self.dataSet[1].PA[i][c, r] += val
                  self.DisplayCurrentImage(self.imv2,self.dataSet[1])

          if self.ui.groupBox_regionSelection.isChecked():            #if region selection is selected
              xCenter, yCenter, radius = int(self.ui.lineEdit_Xpos.text()), int(self.ui.lineEdit_Ypos.text()),
int(self.ui.lineEdit_radius.text())
              # selects a circular region in coordinate system where x and y range from [-ImageSize/2, ImageSize/2]
              if self.ui.checkBox_w1.isChecked():         # only apply editing to window if corresponding checkBox is checked
                  for i in range(1, len(self.dataSet[0].PA)):            # iterate through arrays
                      if (self.ui.radioButton_interior.isChecked()):     # voxel is inside circle
                          lx, ly = self.dataSet[0].PA[i].shape
                          X, Y = np.ogrid[0:lx, 0:ly]
                          mask = (X - lx/2-xCenter)**2 + (Y - ly/2+yCenter)**2 <= radius**2
                          self.dataSet[0].PA[i][mask] += val
                      elif (self.ui.radioButton_exterior.isChecked()):    # voxel is outside circle
                          lx, ly = self.dataSet[0].PA[i].shape
                          X, Y = np.ogrid[0:lx, 0:ly]
                          mask = (X - lx/2-xCenter)**2 + (Y - ly/2+yCenter)**2 > radius**2
                          self.dataSet[0].PA[i][mask] += val
                  self.DisplayCurrentImage(self.imv1,self.dataSet[0])

              if self.ui.checkBox_w2.isChecked():        # only apply editing to window if corresponding checkBox is checked
                  for i in range(1, len(self.dataSet[1].PA)):            # iterate through arrays
                      if (self.ui.radioButton_interior.isChecked()):     # voxel is inside circle
                          lx, ly = self.dataSet[1].PA[i].shape
                          X, Y = np.ogrid[0:lx, 0:ly]
                          mask = (X - lx/2-xCenter)**2 + (Y - ly/2+yCenter)**2 <= radius**2
                          self.dataSet[1].PA[i][mask] += val
                      elif (self.ui.radioButton_exterior.isChecked()):    # voxel is outside circle
                          lx, ly = self.dataSet[1].PA[i].shape
                          X, Y = np.ogrid[0:lx, 0:ly]
                          mask = (X - lx/2-xCenter)**2 + (Y - ly/2+yCenter)**2 > radius**2
                          self.dataSet[1].PA[i][mask] += val
                  self.DisplayCurrentImage(self.imv2,self.dataSet[1])

      #RECREATE COMPLEX ARRAY
      if self.dataSet[0] == self.dsMg:
          for i in range(1, len(self.dsMg.PA)):
              self.dsComplex.PA[i] = np.multiply(self.dsMg.PA[i],np.exp(1j*self.dsPh.PA[i]))
              self.dsRe.PA[i] = self.dsComplex.PA[i].real
```

```
                self.dsIm.PA[i] = self.dsComplex.PA[i].imag
                self.dsComplexImage.PA[i] = np.fft.fft2(self.dsComplex.PA[i])
                self.dsComplexImage.PA[i] = np.fft.fftshift(self.dsComplexImage.PA[i])
                self.dsImageMag.PA[i] = np.absolute(self.dsComplexImage.PA[i])
                self.dsImagePhase.PA[i] = np.angle(self.dsComplexImage.PA[i])

        if self.dataSet[0] == self.dsRe:
            for i in range(1, len(self.dsMg.PA)):
                self.dsComplex.PA[i] = self.dsRe.PA[i] + 1j*self.dsIm.PA[i]
                self.dsMg.PA[i] = np.absolute(self.dsComplex.PA[i])
                self.dsPh.PA[i] = np.angle(self.dsComplex.PA[i])
                self.dsComplexImage.PA[i] = np.fft.fft2(self.dsComplex.PA[i])
                self.dsComplexImage.PA[i] = np.fft.fftshift(self.dsComplexImage.PA[i])
                self.dsImageMag.PA[i] = np.absolute(self.dsComplexImage.PA[i])
                self.dsImagePhase.PA[i] = np.angle(self.dsComplexImage.PA[i])

    def ResetData (self):
        for i in range(1, len(self.dsMg.PA)):
            self.dsRe.PA[i] = self.dsOriginalComplex.PA[i].real
            self.dsIm.PA[i] = self.dsOriginalComplex.PA[i].imag
            self.dsMg.PA[i] = np.absolute(self.dsOriginalComplex.PA[i])
            self.dsPh.PA[i] = np.angle(self.dsOriginalComplex.PA[i])
            self.dsComplexImage.PA[i] = np.fft.fft2(self.dsOriginalComplex.PA[i])
            self.dsComplexImage.PA[i] = np.fft.fftshift(self.dsComplexImage.PA[i])
            self.dsImageMag.PA[i] = np.absolute(self.dsComplexImage.PA[i])
            self.dsImagePhase.PA[i] = np.angle(self.dsComplexImage.PA[i])
        if self.ui.radioButton_mp.isChecked():
            self.dataSet[0] = self.dsRe
            self.dataSet[1] = self.dsIm
        if self.ui.radioButton_ri.isChecked():
            self.dataSet[0] = self.dsMg
            self.dataSet[1] = self.dsPh
        self.dataSet[2] = self.dsImageMag
        self.dataSet[3] = self.dsImagePhase
        self.DisplayCurrentImage(self.imv1, self.dataSet[0])
        self.DisplayCurrentImage(self.imv2, self.dataSet[1])
        if self.windows[2] == 1:
            self.DisplayCurrentImage(self.imv3, self.dataSet[2])
        if self.windows[3] == 1:
            self.DisplayCurrentImage(self.imv4, self.dataSet[3])

    def ReconstructImageData (self):
        self.windows[2] = 1
        self.windows[3] = 1
        self.dataSet[2] = self.dsImageMag
        self.dataSet[3] = self.dsImagePhase
        self.DisplayCurrentImage(self.imv3, self.dataSet[2])
        self.DisplayCurrentImage(self.imv4, self.dataSet[3])

    def ReconstructRawData (self):
        for i in range(1, len(type.PA)):
            self.dsComplexImage.PA[i] = np.multiply(self.dsImageMag.PA[i],np.exp(1j*self.dsImagePhase.PA[i]))
            self.dsComplexImage.PA[i] = np.fft.ifft2(self.dsComplex.PA[i])
            self.dsComplexImage.PA[i] = np.fft.fftshift(self.dsComplexImage.PA[i])
            self.dsRe.PA[i] = self.dsComplex.PA[i].real
            self.dsIm.PA[i] = self.dsComplex.PA[i].imag
        self.windows[2] = 1
        self.windows[3] = 1
        self.dataSet[2] = self.dsImageMag
        self.dataSet[3] = self.dsImagePhase
        self.DisplayCurrentImage(self.imv3, self.dataSet[2])
        self.DisplayCurrentImage(self.imv4, self.dataSet[3])

    def SwitchDisplaytoRI (self):#,data1,data2):   # updates windows 1&2 to display current k-space pair
        if (self.windows[0] == 1):
            self.dataSet[0] = self.dsRe
            self.DisplayCurrentImage(self.imv1, self.dsRe)#data1)
            self.ui.label_k1.setText("K-Space [Real]")
        if (self.windows[1] == 1):
            self.dataSet[1] = self.dsIm
            self.DisplayCurrentImage(self.imv2, self.dsIm)#data2
            self.ui.label_k2.setText("K-Space [Imaginary]")

    def SwitchDisplaytoMP (self):#,data1,data2):   # updates windows 1&2 to display current k-space pair
        if (self.windows[0] == 1):
            self.dataSet[0] = self.dsMg
            self.DisplayCurrentImage(self.imv1, self.dsMg)#data1)
            self.ui.label_k1.setText("K-Space [Magnitude]")
        if (self.windows[1] == 1):
            self.dataSet[1] = self.dsPh
            self.DisplayCurrentImage(self.imv2, self.dsPh)#data2
            self.ui.label_k2.setText("K-Space [Phase]")

    def ImageSlider (self):
        self.nCurrentImage = self.ui.verticalSlider_slice.value()
        if (self.windows[0] == 1):
            self.DisplayCurrentImage(self.imv1, self.dataSet[0])
        if (self.windows[1] == 1):
            self.DisplayCurrentImage(self.imv2, self.dataSet[1])
        if (self.windows[2] == 1):
            self.DisplayCurrentImage(self.imv3, self.dataSet[2])
        if (self.windows[3] == 1):
            self.DisplayCurrentImage(self.imv4, self.dataSet[3])

    def DisplayCurrentImage (self,win,dstype):  # display the current image from the data "dstype" in window "win"
        i = self.nCurrentImage
        self.ui.label.setText(str(self.nCurrentImage))
        if self.dataSetIsNew == False:
            self.ui.lineEdit_date.setText(format(dstype.StudyDate[i]))
            self.ui.textEdit_filename.setText(self.seriesFileNames[i-1]) if i > 0 else self.ui.textEdit_filename.setText("")
            self.ui.lineEdit_manufacturer.setText(dstype.Manufacturer[i])
            self.ui.lineEdit_series.setText(dstype.SeriesDescription[i])
            self.ui.lineEdit_institution.setText(dstype.InstitutionName[i])
            self.ui.lineEdit_fieldT.setText(str(dstype.MagneticFieldStrength[i]))
            self.ui.lineEdit_receivecoil.setText(str(dstype.ReceiveCoilName[i]))
            #self.ui.lblPatient.setText(dstype.PatientName[i])
            self.ui.lineEdit_protocol.setText(str(dstype.ProtocolName[i]))
            self.ui.lineEdit_bandwidth.setText(str(dstype.PixelBandwidth[i]))
            self.ui.lineEdit_TE.setStyleSheet("background-color: white") if (self.checkEqual(dstype.TE))
  else self.ui.lineEdit_TE.setStyleSheet("background-color: yellow")
            self.ui.lineEdit_TE.setText(str(dstype.TE[i]))
            self.ui.lineEdit_TR.setStyleSheet("background-color: white") if (self.checkEqual(dstype.TR))
  else self.ui.lineEdit_TR.setStyleSheet("background-color: yellow")
            self.ui.lineEdit_TR.setText(str(dstype.TR[i]))
            self.ui.lineEdit_imagesize_col.setText(str(dstype.Columns[i]))
            self.ui.lineEdit_imagesize_row.setText(str(dstype.Rows[i]))
            self.ui.lineEdit_TI.setStyleSheet("background-color: white") if (self.checkEqual(dstype.TI))
  else self.ui.lineEdit_TI.setStyleSheet("background-color: yellow")
            self.ui.lineEdit_TI.setText(str(dstype.TI[i]))
            self.ui.lineEdit_slicethick.setText(str(dstype.SliceThickness[i]))
            self.ui.lineEdit_sliceloc.setStyleSheet("background-color: white") if (self.checkEqual(dstype.SliceLocation))
  else self.ui.lineEdit_sliceloc.setStyleSheet("background-color: yellow")
            self.ui.lineEdit_sliceloc.setText(str(dstype.SliceLocation[i]))
            self.ui.lineEdit_pixsize_row.setText(str(dstype.PixelSpacingX[i]))
            self.ui.lineEdit_pixsize_col.setText(str(dstype.PixelSpacingY[i]))
    #         self.ui.lblFA.setStyleSheet("background-color: white") if (self.checkEqual(dstype.FA))
  else self.ui.lblFA.setStyleSheet("background-color: yellow")
    #         self.ui.lblFA.setText(str(dstype.FA[i]))
            self.ui.lineEdit_phasedir.setText(str(dstype.InPlanePhaseEncodingDirection[i]))
            self.ui.lineEdit_FoVX.setText(str(dstype.FoVX[i]))
            self.ui.lineEdit_FoVY.setText(str(dstype.FoVY[i]))
            self.ui.lineEdit_b.setText(str(dstype.bValue[i]))
            self.ui.textEdit_header.setText(dstype.header[i])
        data = dstype.PA[i]  #not sure why we need to transpose]
#         xscale = dstype.PixelSpacingX[i] if (dstype.PixelSpacingX[i] > 0.) else 1
#         yscale = dstype.PixelSpacingY[i] if (dstype.PixelSpacingY[i] > 0.) else 1
#         xmin = -dstype.FoVX[i]/2   #set origin to center of image, need to upgrade to set by DICOM tag
#         ymin = -dstype.FoVY[i]/2
#         textEdit_results was lblUpperLeft in the line below
        #self.ui.textEdit_results.setText("UL=" + "{:.1f}".format(dstype.ImagePosition[i][0]) + ","
  + "{:.1f}".format(dstype.ImagePosition[i][1]) + "," + "{:.1f}".format(dstype.ImagePosition[i][2]))
#         setImage(img, autoRange=True, autoLevels=True, levels=None, axes=None, xvals=None, pos=None, scale=None,
 transform=None, autoHistogramRange=True)
        win.setImage(data, autoRange=True, autoLevels=True, autoHistogramRange=True)#pos = (xmin,ymin),
 scale = (xscale,yscale), autoHistogramRange=True)
#         self.imv1.getView().setLabel(’bottom’,self.DirectionLabel(dstype.RowDirection[i]),"mm")
#         self.imv1.getView().setLabel(’left’,self.DirectionLabel(dstype.ColumnDirection[i]),"mm")

    def checkEqual(self, lst):    #returns True if all elemments (except the 0th element) of the list are equal
        return lst[2:] == lst[1:-1]

    def ClearImages (self):  #Deletes all images except default image at index 1
        self.ds = ImageList()                         #list of data sets, can be dicom, tiff, fdf
        self.dsRe = ImageList()    # Uses separate module to create list of image data sets
        self.dsIm = ImageList()
        self.dsMg = ImageList()
        self.dsPh = ImageList()
        self.dsComplex = ImageList()
        self.dsOriginalComplex = ImageList()
        self.dsComplexImage = ImageList()
        self.dsImageMag = ImageList()
        self.dsImagePhase = ImageList()
        del self.seriesFileNames[:]
        self.windows = [0,0,0,0]
        self.nCurrentImage=0
        self.nImages=0
#         self.image3D.zeros[1,1,1]
        self.DisplayCurrentImage(self.imv1, self.ds)
        self.DisplayCurrentImage(self.imv2, self.ds)
        self.DisplayCurrentImage(self.imv3, self.ds)
        self.DisplayCurrentImage(self.imv4, self.ds)
        self.ui.lineEdit_nimages.setText(str(self.nImages))
        self.ui.verticalSlider_slice.setMaximum(0)

    def deleteCurrentImage(self):
        if self.nCurrentImage > 0:
            self.dsRe.deleteImage(self.nCurrentImage)
            self.dsIm.deleteImage(self.nCurrentImage)
            self.dsMg.deleteImage(self.nCurrentImage)
            self.dsPh.deleteImage(self.nCurrentImage)
            self.dsComplex.deleteImage(self.nCurrentImage)
            self.dsOriginalComplex.deleteImage(self.nCurrentImage)
            self.dsComplexImage.deleteImage(self.nCurrentImage)
            self.dsImageMag.deleteImage(self.nCurrentImage)
            self.dsImagePhase.deleteImage(self.nCurrentImage)
            self.nImages -= 1
            self.ui.lineEdit_nimages.setText(str(self.nImages))
            self.ui.verticalSlider_slice.setMinimum(1)       #set slider to go from 1 to the number of images
            self.ui.verticalSlider_slice.setMaximum(self.nImages)
            if self.nImages == 0:
                self.nCurrentImage=0
                self.windows = [0,0,0,0]
                self.ds = ImageList()
                self.DisplayCurrentImage(self.imv1, self.ds)
            else:
                self.nCurrentImage = 1
            self.ui.verticalSlider_slice.setValue(self.nCurrentImage)
            self.DisplayCurrentImage(self.imv1, self.dataSet[0])
            self.DisplayCurrentImage(self.imv2, self.dataSet[1])

    def ViewDicomHeader (self):
        if self.ui.rbViewDicomHeader.isChecked():
            self.ui.textEdit_header.setHidden(False)
            dh = str(self.ds.header[self.nCurrentImage])
            if dh == ’’:
                dh="DICOM Header"
            self.ui.textEdit_header.setText(dh)
        else:
            self.ui.textEdit_header.setHidden(True)

#     def View3d(self):
#         w = gl.GLViewWidget()
#         w.opts[’distance’] = 200
#         w.show()
#         w.setWindowTitle('3D View')
#         g = gl.GLGridItem()
#         g.scale(10, 10, 10)
#         w.addItem(g)
#         data=self.image3D
#         #positive = np.log(np.clip(data, 0, data.max())**2)
#         #negative = np.log(np.clip(-data, 0, -data.min())**2)
#         d2 = np.empty(data.shape + (4,), dtype=np.ubyte)
#         d2[..., 0] = data * (255./data.max())
#         d2[..., 1] = data * (255./data.max())
#         d2[..., 2] = d2[...,1]
#         d2[..., 3] = d2[..., 0]*0.3 + d2[..., 1]*0.3
#         d2[..., 3] = (d2[..., 3].astype(float) / 255.) **2 * 255
#
#         d2[:, 0, 0] = [255,0,0,100]
#         d2[0, :, 0] = [0,255,0,100]
#         d2[0, 0, :] = [0,0,255,100]
#
#         v = gl.GLVolumeItem(d2)
#         v.translate(-128,-128,0)
#         w.addItem(v)
#         ax = gl.GLAxisItem()
#         w.addItem(ax)

    def mouseMoved(self,evt): #mouse move event to move crosshairs and display location and values
        if self.dataSetIsNew == False:
            if (self.windows[0] == 1):
                self.ds = self.dataSet[0]
                pos = evt[0]  ## using signal proxy turns original arguments into a tuple
                if self.imv1.view.sceneBoundingRect().contains(pos):
                    mousePoint = self.imv1.view.mapSceneToView(pos)
                    self.ui.lineEdit_h.setText("{:.2f}".format(mousePoint.x()))
                    self.ui.lineEdit_v.setText("{:.2f}".format(mousePoint.y()))
                    if abs(mousePoint.x()) < self.ds.FoVX[self.nCurrentImage]/2 and abs(mousePoint.y())
      < self.ds.FoVY[self.nCurrentImage]/2:
                        Xindex = int((mousePoint.x()+self.ds.FoVX[self.nCurrentImage]/2)/self.ds.PixelSpacingX[self.nCurrentImage])
   #if self.ds.PixelSpacingX[self.nCurrentImage] > 0. else Xindex = int(mousePoint.x())
                        Yindex = int((mousePoint.y()+self.ds.FoVY[self.nCurrentImage]/2)/self.ds.PixelSpacingY[self.nCurrentImage])
   #if self.ds.PixelSpacingY[self.nCurrentImage] > 0. else Yindex = int(mousePoint.y())
                        value=  self.ds.PA[self.nCurrentImage][Xindex,Yindex]
                        self.ui.lineEdit_value.setText("{:.1f}".format(value))
                        rc= self.ReltoGlobal(mousePoint.x(), mousePoint.y(), self.nCurrentImage, self.ds)
                        self.ui.lineEdit_x.setText("{:.2f}".format(rc[0]))
                        self.ui.lineEdit_y.setText("{:.2f}".format(rc[1]))
                        self.ui.lineEdit_z.setText("{:.2f}".format(rc[2]))
                    self.imv1.vLine.setPos(mousePoint.x())
                    self.imv1.hLine.setPos(mousePoint.y())

    def mouseMoved2(self,evt): #mouse move event to move crosshairs and display location and values
        if self.dataSetIsNew == False:
            if (self.windows[1] == 1):
                self.ds = self.dataSet[1]
                pos = evt[0]  ## using signal proxy turns original arguments into a tuple
                if self.imv2.view.sceneBoundingRect().contains(pos):
                    mousePoint = self.imv2.view.mapSceneToView(pos)
                    self.ui.lineEdit_h.setText("{:.2f}".format(mousePoint.x()))
                    self.ui.lineEdit_v.setText("{:.2f}".format(mousePoint.y()))
                    if abs(mousePoint.x()) < self.ds.FoVX[self.nCurrentImage]/2 and abs(mousePoint.y())
      < self.ds.FoVY[self.nCurrentImage]/2:
                        Xindex = int((mousePoint.x()+self.ds.FoVX[self.nCurrentImage]/2)/self.ds.PixelSpacingX[self.nCurrentImage])
   #if self.ds.PixelSpacingX[self.nCurrentImage] > 0. else Xindex = int(mousePoint.x())
                        Yindex = int((mousePoint.y()+self.ds.FoVY[self.nCurrentImage]/2)/self.ds.PixelSpacingY[self.nCurrentImage])
   #if self.ds.PixelSpacingY[self.nCurrentImage] > 0. else Yindex = int(mousePoint.y())
                        value=  self.ds.PA[self.nCurrentImage][Xindex,Yindex]
                        self.ui.lineEdit_value.setText("{:.1f}".format(value))
                        rc= self.ReltoGlobal(mousePoint.x(), mousePoint.y(), self.nCurrentImage, self.ds)
                        self.ui.lineEdit_x.setText("{:.2f}".format(rc[0]))
                        self.ui.lineEdit_y.setText("{:.2f}".format(rc[1]))
                        self.ui.lineEdit_z.setText("{:.2f}".format(rc[2]))
                    self.imv2.vLine.setPos(mousePoint.x())
                    self.imv2.hLine.setPos(mousePoint.y())

    def mouseMoved3(self,evt): #mouse move event to move crosshairs and display location and values
        if self.dataSetIsNew == False:
            if (self.windows[2] == 1):
                self.ds = self.dataSet[2]
                pos = evt[0]  ## using signal proxy turns original arguments into a tuple
                if self.imv3.view.sceneBoundingRect().contains(pos):
                    mousePoint = self.imv3.view.mapSceneToView(pos)
                    self.ui.lineEdit_h.setText("{:.2f}".format(mousePoint.x()))
                    self.ui.lineEdit_v.setText("{:.2f}".format(mousePoint.y()))
                    if abs(mousePoint.x()) < self.ds.FoVX[self.nCurrentImage]/2 and abs(mousePoint.y())
      < self.ds.FoVY[self.nCurrentImage]/2:
                        Xindex = int((mousePoint.x()+self.ds.FoVX[self.nCurrentImage]/2)/self.ds.PixelSpacingX[self.nCurrentImage])
   #if self.ds.PixelSpacingX[self.nCurrentImage] > 0. else Xindex = int(mousePoint.x())
                        Yindex = int((mousePoint.y()+self.ds.FoVY[self.nCurrentImage]/2)/self.ds.PixelSpacingY[self.nCurrentImage])
   #if self.ds.PixelSpacingY[self.nCurrentImage] > 0. else Yindex = int(mousePoint.y())
                        value=  self.ds.PA[self.nCurrentImage][Xindex,Yindex]
                        self.ui.lineEdit_value.setText("{:.1f}".format(value))
                        rc= self.ReltoGlobal(mousePoint.x(), mousePoint.y(), self.nCurrentImage, self.ds)
                        self.ui.lineEdit_x.setText("{:.2f}".format(rc[0]))
                        self.ui.lineEdit_y.setText("{:.2f}".format(rc[1]))
                        self.ui.lineEdit_z.setText("{:.2f}".format(rc[2]))
                    self.imv3.vLine.setPos(mousePoint.x())
                    self.imv3.hLine.setPos(mousePoint.y())

    def mouseMoved4(self,evt): #mouse move event to move crosshairs and display location and values
        if self.dataSetIsNew == False:
            if (self.windows[3] == 1):
                self.ds = self.dataSet[3]
                pos = evt[0]  ## using signal proxy turns original arguments into a tuple
                if self.imv4.view.sceneBoundingRect().contains(pos):
                    mousePoint = self.imv4.view.mapSceneToView(pos)
                    self.ui.lineEdit_h.setText("{:.2f}".format(mousePoint.x()))
                    self.ui.lineEdit_v.setText("{:.2f}".format(mousePoint.y()))
                    if abs(mousePoint.x()) < self.ds.FoVX[self.nCurrentImage]/2 and abs(mousePoint.y())
      < self.ds.FoVY[self.nCurrentImage]/2:
                        Xindex = int((mousePoint.x()+self.ds.FoVX[self.nCurrentImage]/2)/self.ds.PixelSpacingX[self.nCurrentImage])
   #if self.ds.PixelSpacingX[self.nCurrentImage] > 0. else Xindex = int(mousePoint.x())
                        Yindex = int((mousePoint.y()+self.ds.FoVY[self.nCurrentImage]/2)/self.ds.PixelSpacingY[self.nCurrentImage])
   #if self.ds.PixelSpacingY[self.nCurrentImage] > 0. else Yindex = int(mousePoint.y())
                        value=  self.ds.PA[self.nCurrentImage][Xindex,Yindex]
                        self.ui.lineEdit_value.setText("{:.1f}".format(value))
                        rc= self.ReltoGlobal(mousePoint.x(), mousePoint.y(), self.nCurrentImage, self.ds)
                        self.ui.lineEdit_x.setText("{:.2f}".format(rc[0]))
                        self.ui.lineEdit_y.setText("{:.2f}".format(rc[1]))
                        self.ui.lineEdit_z.setText("{:.2f}".format(rc[2]))
                    self.imv4.vLine.setPos(mousePoint.x())
                    self.imv4.hLine.setPos(mousePoint.y())

    def ReltoGlobal (self, h,v,n, dstype):   #given relative coordinate x,y of image n returns np vector of global coordinates
        rc= ((h+dstype.FoVX[n]/2) * dstype.RowDirection[n]+(v+dstype.FoVX[n]/2)*dstype.ColumnDirection[n])+dstype.ImagePosition[n]
        return rc

# UNUSED FUNCTION
    def GlobaltoRel(self,r,n, dstype):    #Given r vector in global coordinates returns h,v in image plane of image n
        h=np.dot(r-dstype.ImageCenter[n],dstype.RowDirection[n])
        v=np.dot(r-dstype.ImageCenter[n],dstype.ColumnDirection[n])
        return [h,v]

if __name__ == '__main__':
    app = QtGui.QApplication(sys.argv)
    main = Recon()
    main.show()
    sys.exit(app.exec_())
```

Appendix D

AIP Publication

Accuracy of magnetic resonance based susceptibility measurements

Hannah E. Erdevig,[1,2] Stephen E. Russek,[1] Slavka Carnicka,[1] Karl F. Stupic,[1] and Kathryn E. Keenan[1]

[1] *Physical Measurement Laboratory, NIST, Boulder, Colorado 80305, USA*
[2] *Department of Physics, University of Colorado Boulder, Boulder, Colorado 80309, USA*

(Presented 1 November 2016; received 23 September 2016; accepted 10 November 2016; published online 6 February 2017)

Magnetic Resonance Imaging (MRI) is increasingly used to map the magnetic susceptibility of tissue to identify cerebral microbleeds associated with traumatic brain injury and pathological iron deposits associated with neurodegenerative diseases such as Parkinson's and Alzheimer's disease. Accurate measurements of susceptibility are important for determining oxygen and iron content in blood vessels and brain tissue for use in noninvasive clinical diagnosis and treatment assessments. Induced magnetic fields with amplitude on the order of 100 nT, can be detected using MRI phase images. The induced field distributions can then be inverted to obtain quantitative susceptibility maps. The focus of this research was to determine the accuracy of MRI-based susceptibility measurements using simple phantom geometries and to compare the susceptibility measurements with magnetometry measurements where SI-traceable standards are available. The susceptibilities of paramagnetic salt solutions in cylindrical containers were measured as a function of orientation relative to the static MRI field. The observed induced fields as a function of orientation of the cylinder were in good agreement with simple models. The MRI susceptibility measurements were compared with SQUID magnetometry using NIST-traceable standards. MRI can accurately measure relative magnetic susceptibilities while SQUID magnetometry measures absolute magnetic susceptibility. Given the accuracy of moment measurements of tissue mimicking samples, and the need to look at small differences in tissue properties, the use of existing NIST standard reference materials to calibrate MRI reference structures is problematic and better reference materials are required. [http://dx.doi.org/10.1063/1.4975700]

INTRODUCTION

Quantitative Susceptibility Mapping (QSM)[1] using Magnetic Resonance Imaging (MRI) is increasingly used instead of qualitative techniques, such as susceptibility weighted imaging,[2] to map neurological conditions,[3–5] blood oxygen content,[6] and iron overload in the heart and liver.[7] Some neurodegenerative diseases, such as Parkinson's and Alzheimer's disease, have been associated with excess iron in the brain.[8,9] A reproducible and quantitative method to measure blood-oxygen content via QSM is particularly important for finding and determining the severity of cerebral microbleeds resulting from stroke or traumatic brain injury.[10] QSM may be important for measuring iron overload in the heart and liver, caused by diseases such as hemochromatosis, because iron can catalyze the conversion of hydrogen peroxide into free radicals, causing damage to cell membranes, proteins, and DNA.[11] Tissue property measurements using QSM are also advantageous compared to SQUID (superconducting quantum interference device) magnetometry measurements since the latter are done on excised tissue and are inaccurate due to water loss, blood oxidation, and volume changes. However, there is much left to do to validate the accuracy of QSM and of MRI-based susceptibility measurements in general. Accurate in-vivo measurements of magnetic susceptibility, along with the necessary calibrations and post-processing techniques, are required to use magnetic susceptibility as a quantitative biomarker. Creating standard measurement protocols and a phantom with NIST verified susceptibility samples would help ensure site-to-site comparability of data and allow QSM to be more widely and reliably used in clinical applications. In-vivo MRI susceptibility measurements, if done properly, may become the gold standard for tissue susceptibility quantification. The first step is to verify the accuracy of MRI susceptibility measurements relative to other traditional methods.

TISSUE SUSCEPTIBILITY AND TISSUE MIMICS

Tissue is predominantly diamagnetic at body temperature 310 K and room temperature 300 K. This is seen in Fig. 1a, which shows the magnetic moment vs. field for cow liver. The magnetic susceptibility is dominated by the diamagnetic susceptibilities of water (-9.05 x 10^{-6}) and fat (typically -10.0 x 10^{-6}).[12] All susceptibility values in this paper are reported in SI units. The complex magnetic structure of tissue is seen at lower temperatures. Fig. 1a shows a decrease in the diamagnetic (negative) slope as the temperature decreases indicating the presence of a paramagnetic component. At low temperature (1.8 K) there is a deviation in linearity due to paramagnetic and ferrimagnetic components. The presence of a ferrimagnetic component is seen in Fig. 1b, which plots the moment vs. inverse temperature. If there were only a paramagnetic component, the data would be linear. For liver, the paramagnetic and ferrimagnetic components are predominantly due to blood iron in deoxygenated hemoglobin and iron oxide deposits (ferritin).

To mimic the susceptibility properties of tissue, one can use a solution of paramagnetic salts in water. Fig. 1d demonstrates how the diamagnetic susceptibility of water, with minimal temperature-dependence, and a paramagnetic component can roughly approximate the magnetic properties of tissue. We present data from $GdCl_3$ solutions, whose magnetic properties are shown in Fig. 1c,d

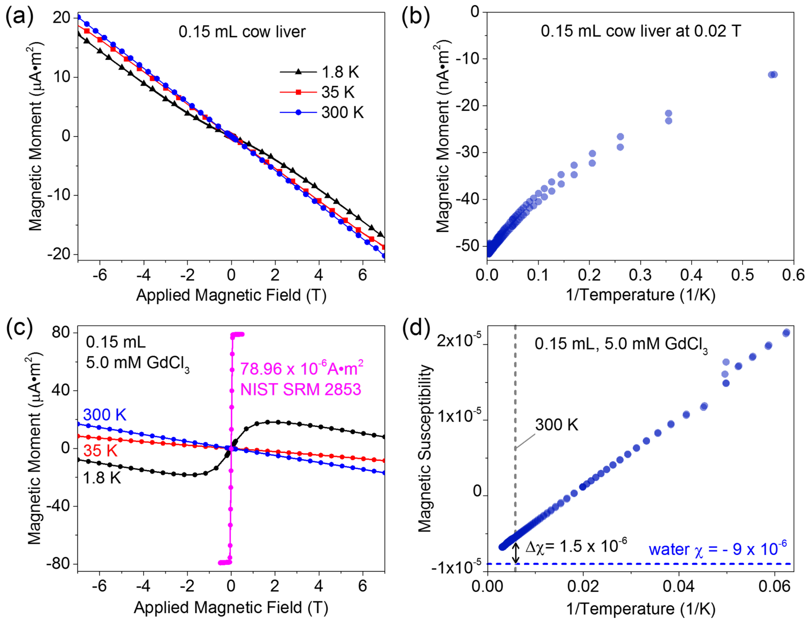


FIG. 1. (a) SQUID magnetometer measurements of magnetic moment vs. applied field for a sample of cow liver. (b) Magnetic moment vs. inverse temperature, upon heating and cooling, of the same sample. (c) SQUID magnetometer measurements of the magnetic moment vs. applied field of the 5.0 mM $GdCl_3$ solution. Also shown is the calibration curve obtained from a NIST moment standard reference material. (d) Magnetic susceptibility vs. inverse temperature for the same solution showing paramagnetic behavior. The horizontal dotted line schematically shows the diamagnetic susceptibility of water. The arrow indicates the susceptibility contribution from the Gd^{3+} ions at 300 K. Comparing the tissue magnetic properties, shown in (a) and (b), to those of the standard Gd solutions, shown in (c) and (d), one can see that the reference solutions are a good starting point to mimic the magnetic properties of tissue, although they lack the full complexity of tissue.

for a 5.0 mM solution in deionized water. The SQUID magnetometer is calibrated with a NIST YIG (yttrium iron garnet) sphere standard reference material (SRM #2852) whose room temperature moment is (79.9 ± 0.3) x 10^{-6} A·m^2. The moment (m) vs. applied field (B_a) data can be fit assuming a paramagnetic component and a diamagnetic component:

$$m = N_{Gd} V g \mu_B J \cdot B_J \left(\frac{gJ\mu_B B_a}{k_B T} \right) - \frac{\chi_w V B_a}{\mu_0} \qquad (1)$$

N_{Gd} is the concentration of Gd^{3+} ions, V is the volume of the sample, g is the Landé g-factor (which is 2.0 for Gd since the angular momentum vanishes), μ_B is the Bohr magneton, J is the ion angular momentum quantum number, B_J is the Brillouin function, k_B is Boltzmann's constant, T is the temperature of the sample, χ_w is the magnitude of the diamagnetic susceptibility of water, and μ_0 is the permeability of free space. The susceptibility due to the Gd^{3+} ions can be calculated from the model (Eq. 1) using the best fit parameters and the measured volume. The measured Gd susceptibility for a 5.0 mM solution at 300 K, shown in Fig. 1d is χ_{Gd} = (1.58 ± 0.16) x 10^{-6}, comparable to the theoretical value of χ_{th} = 1.89 x 10^{-6}. The errors in the measured value come from errors in the moment measurement, the volume measurement and from the extraction of the smaller Gd moment from the larger diamagnetic moment of water. For comparison, the difference in susceptibility between deoxygenated and oxygenated blood, as measured by MRI, is (3.43 ± 0.08) x 10^{-6}.[13]

MRI SUSCEPTIBILITY MEASUREMENTS

MRI susceptibility measurements are typically done by acquiring magnitude and phase data from a gradient echo sequence with multiple echo times. Magnitude and phase images of a phantom are shown in Fig. 2a. The phase image clearly shows distortion of the phase fronts due to the enhanced susceptibility of the paramagnetic salt solution contained within the vial. The imaging was done in a 30 cm bore preclinical scanner designed to image at 1.5 T, 3.0 T, or 7.0 T. The data in this paper were obtained with a static field of B_0 = (1.502102 ± 0.000006) T. The error in the field represents the typical field variation over the active volume with a standard shimming procedure. The phase must be unwrapped and the low-spatial frequency background phase variations subtracted (Fig. 2a). Background phase variations are due to an imperfect shimming of the magnet and to susceptibility discontinuities far from the region of interest.

The difference in proton phase (inside relative to outside the vial), $\delta\phi$, after an echo time, TE, is proportional to the local induced field, δB_L, along the main field direction: $\delta\phi = \gamma_p \cdot \delta B_L \cdot \mathrm{TE}$, where γ_p is the shielded proton gyromagnetic ratio. The local field differs from the macroscopic field and is given by the macroscopic field minus the Lorentz field. The Lorentz field is a correction to the

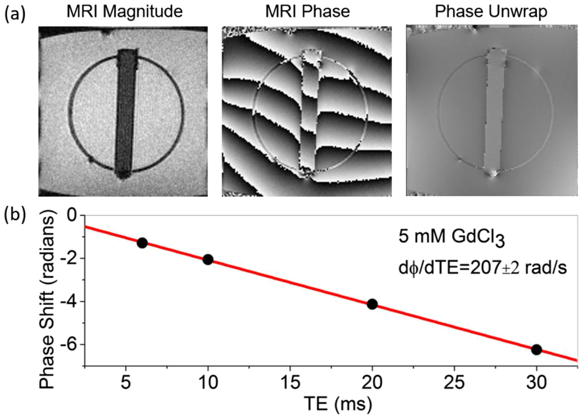


FIG. 2. (a) Magnitude and phase images of a vial containing 5.0 mM GdCl$_3$. The dark circle in the MRI amplitude image is a 76 mm diameter polycarbonate support for the vials. The third image shows the phase after unwrapping and after the long wavelength background has been subtracted. (b) Phase difference as a function of echo time (TE) taken from phase maps.

macroscopic continuum model and attempts to account for the local microscopic distribution of moments. The slope of the measured phase difference vs. echo time, as shown in Fig. 2b, will yield δB_L. The magnetic field distortion is a convolution of the magnetic susceptibility distribution, $\chi(r)$, with the magnetic dipole kernel, d(r): $\delta B_L(\vec{r}) = d(\vec{r}) \otimes \chi(\vec{r})$.[14] The susceptibility map can be obtained by inverting the field profile, although complex methods are required since this inversion is not unique.[15–18] Here, we limit our measurements to simple cylindrical geometries where the induced field is simply related to the susceptibility. For a long cylinder the internal and external field distortion is given by[19]

$$\text{Internal}: \quad \delta B_L = \frac{\Delta\chi B_0}{6}(3\cos^2\theta - 1) \tag{2a}$$

$$\text{External}: \quad \delta B_L = \frac{\Delta\chi B_0}{2} a^2/r^2 \sin^2\theta cos2\phi \tag{2b}$$

where $\Delta\chi$ is the susceptibility difference between the inside and outside of the cylinder, θ is the angle of the cylinder axis with respect to the main field, ϕ is the azimuthal angle of the observation point relative to the plane of the main field and cylinder axis, and a is the radius of the cylinder. For the simple case where the cylinder is aligned with the main field ($\theta = 0$), the susceptibility difference is given by $\Delta\chi = \frac{3\delta\phi}{\gamma_p B_0 TE}$. By measuring the slope of $\delta\phi$ vs. TE, as seen in Fig. 2b, the susceptibility can be determined. The susceptibility difference of the 5.0 mM $GdCl_3$ solution at 300 K, was (1.71 ± 0.02) x 10^{-6}, which, within error bars, agrees with the SQUID magnetometer measurements. The intrinsic errors for the SQUID measurements are larger than the MRI measurements, although the systematic errors for the MRI measurements have not yet been determined.

ANGLE DEPENDENT MEASUREMENTS

To test the orientational dependence, MRI phase maps were obtained from a phantom with vials (80 mm long, 5.0 mL volume) oriented along and perpendicular to the B_0 field. The vials were filled with 5.0 mM $GdCl_3$; the main compartment of the phantom was filled with deionized water. Line scans through the cylinders are shown in Fig. 3a along with the predicted phase change and induced fields obtained from Eq. 2a,b. Good agreement is observed, although there is some deviation at the edges of the vials, in part due to the loss of signal from the plastic vial.

To more precisely verify the orientation dependence, a rotating phantom was constructed in which the 80 mm vials could be continuously rotated while in the MRI scanner. A schematic of the rotating phantom is shown in the inset in Fig. 3b. Four 80 mm vials filled with 1.0 mM and 5.0 mM $GdCl_3$ solutions were placed in the scanner. A rod extended from the outside of the scanner to the internal rotation gears; each revolution corresponded to 19° mechanical rotation of the phantom

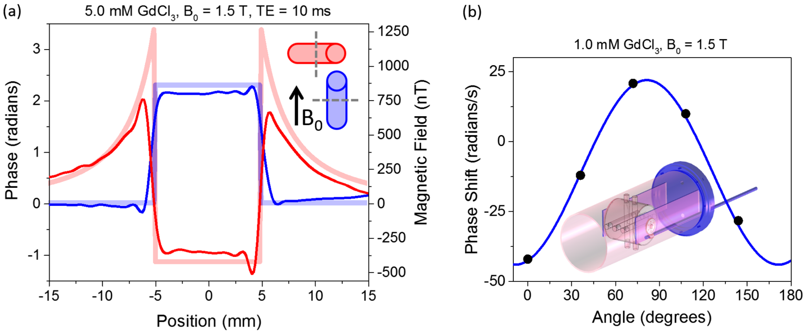


FIG. 3. (a) Line scans (opaque lines) of phase and corresponding field distortions taken with the field parallel (blue) and perpendicular (red) to the cylinder axis. When the field was perpendicular to the cylinder axis, the line scan was taken along B_0 (ϕ = 0). Also shown are the predicted phase shifts (lighter lines) from Eq. 2. (b) Plot of the change of phase with echo time within a cylinder of 1.0 mM $GdCl_3$ as a function of angle of the cylinder axis relative to the B_0 field. Also plotted is a fit using Eq. 2a (blue line). The inset a schematic of the rotating phantom used for the experiment.

insert. The change of phase between the center of each vial and the surrounding water was collected as a function of angle (Fig. 3b). The data were fit using Eq. 2a, yielding $\Delta\chi$ = (0.324 ± 0.005) x 10^{-6} for the 1.0 mM solution.

BEYOND THE SIMPLE MODELS

A multiphysics finite element simulation with a package for modeling magnetic fields without currents was used to compute the macroscopic field of the five perpendicular vials, shown in the inset of Fig. 3b. The vials were filled with a solution with a magnetic susceptibility of 3.0 x 10^{-6} relative to the surrounding water. The numerical accuracy of the field distortion was estimated to be ±7% by varying degrees of freedom from 2 to 5 million. Finite element calculations of extremely small field perturbations on a very large B_0 field gave significant numerical errors. Fig. 4a,b show the field distortions when the B_0 field is parallel and perpendicular to the vial axes, respectively. The field profiles within the vials are not constant, as predicted by the simple models, due to the fields from neighboring vials, the finite length of the vials, and the phantom structure. Determining the local susceptibility from the full inversion of the 3-dimensional phase map should account for these distortions.

One of the main approximations in MRI-based susceptibility measurements is to assume that the local field is given by the macroscopic field, B_m, minus the Lorentz field: $B_L = B_m - \frac{2}{3}\chi B_0$. This assumes that the local microscopic fields average to zero. To determine the local field, precise microscopic calculations are needed. As a simple test, we performed a Monte Carlo simulation where 2.5 x 10^6 Gd spins were randomly distributed in 2 μm diameter sphere and 300 water molecules were allowed to diffuse throughout the volume. The fields sensed by the water molecules after a time of 0.15 ms are plotted in Fig. 4c. The Gd density corresponds to 1.0 mM concentration and an MRI-measured susceptibility of 0.32 x 10^{-6}. The microscopic field calculated from the simulation is 13.5 nT, which is much smaller than the Lorentz field B_L = 320 nT. The simulation supports the assumption that the microscopic fields due to neighboring spins average to zero, and the local field approximation is valid. For tissues, which may have more complex local geometry, this local field assumption may not be valid.

The Monte Carlo simulation gave a Gaussian distribution in microscopic fields, which had a standard deviation of 249 nT. This field distribution gives rise to a short total dephasing time T2*. The T2* value can be measured with the same data set as the susceptibility using the magnitude images and extracting the exponential decrease in the magnitude signal with echo time TE. The T2* value can be used to obtain measurements of the local iron concentration in tissue.[8] While the

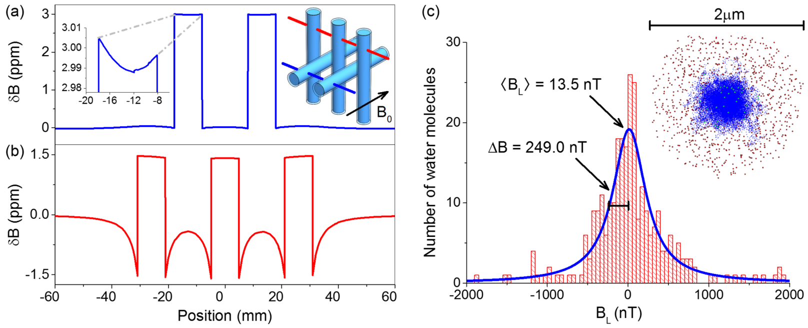


FIG. 4. Numerical calculations of the field distortions produced by the phantom shown in the inset in Fig. 3b, with five vials of paramagnetic salt solution with a susceptibility of 3.0 x 10^{-6}. The macroscopic field distribution is plotted, not the local field, since the macroscopic field is what is calculated using the macroscopic Maxwell equations. (a) The field distortion calculated by a finite element method when the vial axis is parallel to B_0 field. The inset graph shows the variation within the vial due to neighboring vials and structures. (b) The field distortion when the B_0 field is perpendicular to the axis of the vial and the line scan is taken perpendicular to both B_0 field and the vial axis. (c) Monte Carlo simulation generated histogram of microscopic fields experienced by an ensemble of water molecules diffusing (with a diffusion constant of 2.0 x 10^{-3} mm^2/s) in a 1.0 mM Gd solution. The geometry is shown in the inset with the red and blue dots representing Gd^{3+} ions and water, respectively.

decrease in T2* and the change in phase both arise, in the system studied here, from the Gd spins, T2* is strongly affected by the local microscopic structure while the phase shift is not.

CONCLUSIONS

The relative phase shifts and local induced magnetic fields can be measured very precisely with MRI. The relative susceptibilities can be accurately determined from these field shifts for simple geometries and agree with primary measurements of susceptibility where standards exist. More suitable primary standards, however, will be required to validate MRI susceptibility measurements in complex geometries. More extensive investigation into how the local field depends on microscopic tissue geometry is required to determine the accuracy of local field models.

[1] Y. Wang and T. Liu, Magn Reson Med **73**(1), 82–101 (2015).

[2] E. M. Haacke, S. Mittal, Z. Wu, J. Neelavalli, and Y. C. Cheng, AJNR Am J Neuroradiol **30**(1), 19–30 (2009).

[3] J. H. Jensen, K. Szulc, C. Hu, A. Ramani, H. Lu, L. Xuan, M. F. Falangola, R. Chandra, E. A. Knopp, J. Schenck, E. A. Zimmerman, and J. A. Helpern, Magn Reson Med **61**(2), 481–485 (2009).

[4] K. Shmueli, J. A. de Zwart, P. van Gelderen, T. Q. Li, S. J. Dodd, and J. H. Duyn, Magn Reson Med **62**(6), 1510–1522 (2009).

[5] H. Tan, T. Liu, Y. Wu, J. Thacker, R. Shenkar, A. G. Mikati, C. Shi, C. Dykstra, Y. Wang, P. V. Prasad, R. R. Edelman, and I. A. Awad, Invest Radiol **49**(7), 498–504 (2014).

[6] E. M. Haacke, K. Prabhakaran, I. R. Elangovan, Z. Wu, and J. Neelavalli, in *Susceptibility Weighted Imaging in MRI* (John Wiley & Sons, Inc., 2011), pp. 517–528.

[7] S. D. Sharma, D. Hernando, D. E. Horng, and S. B. Reeder, Magnetic Resonance in Medicine **74**(3), 673–683 (2015).

[8] F. Mitsumori, H. Watanabe, and N. Takaya, Magn Reson Med **62**(5), 1326–1330 (2009).

[9] S. Oshiro, M. S. Morioka, and M. Kikuchi, Adv Pharmacol Sci **2011**, 378278 (2011).

[10] T. Liu, K. Surapaneni, M. Lou, L. Cheng, P. Spincemaille, and Y. Wang, Radiology **262**(1), 269–278 (2012).

[11] N. C. Andrews, New England Journal of Medicine **341**(26), 1986–1995 (1999).

[12] J. F. Schenck, Medical Physics **23**(6), 815–850 (1996).

[13] V. Jain, O. Abdulmalik, K. J. Propert, and F. W. Wehrli, Magn Reson Med **68**(3), 863–867 (2012).

[14] F. Schweser, A. Deistung, B. W. Lehr, and J. R. Reichenbach, Med Phys **37**(10), 5165–5178 (2010).

[15] T. Liu, J. Liu, L. de Rochefort, P. Spincemaille, I. Khalidov, J. R. Ledoux, and Y. Wang, Magn Reson Med **66**(3), 777–783 (2011).

[16] L. de Rochefort, R. Brown, M. R. Prince, and Y. Wang, Magn Reson Med **60**(4), 1003–1009 (2008).

[17] T. Liu, P. Spincemaille, L. de Rochefort, B. Kressler, and Y. Wang, Magn Reson Med **61**(1), 196–204 (2009).

[18] Y. C. Cheng, J. Neelavalli, and E. M. Haacke, Phys Med Biol **54**(5), 1169–1189 (2009).

[19] J. Neelavalli and Y.-C. N. Cheng, in *Susceptibility Weighted Imaging in MRI* (John Wiley & Sons, Inc., 2011), pp. 17–31.