

12-2016

Novel Simulation to Avoid Bias in Measurement of Hyperpolarized Pyruvate: Demonstrated in Phantom and In Vivo

Christopher M. Walker

Follow this and additional works at: https://digitalcommons.library.tmc.edu/utgsbs_dissertations



Part of the [Analytical, Diagnostic and Therapeutic Techniques and Equipment Commons](#), and the [Other Physics Commons](#)

Recommended Citation

Walker, Christopher M., "Novel Simulation to Avoid Bias in Measurement of Hyperpolarized Pyruvate: Demonstrated in Phantom and In Vivo" (2016). *The University of Texas MD Anderson Cancer Center UTHealth Graduate School of Biomedical Sciences Dissertations and Theses (Open Access)*. 712.
https://digitalcommons.library.tmc.edu/utgsbs_dissertations/712

This Dissertation (PhD) is brought to you for free and open access by the The University of Texas MD Anderson Cancer Center UTHealth Graduate School of Biomedical Sciences at DigitalCommons@TMC. It has been accepted for inclusion in The University of Texas MD Anderson Cancer Center UTHealth Graduate School of Biomedical Sciences Dissertations and Theses (Open Access) by an authorized administrator of DigitalCommons@TMC. For more information, please contact digitalcommons@library.tmc.edu.

Novel Simulation to Avoid Bias in Measurement of Hyperpolarized Pyruvate: Demonstrated in Phantom and *In Vivo*

by
Christopher Michael Walker

APPROVED:

James A. Bankson, Ph.D. (Advisory Professor)

John Hazle Ph.D.

Dawid Schellingerhout M.D.

Richard Wendt Ph.D.

Steven Millward Ph.D.

Arvind Rao Ph. D.

APPROVED:

Dean, The University of Texas
Graduate School of Biomedical Sciences at Houston

**NOVEL SIMULATION TO AVOID BIAS IN MEASUREMENT OF HYPERPOLARIZED PYRUVATE:
DEMONSTRATED IN PHANTOM AND *IN VIVO***

A

DISSERTATION

Presented to the Faculty of
The University of Texas
Health Science Center at Houston
And

The University of Texas M.D. Anderson Cancer Center
Graduate School of Biomedical Sciences
in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

by

Christopher M. Walker
B.Sc. Trinity University, 2010
Houston, Texas

December 2016

Copyright

Dedication

To my mother and father, Cheryl and Mike Walker, and my sister, Ashley Walker, for their ceaseless support and encouragement.

Acknowledgment

I would like to thank my adviser, Dr. James A. Bankson, for his patience and dedication to my development as an academician. Under his tutelage I have grown tremendously in the way I think about problems and how I present my thoughts, especially using the written word. No one would have predicted that the dyslexic 4th grader would one day write a dissertation, and my thanks to Dr. Bankson is paramount in the achievement of that goal.

I would also like to thank my committee members for their time and dedication to my project and my development throughout all stages of my post-graduate education. Dr. Hazle, who first encouraged me to continue to explore magnetic resonance imaging as a summer student who could barely wrap his head around frequency encoding gradients. I would not be here today if not for his initial support which, much to my benefit, has followed me throughout my graduate training. Dr. Wendt, to whom I owe the bulk of my understanding of the phenomenon of magnetic resonance, thanks to his unflagging didactic efforts. If Haacke is the magnetic resonance bible, then Dr. Wendt's lecture is the MRI sermon on the mount. Statistical analysis is fraught with pitfalls and I have my many meetings with Dr. Rao to thank that this project never found itself hopelessly lost in a statistical quagmire. The extent to which I understand biochemical processes and enzyme kinetics is owed to the instruction of Dr. Milward. Finally, Dr. Schellingerhout, if I am able to achieve any semblance of the enthusiasm you possess for investigative pursuits, while maintaining your grasp of the difficulty inherent in them, I will be truly blessed as an academician.

I have been fortunate to be involved in the Julia Jones Matthews and M.D. Anderson CPRIT scholar program for four years. Not only has this program supported my work and allowed me to present said work at multiple meetings, it has also given me exposure to aspects of cancer biology I simply would not have seen outside of the program. Dr. Watowich, thank you for all your help and effort with the CPRIT program. I hope I am far from the last student to glean such tremendous benefit from it.

I would also like to thank all of those who stewarded me into post-graduate academic pursuits. The faculty at Trinity University began the extensive molding necessary for me to succeed after graduation. I especially thank Dr. Hough and Dr. Ugolini who opened my research career and taught me the foundational understanding of physics I needed to be able to approach the phenomenon of magnetic resonance. To Dr. Lewis, I am ever indebted. To whatever extent I possess the ability to write good software I owe it to him, and it has paid dividends in this project and my life as a whole.

**NOVEL SIMULATION TO AVOID BIAS IN MEASUREMENT OF HYPERPOLARIZED PYRUVATE:
DEMONSTRATED IN PHANTOM AND *IN VIVO***

Christopher M. Walker B.Sc

Advisory Professor: James A. Bankson, Ph.D

Abstract

Dynamic nuclear polarization creates a transient hyperpolarized nuclear state that can dramatically increase the signal detected by magnetic resonance imaging. This signal increase allows real-time spectroscopic imaging of specific metabolites *in vivo* by magnetic resonance. Real-time imaging of both the spatial and chemical fate of hyperpolarized metabolites is showing great promise to meaningfully benefit clinical care of cancer patients. Imaging of hyperpolarized agents will have a larger clinical impact if it can function as a quantitative modality upon which clinical decisions can be made. However, quantitative measurement of hyperpolarized agents is currently difficult due to the restrictions imposed by the transient hyperpolarized state and the complexity inherent in biological systems. As more advanced imaging and measurement techniques are developed for imaging hyperpolarized substrates, it is critical to characterize their effect on any relevant quantitative measure. To assist in accurate quantitative measurement of hyperpolarized agents, an infrastructure where acquisition strategies can be developed, compared, optimized and validated was critically need. A novel simulation architecture was developed that combines classical chemical kinetics with the basic physics of nuclear magnetic resonance and couples them to multiple perfusion models. Simulation results showed that changes in the acquisition strategy used will affect the resulting quantification of chemical exchange rates and suggested that any bias that is imposed by the acquisition strategy can be avoided by using optimized pulse sequences. To validate these predictions, a phantom system was developed that allows controllable chemical conversion of hyperpolarized pyruvate into lactate with a variability less than 20%. Using this phantom system, studies showed that poorly optimized pulse sequences

significantly reduced the measured value of the chemical exchange rates, whereas optimized pulse sequences showed no significant difference in chemical exchange measurements. In order to test simulation predictions for a perfused system, an animal cohort with orthotropic anaplastic thyroid cancer was scanned with multiple sequences. Again, optimized sequences showed no significant difference in measured exchange rates while poorly designed sequences significantly underestimated the exchange rates, which is consistent with the simulation results. These validation studies suggest that this simulation architecture will be a powerful tool for developing and optimizing acquisition and quantization methods for hyperpolarized magnetic resonance imaging.

Table of Contents

Acknowledgment	v
Abstract	vii
Table of Contents	ix
List of Figures	xiv
List of Tables	xvii
List of Abbreviations	xviii
Chapter 1. Introduction and Motivation	1
Hypothesis	9
Chapter 2. Nuclear Magnetic Resonance Physics	11
Section 2.1 Larmor Precession	11
Section 2.2 The Magnetization Vector and Thermal Equilibrium	14
Section 2.3 Spin-Lattice and Spin-Spin Decay	17
Section 2.4 Time-Varying Magnetic Fields and the Rotating Frame	20
Section 2.5 Summary of the Bloch Equation	24
Section 2.6 Chemical Shift	25
Section 2.7 Fourier Spectroscopy	26
Section 2.8 Clinical Magnetic Resonance Spectroscopy	30
Section 2.9 Dynamic Nuclear Polarization	32
Section 2.10 Models of Dynamic Nuclear polarization	37

<u>A. The Well-Resolved Solid Effect</u>	37
<u>B. Thermal Mixing</u>	40
<u>Section 2.11: Detection of Magnetic Resonance Signal</u>	45
<u>Chapter 3. Simulation of Hyperpolarized Studies</u>	49
<u>Section 3.1: Theory</u>	49
<u>Section 3.2: Implementation</u>	58
<u>Section 3.3: Verification</u>	62
<u>Chapter 4. Quantitative Accuracy of Dynamic Spectroscopy</u>	68
<u>Section 4.1 Introduction and Theory</u>	68
<u>Section 4.2 Methods</u>	72
<u>Section 4.3 Results</u>	75
<u>Section 4.4 Discussion</u>	82
<u>Chapter 5. System Validation</u>	87
<u>Section 5.1 Introduction and Theory</u>	87
<u>Section 5.2 Methods</u>	89
<u>A. Hyperpolarization</u>	89
<u>B. Dynamic Spectroscopy Repeatability</u>	90
<u>C. Spectroscopic Phantom Imaging</u>	92
<u>D. Dynamic Spectroscopy Sequence Parameter Dependence</u>	92
<u>E. Dynamic Spectroscopy in Vivo</u>	93

Section 5.3 Results	94
A. Repeatability Studies	94
B. Closed System Parameter Dependence	98
B. In Vivo Studies	100
Section 5.4 Discussion	101
Chapter 6. Conclusion and Future Work	104
B. Future Directions for the Simulation Architecture	105
C. Future Directions for the Dynamic Phantom	107
Appendix A: Hyperpolarized Exchange Kinetics	110
Section A.1 Label Exchange	110
Section A.2 Enzyme Flux Kinetics	112
Section A.3 Enzyme Exchange Kinetics	115
Section A.3 Enzyme Exchange Kinetics of Hyperpolarized Pyruvate	116
Appendix B: Source Code for HypWright	118
Section B.1 Higher Level Structures	118
World Class	118
Voxel Class	124
Section B.2: Pulse Sequence	130
Pulse Sequence Class	130
RF Pulse Class	139

Block Pulse Class	142
Sinc Pulse Class	144
Gradient Pulse Class	147
Linear Gradient Pulse	149
Section B.3: Spin Groups	150
Spin Group Class	150
Isolated Spin Group Class	151
Two-Site Exchange Group Class	153
Two-Site Perfusion Exchange Group Class	156
Bankson Spin Group Class	160
Section B.4 Signal Curve Modeling and Fitting	162
MultiPool Class	162
MultiPool Tofts Class	167
MultiPool Tofts Gamma VIF Class	170
Gamma Bankson Model Class	173
Section B.5 Example Scripts	177
Example Script for Simulating and Processing single pulse dynamic spectroscopy	177
Example Script for a Multiband Frequency Encode Snapshot	182
References	185
Vita	199

List of Figures

Figure 2-1. Visualization of the Bloch equations	25
Figure 2-2. Free induction decay and Lorentzian line shape.	28
Figure 2-3. Fourier analysis and phase correction of a free induction decay.....	29
Figure 2-4. Dynamic nuclear polarization.....	33
Figure 2-5. Nucleus electron pair energy diagram.	34
Figure 2-6. Nucleus electron energy diagram with flip-flops transitions.	35
Figure 2-7. An energy diagram of an electron-nuclear pair with all dipolar interaction.....	38
Figure 2-8. Polarization from the well-resolved solid effect..	40
Figure 2-9. Energy diagram of the thermal mixing process.....	41
Figure 2-10. Microwave sweep of thermal mixing.	43
Figure 2-11. A comparison of a conventional magnetic resonance signal and hyperpolarized magnetic resonance signal	46
Figure 3-1. Outline of the simulation architecture.....	59
Figure 3-2. A comparison of numerical methods for solving ordinary differential equations.	61
Figure 3-3. A comparison of the analytical and numerical solutions for a system of two spins coupled by chemical exchange.....	62
Figure 3-4. The computational performance of two spins either isolated or coupled by chemical exchange as a function of the difference in their chemical shifts	63
Figure 3-5. Exchange rate fitting for the longitudinal magnetization of two exchanging spin groups.	64
Figure 3-6. Perfusion fitting of the longitudinal magnetization of a single perfused spin group assuming two spatial compartments.....	65
Figure 3-7. Comparison of excitation profiles for doped C ¹³ urea.....	66
Figure 3-8. A comparison of dynamic spectroscopy for a dynamic phantom and a simulation	67

Figure 4-1. Workflow of simulation, processing and fitting	74
Figure 2-2. Percent error plots of driving versus fit exchange rates for the closed system approximation..	75
Figure 4-3. Percent error plots of driving versus fit exchange rates for the perfused system approximation.....	76
Figure 4-4. Relative total SNR of each study for the high driving exchange rate for both closed and perfused system.....	77
Figure 4-5. Average SNR per excitation for the closed and perfused systems with a high driving exchange rate.....	78
Figure 4-6. Normalized Square-2 Norms of the fits of both closed and perfused systems, with a high driving k_{pl}	79
Figure 4-7. Contrast error maps for the closed and perfused system approximations..	80
Figure 4-8. Percent error plots of driving versus fit exchange rates for the closed system approximation	81
Figure 4-9. Percent error plots of driving versus fit exchange rates for the perfused system.....	82
Figure 5-1. A schematic view of the dynamic chemical phantom structure.	91
Figure 5-2. Dynamic signal evolution across (N= 7) injections into enzyme phantom	95
Figure 5-3. Spectroscopic images of the reaction carried out in a standard imaging phantom.	98
Figure 5-4. Comparing simulation to dynamic phantom data for the closed system.	99
Figure 5-5. Comparison of <i>in vivo</i> vs. simulation kinetic data analysis from data acquired using different acquisition parameter combinations.	100
Figure 6-1. Simulated radial multi-band frequency encoded snap shot image of a square of perfused tissue converting pyruvate to lactate.	107
Figure 6-2. The reaction schematic of pyruvate and peroxide.....	108

Figure A-1. Comparison of label exchange modeling.	111
Figure A-2. A schematic of the Theorell-Chance mechanism for lactate dehydrogenase.	112

List of Tables

Table 2-1. Properties of nuclei commonly detected by magnetic resonance spectroscopy.....	31
Table 4-1. Parameters used for simulation and fitting.....	74
Table 5-1. The mean, standard deviation and coefficient of variation of all repetitions (N = 7) of the dynamic phantom	96
Table 5-2. Survey of HP parameter variation in recent animal studies	96

List of Abbreviations

ADP – Adenosine Diphosphate

ATP – Adenosine Triphosphate

DNA - Deoxyribonucleic Acid

DNP – Dynamic Nuclear Polarization

EDTA - Ethylenediaminetetraacetic Acid

EPSI - Echo Planar Spectral Imaging

FID – Free Induction Decay

FWHM – Full Width at Half Maximum

HEPES - 4-(2-hydroxyethyl)-1-Piperazineethanesulfonic Acid

HP - Hyperpolarized

IDH – Isocitrate Dehydrogenase

LDH – Lactate Dehydrogenase

MCT - Monocarboxylate Transporter

MRI – Magnetic Resonance Imaging

MRS – Magnetic Resonance Spectroscopy

NAD⁺ - Oxidized Nicotinamide Adenine Dinucleotide

NADH – Reduced Nicotinamide Adenine Dinucleotide

NMR – Nuclear Magnetic Resonance

ppm – parts per million

ROS – Reactive Oxygen Species

RF – Radio Frequency

SNR – Signal to Noise Ratio

TCA – Tricarboxylic acid

TIGAR - TP53-Inducible Glycolysis and Apoptosis Regulator

TR – Repetition Time

VIF – Vascular Input Function

Chapter 1. Introduction and Motivation

Cancer is most fundamentally characterized by a cell or a population of cells that sustains chronic uncontrolled proliferation¹. While sustained proliferation is quite common in single cell organisms, in more complex multi-cellular organisms, proliferation is tightly regulated to maintain homeostasis. In the case of cancer, such regulation breaks down through many mechanisms and the neoplastic population can become a threat to the survival of the organism. The loss of tight genetic control is characterized by, among other traits, genotypic, phenotypic and cellular heterogeneity^{2,3}.

Such heterogeneity is a driving factor among the many challenges associated with the management of cancer. This point is highlighted by the relatively poor improvement in cancer mortality in the United States since 1930⁴, despite the massive investments in research and treatment and the resulting breakthroughs in human understanding of the disease and how to treat it. Additionally, cancer represents the second, likely soon to be first, largest cause of death in the United States⁴. In order to be properly managed, cancers have to be well understood. Recently this core concept has been taken to its most extreme interpretation with the advent of personalized care. The aging concept of managing cancers based on their stage and organ of presentation is being replaced with a paradigm of characterization and treatment of a tumor on a patient-specific basis. To achieve such a goal, technology will have to be leveraged to give clinicians the specific information about a particular patient's tumor such that a treatment strategy can be devised and continuously revised. To that end, medical imaging will play a key complementary role in detection, characterization and monitoring of disease⁵⁻⁹.

Imaging strategies have multiple advantages, yet in order to be useful, clinical imaging needs to be sensitive and specific to particular biologic functions of interest¹⁰. In the case of profiling cancer, imaging with sensitivity to cellular characteristics and processes on a molecular level, known as molecular imaging, is particularly attractive. This is mostly because the drivers of progression in cancer are themselves cellular processes and therefore sensitivity on a cellular level is critical to monitoring

such drivers¹. However, in order to characterize cellular processes noninvasively, imaging methods either require extreme detection sensitivity or must target processes that involve plentiful agents for detection. One such cellular process that involves large set of pathways through which a multitude of molecules are processed is cellular energetics, or metabolism. Fortunately, one of the primary cellular functions altered by the dysregulations associated with cancers is metabolism². Therefore, cellular metabolism has the potential sensitivity and specificity to make it an effective molecular imaging target.

Most normal mammalian cells metabolize glucose into CO_2 in order to produce adenosine triphosphate (ATP), which is used in intercellular energy transfer¹¹. The breakdown of glucose follows a multi-step pathway with many branching points, but it commonly progresses to pyruvate through a process known as glycolysis. Glycolysis is composed of ten reactions catalyzed by enzymes and converts glucose, along with the cofactors adenosine diphosphate (ADP) and oxidized nicotinamide adenine dinucleotide (NAD^+), into pyruvate along with the higher energy compounds ATP and reduced nicotinamide adenine dinucleotide (NADH). In well differentiated cells, pyruvate is normally transported into the mitochondria where it is further broken down into CO_2 by a process known as the tricarboxylic acid (TCA) cycle, which produces NADH from NAD^+ . The excess NADH is then used to drive oxidative phosphorylation, which generates a large amount of ATP. In total, glycolysis coupled with the TCA cycle and oxidative phosphorylation produces 36 ATP molecules from a single glucose molecule. Oxidative phosphorylation requires oxygen, so under anaerobic conditions pyruvate is normally shunted into lactate through a process known as anaerobic glycolysis. Anaerobic glycolysis is a single step reaction that oxidizes NADH to produce lactate. It takes place outside of the mitochondria in the cell's cytosol. Generally, lactate is exported outside of the cell after anaerobic glycolysis, where it is used by the Cori cycle in the liver.

Neoplastic tissue preferentially converts pyruvate into lactate even in the presence of oxygen¹². The conversion of pyruvate to lactate in the presence of oxygen is commonly referred to as aerobic

glycolysis or the Warburg effect¹³. Aerobic glycolysis, while much less energy efficient than oxidative phosphorylation, still produces 2 ATP molecules from each molecule of glucose. It is initially non-intuitive that highly proliferative neoplastic cells would select for a less efficient method of producing ATP. This was first theorized to be driven by defects in the mitochondria limiting the cells' ability to engage in oxidative phosphorylation. However, it has been shown that mitochondria in neoplastic cells do not often have impaired function. It has also been proposed that highly proliferative cells are not limited by energy production¹⁴. Most neoplastic cells are in glucose-rich environments and can increase glucose uptake, thereby offsetting the energy restrictions they might incur by favoring less efficient anaerobic glycolysis. Therefore, if the cells can gain other benefits from favoring aerobic glycolysis, then the less efficient ATP production might not be detrimental. Indeed, most tumors do upregulate glucose uptake through phosphoinositide 3-kinase activation, which has been well studied, including in clinical disease by ¹⁸F-deoxyglucose positron emission tomography.

There are many potential proliferative benefits associated with the reduction of the TCA cycle and the increased glucose uptake that is associated with aerobic glycolysis. Oxidative phosphorylation is the largest generator of reactive oxygen species (ROS) in normal cells. During cell division, DNA must be replicated and is vulnerable to damage by ROS. Therefore, a reduction in oxidative phosphorylation could protect cells as they move through the cell cycle. Additionally, glucose can be catabolized into other metabolic intermediates, as opposed to the complete breakdown to CO₂ which maximizes ATP production. Proliferating cells need to replicate their entire cellular content. This places a high demand on the biosynthesis of nucleotides, lipids, and amino acids. While the production of these intermediates and their use to build macromolecules requires ATP, it also requires biomass, both of which can be supplied by glucose. There are many metabolic products that require more carbon biomass than ATP to be assembled. To generate these, it is more efficient to limit the ATP produced by glucose and allow it to be used to generate metabolic precursors. Overall, cells in a proliferative state reduce the amount of

nutrients catabolized for energy production to allow some carbon to be used in the production of the macromolecular structures that are needed to form two viable daughter cells¹⁵.

Another feature of anaerobic glycolysis is the production of a large amount of lactate. While the generation and export of large amounts of lactate from the cell seems like a waste of either potential ATP production or carbon biomass, it is important to consider the selective pressures on a proliferating cell. When selecting for rapid proliferation, the most efficient utilization of nutrients might not be preferred. In an organism, cells are in a nutrient-rich environment and therefore can be less efficient in the generation of ATP and other metabolic intermediates, which can be offset by increased nutrient uptake. Additionally, there are other tissues and organs that can utilize lactate for energy production, thus recapturing the potential loss of energy through mechanisms such as the Cori cycle.

Generally, aerobic glycolysis confers many potential benefits onto rapidly proliferating cells. They can take up much more glucose without having to produce the associated ROS, which could be devastating to DNA replication. The excess glucose can be rapidly shunted into macromolecular precursors, which will be needed to replicate the entire content of a cell. Only the most rapid steps in glucose catabolism will be favored as lactate is exported from the cell limiting the more efficient, yet not so rapid, later steps in glucose catabolism. Finally, anaerobic glycolysis is still energy-positive, generating the ATP and NADH that are needed for the replication and assembly of cellular content.

A critical step in anaerobic glycolysis is the conversion of pyruvate to lactate. This is catalyzed by the enzyme lactate dehydrogenase (LDH) and requires the co-enzyme NADH, which donates a proton to become NAD⁺ ¹¹. The conversion from pyruvate to lactate is reversible, with the direction of favorability determined partially by the LDH isoform catalyzing the reaction. LDH is a tetramer composed of four sub-units that can be from either or both of two genes, LDH-A and LDH-B. The combination of these two subunits forms five possible LDH isoforms LDH (1-5), LDH1 is composed entirely of subunits encoded by LDH-B and it favors the conversion of lactate into pyruvate while LDH5 is composed of subunits encoded

by LDH-A and strongly favors the conversion of pyruvate into lactate. In most cancer cells LDH-A is strongly upregulated, creating an abundance of LDH5. This helps drive anaerobic glycolysis.

Large production of lactate will tend to occur only in stressed muscles or in tumor tissue. Therefore, in a rested subject, the production of lactate through aerobic glycolysis would be a specific marker of cancer. The production of lactate is also dependent upon a cell's redox state and therefore it is coupled with many cellular processes. Most obvious would be cell viability. Dead and dying cells will be unable to produce lactate as cellular functions are shutting down. A cell's response to cellular damage via reactive oxygen species has also been shown to correlate with reduced lactate production. This is mediated by ROS scavenger compounds, which deplete a cell's reducing potential invoked to protect the genome. The metabolic alterations in cells as a response to insult can be rapid, allowing response characterization long before significant physiologic or morphologic changes occur¹⁶⁻¹⁹.

Recently it has been shown that the aberrant metabolism that is displayed by cancer is not simply a byproduct of rapid proliferation but is closely tied to tumorigenesis²⁰. Some tumors show a dependence on upregulated phosphoinositide 3-kinase, which upregulates glucose transport. Additionally, oncogenes RAS and MYC correlate with an upregulation of glycolysis. MYC also regulates proteins that control glutamine metabolism and can lead to a phenotypic dependence on glutamine metabolism. Tumors cell frequently experience hypoxic conditions, which along with RAS can increase the expression of hypoxia-inducible factor1 α and 2 α which in turn upregulate glycolysis. Isocitrate dehydrogenase 1 and 2 (IDH 1 and 2) have been shown to be activated in a subset of gliomas. IDH1 and 2 catalyze the conversion between isocitrate and α -ketoglutarate, resulting in a unique metabolic phenotype in the gliomas with mutant IDH 1 or 2. Tumor suppressor genes also play a guiding role in cellular metabolism. The tumor suppressor gene p53 can drive more glucose into the pentos phosphate shunt by regulating the expression of TIGAR. While it is becoming clear that the mutations involved in tumorigenesis play a role in regulating metabolism, the exact mechanisms are still under investigation.

Despite the ongoing investigation into how oncogenic driving mutations alter metabolism, there has been progress in leveraging unique metabolic phenotypes, such as the production of 2-hydroxyglutarate in IDH1 mutant glioblastomas, to characterize cancer mutations²¹. Such characterization could be used to assist tumor profiling, which is becoming critical for treatment decisions in the age of personalized therapies²². Additionally, cancer cells can be dependent upon their altered metabolism, providing an opportunity for pharmacologic intervention²³. With primary or adjuvant therapies targeting metabolism specifically, sensitive probes of cancer metabolism would be an unparalleled tool in monitoring therapy response and efficacy.

While pyruvate is a downstream product of glycolysis, it is also taken up by cells via the monocarboxylate transporter 1 (MCT-1). The MCT family of proteins transport monocarboxylates in a proton-linked manner²⁴. Two isoforms, MCT-1 and MCT-4, are important for lactate efflux. MCT-1 is less specific for lactate than MCT-4, and can also result in pyruvate flux into the cell. Cancer cells frequently upregulate MCT-1 and MCT-4, which remove the excess lactate produced by anaerobic glycolysis. Additionally, cancer cells can rapidly take up pyruvate through the MCT-1 transporter.

Because of these biologic factors, pyruvate would be an ideal target for probing anaerobic glycolysis of tumors. Pyruvate is an organic compound consisting of a carboxylic acid and a keto group. Pyruvate is converted to lactate by reducing the number 2 carbon with an H^- from NADH, thus altering the chemical structure of the molecule. This chemical change will result in a change in the frequency of its carbon magnetic resonance signal. Therefore, magnetic resonance spectroscopy of pyruvate could be a promising tool for assessing anaerobic glycolysis and thus cancer metabolism.

The carbons that make up the backbone of a pyruvate molecule have the potential to be detected via magnetic resonance imaging (MRI), but the magnetic resonance signal of carbon is extremely weak. However, using the technique of hyperpolarization, the magnetic resonance signal of a compound can be increased by many orders of magnitude²⁵. Additionally, pyruvate has numerous

advantages as a hyperpolarized agent²⁶. The carbon in the one position of pyruvate has a detection lifetime on the order of a few minutes. Pyruvate is rapidly distributed after an intravenous injection where it is quickly taken up by cells through the MCT-1. Once it is in the cytosol, pyruvate is quickly converted to its downstream products²⁷. The delivery and conversion of pyruvate can happen on the order of a few seconds, depending on the tissue. In the case of cancer specifically, the delivery, uptake, and conversion of pyruvate is generally quick, due in part to the Warburg effect. Fortunately, carbon 13 labeled pyruvate is non-toxic even at high dosages, allowing large amounts of pyruvate to be safely administered²⁸. In summary, as a hyperpolarized magnetic resonance agent, pyruvate is an ideal probe of cancer metabolism due to its physical as well as its physiologic and biochemical properties.

Detection of hyperpolarized pyruvate by magnetic resonance is quite different from the techniques of conventional MRI and magnetic resonance spectroscopy (MRS), which focus on detection of hydrogen atoms normally bound in a water molecule. These differences result in acquisition strategies that are divergent from conventional MRI and MRS^{16,26-30}. Therefore, much of the development and optimization of acquisition and processing associated with conventional MRI and MRS cannot be applied to hyperpolarized agents. Additionally, as a molecular imaging strategy, the ability to quantify results in some way that is comparable with other measurements and is intrinsically related to underlying biology is critical for clinical utility. Given these constraints, it is imperative that hyperpolarized MRI and MRS be thoroughly characterized and optimized. However, due to practical limitations such characterization cannot be performed by experimentation alone. The parameter space that needs to be explored to ensure efficient detection and quantization fidelity is far too extensive to be thoroughly explored in the lab. There was thus a critical need for a simulation architecture that could rapidly explore the numerous detection methods and quantization techniques proposed for hyperpolarized MRI and MRS. In this work such a system has been developed from first principals and validated in multiple physical models. The simulation architecture described herein is a flexible tool for

designing, comparing and optimizing acquisition methods related to hyperpolarized agents and will serve as a powerful tool as hyperpolarized MRI and MRS move from developing pre-clinical techniques to robust and routine clinical modalities.

Hypothesis

The value of the pyruvate-to-lactate exchange rate that is measured by hyperpolarized MR is significantly altered by the MR data acquisition strategy that is employed. By utilizing a novel simulation, pulse sequences can be designed such that any biases imposed by the acquisition strategy can be removed for both phantom and *in vivo* studies.

Aim 1: Development of a novel perfused Bloch-McConnell simulator

The governing equations of the classical model of nuclear magnetic resonance (NMR), the Bloch equations, can be solved numerically. The Bloch equations are widely used to simulate and optimize MRI pulse sequences and acquisition strategies. It should be noted that although many conventional Bloch simulators do not account for a hyperpolarized state, the generalized Bloch equations do model this situation. To meaningfully simulate hyperpolarized imaging, a physical model of tracer delivery and conversion was implemented, modeling both perfusion and chemical exchange.

Aim 2: Compare the effects of excitation angles and repetition times using the perfused Bloch simulator

Due to the non-renewable nature of hyperpolarized magnetization, each signal excitation will affect all subsequent measurements. Therefore, the detected signal will be inherently linked to the excitation scheme used in acquisition. Most hyperpolarized studies are processed to yield apparent rates of chemical exchange between multiple chemical pools. Using the simulation architecture from aim one to compare the rate constant resulting from the processing of simulation data to the actual rate constant used in simulation, the accuracy and repeatability of the measured exchange rate was determined across a range of sequence, physiologic, and modeling conditions.

Aim 3: Using a novel dynamic enzyme phantom and *in vivo* models, errors introduced by the acquisition method, as predicted by simulation, were demonstrated for dynamic spectroscopy and compared to sequences designed to avoid such errors.

In order to validate the simulation predictions from aim 2, physical phantoms must be used. These model systems would need to convert hyperpolarized pyruvate into lactate in a repeatable manner. However, quantification of agent delivery and exchange rates has been difficult due to the complexity of the *in vivo* environment and the constraints inherent in hyperpolarized agents. The conversion of pyruvate to lactate can be run in the controlled environment of an isolated buffer, allowing imaging and quantification without the complexity of a biological system. Using the novel dynamic enzyme phantom, pulses sequences predicted by simulation to introduce errors in the measured apparent exchange rate were compared to sequences designed to avoid such errors. Additionally, in order to account for perfusion, similar validation studies were undertaken in a mouse model of thyroid cancer.

Chapter 2. Nuclear Magnetic Resonance Physics

The phenomenon of nuclear magnetic resonance is well described by the equation first presented by Felix Bloch^{31,32}.

$$\frac{d\vec{M}}{dt} = \gamma \vec{M} \times \vec{B} + \frac{1}{T_1} (\vec{M}_0 - \vec{M}_{\parallel}) - \frac{1}{T_2} M_{\perp}$$

The Bloch equation is composed of multiple terms which will be developed independently and then brought together. The precise interpretation of each term will be developed over the next chapter but briefly; the first term relates the precessional motion of the net magnetization vector \vec{M} to the external magnetic field \vec{B} and the particles' inherent gyromagnetic ratio γ ; the second term describes the system's tendency to return to its thermal equilibrium \vec{M}_0 with a time constant T_1 ; finally, the third term describes a dephasing of the net magnetization vector due to molecular tumbling that is characterized by a decay time T_2 .

Section 2.1 Larmor Precession

Elementary particles or groups of particles with a non-zero spin give rise to magnetic dipoles³³. Such dipoles will precess when in the presence of an external magnetic field. Consider a spinning charged body with mass m and charge q . For mathematic simplicity it can be assumed that this object is a uniformly charged infinitesimally thin hoop. If the hoop is spinning with some angular velocity ω it has a current³⁴:

$$J = \frac{q\vec{\omega}}{2\pi} \tag{2.1}$$

Current loops give rise to magnetic dipoles by the relation:

$$\mu = JA \tag{2.2}$$

where μ is the magnetic dipole moment, J is the current and A is the area enclosed by the hoop.

Combining equations (2.1) and (2.2) and assuming a perfectly circular hoop leads to³¹:

$$\mu = \frac{q\vec{\omega}}{2\pi} * \pi r^2 \quad 2.3$$

where r is the radius of the hoop. To relate this back to a fundamental particle, the angular momentum must be defined. The angular momentum of a hoop will follow³⁵:

$$\vec{L} = I\vec{\omega} \quad 2.4$$

where, I is the Inertia of the system. Using the inertia of a spinning hoop of infinitesimal thickness yields:

$$\vec{L} = mr^2\vec{\omega} \quad 2.5$$

Rearranging equation (2.3) and substituting in equation (2.5) gives the relation for the dipole moment of the form:

$$\vec{\mu} = \frac{1}{2} \frac{q}{m} \vec{L} \quad 2.6$$

Thus the dipole moment of a spinning hoop of charge relates to its angular momentum only by its charge-to-mass ratio. Note that the factor of $\frac{1}{2}$ is purely a function of the object's geometry. If it were assumed to be a spinning disk that factor would be unity and it would be $\frac{2}{3}$ and $\frac{3}{5}$ for a sphere and ball respectively. The key idea is that if the object is spherically symmetric the magnetic dipole moment is determined by the charge-to-mass ratio scaled by a constant that is determined by the geometry.

A fundamental property of elementary particles is their inherent angular momentum, which arises from a fundamental property of particles called spin. Spin is quantized into discrete states denoted by the quantum number s . A particle's intrinsic angular momentum \vec{S} relates to its spin s and the reduced Planck constant \hbar by³⁶:

$$\vec{S} = \hbar\sqrt{s(s+1)} \quad 2.7$$

Particles, either elementary or composite, can be classified by their possible spin states. Bosons are particles with integer spin, that is, they are symmetric about a 360° rotation. Fermions have half integer spin states. With half integer spin states particles that have been rotated by 360° are distinct. Thus asymmetry of Fermions results in their states being paired based on their spin. For Fermions, the relation defined in equation (2.6) does not strictly hold and must be modified to:

$$\vec{\mu} = g \frac{q}{2m} \vec{S} \quad 2.8$$

where g , or g-factor, is a constant of proportionality that relates a particles' magnetic moment to its inherent angular momentum. Interestingly, in the case of electrons, the g-factor is 2.00038 and the above classical derivation, assuming a disk geometry, gives nearly the exact form of the quantum relation for electrons. In the case of protons, the g factor is about 5.59 and such a classical derivation breaks down. This illustrates that particle spin is a purely quantum phenomenon with no robust classical analog. This should make some sense, as imagining a geometric system with spin $\frac{1}{2}$, that is symmetric about a 720° rotation and not one of 360° , is impossible.

Commonly, the charge-to-mass ratio and g-factor are combined into a quantity known as the gyromagnetic ratio γ . With this addition equation (2.8) becomes:

$$\vec{\mu} = \gamma \vec{S} \quad 2.9$$

In the presence of an external magnetic \vec{B} field magnetic dipoles will tend to align with the field. This is made manifest by a torque $\vec{\tau}$ given by³⁴:

$$\vec{\tau} = \vec{\mu} \times \vec{B} \quad 2.10$$

Torque is equal to the moment of inertia times the angular acceleration $\vec{\alpha}$:

$$\vec{\tau} = I \vec{\alpha} \quad 2.11$$

where angular acceleration is the time rate of change in angular momentum or:

$$\vec{\alpha} = \frac{d\vec{L}}{dt} \quad 2.12$$

Taking the time derivative of angular momentum as defined in equation (2.4) results in:

$$\frac{d}{dt}(\vec{L}) = \frac{d}{dt}(I\vec{\omega}) = \frac{dI}{dt}\vec{\omega} + I\frac{d\vec{\omega}}{dt} \quad 2.13$$

Assuming no change in the moment of inertia, equation (2.13) becomes:

$$\frac{d\vec{L}}{dt} = I\frac{d\vec{\omega}}{dt} = I\vec{\alpha} = \vec{\tau} \quad 2.14$$

Finally taking the time derivative of equation (2.9) with γ assumed to be constant yields:

$$\frac{d}{dt}(\vec{\mu}) = \frac{d}{dt}(\gamma\vec{S}) = \gamma\frac{d\vec{S}}{dt} \quad 2.15$$

Assuming angular momentum \vec{L} arises solely from the particles' intrinsic angular momentum \vec{S} requires:

$$\gamma\frac{d\vec{S}}{dt} = \gamma\frac{d\vec{L}}{dt} \quad 2.16$$

Combining equations (2.10), (2.14), (2.15) and (2.16) yields the following relation³¹:

$$\frac{d\vec{\mu}}{dt} = \gamma\vec{\mu} \times \vec{B} \quad 2.17$$

Assuming that \vec{B} is constant, referred to as \vec{B}_0 , equation (2.17) results in a precession about the \hat{B} axis with a precessional frequency ω following

$$\omega = -\gamma\vec{B}_0 \quad 2.18$$

The precessional frequency defined in equation (2.18) is referred to as the Larmor frequency.

Section 2.2 The Magnetization Vector and Thermal Equilibrium

Equation (2.17) assumes an isolated spin in a uniform magnetic field. Conventional magnetic resonance is a bulk phenomenon and so an ensemble of spins must be considered. Assume some small volume element or voxel that is large enough to contain a large number of fundamental particles but has a negligible change in B_0 . This is reasonable when considering small molecules like water. In one cubic micron there are over a billion water molecules. The B_0 fields in the following section generally

refer to large man-made fields and as such will not vary significantly over distances on the order of a micron, well satisfying the voxel conditions outlined above. When dealing with an ensemble of magnetic material, such as a group of dipoles, a quantity called magnetization is used³⁴.

Magnetization is defined as the differential magnetic moment $d\vec{m}$ for some differential volume dV :

$$\vec{M} = \frac{d\vec{m}}{dV} \quad 2.19$$

When some finite number of magnetic moments is being considered over some finite space V equation (2.19) becomes:

$$\vec{M} = \frac{1}{V} \sum_{i=\text{dipoles in } V} \vec{\mu}_i \quad 2.20$$

If only the dipole interaction with an external field is considered, then equation (2.17) can be combined with equation (2.20) to yield:

$$\frac{d\vec{M}}{dt} = \frac{1}{V} \sum_{i=\text{dipoles in } V} \gamma \vec{\mu}_i \times \vec{B}_{ext} = \gamma \vec{M} \times \vec{B}_{ext} \quad 2.21$$

However, equation (2.17) was derived for an isolated magnetic moment. With the inclusion of multiple spins more interactions must be considered. In practice the voxel that defines the magnetization above will be in thermal contact with a surrounding lattice. Additionally, the lattice can be assumed to be large compared to the magnetization voxel, which was assumed to be small. Systems in thermal contact with larger lattices, or reservoirs, will follow the Boltzmann distribution:

$$P(\epsilon) = \frac{e^{-\frac{\epsilon}{kT}}}{Z} \quad 2.22$$

where ϵ is the energy of a state, in this case the magnetic potential energy of a magnetization vector in a B field, $P(\epsilon)$ is the probability of a particle to be in that state, T is the temperature of the reservoir, k is the Boltzmann constant and Z is a normalization constant that ensures that the total probability of all

states is unity. Conceptually, the thermal energy of the lattice will continuously perturb the magnetization from its lowest energy state, which is aligned with the external magnetic field. Therefore, the magnetic moments will be distributed across a range of states and subsequent energy levels driven by the temperature of the contacting lattice. The lattice must be large compared to the magnetization voxel so the energy loss from the lattice that perturbs the magnetization can be ignored. Classically equation (2.22) would be a continuous distribution with the magnetic energy ϵ defined as:

$$\epsilon = \vec{\mu} \cdot \vec{B} \quad 2.23$$

which is simply the magnetic energy of a bar magnet in an external field. However, the spin quantum number of Fermions is quantized into half integer states and there will only be a discrete number of possible states for equation (2.22)³⁶. In a spin $\frac{1}{2}$ system the spin quantum number s can have two possible states $+\langle\frac{1}{2}\rangle$ and $-\langle\frac{1}{2}\rangle$. Using intrinsic angular momentum \vec{S} , equation (2.7), from those two spin states and the resulting dipole moment $\vec{\mu}$, equation (2.9) and equation (2.23) become:

$$\epsilon_+ = \frac{1}{2}\gamma\hbar \cdot B \text{ and } \epsilon_- = -\frac{1}{2}\gamma\hbar \cdot B \quad 2.24$$

The energy difference between these two states will be

$$\Delta E = \epsilon_+ - \epsilon_- = \hbar\gamma \cdot B = \hbar\omega \quad 2.25$$

Note that the frequency of such an energy transition is exactly the Larmor frequency. The energy required to induce a change in the distribution will be at or near the Larmor frequency. The magnetization at thermal equilibrium M_0 will be, as defined by the Boltzmann distribution equation (2.22), a weighted average of all possible dipole moments over some voxel with spin density ρ_0 ³¹:

$$M_0 = \rho_0 \frac{\sum_{m=-\frac{1}{2}}^{\frac{1}{2}} \mu(m) e^{\frac{-m\hbar\gamma \cdot B}{kT}}}{\sum_{m=-\frac{1}{2}}^{\frac{1}{2}} e^{\frac{-m\hbar\gamma \cdot B}{kT}}} \quad 2.26$$

In all but the super-cooled temperature ranges and for field strengths on the order of Tesla or lower

$$kT \gg \hbar\gamma \cdot B \quad 2.27$$

As an example assume protons at room temperature and a 3 Tesla field. kT is ~ 25.9 meV whereas $\hbar\gamma B$ is $\sim 0.25 \times 10^{-4}$ meV satisfying the relationship in equation (2.27).

Under the conditions in the relationship (2.27), a Taylor series expansion of the exponent in equation (2.22) is:

$$e^{\frac{-m\hbar\gamma B}{kT}} = 1 + \frac{-m\hbar\gamma \cdot B}{kT} + \mathcal{O}\left\{\left(\frac{-m\hbar\gamma \cdot B}{kT}\right)^2\right\} + \dots \quad 2.28$$

Using the first two terms of the expansion, equation (2.26) becomes:

$$M_0 \cong \rho_0 \frac{\sum_{m=-\frac{1}{2}}^{\frac{1}{2}} \mu(m) 1 + \frac{-m\hbar\gamma \cdot B}{kT}}{\sum_{m=-\frac{1}{2}}^{\frac{1}{2}} 1 + \frac{-m\hbar\gamma \cdot B}{kT}} \quad 2.29$$

Expanding the summation and noting that for a spin $\frac{1}{2}$ system $\mu(m) = \pm \frac{1}{2} \hbar\gamma$, equation (2.29) reduces to:

$$M_0 \cong \frac{\rho_0 \gamma^2 \hbar^2 B}{4kT} \quad 2.30$$

with the direction of M_0 parallel to the direction of B , which in most cases will be along \hat{z} as the direction of the static B field that conventionally defines the \hat{z} axis.

Section 2.3 Spin-Lattice and Spin-Spin Decay

If the ensemble spin system described by equation (2.21) has some thermal equilibrium M_0 which is approximated by equation (2.30), it should follow that any perturbation from M_0 will result in an unstable system that will return over some length of time, governed by a decay constant T_1 , to M_0 . The quantum mechanical perturbation theory behind this relaxation phenomenon is too lengthy to be outlined here in full detail³⁷. Conceptually, the energy either gained or lost to achieve the return to M_0

from some perturbed state is provided by numerous interactions with the lattice. Assuming that the interactions between the spin system and the lattice are proportional to the magnitude of the perturbation from equilibrium, the following differential equation and solution³¹ hold:

$$\frac{dM}{dt} = \frac{1}{T_1} (M_0 - M) \quad 2.31$$

$$M(t) = M(0)e^{-\frac{t}{T_1}} + M_0 \left(1 - e^{-\frac{t}{T_1}}\right) \quad 2.32$$

While T_1 decay governs how a spin system returns to equilibrium based on energy exchange with its lattice, the spins within the spin system can interact with each other. Such interactions are known as spin-spin interaction and are characterized by a constant conventionally referred to as T_2 . Since the temperature of the spin system is non-zero, there will be some molecular tumbling of the atoms or molecules in the voxel that defines a magnetization vector. The tumbling molecules which could be charged will create fluctuations in the magnetic field. Such fluctuations will be random, small and depend on the tumbling rate of the molecules. Small fluctuations in the magnetic field will create small deviations from the larger applied magnetic field that will result in small changes in the Larmor frequency. Therefore, the Larmor frequency of the spins within a voxel will be a constant number but will be distributed over a small range of frequencies. The precessional frequency characterizes the component of the magnetization that is perpendicular to the B_0 field. Conventionally this plane is considered to define the \hat{x} and \hat{y} axes and simply referred to as the transverse axis or M_\perp . Adding this dephasing term to equation (2.21) for only the transverse components:

$$\frac{d\vec{M}_\perp}{dt} = \gamma \vec{M}_\perp \times \vec{B} - \frac{1}{T_2} \vec{M}_\perp \quad 2.33$$

Note that, unlike spin lattice interaction, spin-spin interactions are uniformly a loss of magnetization.

They also involve no energy exchange; they simply arise from a dispersion of precessional frequencies,

which will slowly decay the constituent spins of the transvers magnetization. If Larmor precession is ignored, the solution to equation (2.33) is of the form:

$$\vec{M}_\perp(t) = \vec{M}_\perp(0)e^{-\frac{1}{T_2}t} \quad 2.34$$

which is a simple exponential decay in the transverse magnetization. Practically speaking, there are more deviations in the magnetic field than those caused by molecular tumbling, these will be referred to as T_2' . These additional spatial variations in the magnetic field are generally considered to be time-independent, and therefore the dephasing they cause could be reversible. The combination of the irreversible spin-spin dephasing and the reversible interactions with time-invariant non-uniform magnetic field distortions is referred to as T_2^* relaxation and follows³⁷:

$$\frac{1}{T_2^*} = \frac{1}{T_2} + \frac{1}{T_2'} \quad 2.35$$

Note that the deviation in the magnetic field caused by T_2' are assumed to be local and isotropic. If there is some coherency to the deviation of the external field that drives T_2' then it would not generally lead to an exponential decay and is not included in T_2' . For simplicity it will be assumed that the external magnetic field is uniform and thus $T_2' = \infty$ and $T_2^* = T_2$, although under most conditions the following equations will still hold by simply replacing T_2 with T_2^* .

Combining equations (2.31) and (2.33) into a single vector equation results in the phenomenological equation first described by Bloch^{31,32}:

$$\frac{d\vec{M}}{dt} = \gamma \vec{M} \times \vec{B} + \frac{1}{T_1} (\vec{M}_0 - \vec{M}_\parallel) - \frac{1}{T_2} \vec{M}_\perp \quad 2.36$$

where \vec{M} has been split into two components; the longitudinal component M_\parallel which is parallel with the thermal equilibrium magnetization \vec{M}_0 , and the transverse magnetization M_\perp which is perpendicular to the longitudinal magnetization. Note, that the spin-spin relaxation only acts on the transverse magnetization as the longitudinal component of the magnetization will not precess around the

longitudinal axis. Additionally, because spin-spin dephasing is generally faster than the spin lattice relaxation the transverse components only decay with T_2 .

Section 2.4 Time-Varying Magnetic Fields and the Rotating Frame

Larmor precession adds complexity to the motion of the magnetization vector. Through a coordinate change into a frame of reference F' that is rotating with respect to the laboratory frame F the Larmor precession can be reduced or eliminated. Based on relative motion, the time-derivative $d\vec{V}'/dt$ of any time-dependent vector $\vec{V}(t)$ with time derivative $d\vec{V}/dt$ in the lab frame will be³⁵:

$$\frac{d\vec{V}'}{dt} = \frac{d\vec{V}}{dt} - \vec{\omega}_r \times \vec{V}(t) \quad 2.37$$

where $\vec{\omega}_r$ is the angular velocity vector for the rotating frame. Ignoring T_1 and T_2 decay for now for simplicity, and combining equations (2.21) and (2.37) leads to:

$$\frac{d\vec{M}'}{dt} + \vec{\omega}_r \times \vec{M} = \gamma \vec{M} \times \vec{B}_{ext} \quad 2.38$$

Equation (2.38) can be reduced to:

$$\frac{d\vec{M}'}{dt} = \gamma \left\{ \vec{\mu} \times \left(\vec{B}_{ext} + \frac{\vec{\omega}_r}{\gamma} \right) \right\} \quad 2.39$$

By redefining $\frac{\vec{\omega}_r}{\gamma}$ as a fictitious B field³¹ arising from the coordinate change $\frac{\vec{\omega}_r}{\gamma} = \vec{B}_{Fict}$ then the external B field can be combined with B_{Fict} to yield an effective B field in the rotating frame B_{eff} ³¹:

$$\frac{d\vec{\mu}'}{dt} = \gamma \{ \vec{\mu} \times \vec{B}_{eff} \} \quad 2.40$$

$$\vec{B}_{eff} = \vec{B}_{ext} + \frac{\vec{\omega}_r}{\gamma} \quad 2.41$$

Notice that if B_{eff} is zero then $\frac{d\vec{\mu}'}{dt} = 0$ and the magnetic moment is unchanged. This occurs when:

$$\frac{\vec{\omega}_r}{\gamma} = -\vec{B}_{ext} \quad 2.42$$

Or

$$\vec{\omega}_r = -\gamma \vec{B}_{ext} \quad 2.43$$

Which is the Larmor frequency. Therefore, if a rotating reference frame is chosen to rotate at the Larmor frequency then the motion of the system is greatly simplified. A classic analogy for this frame shift is the idea of a carousel. The motion of an object on a carousel is quite complex when observed from the ground next to the carousel. However, if the observer were to step onto the carousel, the motion would be greatly simplified relative to the observer.

In most cases, to detect the magnetization vector the vector must be excited away from thermal equilibrium into the transverse plane. Generally, the perturbation away from the thermal equilibrium is provided by a transient magnetic pulse. This will require splitting B into a time-invariant component B_0 and some short-lived component B_1 . Splitting up B into these components, equation (2.36) becomes:

$$\frac{d\vec{M}}{dt} = \gamma(\vec{M} \times \{\vec{B}_0 + \vec{B}_1(t)\}) + \frac{1}{T_1}(\vec{M}_0 - \vec{M}_{\parallel}) - \frac{1}{T_2}\vec{M}_{\perp} \quad 2.44$$

Assuming that \vec{B}_1 is left-handed circularly polarized electromagnetic field rotating about the \hat{z} axis with a frequency ω and initial phase ϕ_0 following:

$$\vec{B}_1(t) = B_1\{\cos(\omega t + \phi_0)\hat{x} - \sin(\omega t + \phi_0)\hat{y}\} \quad 2.45$$

In the rotating frame rotating with an angular frequency ω_{ref} , equation (2.45) will become:

$$\vec{B}'_1(t) = B_1\left\{\cos\left((\omega - \omega_{ref})t + \phi_0\right)\hat{x} - \sin\left((\omega - \omega_{ref})t + \phi_0\right)\hat{y}\right\} \quad 2.46$$

If the reference frame is rotating with the same frequency as the circularly polarized $\vec{B}_1(t)$, or, $\omega_{ref} = \omega$ then equation (2.46) collapses to:

$$\vec{B}'_1(t) = B_1\hat{x} \quad 2.47$$

Also assuming that the duration of the magnetic pulse is short compared to the relaxation effects, equation (2.44) can then be reduced to:

$$\frac{d\vec{M}'}{dt} = \gamma(\vec{M} \times \{\vec{B}_0 - \gamma\omega_{eff}\}) + \gamma(\vec{M}' \times B_1(t)\hat{x}) \quad 2.48$$

If the B_1 field also is rotating at the Larmor frequency, then equation (2.48) is further reduced to:

$$\frac{d\vec{M}'}{dt} = \gamma(\vec{M}' \times B_1(t)\hat{x}) \quad 2.49$$

Equation (2.49) is in a similar form to equation (2.21) and will create a rotation about an axis. However, since the B_1' field is along the \hat{x} axis the rotation will also be an angle θ about \hat{x} given by:

$$\vec{M}'(t) = \vec{M}'(0)R_x(\omega t) \quad 2.50$$

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix} \quad 2.51$$

where $\omega = \gamma B_1$ and the excitation angle θ will be discussed later in this section.

For atomic nuclei, the Larmor frequency is in the radiofrequency range and therefore the magnetic pulses used to excite such spin systems are referred to as radiofrequency pulses (RF-pulses). When the frequency of the RF-pulse ω is offset from the Larmor frequency, which will still be considered the reference frequency for the rotating frame $\omega_{ref} = \omega_0$, B_1 is no longer constant in time and (2.48) becomes:

$$\frac{d\vec{M}'}{dt} = \gamma(\vec{M} \times \{\vec{B}_1(t) + \vec{B}_0 - \gamma\vec{\omega}_{eff}\}) \quad 2.52$$

which can be distributed into its three component vectors assuming that the \vec{B}_0 field is along the \hat{z} direction, and that \vec{B}_1 is within the $\hat{x}\hat{y}$ plane and defining $\Delta\vec{\omega} = \gamma\vec{B}_0 - \vec{\omega}_{ref} = \gamma\vec{B}_{eff}$:

$$\frac{dM'_x}{dt} = M'_y\Delta\omega - M'_z\gamma B_{1x}(t) \quad 2.53$$

$$\frac{dM'_y}{dt} = -M'_x\Delta\omega + M'_z\gamma B_{1x}(t) \quad 2.54$$

$$\frac{dM'_z}{dt} = -\gamma(B_{1,x}(t)M'_y + B_{1,y}(t)M'_x) \quad 2.55$$

Defining the transverse plane as a pair of complex numbers then:

$$M_{\perp} = M_x + iM_y \quad 2.56$$

$$B_{1,\perp} = B_{1,x} + iB_{1,y} \quad 2.57$$

Combining equations (2.53-2.57), a differential equation results:

$$\frac{dM'_{\perp}}{dt} = -i\Delta\omega M'_{\perp} + i\gamma M'_z B_{1,\perp}(t) \quad 2.58$$

From the solution to this ordinary differential equations some of the excitation behavior of the Bloch equation (2.44) becomes clear³⁷:

$$M'_{\perp}(t) = i\gamma e^{-i\Delta\omega t} \int_0^t M'_z(\tau) B_{1,\perp}(\tau) e^{i\Delta\omega\tau} d\tau \quad 2.59$$

From inspection, the complex transverse magnetization in the reference frame of the B_1 field will be dependent on the frequency relation of the B_1 field and the rate of precession. Equation (2.59) governs the initial phase of the transverse magnetization similar to ϕ_0 in equation (2.45) as well as the excitation angle, θ , defined in (2.51). The idea that the relative frequency between the excitation pulse and the Larmor frequency determines the perturbation from equilibrium also agrees with the quantum nature of the transition between states and the energy needed briefly mentioned after equation (2.25).

In the simplified case of a rectangular envelope pulse matched to the Larmor frequency, following equation (2.59), the tip angle θ will be³¹:

$$\theta_{tip} = \gamma B_1 \tau \quad 2.60$$

where B_1 is the amplitude of the block pulse and τ is its duration. Additionally, the phase of the transverse magnetization will be matched to the phase of the block pulse. While these relationship hold for a rectangular pulse, more complicated relationship between the pulse characteristics and the resulting excitation angle and phase exist for other pulse shapes.

Section 2.5 Summary of the Bloch Equation

In summary the key aspects of the Bloch equation (2.44) will be restated:

$$\frac{d\vec{M}}{dt} = \gamma(\vec{M} \times \{\vec{B}_{eff} + \vec{B}_1(t)\}) + \frac{1}{T_1}(\vec{M}_0 - \vec{M}_{\parallel}) - \frac{1}{T_2}M_{\perp} \quad 2.44$$

A uniform ensemble of magnetic moments in the presence of a magnetic field will precess about the magnetic field at the Larmor frequency, equation (2.18). Additionally, there is a thermal equilibrium magnetization M_0 due to Boltzmann distribution energy states that is caused by the applied field, equation (2.22 and 2.26). The magnetization will return to its equilibrium magnetization through energy exchange with its lattice with a characteristic spin lattice relaxation time T_1 , equation (2.32). Combined with the relaxation caused by energy exchange, the coherence of the individual spins that make up the magnetization will decay due to interaction between spins. This will cause a reduction in the transverse magnetization characterized by the spin-spin relaxation time T_2 equation (2.34). In order to excite the magnetization out of its thermal equilibrium, energy must be added at or near the Larmor frequency, equation (2.59). This is generally accomplished with a brief radiofrequency pulse or RF-pulse. After excitation, the magnetization in the transverse plane will oscillate at the Larmor frequency within a decay envelop defined by T_2 . This oscillating decaying signal is referred to as the free induction decay (FID) and is the fundamental signal detected for all nuclear magnetic resonance phenomenon. The solution to equation (2.44) is displayed in figure 2-1.

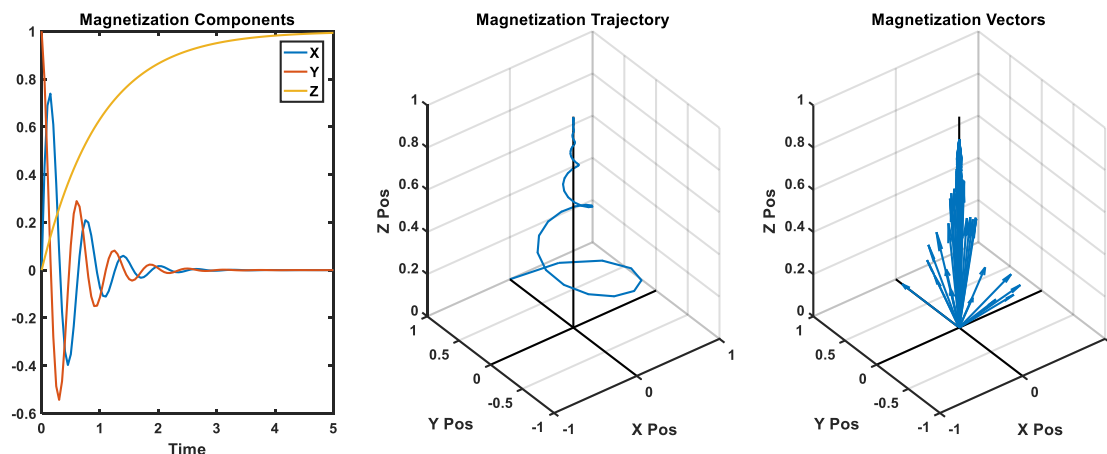


Figure 2-1. Visualization of the Bloch equations. Magnetization vector during a free induction decay visualized three different ways. (Left) the components of a magnetization in three space. Note that only the X and Y components will be detectable in a conventional magnetic resonance signal. (Center) the trajectory of a magnetization vector illustrating the classical corkscrew shape. (Right) a sampling of the magnetization vector over time as it follows the trajectory in the center panel.

Section 2.6 Chemical Shift

The previous sections have all dealt with the effects that magnet fields have on dipole moments from nuclear spins. However, the electrons that orbit nuclei generate dipole moments of their own. There are two main effects to consider when discussing how orbital electrons interact with external magnetic fields³⁴; the effect from the electrons' inherent spin which gives rise to paramagnetism; and the effect that the magnetic field has on orbital motion, which gives rise to diamagnetism. Paramagnetism arises in much the same way that nuclear magnetization arises, the electron's inherent spin results in a dipole moment that aligns with the applied field. However, this effect is only dominant in particular atoms due to the Pauli exclusion principle for electrons, and even for those atoms it is reduced by thermal energy affecting the fraction of alignment. Therefore, paramagnetic effects are quite rare and will be ignored.

Diamagnetism, arises from the change in the electron orbital motion due to the applied magnetic field and results in an overall reduction of the magnetic field. A quantum mechanical derivation of diamagnetism is beyond the scope of this discussion, but for a classical approximations readers are directed to reference [34]³⁴. The reduction of the external magnetic field can be conceptualized to result from an induced magnetic field B_{ind} that oppose B_0 ³⁴. The reduced magnetic field at a nuclei caused by such induced fields is referred to as chemical shielding and is given by:

$$B_{shifted} = B_0 - B_{ind} \quad 2.61$$

Since the induced field B_{ind} is determined by the external magnetic field B_0 , equation (2.61) is normally simplified to relate $B_{shifted}$ to B_0 with a chemical shielding constant σ ³⁷:

$$B_{shifted} = (1 - \sigma)B_0 \quad 2.62$$

Chemical shielding will depend only on the chemical structure of the molecule containing the nuclei of interest. Chemical shielding will not be similar to the random isotropic fields that gave rise to T_2 decay but will be constant and identical for all nuclei in a particular position in a molecule. If there is a chemical species-dependent deviation to the magnetic field, then by equation (2.18) there should be a shift in Larmor frequency³¹:

$$\vec{\omega} = -\gamma(1 - \sigma)\vec{B}_0 \quad 2.63$$

Equation (2.63) relates the frequency of the detectable Larmor precession to the chemical composition of the molecules that produce them. Therefore, spectral analysis of the signal resulting from the Larmor precession will yield information on the chemical structure of the compounds giving rise to the nuclear magnetic resonance signal.

Section 2.7 Fourier Spectroscopy

The Fourier transform can be used to spectrally analyze the frequency components of a signal. Conceptually the Fourier transform, as it relates to NMR, decomposes a time domain signal into its

frequency components. Consider a function of time $g(t)$ the Fourier transform of the function $\mathcal{F}\{g(t)\}$ is defined as³⁷:

$$G(\xi) = \mathcal{F}\{g(t)\} \equiv \int_{-\infty}^{\infty} g(t) e^{-i2\pi\xi t} dt \quad 2.64$$

When relating ξ and t by the Fourier transform as above they are referred to as Fourier conjugates.

Again, for magnetic resonance spectroscopy the two domains related by the Fourier transform are the time domain, normally in units of seconds, and the frequency domain, normally in units of Hz. Fourier transforms are invertible and the inverse Fourier transform \mathcal{F}^{-1} will be:

$$g(t) = \mathcal{F}^{-1}\{G(\xi)\} \equiv \int_{-\infty}^{\infty} G(\xi) e^{i2\pi t \xi} d\xi \quad 2.65$$

Note that by equation (2.64) and (2.65) $g(t) = \mathcal{F}^{-1}[\mathcal{F}\{g(t)\}]$ and that either the time domain or the frequency domain signals are sufficient to determine the other. Additionally, equation (2.18) is a relationship between the Larmor frequency in units of *radians/second* and not Hz which is the frequency domain defined in equation (2.64). Angular frequency and Hz are easily related by a constant of 2π , and the Fourier transform between the time domain and angular frequency is only slightly different than equations (2.64) and (2.65). As a final note, the Fourier transforms described above are continuous with all possible frequencies or time points in the integral, whereas in practice the data from NMR are discrete and therefore the integral is replaced with a sum over all measured time points and the corresponding sampling bandwidth. This discrete form of the Fourier transform is called the discrete Fourier transform.

The FIDs associated with the magnetic resonance signal of a single chemical species will be a damped sinusoid as determined by equation (2.44) following:

$$f(t) = e^{-i\omega_0 t - \frac{t}{T_2}} \quad 2.66$$

$$g(\omega) = \mathcal{F}\{f(t)\} = \frac{c}{\frac{1}{T_2} + i(\omega - \omega_0)} \quad 2.67$$

where c is a constant related to the initial magnitude of $f(t_0)$. Equation (2.67) is the classic Lorentzian line shape as seen in figure 2.2.

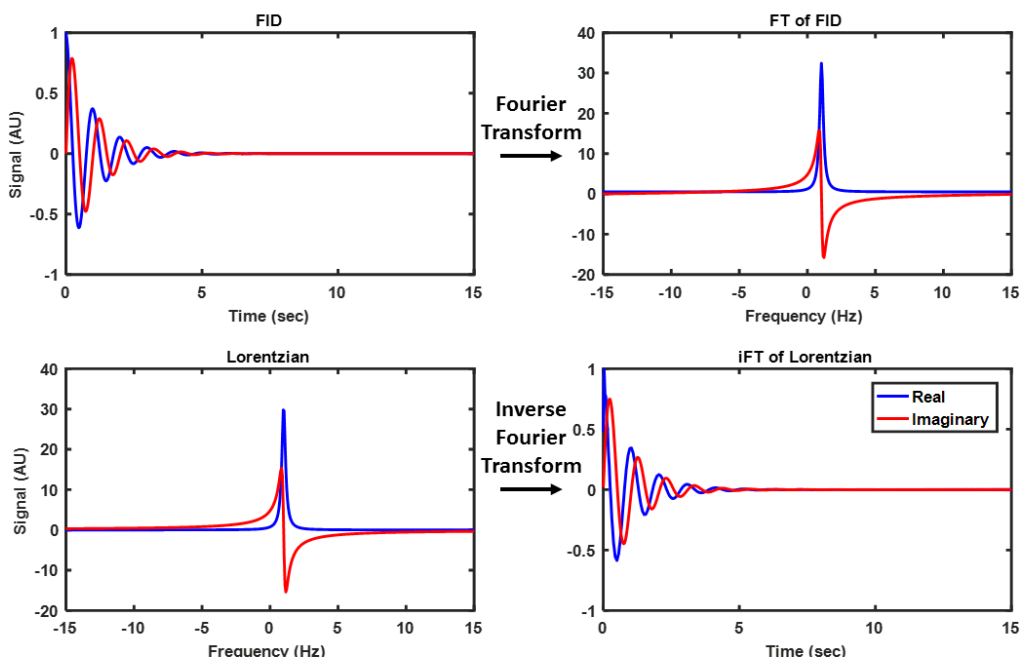


Figure 2-2. Free induction decay and Lorentzian line shape. The top panel show the real and imaginary parts of a FID following equation (2.66) and the resulting Fourier transform. The lower panel shows the resulting Lorentzian, equation (2.67), with the same T_2 and ω as the top panel, as well as the resulting inverse Fourier transform.

Equations (2.66 and 2.67) are complex functions and give rise to what are referred to as the absorption and dispersion parts of the signal. The relationship between the absorption and dispersion, or just the real and imaginary parts of an NMR spectrum is determined by the phase of the signal. The phase of the signal relates to the position of the magnetization vector in the transverse plane. If there is

a synchronization mismatch between the frequency of the excitation pulse and the frequency reference of the receive system there will be a phase shift in the entire signal of a constant value. This is referred to as the 0th order phase. Additionally, if there are multiple resonance signals in a single FID, then any temporal delay will give rise to an additional 1st order phase shift as each resonance signal will have a different resonance frequency and therefore impart a slightly different phase shift over the same timing mismatch. Such phase discrepancies can be corrected by adding a phase shift to the FID prior to Fourier transform given by:

$$f'(t) = e^{-i(\phi_0 + \phi_1 t)} \quad 2.68$$

where ϕ_0 and ϕ_1 are the zeroth and first order phase correction terms respectively. Phase correction of a NMR signal is shown in figure 2-3.

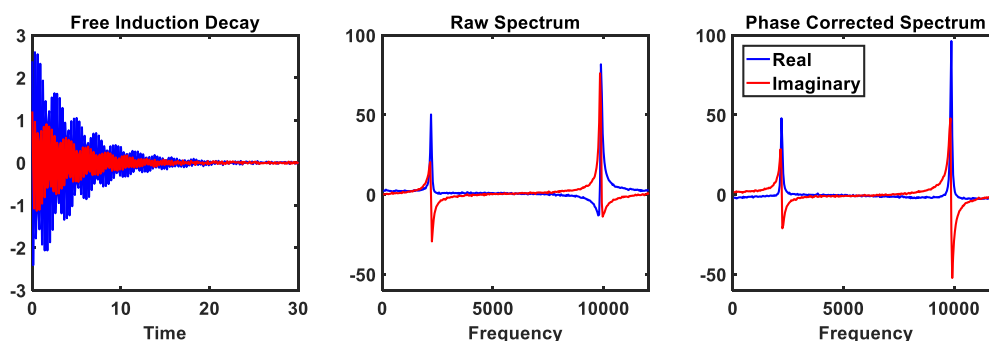


Figure 2-3. Fourier analysis and phase correction of a free induction decay. The complex FID (left) is decomposed into its spectral components by a Fourier transform. The peaks are then phase corrected (right) so its real component (blue) is completely positive.

Using the Fourier transform the time domain FID can be converted into its corresponding frequency components. Relative peak intensities determined by Fourier spectroscopy can be used to determine the relative concentrations of chemicals in some sample. This is the fundamental concept behind magnetic resonance spectroscopy (MRS). MRS is a powerful tool for determining a substance's chemical composition non-invasively³⁷.

Section 2.8 Clinical Magnetic Resonance Spectroscopy

In clinical practice most of the magnetic resonance signal comes from protons in either water or some lipid compound. While comparatively much smaller than the water or fat signal, other protons in the body will generate magnetic resonance signals. Using magnetic proton resonance spectroscopy to non-invasively probe a tissues' chemical composition is a powerful clinical procedure for specific diseases and anatomical locations. However, MRS is generally signal-limited, making it difficult to perform in most of the body. Clinical MRS usually performed on relatively homogeneous organs that are either naturally stationary or can be easily immobilized, including the brain and the prostate. This allows multiple spectroscopic scans to be performed to allow signal averaging. By averaging the signals from repeated measurement, spectral peaks can be enhanced, as they will add coherently while the random noise will add incoherently. Even with large numbers of averaged acquisitions clinical MRS focuses on only a few compounds that are relatively abundant in some tissue or disease type.

The reason clinical MRS is so limited can be found in the Boltzmann distribution equation (2.22). With a spin $\frac{1}{2}$ particle, the probabilities of being in the spin up (P_+) and spin down (P_-) position given by:

$$P_{\pm} = \frac{e^{\pm \frac{u}{2}}}{e^{\frac{u}{2}} + e^{-\frac{u}{2}}} \quad 2.69$$

where $u = \frac{\hbar \gamma B_0}{kT}$. The number of excess spins (N_{excess}) will be the difference in the numbers of spin up and spin down ($N(P_+)$ and $N(P_-)$ respectively)³¹:

$$N_{excess} \equiv N(P_+) - N(P_-) \quad 2.70$$

$$N_{excess} = N \left(\frac{e^{\frac{u}{2}}}{e^{\frac{u}{2}} + e^{-\frac{u}{2}}} - \frac{e^{-\frac{u}{2}}}{e^{\frac{u}{2}} + e^{-\frac{u}{2}}} \right) = N \left(\frac{e^{\frac{u}{2}} - e^{-\frac{u}{2}}}{e^{\frac{u}{2}} + e^{-\frac{u}{2}}} \right) = \tanh \left(\frac{u}{2} \right) \quad 2.71$$

where N is the total number of spins. Notice that as the term in the exponent approaches zero, normally by B_0 approaching zero, N_{excess} also approaches zero, and that as the exponential term gets larger N_{excess} approaches N . For protons with a clinically reasonable $B_0 = 1.5 \text{ T}$ at body temperature (310K), $u = 6.6 \times 10^{-6}$ then by equation (2.71) the spin excess is nearly 5×10^{-6} or 5 parts per million (ppm)³¹. Therefore, out of every million protons in the body only about five can give rise to any detectable signal by magnetic resonance. This is not catastrophic from compounds such as water or fat, which are abundant in the body. However, for other biologic compounds like metabolites there are simply not enough molecules in the body to generate a robust magnetic resonance signal with a high signal-to-noise ratio (SNR) and good spatial resolution.

The problems of a weak signal due to the low biologic abundance of compounds is exacerbated when nuclei other than protons are considered. Other nuclei can have a non-zero spin and therefore can be detected by magnetic resonance. Table 2.1 briefly summarizes the commonly detected nuclei for biologic magnetic resonance spectroscopy.

Nuclei Natural Abundance Gyromagnetic Ratio Relative Sensitivity

^1H	99.98 %	42.6 MHz/T	100 %
^{13}C	1.11 %	10.7 MHz/T	1.6 %
^{19}F	100 %	40.1 MHz/T	83 %
^{23}Na	100 %	11.3 MHz/T	9.3 %
^{31}P	100%	17.2 MHz/T	6.63%

Table 2-1. Properties of nuclei commonly detected by magnetic resonance spectroscopy.

While nuclei other than protons give rise to less relative signal, they additionally tend to be far less numerous in the body compared to protons. Carbon, with its central role in organic chemistry, could

provide critical information about the biochemical state of tissue. The potential to detect usable magnetic resonance signal from ^{13}C containing molecules has driven technical advances seeking to increase the excess spin population of ^{13}C nuclei far surpassing its thermal equilibrium magnetization. Increasing the excess spin population beyond thermal equilibrium can overcome the signal limitations imposed by equation (2.71) and would allow real time MRS of select ^{13}C compounds.

Section 2.9 Dynamic Nuclear Polarization

A brief outline of dynamic nuclear polarization will be presented followed by a description of a few quantitative models³⁸. Dynamic nuclear polarization (DNP) is a process that can transiently increase the polarization of a spin population to near unity³⁹⁻⁴¹. This is achieved by mixing a small amount of paramagnetic impurities with a diamagnetic material and cooling the mixture well into the solid state. If the paramagnetic impurities contain unpaired electrons and the diamagnetic material is a ^{13}C -enriched compound, both the electrons and the ^{13}C nucleus will be particles of spin 1/2. Due to a large discrepancy in the charge-to-mass ratio between the electron and the carbon nucleus and their subsequent gyromagnetic ratios, equation (2.8 and 2.71) shows that electrons are polarized to near unity (99.8%) at around 1.4 Kelvin while the ^{13}C nucleus will remain relatively un-polarized (0.13%) at a field strength of 3.35 T as shown in figure 2-4.

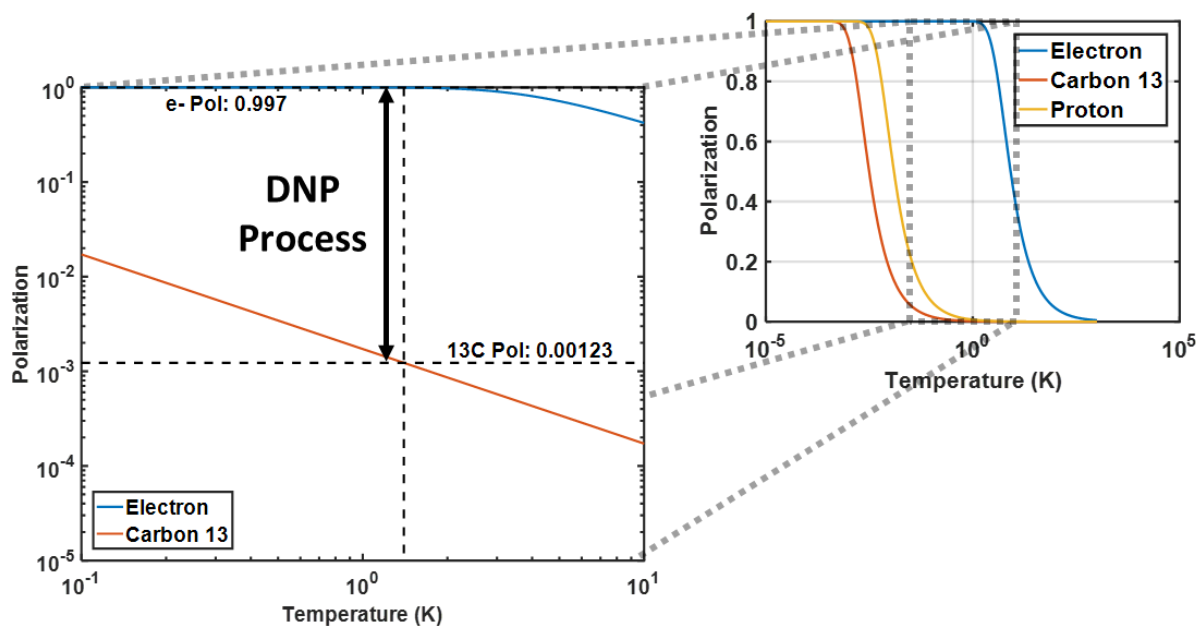


Figure 2-4. Dynamic nuclear polarization. Polarization as a function of temperature at 3.35T as predicted by equation (2.71). At 1.4K electrons, protons and ^{13}C polarize to 0.998, 0.00527 and 0.00132 respectively due to differences in their gyromagnetic ratios.

Consider an isolated electron and ^{13}C nucleus as a dipole pair as shown in figure 2-5. The coupled system will have four possible spin states with moments $\langle \uparrow_e \uparrow_n \rangle$, $\langle \uparrow_e \downarrow_n \rangle$, $\langle \downarrow_e \uparrow_n \rangle$, and $\langle \downarrow_e \downarrow_n \rangle$, where the electron dipole moment is \uparrow_e and the nuclear dipole moment is \uparrow_n and \uparrow is aligned with the field while \downarrow is aligned against. Transitions between these possible energy states will be governed by the energy added to the system by T_1 relaxation. Assuming that the electrons are fully polarized, the states with spin down electrons will be completely unpopulated leaving just the states $\langle \uparrow_e \uparrow_n \rangle$ and $\langle \uparrow_e \downarrow_n \rangle$ as the primary states.

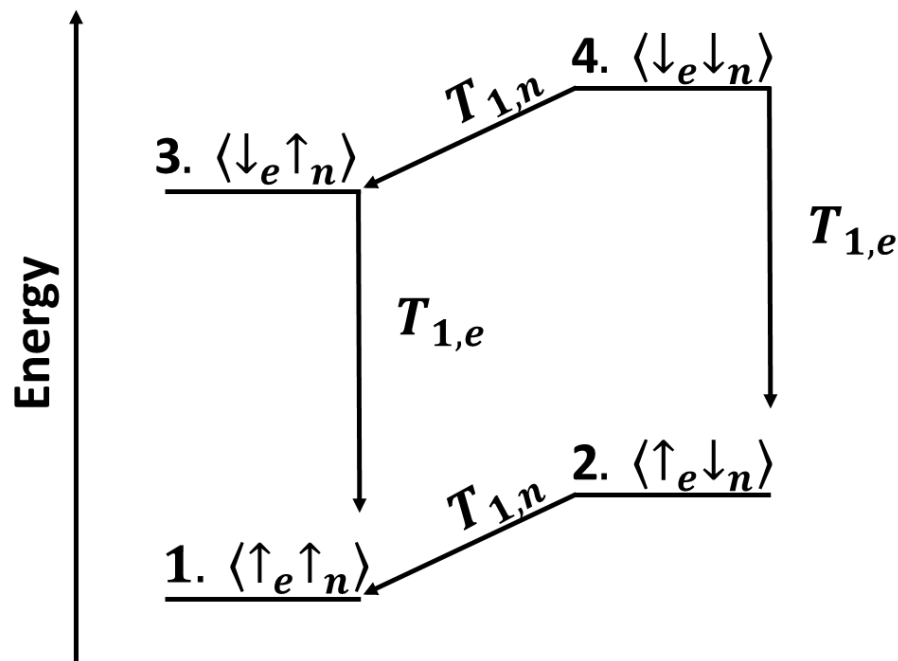


Figure 2-5. Nucleus electron pair energy diagram. An energy diagram of an electron-nuclear pair in a high magnetic field with no di-polar interactions considered.

From these states, there are two possible dipolar interactions. So called flip-flips will completely reverse the entire spin state or $\langle \uparrow_e \uparrow_n \rangle \rightarrow \langle \downarrow_e \downarrow_n \rangle$ and will require energy $\hbar(\omega_E + \omega_C)$, where ω_E and ω_C are the Larmor frequencies of the electrons and carbon nuclei, respectively. The other transition, so called flip-flops, will be of the form $\langle \uparrow_e \downarrow_n \rangle \rightarrow \langle \downarrow_e \uparrow_n \rangle$ and will require energy $\hbar(\omega_E - \omega_C)$. The energy required to induce these transitions is supplied by microwave irradiation with a either frequency $\Omega = \omega_E \pm \omega_C$. If the line width of the electron's Larmor frequency $\Delta\omega_E$ is much smaller than the resonance frequency for carbon, the energy spectrum of flip-flips and flip-flops will not overlap and only flip-flips or flip-flops transitions can be driven.

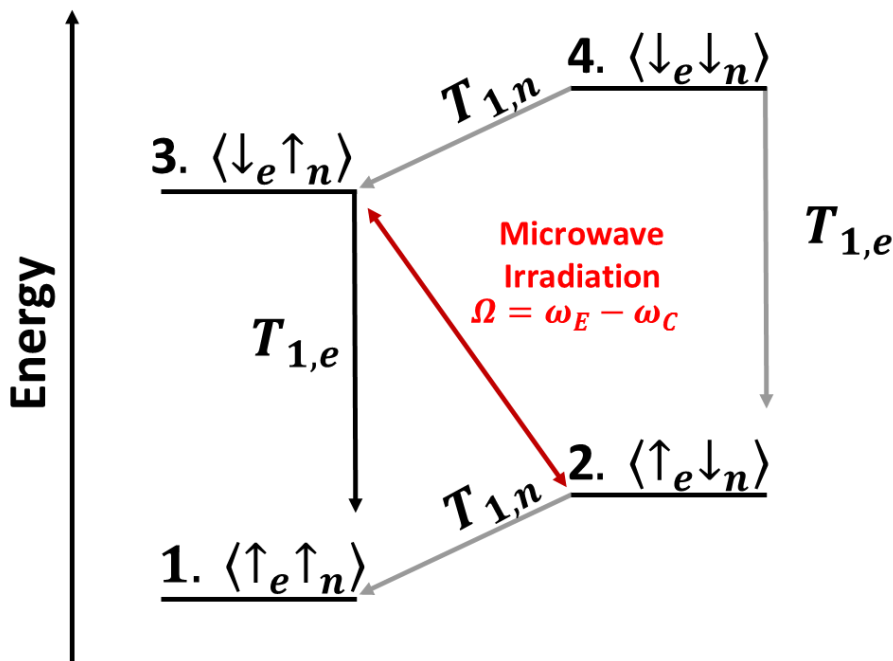


Figure 2-6. Nucleus electron energy diagram with flip-flops transitions. Dipole pairs are excited by an external microwave source from energy level 2. to level 3. Due to rapid $T_{1,e}$ relaxation the electron quickly relaxes transitioning the pair from level 3. to 1. At energy level 1. the electron can form a new dipole pair with a different carbon nucleus and the new pair would be at energy level 2.

If the driving microwave irradiation is tuned to only excite flip-flop transitions, as shown in figure 2-6, then after a flip-flop there will be an electron in the spin down state which could potentially induce a reverse flip-flop of the form $\langle \downarrow_e \uparrow_n \rangle \rightarrow \langle \uparrow_e \downarrow_n \rangle$. However, due to the interaction strength between the electron and the magnetic field, relaxation of an electron back to its low energy state is so rapid that the probability of reverse flip-flop transitions becomes vanishingly small. The relaxed spin up electron is then able to participate in another flip-flop interaction with a different carbon nucleus driven by the microwave irradiation. As this processes continues it becomes clear that over time the high polarization of the electrons will eventually be transferred to the carbon nuclei.

Thermal interactions between the carbon nuclei and the lattice will be acting to relax them back to their thermal equilibrium distribution. An equilibrium carbon polarization will eventually be achieved once the lower energy carbon spin population is so large that there are as many low energy carbons relaxing by T_1 relaxation as there are dipole flip-flop transitions being induced by the microwave irradiation. Qualitatively, the factors that affect this equilibrium value are, the T_1 relaxation times of both the carbon nuclei and the electron impurities, the ratio of electron impurities to carbon nuclei, the strength of the microwave driver, and the thermal equilibrium polarization of the electrons and carbons. Due to the vast difference in gyromagnetic ratio between the carbons and electron impurities, the T_1 relaxation times of the electrons will be much shorter than carbons, allowing a single electron to induce many flip-flop dipolar transitions before the resulting low energy carbons decay back to their thermal equilibrium values. Therefore, a single electron can facilitate polarization of many carbons.

For such an energy structure to exist, the system must be frozen well into the solid state and the electrons be spatially limited in the number of carbons they can interact with. The interaction between a low energy carbon and an adjacent higher energy carbon somewhat diminishes the effect of spatial isolation. A similar flip-flop dipolar transition is possible. However, it will not require any external energy because the total energy status of the dipole pair is unchanged. These carbon-to-carbon flip-flop transitions, also referred to as spin diffusion, allow relatively small numbers of electron impurities to hyperpolarize a large number of carbon nuclei. To hyperpolarize a large number of carbon nuclei with great efficiency the electrons need to be evenly distributed throughout the solid lattice. Compounds that form structured crystal lattices frequently will not uniformly distribute the electron impurities and therefore glassing solids are used for the majority of hyperpolarization preparations.

If the driving microwave irradiation is shut off, then the equilibrium maintained by the flip-flop dipolar transitions will be disrupted. The carbons will decay back to their thermal equilibrium distributions with their native T_1 ⁴². Depending on the compound and the relaxation enhancing

impurities, this can be on the order of hours in the solid state. A brilliant insight by Ardenkjaer-Larsen and others that even in the liquid state a hyperpolarized agent could have a T_1 on the order of tens of seconds^{25,43} has brought hyperpolarization to medical and imaging science. This allows a total lifetime of minutes, which is enough time to be used in liquid state magnetic resonance spectroscopy²⁷. In order to perform MRS on hyperpolarized agents in the liquid state, the solid state agent must be rapidly heated and delivered to the magnetic resonance system that will perform the measurements. In a process called dissolution DNP, the solid state system is flushed by a superheated fluid that allows rapid melting and delivery to an external system for subsequent measurement.

Section 2.10 Models of Dynamic Nuclear polarization

A. The Well-Resolved Solid Effect

The previous section described, in qualitative terms, a model of the nuclear Overhauser effect first proposed by Overhauser in 1953⁴⁰ and demonstrated by Slichter in conducting solids the same year³⁹. Dynamic nuclear polarization is generally achieved through four theoretical mechanisms^{38,41}, two of which apply to clinical and pre-clinical DNP: the well-resolved solid effect^{44,45} and thermal mixing^{46,47}. The derivation of each mechanism requires a full development of spin-temperature theory or density matrix formalism that is beyond the scope of this discussion. Additionally, the resulting models require assumptions that do not always hold for DNP in practice, and a general theoretical treatment of DNP under all conditions is still an area of active study⁴⁸. With these limitations in mind, some of the important components of the classical models will be presented and related to the clinical and preclinical use of DNP for ^{13}C nuclei.

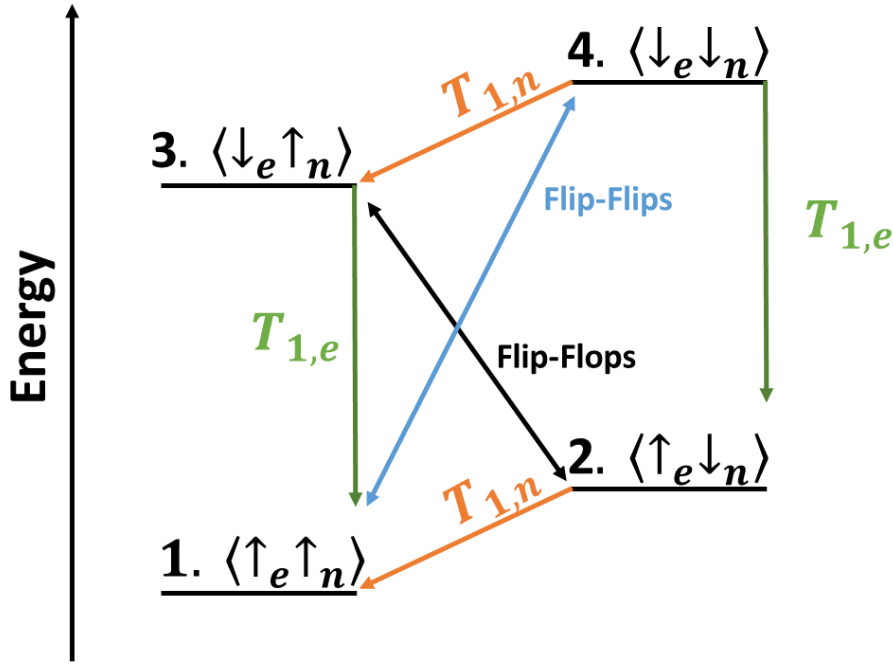


Figure 2-7. An energy diagram of an electron-nuclear pair with all dipolar interaction.

A two spin system composed of an electron and proton, as shown in figure 2-7, will have a Hamiltonian⁴⁵:

$$\mathcal{H} = \mathcal{H}_z + \mathcal{H}_{hf} + \mathcal{H}_{nn} + \mathcal{H}_{ee} + \mathcal{H}_{MW} \quad 2.72$$

where H_z corresponds to the Zeeman interactions, H_{hf} corresponds to the hyperfine interactions, H_{nn} and H_{ee} correspond to the dipole interactions for electrons and nuclei respectively, and H_{MW} corresponds interactions driven by an external microwave irradiation. If the microwave source is off ($H_{MW} = 0$) and equation (2.69) is solved,⁴⁵ the population, p_i of states 1-4 in figure 2-7 will be:

$$p_1 = p_2 = \frac{1}{2} \frac{1}{1 + e^{-\frac{\hbar \gamma_e B_0}{kT}}} \quad 2.73$$

$$p_3 = p_4 = \frac{1}{2} \frac{e^{-\frac{\hbar \gamma_e B_0}{kT}}}{1 + e^{-\frac{\hbar \gamma_e B_0}{kT}}}$$

This assumes that the nuclear Zeeman splitting is negligible compared to the electron splitting. The polarization of the nuclei, P_n , and electron, P_e , will follow:

$$\begin{aligned}
 P_n &= (p_1 - p_2 + p_3 - p_4) \\
 P_e &= (p_1 - p_3 + p_2 - p_4)
 \end{aligned}
 \tag{2.74}$$

By combining equations (2.73 and 2.74):

$$\begin{aligned}
 P_n &\cong 0 \\
 P_e &= \frac{1 - e^{-\frac{\hbar\gamma_e B_0}{kT}}}{1 + e^{-\frac{\hbar\gamma_e B_0}{kT}}} = -\tanh\left(-\frac{\hbar\gamma_e B_0}{2kT}\right) = \tanh\left(\frac{u_e}{2}\right)
 \end{aligned}
 \tag{2.75}$$

Essentially at thermal equilibrium the nuclei are completely unpolarized while the electrons follow the polarization predicted by equation (2.71).

In order to drive flip-flop transitions the microwave irradiation would need to provide $(\omega_{MW} = \omega_e - \omega_n)$. If the power of the microwave source is high enough to saturate the flip-flop transition, then $p_2(t) = p_3(t)$, and following equation (2.74):

$$P_n = P_e = \tanh\left(\frac{u_e}{2}\right) \tag{2.76}$$

If the microwave source was tuned to induce flip-flips $(\omega_{MW} = \omega_e + \omega_n)$, then $p_1(t) = p_4(t)$ and equation (2.74) shows:

$$P_n = -P_e = -\tanh\left(\frac{u_e}{2}\right) \tag{2.77}$$

Equations (2.76 and 2.77) give rise to classic signatures of the well-resolved solid effect. That is that properly tuned narrow band microwave irradiation will give rise to either positive or negative enhancement of the nuclear polarization. Additionally, the difference in the frequencies leading to enhancement will be twice the Larmor frequency of the nuclei, and they will be centered about the electron's paramagnetic resonance.

In practice there are far more interactions than the simplified model described above. Mostly they arise from the multitude of nuclei interacting with each electron. Interactions between electrons are weak because the well-resolved solid effect tends to occur when the electrons are dilute compared

to the nuclei and therefore spaced far apart. These additional nuclear interactions have two primary effects on the above results: they broaden the range of frequencies that give rise to enhancement, and they serve as an energy sink that reduces the efficiency of each electron to polarize surrounding nuclei⁴⁵. While not derived, the results of reference [38]³⁸ are shown in figure 2-8 outlining the two classical hallmarks of the well-resolved solid effect.

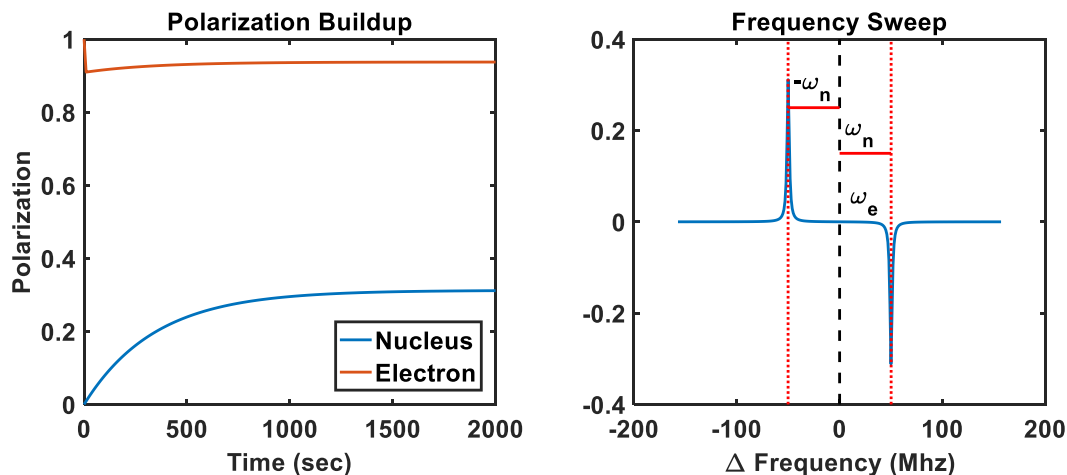


Figure 2-8. Polarization from the well-resolved solid effect. Theoretical polarization build up for an arbitrary electron nucleus system (left). Theoretical frequency sweep of the same system showing the two polarization peaks offset by ω_n from ω_e (right).

B. Thermal Mixing

While the well-resolved solid effect described by equations (2.72 – 2.77) and outlined in figures 2.7 and 2.8 can be the dominant effect in theory, the conditions are quite rare in practice. This is because situations where the electron spectral resonance line width is much narrower than the Larmor frequency of a nuclei are difficult to achieve and often require specific crystal lattices⁴¹. It is much more common that the electron spectral width will span both the flip-flip and flip-flop transition, leading to $\omega_n \ll \Delta\omega_e$. Under such conditions, driving a microwave source at any particular frequency near $\omega_E \pm \omega_C$ will induce flip-flips and flip-flops. The overlap of these transitions will degrade the nuclear

polarization achievable, and if the electron spectral width is much bigger than the Larmor frequency, any significant dynamic nuclear polarization by the solid effect becomes impossible. However, due to the interaction between some groups of electrons and some nuclei, it is still possible to achieve dynamic nuclear polarization, although the mechanism is different than the well-resolved solid effect. Dynamic nuclear polarization when the spread of the electron resonance is much larger than the Zeeman splitting of a nuclei is called thermal mixing^{46,47}.

Because thermal mixing requires the interaction of a large number of electrons a formalism has been developed that draw parallels to statistical mechanics. Spin temperature (β), which was first introduced by Redfield⁴⁹, is defined as:

$$\beta \equiv \frac{1}{kT} \quad 2.78$$

The population of each state defined by the Hamiltonian (\mathcal{H}) and the spin temperature given by:

$$P = \frac{e^{-\beta\mathcal{H}}}{A} \text{ \& } 1 = \sum P \quad 2.79$$

where A is simply a normalization constant.

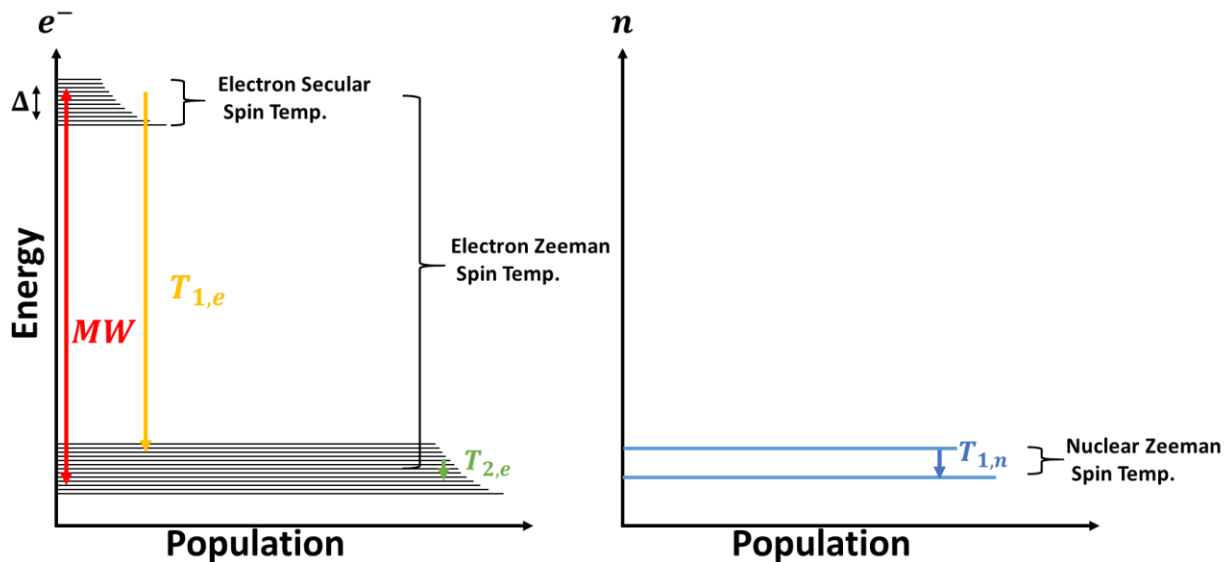


Figure 2-9. Energy diagram of the thermal mixing process. Due to the similar spin temperatures of the nuclear Zeeman bath, and the secular electron bath they are considered to be in strong thermal contact

($\beta_{eS} = \beta_{nZ}$) allowing their population distributions to match. Therefore, when the secular electron bath is cooled by interactions with the electron Zeeman bath facilitated by microwave ($MW = \omega_e + \Delta$) irradiation that cooling is transferred to the nuclear Zeeman bath.

Dynamic nuclear polarization arises from the interaction of the three spin temperatures that are outlined in figure 2-9^{38,46,47}. The first two spin temperatures describe the Zeeman splitting of the electrons and the nuclei, β_{eZ} and β_{nZ} respectively. These spin temperatures give rise to the population distributions discussed for the well-resolved solid effect when the microwave irradiation was off, equation (2.75). Additionally, now that the ensemble of electrons with a range of resonance frequencies is being considered, there is a third spin temperature (β_{eS}) referred to as the secular or non-Zeeman spin temperature that needs to be considered. Finally, similar to the solid effect, thermal mixing requires microwave irradiation. The frequency of irradiation, by contrast, will be shifted a small amount Δ from the electron's resonance and not necessarily ($\omega_e \pm \omega_n$), which was needed for the well-resolved solid effect. The evolution of all three spin temperatures has been described by Provotorov⁵⁰:

$$\begin{aligned} \frac{d\beta_{eZ}}{dt} &= -W(\beta_{eZ} - \beta_{nZ}) - \frac{1}{T_{1e}}(\beta_{eZ} - \beta'_{eZ}) \\ \frac{d\beta_{nZ}}{dt} &= \frac{d\beta_{eS}}{dt} = W\left(\frac{\Delta^2}{D^2}\right)(\beta_{eZ} - \beta_{nZ}) - \frac{1}{T_{1n}}(\beta_{nZ} - \beta'_{nZ}) \end{aligned} \quad 2.80$$

where W is the transition probability induced by the microwave irradiation, D is the electron linewidth and β' denotes the Zeeman spin temperatures due to interactions with the external lattice and defines thermal equilibrium. Because the spread in the electron resonance linewidth is considered to be comparable or large compared to the nuclear Zeeman splitting, the spin temperatures β_{eS} and β_{nZ} are considered to be in strong thermal contact^{38,46} and therefore equal. In the steady state $\frac{d\beta_{eZ}}{dt} = \frac{d\beta_{nZ}}{dt} = 0$ and the equilibrium value of the nuclear spin temperature is³⁸:

$$\beta_{nz} = \beta'_{nz} \frac{\omega_e}{\Delta} \frac{WT_{1n} \left(\frac{\Delta^2}{D^2} \right) / (1+f)}{1 + WT_{1e} + WT_{1n} \left(\frac{\Delta^2}{D^2} \right) / (1+f)} \quad 2.81$$

where f is an additional leakage term. Equation (2.81) yields the two hallmarks of DNP by thermal mixing. An antisymmetric response around the electron's resonance and a linear sloping zero crossing seen in figure 2-10.

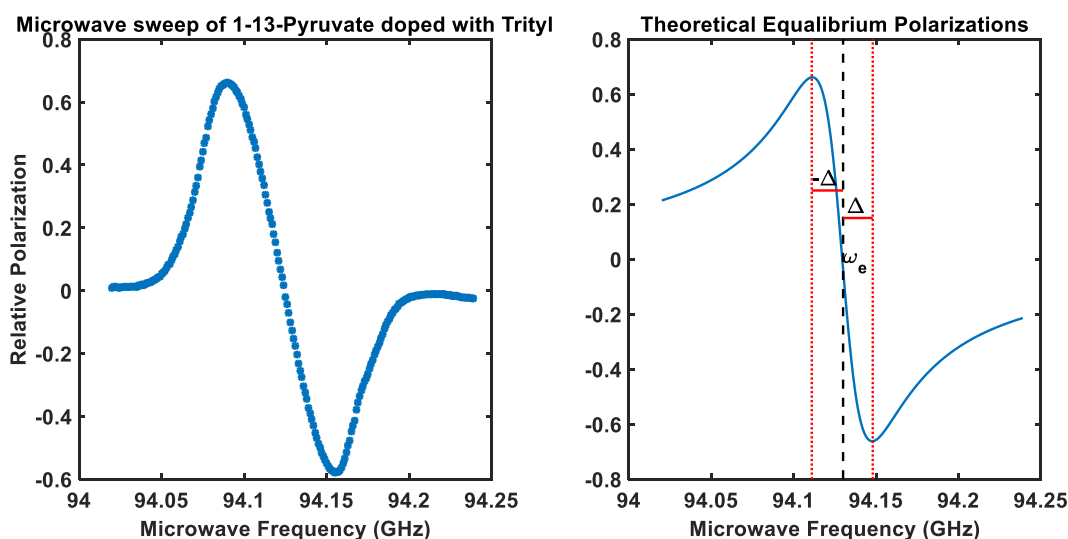


Figure 2-10. Microwave sweep of thermal mixing. Qualitative comparison of the polarization of 1-13C-pyruvic acid doped with 15 mM Ox063 Trityl as a function of microwave frequency and polarization as predicted by equation (2.81). The theoretical curve does not approach zero as quickly as the measured data as the microwave frequency diverges from the electron paramagnetic resonance. This is a limitation of the high temperature assumption³⁸ that leads to equation (2.81), for a more rigorous treatment of thermal mixing references [42] and [47] should be consulted.^{42,47}

Practically, dynamic nuclear polarization of carbon-13 nuclei is performed at temperatures below that of liquid helium with field strengths on the order of Tesla. Under these conditions, unpaired electrons in the polarizing radicals polarize to near unity. The paramagnetic impurity most commonly

used is the unpaired electron on a persistent radical, normally a proprietary triphenylmethyl derivative called Ox063. The large structure surrounding the central carbon of Ox063 is sterically crowded and therefore cannot easily react. This leaves a single unpaired valence electron radical that is chemically stable⁴². Using mechanisms described above, the high polarization of such radicals can be transferred to nuclei.

It has been shown that at a temperature of 1.4 K and a field strength of 3.35 T the electron linewidth of Ox063 is about 60 MHz, which is caused primarily by g-factor anisotropy when in a solution of 1-¹³C-pyruvic acid at a concentration of 15 mM⁴². The Larmor frequency of ¹³C at 3.35 T is 38.55 MHz and therefore the solid effect is not a plausible mechanism to polarize ¹³C. Protons however, have a Larmor frequency of 142.7 MHz and could be well polarized by an Ox063 radical using the solid effect⁴². With the C¹³ Larmor frequency well below the electron line width, thermal mixing will drive effective polarization enhancement of the C¹³. In order to increase polarization via thermal mixing, spin-spin interactions between the paramagnetic impurities need to be rapid and numerous. Therefore, the distance between the paramagnetic impurities cannot be excessive. As long as the solid state system is a glass the distance between the free radicals in Ox063 at 15 mM will be 5 nm, which is close enough to allow them to magnetically couple. If there is a crystalline structure in the solid state, the distance between the free radicals could be much larger and inhibit polarization. If the solute is not glass-forming, a glassing agent such as glycerol is often used. Additionally, it has been shown that small amounts of a Gd³⁺ compound can increase the steady state dynamic nuclear polarization of 1-C¹³-Pyruvic acid⁵¹. This is likely caused by a shortening of the electron's T_1 while the C¹³ T_1 is unaffected⁴². This will lead to an increased polarization as predicted by equation (2.81). The addition of Gd³⁺ has been shown to increase the polarization by a factor of as much as two. However, such effects are reduced at higher field strengths⁵².

In practice, the above considerations lead to the following general hyperpolarized setup. A single C^{13} -enriched compound, such as 1- ^{13}C -Pyruvic acid, is doped with ~15 mM of a radical and ~0.1 mM Gd^{+} chelate. The solution is rapidly cooled below 1.5K and is irradiated via microwaves while under a strong magnetic field. Typically build-up times are on the order of an hour, and nuclear polarizations of ~30% are common⁴³.

Polarizations of nuclei on the order of tens of percent represent a massive increase in the potential NMR signal. However, to be useful clinically, the polarized nuclei need to interact with some target biology. This will involve the removal from the microwave irradiation and significant heating to reach body temperature. Fortunately, once a polarization level is achieved it will return to thermal equilibrium at its T_1 relaxation rate. In the solid state, the T_1 of C^{13} enriched compounds is on the order of hours. However, once heated it is on the order of a minute. Therefore, if the process of heating, delivery to target biology and scanning are rapid there will be significant polarization remaining from the process of dynamic nuclear polarization. The rapid melting and delivery of highly dynamic nuclear polarized agents is referred to as dissolution dynamic nuclear polarization.

Section 2.11: Detection of Magnetic Resonance Signal

Detecting hyperpolarized agents through magnetic resonance is substantially different than conventional magnetic resonance imaging or spectroscopy, even though they operate according to the same principles. In conventional magnetic resonance, after excitation the excited spin system will return to thermal equilibrium as it interacts with its lattice through the process known as the spin-lattice relaxation or T_1 relaxation³¹. Additionally, there will be some dephasing of the transverse magnetization caused by spin-spin interactions, also referred to as the T_2 decay. These effects combine to generate the signal depicted in figure 2-11. The net magnetization, M_0 is excited to create a transverse magnetization, and longitudinal magnetization is initially reduced but will then recover following T_1 relaxation. The

transverse magnetization will oscillate at the Larmor frequency while decaying with time constant T_2 towards zero. Note that $T_2 \leq T_1$ or, the spin-spin relaxation will always be as fast or faster than the spin lattice relaxation. If serial excitation is performed faster than a few T_1 times, then a steady state magnetization will be achieved as seen in figure 2-11. This steady state magnetization is foundational to conventional magnetic resonance spectroscopy or imaging as it imparts contrast and allows the assumption of consistency between measurements.

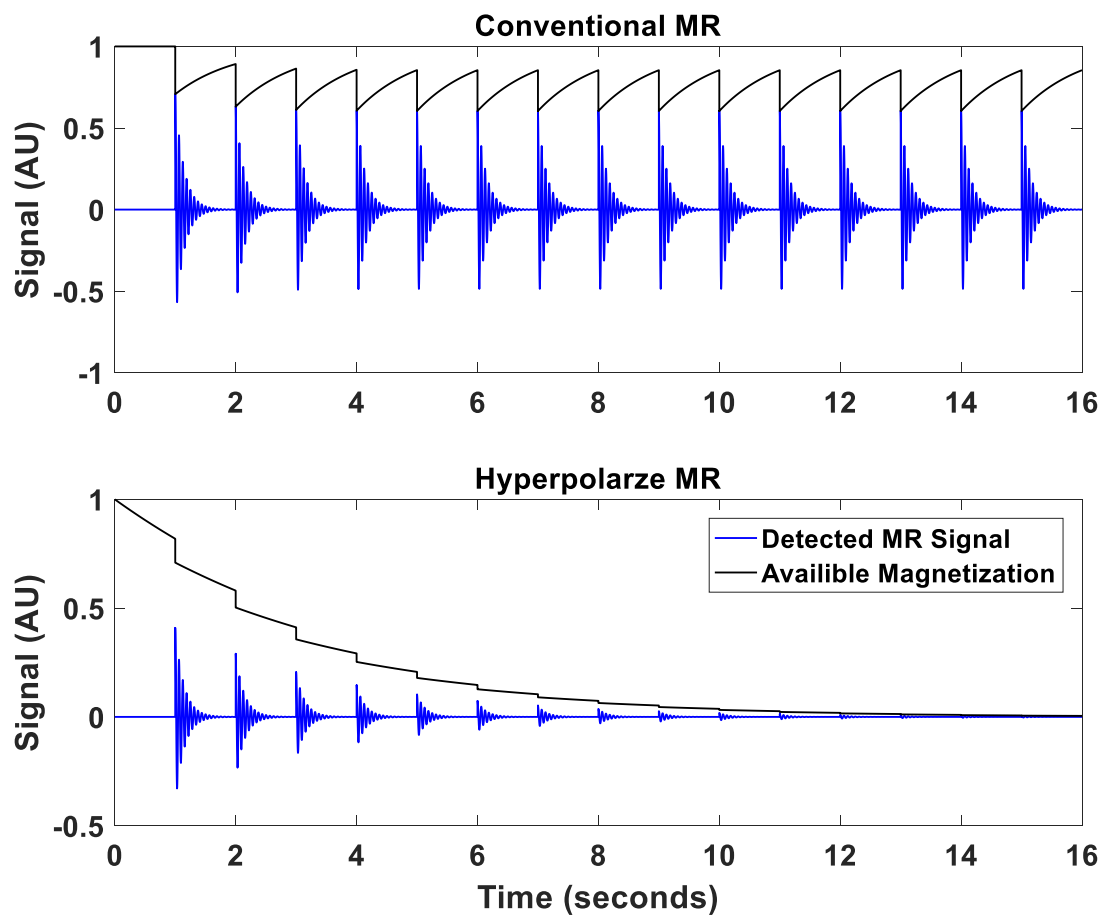


Figure 2-11. A comparison of a conventional magnetic resonance signal (top) and a hyperpolarized magnetic resonance signal (bottom). The conventional signal recovers with T_1 , eventually reaching a steady state signal. The hyperpolarized signal, by contrast, is constantly decaying and no steady state signal is achieved.

Unlike conventional magnetic resonance, hyperpolarized agents derive their longitudinal magnetization from the process of dynamic nuclear polarization. Once the DNP process is terminated the hyperpolarized will begin to decay with its inherent spin lattice relaxation time back to thermal equilibrium. As in conventional magnetic resonance, excitation of a hyperpolarized spin state will excite some or all of the longitudinal magnetization into the transverse plane where it can then be detected by a loop receiver coil. However, unlike conventional magnetic resonance, the hyperpolarized longitudinal magnetization will not recover after excitation. It will continue to decay. Due to this fundamentally transient magnetization, the steady state magnetization which is so often fundamental to conventional magnetic resonance is achieved only after longitudinal magnetization has decayed to undetectable levels. Therefore, while much of conventional wisdom and techniques associated with magnetic resonance do apply to hyperpolarized agents, many do not. A helpful analogy is to conceptualize the longitudinal magnetization of a hyperpolarized agent as a diminishing resource. Excitation into the transverse plane is necessary for signal detection and the amount of the resource consumed during an excitation will directly correlate to the strength of the signal detected. However, if serial measurements are to be made, some longitudinal magnetization will have to be conserved to be available for subsequent excitations and detection.

The simplest magnetic resonance study of a hyperpolarized agent is a single spectroscopic acquisition. This would be no different than the spectral acquisitions described in section 2.7 except for the substantially increased signal due to hyperpolarization. With the massive signal increase made possible by dynamic nuclear polarization, it is possible to serially excite the spin system using excitations that do not completely consumed the longitudinal magnetization. Such a serial excitation would allow for multiple spectral readouts. Each spectral readout could be treated independently using the same methods described in section 2.10. The only exception would be that the previous excitation would

diminish the remaining longitudinal magnetization for subsequent excitations. This can be partially accounted for either by keeping the excitation angles so low that they have negligible effect on future excitations compared to T_1 decay, but such low excitation could severely limit the signal. Alternatively, the resulting signal could be corrected with a simple scaling factor that accounts for all previous excitations or signal losses due to excitation which could be accounted for in quantification methods^{53,54}. This fundamental link between the detection strategy and the resulting signal evolution must be well characterized if reliable quantification methodologies are to be applied to hyperpolarized studies.

Chapter 3. Simulation of Hyperpolarized Studies

This chapter is intended to address Aim 1.

Section 3.1: Theory

Once the hyperpolarized signal is detected it needs to be processed. For hyperpolarized pyruvate, the study endpoint of interest is normally some metric of the rate of metabolic conversion of pyruvate to lactate²⁶. While some sense of metabolic rate can be determined from a qualitative analysis of signal curves from simple dynamic spectroscopy, a quantitative measure of the rate of conversion would allow much more specific information on the underlying biology. Many methods have been proposed to quantify the rate of conversion of hyperpolarized pyruvate to lactate. Simple methods such as the ratio of the pyruvate signal to the lactate signal⁵⁵, to more advanced methods that attempt to fit the signal evolution to some model of conversion^{53,56-60} have been proposed. Due to the non-renewable nature of a hyperpolarized signal, excitation for detection will affect all subsequent measurements. It is still unclear to what the extent such perturbations in signal evolution caused by detection alter quantitative strategies for detecting metabolic conversion of hyperpolarized pyruvate⁶¹.

With current technology, the process of generating hyperpolarized pyruvate is lengthy due to the need to build up a significant hyperpolarized state. Additionally, due to the hardware and reagent requirements, the creation of hyperpolarized pyruvate is still relatively costly compared to other magnet resonance agents²⁶. With these practical concerns in mind, exhaustively testing a range of acquisition strategies experimentally would be exceedingly expensive and difficult. Additionally, the quantitative parameter of interest, the apparent rate constant for chemical conversion, at a minimum will require dynamic chemical conversion. These requirements limit the systems available to explore how the sequence used in detection affects the quantitative results. Fortunately, there exist well-accepted

numeric models for all of the above considerations⁶². The physics behind the magnetic resonance phenomenon is well described using the Bloch equations. The Bloch equations can be adapted to account for chemical exchange between two distinct chemical species, in that form, they are referred to as the Bloch-McConnell equations⁶³. Delivery of a magnetic resonance contrast agent via endogenous vasculature has been described by Tofts in the case of gadolinium^{64,65} and adapted for hyperpolarized agents by Bankson⁵³. By combining all of these models, it should be possible to numerically simulate the critical aspects of a realistic magnetic resonance study of hyperpolarized pyruvate. Such a simulation platform would be able to explore how detection strategies affect the resulting hyperpolarized signal and any subsequent quantization across a wide range of biologic and sequence parameters⁶².

Recall that the Bloch equation described in chapter 2:

$$\frac{d\vec{M}}{dt} = \gamma(\vec{M} \times \{\vec{B}_0 + \vec{B}_1(t)\}) + \frac{1}{T_1}(\vec{M}_0 - \vec{M}_{\parallel}) - \frac{1}{T_2}M_{\perp} \quad 2.44$$

where \vec{M} is the magnetization vector with longitudinal and transvers components \vec{M}_{\parallel} and M_{\perp} respectively, γ is the gyromagnetic ratio, \vec{B}_0 is the static magnetic field, $\vec{B}_1(t)$ is some time varying magnetic field, T_1 is the spin lattice relaxation time, \vec{M}_0 is the equilibrium magnetization, and T_2 is the spin-spin relaxation time. In the case of hyperpolarized carbon, it is generally assumed that the contribution to the signal from thermal polarization is negligible i.e., $\vec{M}_0 \ll \vec{M}(t)$, even as $\vec{M}(t)$ approaches zero due to T_1 relaxation. Therefore, \vec{M}_0 can be neglected and the Bloch equation for hyperpolarized C^{13} agents then becomes:

$$\frac{d\vec{M}}{dt} = \gamma(\vec{M} \times \{\vec{B}_0 + \vec{B}_1(t)\}) + \frac{1}{T_1}(\vec{M}_{\parallel}) - \frac{1}{T_2}M_{\perp} \quad 3.1$$

In order to account for multiple chemical species, equation (3.1) needs to be expanded into a matrix form. Additionally, with the removal of M_0 , the T_1 and T_2 terms can be combined resulting in:

$$\frac{\delta}{\delta t} \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = \gamma \begin{bmatrix} M_y B_z - M_z B_y \\ M_z B_x - M_x B_z \\ M_x B_y - M_y B_x \end{bmatrix} + \begin{bmatrix} \frac{1}{T_2} & 0 & 0 \\ 0 & \frac{1}{T_2} & 0 \\ 0 & 0 & \frac{1}{T_1} \end{bmatrix} \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} \quad 3.2$$

For simplicity equation (3.2) combines B_0 and $B_1(t)$ into a single B . With the inclusion of two chemical species equation (3.2) becomes:

$$\frac{\delta}{\delta t} \begin{bmatrix} M_{x,a} \\ M_{y,a} \\ M_{z,a} \\ M_{x,b} \\ M_{y,b} \\ M_{z,b} \end{bmatrix} = \gamma \begin{bmatrix} (1 - \sigma_a)(M_{y,a} B_z - M_{z,a} B_y) \\ (1 - \sigma_a)(M_{z,a} B_x - M_{x,a} B_z) \\ (1 - \sigma_a)(M_{x,a} B_y - M_{y,a} B_x) \\ (1 - \sigma_b)(M_{y,b} B_z - M_{z,b} B_y) \\ (1 - \sigma_b)(M_{z,b} B_x - M_{x,b} B_z) \\ (1 - \sigma_b)(M_{x,b} B_y - M_{y,b} B_x) \end{bmatrix} + \begin{bmatrix} \frac{1}{T_{2,a}} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{T_{2,a}} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{T_{1,a}} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{T_{2,b}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{T_{2,b}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{T_{1,b}} \end{bmatrix} \begin{bmatrix} M_{x,a} \\ M_{y,a} \\ M_{z,a} \\ M_{x,b} \\ M_{y,b} \\ M_{z,b} \end{bmatrix} \quad 3.3$$

where σ is the chemical shielding term and the second subscript a or b is used to denote the separate chemical species. Equation (3.3) is little more than a combination two forms of equation (3.2). Without any cross terms, the chemical species a and b are completely independent. In order to couple the two chemical species, some chemical exchange term needs to be added.

Chemical exchange between two chemical pools will be given by:⁶⁶



where k_1 and k_2 are the forward and reverse apparent exchange rates respectively. Equation (3.4) can be written in matrix form following:

$$\frac{\partial}{\partial t} \begin{bmatrix} [A](t) \\ [B](t) \end{bmatrix} = \begin{bmatrix} -k_1 & k_2 \\ k_1 & -k_2 \end{bmatrix} \begin{bmatrix} [A](t) \\ [B](t) \end{bmatrix} \quad 3.5$$

where $[A]$ and $[B]$ are the concentrations of A and B respectively. Equations (3.5) and (3.3) can be combined into^{63,67-69}:

$$\frac{\delta}{\delta t} \begin{bmatrix} M_{x,a} \\ M_{y,a} \\ M_{z,a} \\ M_{x,b} \\ M_{y,b} \\ M_{z,b} \end{bmatrix} = \gamma \begin{bmatrix} (1 - \sigma_a)(M_{y,a}B_z - M_{z,a}B_y) \\ (1 - \sigma_a)(M_{z,a}B_x - M_{x,a}B_z) \\ (1 - \sigma_a)(M_{x,a}B_y - M_{y,a}B_x) \\ (1 - \sigma_b)(M_{y,b}B_z - M_{z,b}B_y) \\ (1 - \sigma_b)(M_{z,b}B_x - M_{x,b}B_z) \\ (1 - \sigma_b)(M_{x,b}B_y - M_{y,b}B_x) \end{bmatrix} - \begin{bmatrix} \frac{1}{T_{2,a}} + k_1 & 0 & 0 & -k_2 & 0 & 0 \\ 0 & \frac{1}{T_{2,a}} + k_1 & 0 & 0 & -k_2 & 0 \\ 0 & 0 & \frac{1}{T_{1,a}} + k_1 & 0 & 0 & -k_2 \\ -k_1 & 0 & 0 & \frac{1}{T_{2,b}} + k_2 & 0 & 0 \\ 0 & -k_1 & 0 & 0 & \frac{1}{T_{2,b}} + k_2 & 0 \\ 0 & 0 & -k_1 & 0 & 0 & \frac{1}{T_{1,b}} + k_2 \end{bmatrix} \begin{bmatrix} M_{x,a} \\ M_{y,a} \\ M_{z,a} \\ M_{x,b} \\ M_{y,b} \\ M_{z,b} \end{bmatrix} \quad 3.6$$

The coupling between the chemical pools has computational implications when attempting to solve equation (3.6) numerically. During times when there is no radiofrequency excitation, $B_1(t) = 0$, and equation (3.2) becomes a well behaved exponential decay in the rotating frame. For most conventional sequences, the excitation pulses are short and relatively infrequent compared to times when they are not present. This allows a substantial speeding up of computation in the rotating frame by many orders of magnitude when physical values for B_0 and γ are used. Even when there are $B_1(t)$ pulses, if they are close to the Larmor frequency then the transformed field B_{eff} will cause only minor deviations from a simple exponential decay and the computational burden will be minimal. However, once the two chemical pools have been coupled by an exchange term, moving into a rotating frame begins to offer a

reduced computational advantage. This is because there are now two Larmor frequencies to account for in the frame shift corresponding to each chemical species. If the chemical species are well-separated such that there is a few ppm of separation between them, even a moderate magnetic field can result in a few kHz of difference in their Larmor frequencies. Solutions with rapid oscillation pose a sizeable computational burden when solved numerically. The step size for a numeric computation has to be small compared to the frequency of oscillation resulting in step sizes on the order of microseconds for well-separated chemical species.

Fortunately, under certain conditions, there exists a closed form solution to equation (3.6). In order to arrive at the closed form solution to equation (3.6), the cross product terms can be combined with the decay terms yielding:

$$\frac{\delta}{\delta t} \begin{bmatrix} M_{x,a} \\ M_{y,a} \\ M_{z,a} \\ M_{x,b} \\ M_{y,b} \\ M_{z,b} \end{bmatrix} = A \begin{bmatrix} M_{x,a} \\ M_{y,a} \\ M_{z,a} \\ M_{x,b} \\ M_{y,b} \\ M_{z,b} \end{bmatrix} \quad 3.7$$

where:

$$A = \gamma \begin{bmatrix} -\frac{1}{T_{2,a}} - k_1 & (1 - \sigma_a)B_{eff,x}(t) & (1 - \sigma_a)B_{eff,y}(t) & k_2 & 0 & 0 \\ (1 - \sigma_a)B_{eff,x}(t) & -\frac{1}{T_{2,a}} - k_1 & -(1 - \sigma_a)B_{eff,x}(t) & 0 & k_2 & 0 \\ -(1 - \sigma_a)B_{eff,y}(t) & (1 - \sigma_a)B_{eff,x}(t) & -\frac{1}{T_{1,a}} - k_1 & 0 & 0 & k_2 \\ k_1 & 0 & 0 & -\frac{1}{T_{2,b}} - k_2 & (1 - \sigma_b)B_{eff,x}(t) & (1 - \sigma_b)B_{eff,y}(t) \\ 0 & k_1 & 0 & (1 - \sigma_b)B_{eff,x}(t) & -\frac{1}{T_{2,b}} - k_2 & -(1 - \sigma_b)B_{eff,x}(t) \\ 0 & 0 & k_1 & -(1 - \sigma_b)B_{eff,y}(t) & (1 - \sigma_b)B_{eff,x}(t) & -\frac{1}{T_{1,b}} - k_2 \end{bmatrix} \quad 3.8$$

B_{eff} is simply the magnetic field in any arbitrary rotating frame, and σ_a and σ_b are the chemical shielding terms for the chemical species a and b respectively. If there is no active radio frequency pulse, then B_{eff} is no longer time-dependent and A also becomes time independent. The closed form solution to equation (3.7) when A is constant in time is⁷⁰:

$$\vec{M}(t) = e^{At} \vec{M}(0) \quad 3.9$$

where, \vec{M} is the combined vector for both chemical species. Note that in equation (3.9) the exponent represents matrix exponentiation. If there is some time-varying magnetic field, then equation (3.9) breaks down. However, equation (3.6) can still be solved numerically during such times. Therefore, by combining a numerical solver with the analytical solution, a large reduction in computation time can be achieved when the radio frequency pulses do not occupy a majority of the calculation time. By solving equation (3.7) or its closed form under the right conditions, equation (3.9), it is possible to simulate the chemical exchange of a hyperpolarized agent.

The simplest model for pyruvate delivery would be to assume instantaneous delivery as a delta function bolus, or that all of the pyruvate that will arrive during the study does so at $t = 0$. This approximation is little more than a boundary condition on equation (3.7) and does not represent a good model of perfusion. A second model would be to allow a driving input function for the pyruvate or lactate magnetizations over time, $\vec{b}(t)$. This would change equation (3.7) to:

$$\frac{\delta \vec{M}}{\delta t} = A\vec{M} + c\vec{b}(t) \quad 3.10$$

where c is an exchange constant between the vascular delivery and the system of interest. Equation (3.10) also has a closed form solution given by⁷⁰:

$$\vec{M}(t) = e^{At}\vec{M}(0) + c \int_0^t e^{A\tau}\vec{b}(\tau) d\tau \quad 3.11$$

Equation (3.11) has a computational consideration as the integral will need to be evaluated numerically. However, for most cases the computation of a single definite integral will be much more efficient than numerically solving a rapidly oscillating system, and therefore equation (3.11) still represents a significant speedup over (3.10). Note there are some forms of A that run into discretization issues when computed. The decay terms in A act as forcing functions that drive any magnetization, either transverse or longitudinal, eventually to zero over a long enough time. These forcing terms eventually become so large, that, depending on the programming language used, they can result in infinite numbers that destroy

computational fidelity. However, under these unstable conditions it is generally safe to assume that there is no signal, as the initial signal would have to have been huge in order to be able to outlast a forcing function that pushed the discretization limits of a system. It would have to be so large that it would likely have its own discretization issues. Therefore, simply replacing any poorly defined numeric results in the computation of the integral in equation (3.11) with zeros sufficiently resolves this issue. Alternatively, computation intervals that are long enough to push the forcing terms to their discretization limits can also be split and recalculated. By combining such splitting with a recursive algorithm, it is possible to solve for an arbitrarily long time interval using equation (3.11).

If a more complicated model of perfusion is to be implemented, equation (3.11) needs to be altered. Tofts has proposed a multi-compartment model for perfusion of magnetic resonance imaging contrast agents^{64,65} that was adapted for hyperpolarized agents by Bankson⁵³. Following these models, a tissue is divided into two spatially separated compartments; the vascular space, and the extravascular space, although additional compartments could be considered. When the agent is injected intravenously, it arrives to the tissue via the vasculature, and thus the concentration of the agent follows a vascular input function that is a function of the vascular system and the injection bolus. Once in the vasculature, the agent would move across the vessel walls with a transfer constant K_{ve} . The rate at which the agent crossing from the blood into the extravascular space will simply be the K_{ve} divided by the volume fraction of the extravascular-extracellular space v_e . With these constants, the transfer of an agent out of the vasculature would be given by:

$$\frac{\delta}{\delta t} \begin{bmatrix} C_v \\ C_{ev} \end{bmatrix} = \begin{bmatrix} -\frac{K_{ve}}{v_e} & \frac{K_{ve}}{v_e} \\ \frac{K_{ve}}{v_e} & -\frac{K_{ve}}{v_e} \end{bmatrix} \begin{bmatrix} C_v \\ C_{ev} \end{bmatrix} \quad 3.12$$

where C_v and C_{ev} are the agent concentrations in the vasculature and extravascular spaces respectively.

Note that equation (3.12) assumes that the rate constant for transfer of an agent from the vascular space to the extravascular-extracellular space is the same as the reverse transfer constant, that is to say

there is no biological preference for uptake or clearance of the agent across the vasculature in the tissue of interest.

Notably only the pyruvate in the cytosol would be converted into lactate as the enzyme lactate dehydrogenase is confined in the cytosol. To account for this, the extravascular space could be divided into two compartment, the cellular compartment and an extravascular extracellular space. Such compartmentalization requires a three-step transport from the vasculature to the cells involving not only leakage from the vasculature but cellular uptake which would likely be mediated by MCT-1. If cellular uptake is so quick that the two spatially separated pools can be considered to be in quasi-equilibrium, then all pyruvate outside of the vascular pool can be assumed to be available for conversion into lactate. Indeed, initial modeling results suggest that the simpler, two physical pool model of perfusion is sufficient to model *in vivo* delivery of pyruvate and its subsequent conversion to lactate⁵³. With the assumption of two physical pools and two chemical pools, equation (3.12) can be combined with equation (3.10) to yield:^{53,62}

$$\frac{\delta \vec{M}}{\delta t} = v_e \left\{ A_{ev} \vec{M}_{ev} + \frac{k_{ve}}{v_e} \vec{M}_v \right\} + (1 - v_e) A_v \vec{M}_v \quad 3.13$$

$$A_{ev} = \gamma \begin{bmatrix} -\frac{1}{T_{2,a}} - k_1 - \frac{k_{ev}}{v_e} & (1 - \sigma_a) B_{eff,x}(t) & (1 - \sigma_a) B_{eff,y}(t) & k_2 & 0 & 0 \\ (1 - \sigma_a) B_{eff,x}(t) & -\frac{1}{T_{2,a}} - k_1 - \frac{k_{ev}}{v_e} & -(1 - \sigma_a) B_{eff,x}(t) & 0 & k_2 & 0 \\ -(1 - \sigma_a) B_{eff,y}(t) & (1 - \sigma_a) B_{eff,x}(t) & -\frac{1}{T_{1,a}} - k_1 - \frac{k_{ev}}{v_e} & 0 & 0 & k_2 \\ k_1 & 0 & 0 & -\frac{1}{T_{2,b}} - k_2 - \frac{k_{ev}}{v_e} & (1 - \sigma_b) B_{eff,x}(t) & (1 - \sigma_b) B_{eff,y}(t) \\ 0 & k_1 & 0 & (1 - \sigma_b) B_{eff,x}(t) & -\frac{1}{T_{2,b}} - k_2 - \frac{k_{ev}}{v_e} & -(1 - \sigma_b) B_{eff,x}(t) \\ 0 & 0 & k_1 & -(1 - \sigma_b) B_{eff,y}(t) & (1 - \sigma_b) B_{eff,x}(t) & -\frac{1}{T_{1,b}} - k_2 - \frac{k_{ev}}{v_e} \end{bmatrix}$$

$$A_v = \gamma \begin{bmatrix} -\frac{1}{T_{2,a}} - \frac{k_{ev}}{v_e} & (1 - \sigma_a) B_{eff,x}(t) & (1 - \sigma_a) B_{eff,y}(t) & 0 & 0 & 0 \\ (1 - \sigma_a) B_{eff,x}(t) & -\frac{1}{T_{2,a}} - \frac{k_{ev}}{v_e} & -(1 - \sigma_a) B_{eff,x}(t) & 0 & 0 & 0 \\ -(1 - \sigma_a) B_{eff,y}(t) & (1 - \sigma_a) B_{eff,x}(t) & -\frac{1}{T_{1,a}} - \frac{k_{ev}}{v_e} & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{T_{2,b}} - \frac{k_{ev}}{v_e} & (1 - \sigma_b) B_{eff,x}(t) & (1 - \sigma_b) B_{eff,y}(t) \\ 0 & 0 & 0 & (1 - \sigma_b) B_{eff,x}(t) & -\frac{1}{T_{2,b}} - \frac{k_{ev}}{v_e} & -(1 - \sigma_b) B_{eff,x}(t) \\ 0 & 0 & 0 & -(1 - \sigma_b) B_{eff,y}(t) & (1 - \sigma_b) B_{eff,x}(t) & -\frac{1}{T_{1,b}} - \frac{k_{ev}}{v_e} \end{bmatrix}$$

where \vec{M}_{ev} is the magnetization in the extravascular pool, and \vec{M}_v is the magnetization in the vascular pool. Equation (3.13), assumes a closed vascular system, which is not the case for perfused tissue. In

perfused tissue, the vascular compartment will contain rapidly flowing blood and therefore the signal from that compartment, and it should be modeled differently than the extravascular pool. A vascular input function is commonly used to model the signal from an agent in the blood⁷¹. The rapid flow in the vascular pools allows some simplifications of \vec{M}_v . First, since any agents in the vascular pool rapidly leave the tissue, agent washout from the extravascular space can be modeled as a loss term with no increase in the signal in the vascular compartment. Additionally, since agents in the vascular pool are constantly being supplied by fresh flowing blood, the local signal loss terms such as T_1 and k_{ve} can be ignored. As a consequence of these assumptions, the signal from the vascular pool is governed solely by the concentration in the total blood pool defined by the vascular input function, $\vec{M}_v \equiv \vec{VIF}(t)$.

Following this assumption, equation (3.13) reduces to:

$$\frac{\delta \vec{M}}{\delta t} = v_e \left\{ A_{ev} \vec{M}_{ev} + \frac{k_{ve}}{v_e} \vec{VIF}(t) \right\} + (1 - v_e) \{ \vec{VIF}(t) \} \quad 3.14$$

When there is not time varying B field there is a closed form solution to equation (3.14) given by⁵³:

$$\vec{M}(t) = v_e \left\{ e^{A_{ev}(t)} \vec{M}_0 + \frac{k_{ve}}{v_e} \int_0^t e^{A(t-\tau)} \vec{VIF}(\tau) d\tau \right\} + (1 - v_e) \{ \vec{VIF}(t) \} \quad 3.15$$

Equation (3.15) accounts for the basic physics behind magnetic resonance, as well as the chemical exchange taking place in an isolated extravascular compartment that is fed by the vasculature via exchange across vessel walls. Both the vascular and extravascular compartments contribute to the detected magnetic resonance signal in amounts that depend on their volume fractions of the tissue of interest. Inspection of equation (3.15) reveals that the terms from the Bloch equation are exclusively in the extravascular compartment and do not affect the vascular compartment. As a result, the magnetization vectors in the vascular pool would be unaffected by an excitation pulses would not contribute detectable magnetic resonance signal. If the magnetization in the vascular pool were

sensitive to excitation, it could contribute a signal but could be significantly reduced below the vascular input function, a violation of the assumption leading to equation (3.14). In order to account for this discrepancy, it can be assumed that the excitation pulses are short compared to the flow rate in the vascular compartment. Under a short excitation assumption, the effects of perfusion and excitation can be considered separately. If there is no time-varying magnetic field, then the intravascular longitudinal magnetization can be set to a value defined by the vascular input function in order to preserve fidelity with the perfusion model. Additionally, since it is assumed that perfusion of transverse magnetization is negligible, the transverse components of the magnetization can evolve according to the Bloch equations. During excitation, perfusion can be ignored, again because the duration of the pulse is so short compared to the perfusion timescale. This allows the longitudinal magnetization in the vascular pool to be excited into the transverse plane where it will be detected as a signal without needing to account for blood flow. Once the pulse has played out, the longitudinal magnetization is returned to the value dictated by the vascular input function while the transverse magnetization will continue to evolve according to the Bloch equations. Such modifications to the VIF allows the signal from the blood pool to be accounted for without the need to model complex flow of blood in an arbitrary vasculature. If the radiofrequency pulses are long, or significantly affect the vascular input function, then non-negligible errors in the perfusion model would be introduced by this assumption.

Section 3.2: Implementation

Equation (3.15) and its underlying differential equation were coded in Matlab (The MathWorks Natick MA). The basic structure of the object-oriented architecture is displayed in figure 3-1 and documented in detail in Appendix B. Instantiation of the simulation environment is performed by a singleton⁷² world object which stores references to all the information about the spin systems and the pulse sequence. The pulse sequence is a series of gradient and radiofrequency pulses that are stored as arbitrary $b(t)$ allowing any pulse shape to be used. To assist in usability, helper functions have been

created for the construction of the most standard gradient and RF-pulse waveforms that are used in MRI/MRS. Spin groups logically represent an isochromat and differ based on the underlying assumptions about which model they follow. Once a world system has been populated with both a pulse sequence and a set of spins, it is then calculated and finally evaluated to yield a set of free induction decays or echoes.

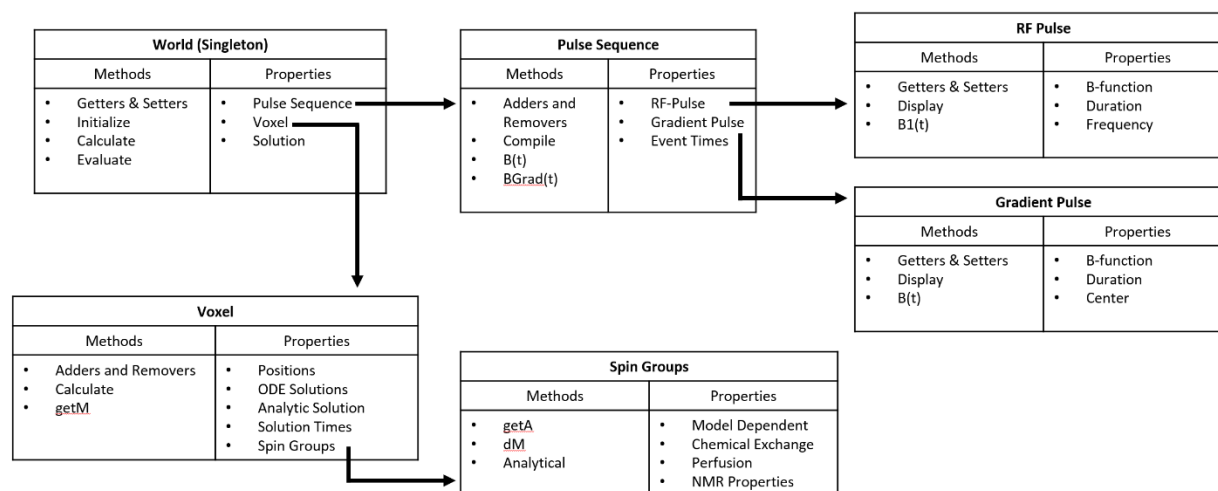


Figure 3-1. Outline of the simulation architecture. The World stores an array of voxels and a single pulse sequence. The Pulse Sequence stores a list of radio frequency pulses and gradient pulses and has logic to more efficiently organize them for rapid query of the magnetic field as a function of time, as well as flags for when the analytic solution (equation 3.15) does not hold. Voxel stores arrays of spin groups and has functions to calculate their solutions, which are stored for fast evaluation at arbitrary times. Spin Groups store all the biophysical parameters and the details of the underlying model. Spin models must be of the form in equation (3.7) and need to have a valid analytical solution and the logic to determine if it applies.

The calculation simply solves equation (3.14) and other preceding equations based on the spin groups present in the simulation. When there are no excitation pulses, the analytical solution can be

used. During the calculation step the analytical solution is defined as a function of t over a time frame that is determined by the pulse sequence. If, however, there is a time varying magnetic field, then the closed form solution that was presented above does not hold and the differential equation must be solved numerically. The simplest way to solve an ordinary differential equation such as equation (3.8) is known as Euler's method⁷⁰:

$$y_{n+1} = y_n + \Delta t * y'_n(t_n) \quad 3.16$$

where the derivative, y' , of a function y is calculated at a particular time t_n and then advanced by some small Δt to another time point t_{n+1} to find an approximate solution y_{n+1} . This process is repeated until the solution has been found for all time points of interest. This method is rarely used in practice for two main reasons (i) it is not very stable, that is, if there are regions of the solution that are changing rapidly then the derivative will be large and therefore the step can be quite large leading to sizeable errors and (ii) it is normally slower than other methods with the same accuracy when variable step sizes are used. If, however, a "trial" step or steps are used in between each step, the error in the function can be minimized. If these trial steps are based on reducing the error order in a Taylor series expansion they are referred to as *Runge-Kutta* methods⁷⁰. These trial points allow for a better sampling of the function along its solution, and since the location of their evaluation is derived from the Taylor series expansion they are generally more efficient than the brute force Euler method with a similar number of function calls. This increased efficiency normally allows for larger step sizes with the same accuracy as with lower order methods for most practical problems, and with larger steps sizes comes faster evaluation. The most commonly used *Runge-Kutta* method uses four test points and is general faster and more accurate than Eulers method or even a *Runge-Kutta* method with only a single additional trial point⁷⁰. A comparison of the different methods is shown in figure 7-2.

Once the numeric solutions have been calculated, they are stored along with the analytical solution in a series of objects that allow for evaluation at arbitrary time points within the calculated

solution space. This set of solutions can then be evaluated at the desired sampling time points to yield a series of free induction decays or echoes. This series of free induction decays can then be evaluated using any processing methods that are applicable to real magnetic resonance data sets, as will be discussed in Chapter 4.

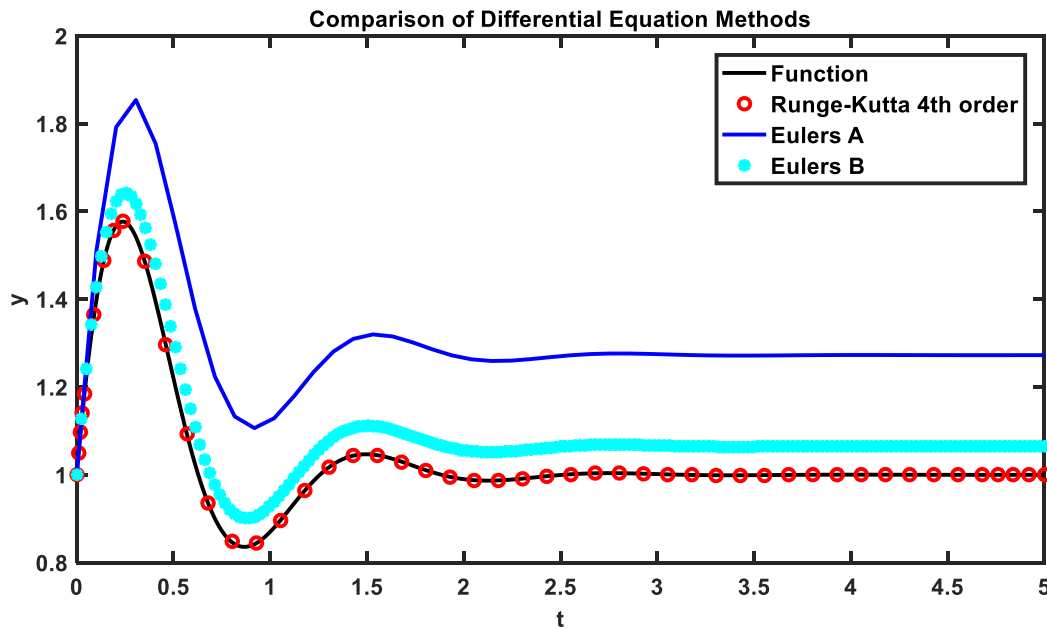


Figure 3-2. A comparison of numerical methods for solving ordinary differential equations. The black line is the actual function $y(t) = e^{\frac{-t}{b}} * \sin(at)$ with a derivative of the form $\frac{dy}{dt} = -\frac{1}{b}e^{\frac{-t}{b}} * \sin(at) + ae^{\frac{-t}{b}} * \cos(at)$. The red dots are the sample points used in a *Runge-Kutta* 4th order solution. The blue line is the sample point used in an *Euler's* method solution with a step size Δt that was set to ensure the same number of sample points as the 4th order *Runge-Kutta*. The cyan asterisks are the evaluation points used in a *Euler's* method solution with a step size that was set to ensure the same number of function evaluations as the 4th order *Runge-Kutta*. Within the same computational burden, the 4th order *Runge-Kutta* shows superior accuracy.

Section 3.3: Verification

To ensure consistency between equations (3.14) and (3.15), a system consisting of two exchanging spins was evaluated using both the analytic solution, equation (3.15), and using an adaptive 4th order *Runge-Kutta* method to solve equation (3.14). The differences between the results were many orders of magnitude lower than the solution values as seen in figure 3-3. As the error tolerance for the *Runge-Kutta* method was tightened, the difference between the two separate solutions was reduced, as was the L^2 norm, which is also depicted in figure 3-3.

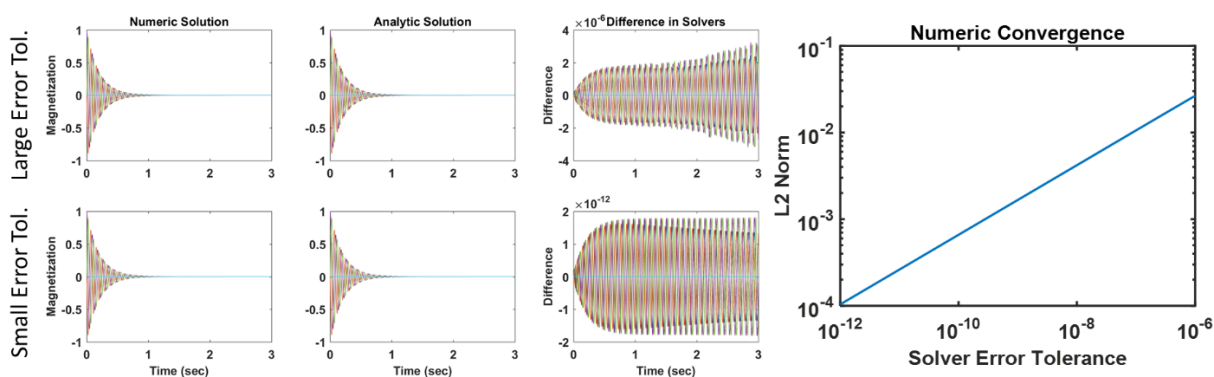


Figure 3-3. A comparison of the analytical and numerical solutions for a system of two spins coupled by chemical exchange. The top series of plots are the resulting free induction decay signals using either equation (3.14) or (3.15) and the difference between the solutions using a relatively high error tolerance of 1×10^{-6} and the lower plot series is identical plots but with the error tolerance reduced to 1×10^{-12} . The plot to the far right is the L^2 norm of the difference between the analytical and numerical solvers as a function of the error tolerance of the *Runge-Kutta* method.

More than providing higher numeric precision, the analytical solution also results in better computational efficiency. As shown in figure 3-4, the computational time for two isolated spins is independent of their chemical shifts. The independence of isolated spins is possible because they can have separate reference frames for calculation and can avoid the computational burden of Larmor

precession via a frame shift. If the two spins are coupled by chemical exchange, it becomes impossible to completely remove the oscillatory motion with a frame shift. The computational burden imposed by the oscillatory motion will be directly related to the difference in chemical shift of the coupled spins, with larger differences in chemical shift resulting in faster oscillatory motion and therefore, longer computational times as shown in figure 3-4. By utilizing the analytical solution, the computational impacts associated with oscillatory motion are removed as the solution is not found iteratively, and computation time is independent of chemical shift just like the isolated spins that are also illustrated in figure 3-4.

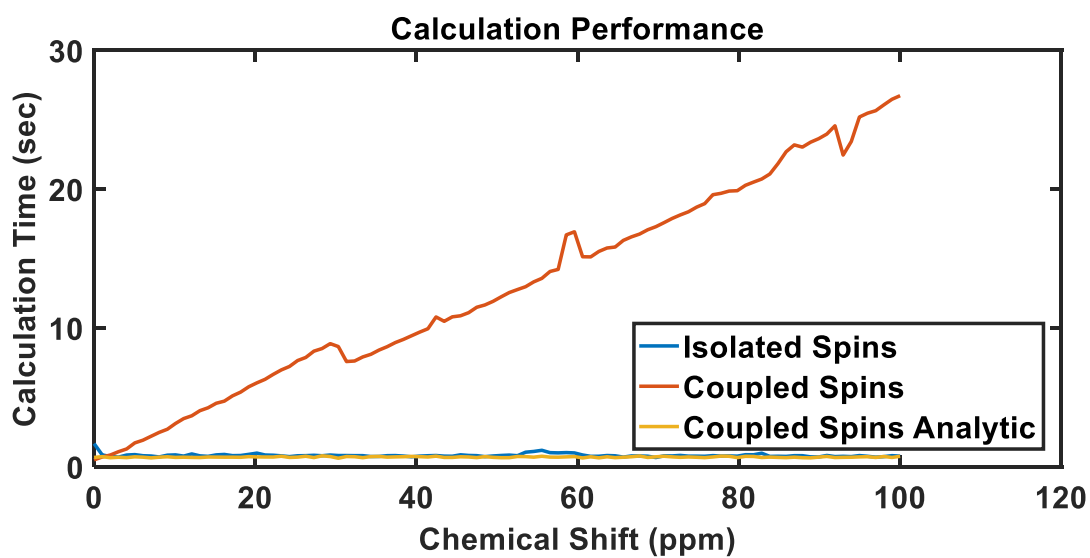


Figure 3-4. The computational performance of two spins that are either isolated or are coupled by chemical exchange as a function of the difference in their chemical shifts. The blue line shows the computational time for two isolated spins and is stable at 1.5 seconds. The red line shows the increasing computational time for two coupled spins when solved numerically as the difference between the two chemical shifts is increased. The yellow line shows that using the analytical solution removes any dependence on the chemical shift and returns the computation to a stable 1.5 second.

In order to verify that the chemical exchange between two spin groups is computationally sound, a pair of coupled spin groups were simulated in the absence of any excitation into the transverse plane. The resulting longitudinal magnetization was fit to equation (3.15) using a least squares method and only fitting for chemical exchange. As seen in figure 3-5 the fit exchange rate matches the simulated exchange rate with good numeric fidelity across a range of exchange rates.

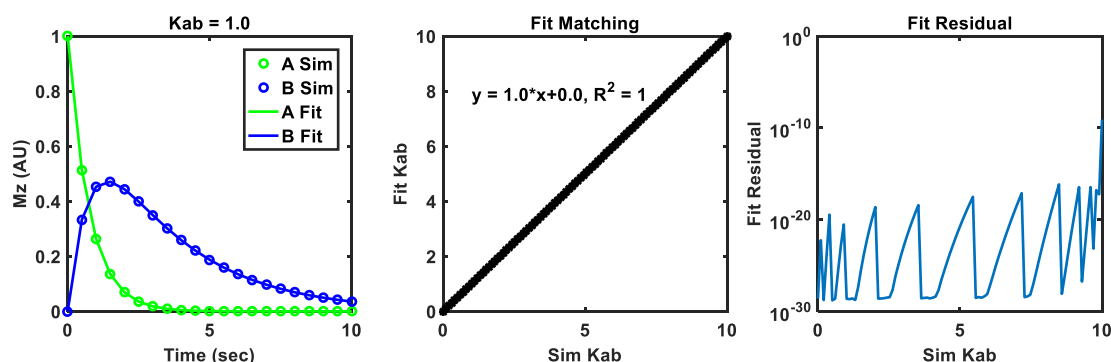


Figure 3-5. Exchange rate fitting for the longitudinal magnetization of two exchanging spin groups. Good qualitative agreement between the fitting function and the simulated longitudinal magnetization with an exchange rate of 0.1 sec^{-1} as seen in the left-most plot. In the center plot the fitted exchange rates are plotted against the simulated exchange rates showing good quantitative agreement with both the slope and R^2 equal to unity. In the right-most plot the residual of the fit for a range of exchange terms is shown.

Similar fitting of the longitudinal magnetization was performed to assess fidelity of the perfusion parameters. A single spin was simulated using the two physical compartment model described by equation (3.12). The resulting longitudinal magnetization was fit with a least squares method allowing both k_{ve} and v_e as fit parameters for a range of values as seen in figure 5-6. The fit results matched the simulated longitudinal magnetization with good qualitative and quantitative accuracy.

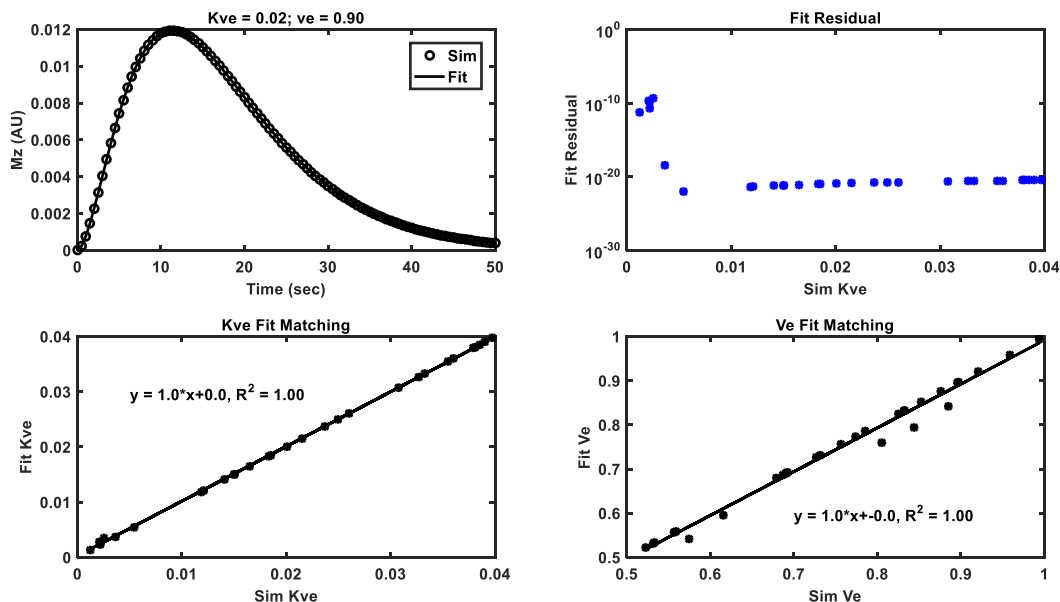


Figure 3-6. Perfusion fitting for the longitudinal magnetization of a single perfused spin group assuming two spatial compartments. In the top left plot good qualitative fitting is shown between the simulated longitudinal magnetization and the fitting function with $k_{ve} = 0.02$ and $v_e = 0.9$. The bottom left and right plots show strong correlation between the simulated k_{ve} and v_e and the resulting fit values with slope and R^2 equal to unity. The top right plot shows the fit residual as a function of the k_{ve}

In order to assess that radiofrequency excitations were being properly modeled, a single isolated spin was simulated with a simple pulse acquire experiment. The spin parameters were set to match those of C^{13} Urea doped with Gd^{3+} with $\delta = 173.5$ ppm, $T_1 = 3$ sec and $T_2 = 20$ msec. The pulse sequence, both simulated and actual, used a 1500 msec block pulse and a 90° excitation angle. A 5 kHz readout bandwidth with 2048 points was used. The center frequency of the pulse was swept from -25 ppm to 25 ppm with a full 15 seconds between each excitation to ensure complete T_1 recovery. Finally, the simulated data were corrected to match the initial phase of the on-resonance excitation with an identical correction factor used for all off-resonance excitations. All dynamic spectroscopy was performed on a 7-T/30-cm Biospec System (Bruker Biospin Corp., Billerica, MA) using B-GA12SHP

gradients and a dual-tuned $^1\text{H}/^{13}\text{C}$ volume coil (72-mm ID, Bruker Biospin MRI). Figure 3-7 shows good agreement between the simulated magnitude and phase of the signal as a function of the center frequency of the excitation pulse and the measured data. This suggests that excitation pulses are indeed well modeled in the simulation architecture.

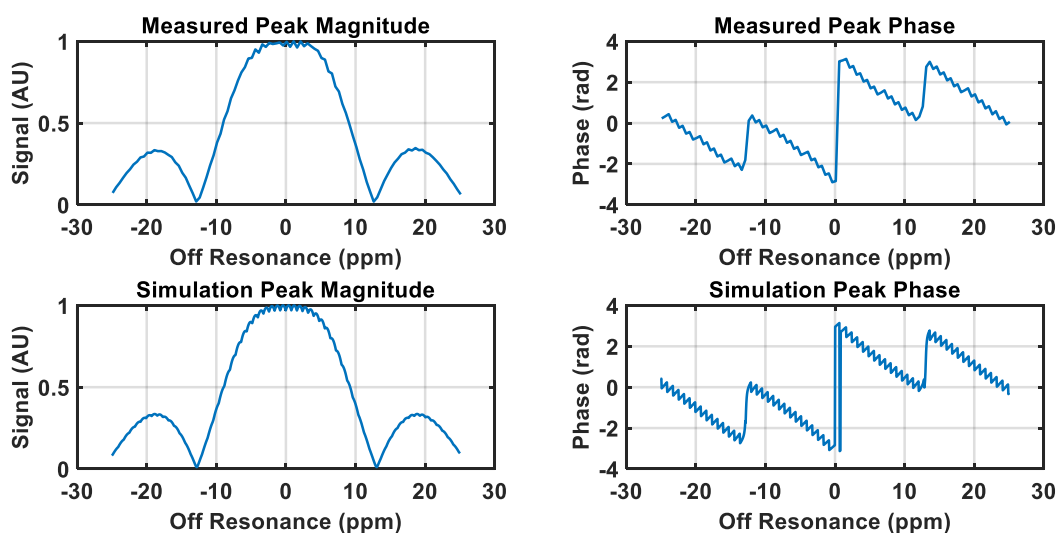


Figure 3-7. Comparison of excitation profiles for doped C^{13} urea. The top two plots show the measured phase and magnitude of a C^{13} urea bulb as a function of the center frequency of the excitation pulse. The lower two plots show the simulation phase and magnitude using an identical simulation pulse sequence.

Finally, simulated dynamic spectroscopy was qualitatively compared to phantom studies. Free induction decays acquired by dynamic spectroscopy of a dynamic enzyme phantom,^{68,73} which will be described in chapter 5 are compared to the simulation results with matching chemical and sequence parameters in figure 3-8. There is strong qualitative agreement between the simulated data and that acquired from a phantom. The only minor sources of disagreement are slightly different noise factors, minor peak splitting in the phantom data caused by imperfect shimming not modeled in the system and

the presence of pyruvate-hydrate in the phantom data. Pyruvate hydrate is normally a small metabolic inactive signal that is generally ignored and therefore is not simulated.

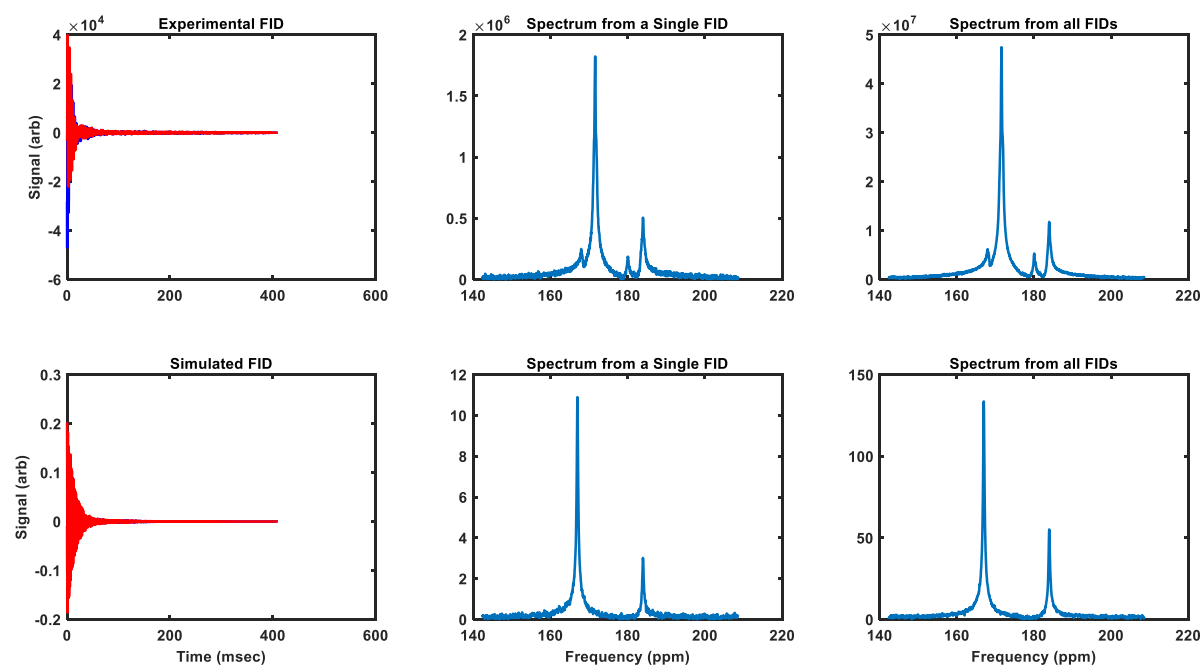


Figure 3-8. A comparison between dynamic spectroscopy data acquired from a phantom (top) and data resulting from a simulation using the same physical and sequence parameters.

These verification studies demonstrate that the simulation architecture is mathematically consistent with the models outlined, specifically equations (3.8) and (3.14). With such a numerically sound platform more complicated biology or sequences can be explored to ensure that measured exchange rates are not skewed by detection methods.

Chapter 4. Quantitative Accuracy of Dynamic Spectroscopy

This Chapter is based upon

Walker, C. M., Chen, Y., Lai, S. Y. & Bankson, J. A. A novel perfused Bloch-McConnell simulator for analyzing the accuracy of dynamic hyperpolarized MRS. *Med Phys* 43, 854, doi:10.1118/1.4939877 (2016).

Copyright © 2016 American Association of Physicists in Medicine. Reproduced with permission of American Association of Physicists in Medicine.

This chapter is intended to address Aim 2.

Section 4.1 Introduction and Theory

Equation (3.14) can be used to generate a single free induction decay or a series of them. After processing as described in chapter 2, a series of time value curves that represent the relative signal intensities of the distinct chemical species can be generated. If hyperpolarized pyruvate is the agent of interest for the simulation study, then the expected downstream product would be hyperpolarized lactate. In that case, the signal curves would show some initial pyruvate signal or early pyruvate perfusion followed by its subsequent conversion to lactate. Eventually, due to wash out or T_1 decay and excitation losses, all of the signal will dissipate. In order to quantify the rate of conversion of pyruvate to lactate a model similar to the equations described in chapter 3 can be fit to the resulting curves. The equations will be slightly different as the fitting will only address two relative signal intensities and not a series of magnetization vectors in 3-space.

Assuming no perfusion, referred to as a closed system approximation, the signal curves will follow^{62,68}:

$$\frac{\delta}{\delta t} Pyr(t) = - \left(\frac{1}{T_{1,Pyr}} + \frac{\cos(\theta)}{TR} + k'_{pl} \right) Pyr(t) \quad 4.1$$

$$\frac{\delta}{\delta t} Lac(t) = - \left(\frac{1}{T_{1,Lac}} + \frac{\cos(\theta)}{TR} \right) Lac(t) + k'_{pl} Pyr(t)$$

where $Pyr(t)$ and $Lac(t)$ are the pyruvate and lactate signal magnitudes respectively, $T_{1,Pyr}$ and $T_{1,Lac}$ are the longitudinal relaxation times for pyruvate and lactate respectively, θ is the excitation angle, TR is the repetition time, and k'_{pl} is the apparent exchange rate between pyruvate and lactate. Note that k'_{pl} is not the same term as k_1 in equations (3.4-13); k_1 describes a physical exchange of chemical species between two pools, k'_{pl} describes the evolution of the signal resulting from k_1 exchange. The distinction is subtle, but important. Ideally k'_{pl} would relate to k_1 in some logical way, or even be identical to it. A final note on k'_{pl} , k'_{pl} is an apparent exchange rate, or it is the observed rate exchange of the hyperpolarized signal and might not be identical to the actual rate of conversion of metabolites. Therefore, k'_{pl} is independent of the underlying process as long as the resulting signal curves are the same. No conversion from lactate to pyruvate is assumed, e.g. $k_2 = 0$. However, that can be added to equation (4.1) without much more complexity. Additionally, note that T_1 decay and excitation losses will be combined into a single term^{28,53}. This assumes that excitation losses can be averaged over the entire repetition time as opposed to being accounted for during the short time interval that the excitation pulse is interacting with the spin system. This assumption has a few conditions under which it is accurate; the excitation angles are not changing during the acquisition, the repetition time is constant throughout the acquisition and the excitation pulse is small. If these conditions are not met, there is a good chance that numerical solutions to equation (4.1) will have reduced accuracy. Additionally, the analytical solution with averaged signal loss will require the excitation angle to be uniform for all excitations. It is possible to account for excitation pulses instantaneously and thus to remove the dependence on such assumptions. Such an instantaneous loss modeling strategy will be presented at the end of this section.

Accounting for perfusion following using a model similar to Toft's model^{53,64,65} requires an adaptation of equation^{53,62} (4.1):

$$\begin{aligned}
 \frac{\delta}{\delta t} Pyr_e(t) &= - \left(\frac{1}{T_{1,Pyr}} + \frac{\cos(\theta)}{TR} + k'_{pl} + \frac{k_{ve}}{v_e} \right) Pyr_e(t) + \frac{k_{ve}}{v_e} Pyr_v(t) \\
 \frac{\delta}{\delta t} Lac(t) &= - \left(\frac{1}{T_{1,Lac}} + \frac{\cos(\theta)}{TR} + \frac{k_{ve}}{v_e} \right) Lac(t) + k'_{pl} Pyr(t) + \frac{k_{ve}}{v_e} Lac_v(t) \\
 Pyr_v(t) &= b_p(t) \\
 Lac_v(t) &= b_L(t)
 \end{aligned} \tag{4.2}$$

where the subscripts e and v are used to denote the vascular and extravascular spaces. Generally, it is assumed that lactate is not carried into the tumor via vasculature, and under these conditions all the $Lac_v(t)$ terms would be removed. The vascular input term for pyruvate $b_p(t)$ was modeled as a gamma variate⁷¹:

$$Pyr_v(t) = \Gamma(t) = t^\alpha \exp\left(\frac{-t}{\beta}\right) \tag{4.3}$$

where α and β are shape terms.

Under these assumptions equations (3.14) and (4.2) reduce to:

$$\begin{aligned}
 \frac{\partial \overrightarrow{M_{P_{ev}}}}{\partial t} &= \gamma(\overrightarrow{M_{P_{ev}}} \times \vec{B}) - \left(\overrightarrow{R_P} + k_{pl} + \frac{k_{ve}}{v_e} \right) \overrightarrow{M_{P_{ev}}} + \frac{k_{ve}}{v_e} \overrightarrow{M_{P_v}} \\
 \frac{\partial \overrightarrow{M_{L_{ev}}}}{\partial t} &= \gamma(\overrightarrow{M_{L_{ev}}} \times \vec{B}) - \overrightarrow{R_L} \overrightarrow{M_{L_{ev}}} + k_{pl} \overrightarrow{M_{L_{ev}}} \\
 \overrightarrow{M_{P_v}} &= \Gamma(\alpha, \beta, t) \\
 \frac{\partial Pyr_E(t)}{\partial t} &= - \left(\frac{k_{ve}}{v_e} + \widehat{k_{pl}} + R_{Pyr} \right) Pyr_E(t) + \frac{k_{ve}}{v_e} Pyr_v(t) \\
 \frac{\partial Lac_E(t)}{\partial t} &= \widehat{k_{pl}} Pyr_E(t) - R_{Lac} Lac_E(t) \\
 Pyr_v(t) &= \Gamma(\alpha, \beta, t)
 \end{aligned} \tag{4.4}$$

4.5

Equation (4.4 and 4.5) implicitly assume that the signal losses due to excitation can be approximated by effectively reducing the T_1 by a factor $\frac{\cos(\theta)}{TR}$, which essentially averages the excitation losses over the entire repetition time. In reality a closer approximation would be to track the transverse and longitudinal magnetization even though only the transverse magnetization can be detected. By relating the longitudinal magnetization to the detected signal it becomes possible to treat signal excitation as an instantaneous event. Creating a discontinuity in the non-detectable longitudinal magnetization which will subsequently affect the transverse magnetizations. If the excitation losses are not averaged over the entire repetition time but are modeled as instantaneous losses equations (4.5) would be modified to:

$$\begin{aligned} \begin{bmatrix} M'_{P,i} \\ M'_{L,i} \end{bmatrix} &= \begin{bmatrix} M_{P,(i-1)} \\ M_{L,(i-1)} \end{bmatrix} e^{A*(t_{i-1}-t_i)} + \frac{k_{ve}}{v_e} \int_{t_{i-1}}^{t_i} e^{A(t_{i-1}-\tau)} \begin{bmatrix} b_P(\tau) \\ b_L(\tau) \end{bmatrix} d\tau \\ A &= \begin{bmatrix} -\frac{1}{T_{1,P}} - k_{pl} - \frac{k_{ve}}{v_e} & k_{lp} \\ k_{pl} & -\frac{1}{T_{1,L}} - k_{lp} \end{bmatrix} \\ \begin{bmatrix} S_{pyr,i} \\ S_{lac,i} \end{bmatrix} &= \frac{\sin(\theta_{pyr,i})}{\sin(\theta_{lac,i})} * \left\{ v_e \begin{bmatrix} M'_{P,i} \\ M'_{L,i} \end{bmatrix} + (1 - v_e) \begin{bmatrix} b_P(i * TR) \\ b_L(i * TR) \end{bmatrix} \right\} \\ \begin{bmatrix} M_{P,i} \\ M_{L,i} \end{bmatrix} &= \frac{\cos(\theta_{pyr,i})}{\cos(\theta_{lac,i})} \begin{bmatrix} M'_{P,i} \\ M'_{L,i} \end{bmatrix} \end{aligned} \tag{4.6}$$

where $M_{P,i'}$ and $M_{L,i'}$ are the longitudinal magnetizations for pyruvate and lactate before the i th excitation, $T_{1,P}$ and $T_{1,L}$ are the longitudinal relaxation times for pyruvate and lactate respectively, k_{pl} and k_{lp} are the forward and reverse exchange rates between pyruvate and lactate respectively, $b_P(t)$ and $b_L(t)$ are the vascular input functions for pyruvate and lactate respectively, k_{ve} and v_e are the

extravasation rate and extravascular volume fraction respectively as described by Tofts⁶⁵ and t_i is the time of the i th RF excitation.

Section 4.2 Methods

The above Bloch-McConnell equations were numerically solved in a custom-built simulation environment developed using the MATLAB computing language (MathWorks, Natick, MA) as outlined in chapter 3. Specifically, a perfectly homogeneous B_0 of 7 Tesla was assumed with a radiofrequency excitation pulse modeled as a five-lobed sinc pulse with 5-kHz bandwidth centered halfway between the lactate and pyruvate resonances. The spectral readout had a 4096-Hz bandwidth and 2048 points. Excitation angles were varied from 5° to 80° , and repetition times (TRs) ranging from 1-s to 10-s were tested. All simulations were carried out for 100 seconds, which ensured that all of the hyperpolarized signal had decayed below the noise floor. For the closed system, equation (3.7) assuming a single physical compartment was used to generate the signal curves with some initial pyruvate signal at the beginning of the acquisition and no additional signal entering the system. To simulate a perfused tissue equation (3.14) was used, and the initial pyruvate was assumed to be zero and that all of the pyruvate was assumed to have arrived in the system via perfusion. High and low driving model exchange rate constants of 0.1-s^{-1} and 0.02-s^{-1} , respectively, were used. The T_1 values used for pyruvate and lactate were assumed to be 43-s and 33-s respectively following the results in⁷⁴, and T_2^* was set to 20-ms for both metabolites⁷⁵. In the perfused system, the vascular input function was modeled as a gamma-variate; the shape terms for pyruvate's gamma variate were $\alpha = 2.8$ and $\beta = 4.5$; the extravasation rate (k_{ve}) was assumed to be 0.02-s^{-1} ,⁷⁶ and $v_e = 0.91$. Total SNR for these dynamic data sets was defined as the sum of the half-height full-width area of noise-free spectral peaks over all time points divided by the standard deviation of the Gaussian noise that was subsequently added. The average signal-to-noise ratio per excitation for each combination of parameter values was calculated as the total SNR divided by the number of excitations.

Gaussian noise was added to noise-free simulation results to achieve a total SNR of $\sim 1,000$ for the perfused and closed systems under reference conditions with 20° excitations and $TR = 2$ -s. This yielded metabolite curves that are consistent with our prior observations *in vivo*. The same noise amplitude was added to simulation results for all other parameter combinations. After Fourier transformation, phase correction was applied and the full-width at half-max (FWHM) area of the spectral peaks was calculated for each point in time. The resulting dynamic curves were fit to equations (4.1), (4.5) or (4.6) by minimizing the mean square error using a trust region reflective algorithm. For this analysis, only the fit exchange term $\widehat{k_{pl}}$ was allowed to vary; all other parameters in the analysis model equation (4.1), (4.5) or (4.6) were assumed to be identical to those used in the driving model equation (4.4). This process was repeated 20 times for all parameter combinations, and the average apparent fit exchange rate resulting from the analysis was compared with the driving exchange rate used by the

Parameter	Symbol	Value
Gyromagnetic Ratio	γ	$67.262 \times 10^6 \frac{rad}{sec * T}$
Vascular extravasation rate	k_{ve}	$0.02 sec^{-1}$
Extravascular volume fraction	v_e	0.91
Pyruvate T1 Relaxation Time	$T_{1,pyr}$	43 sec
Lactate T1 Relaxation Time	$T_{1,Lac}$	33 sec
T_2^* Relaxation Time for both Pyruvate and Lactate	T_2^*	20 msec
Vascular Input Function	$\Gamma(t)$	$t^{1.8} exp\left(\frac{-t}{4.5}\right)$

driving model.

Table 4-1. Parameters used for simulation and fitting.

This workflow is represented schematically in Fig. 4-1 and the set of constants used for simulation are summarized in Table 4-1. A similar process was used to explore contrast, which we define as the difference between the two exchange rates observed using identical acquisition parameters. To assess fit quality, the squared 2-norm of each fit was calculated and averaged for each of the 20 fitting repetitions. The squared 2-norm of the fits was normalized for the total number of excitations to remove dependence on the number of data points.

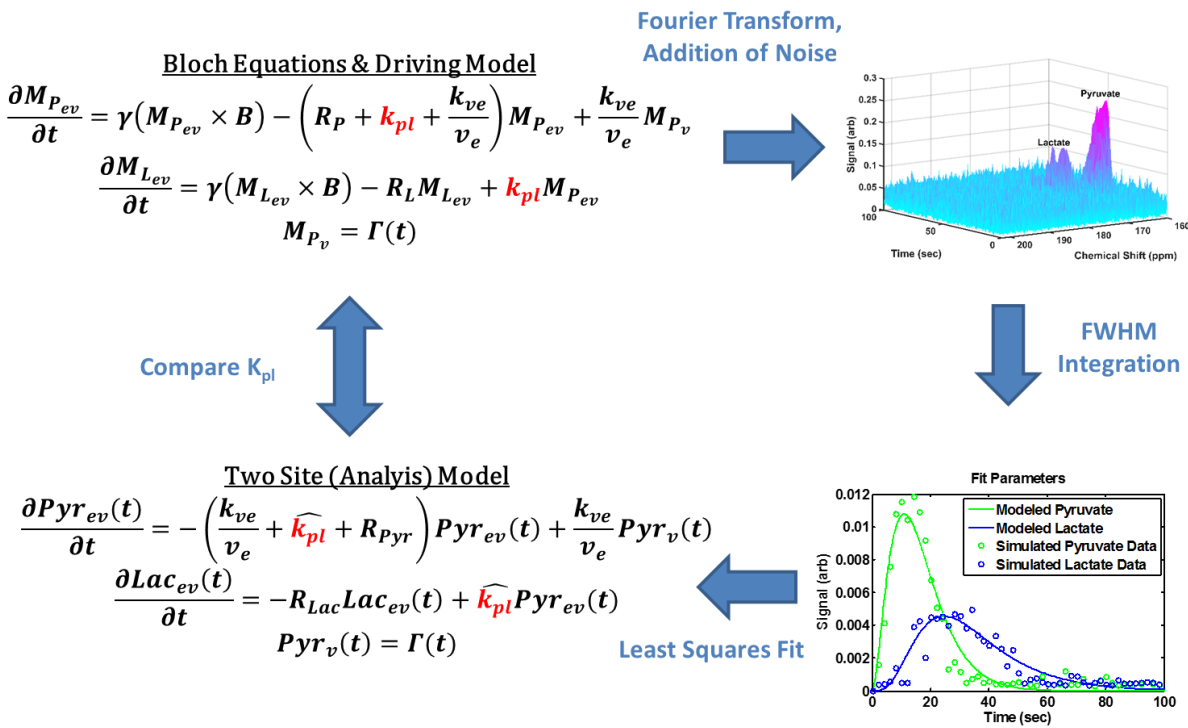


Figure 4-1. Workflow of simulation, processing and fitting: The Bloch-McConnell equations coupled with perfusion were solved for a range of sequence parameters. Noise was added to the resulting free induction decay signals. The signal of each metabolite was estimated via FWHM integration of the spectral peak at each time point. The signal evolution curves were fit using a two-site model to determine fit exchange. The fitted k'_{pl} was then compared to the assumed (driving) value to determine the accuracy of the exchange rate measurements for a given combination of sequence parameters.

Section 4.3 Results

Heat maps of the percent error for each combination of excitation angle and TR for the closed system are shown in Figure 4-2. A wide range of excitation angle and TR combinations resulted in accurate measurement of exchange rates, nominally with errors less than about 10% of the driving exchange rate. The range of sequence parameters that resulted in accurate fits was larger when a higher apparent exchange rate was used in the driving model. Estimates of the driving exchange rate began to result in inaccurate rate constants at very low and relatively high excitation angles, with a weaker dependence on TR. Accurate fit exchange rates were achieved with excitation angles of 10° to 40° for nearly all TRs at both high and low simulation exchange rates. Notably, the accuracy of analysis degrades precipitously for combinations with low exchange and large excitation angles as excitation losses suppressed the entire hyperpolarized signal before a significant lactate signal could be produced.

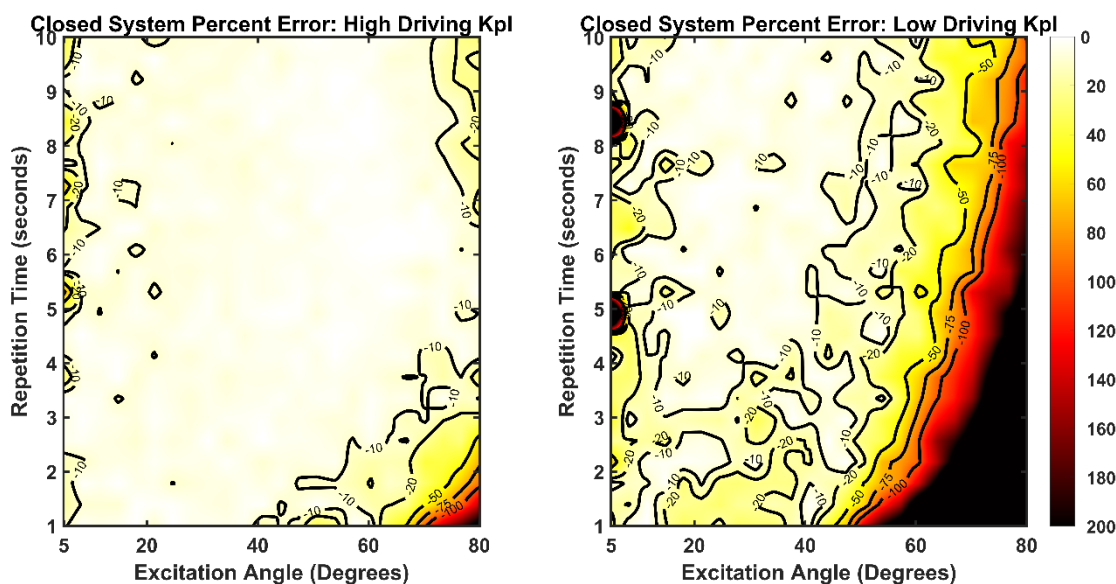


Figure 2-2. Percent error plots of driving versus fit exchange rates for the closed system approximation.

Left, high simulation exchange rate of 0.1 s^{-1} ; right, low simulation exchange rate of 0.02 s^{-1} . Errors

ranged from 1% to greater than 250%. A wide range of sequence parameters provided accurate estimations of $\widehat{k_{pl}}$, especially for the high simulation exchange rate data.

When perfusion was included, the accuracy of these measurements (Figure 4-3) at the lower driving exchange rate did not deteriorate to the same extent as was seen in the closed system. Generally, a more limited range of sequence parameter combinations yielded accurate observations though the maximum overall error was reduced. Regarding the high driving exchange rate, accuracy of measurements begins to degrade along a boundary extending approximately from an excitation angle of 30° and TR of 2 seconds to an excitation angle of 70° and TR of 10 s. Data assuming a lower driving exchange rate resulted in substantial error (~30% or greater) except over a narrow band from excitation angle 20° and TR of 2 to excitation angle 30° and TR of 10 as the limited lactate signal produced by slow exchange was more sensitive to the effects of excitation on signal evolution.

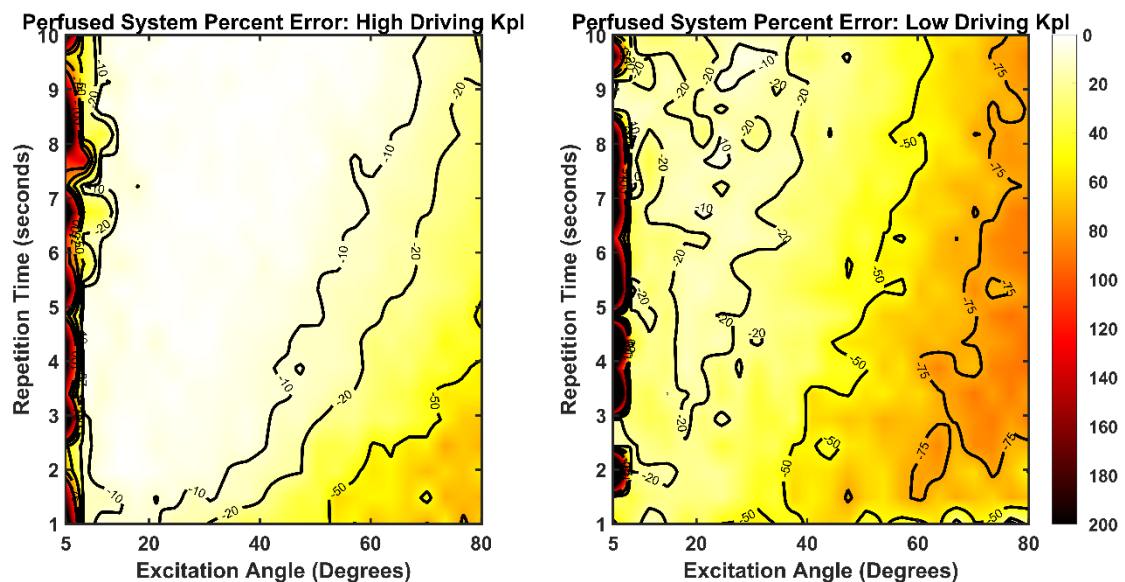


Figure 4-3. Percent error plots of driving versus fit exchange rates for the perfused system approximation. Left, high simulation exchange rate of 0.1 s^{-1} ; right, simulation exchange conversion rate

of 0.02 s^{-1} . The errors ranged from 1% to over 200%. Generally, the errors were less drastic than those for the closed system. However, there were fewer combinations of sequence parameters that yielded highly accurate exchange rate estimations.

Total SNR and average SNR per excitation were used as metrics of signal quality. The effects of excitation angle and TR on these metrics are summarized in Figures 4-4 and 4-5. The total SNR is maximized at fast repetition times and relatively low excitation angles for the closed system (Figure 4-4). In contrast to the closed system, a wider range of excitation angles resulted in maximal total SNR for the perfused system likely due to vascular delivery of fresh pyruvate offsetting the signal losses at higher excitation angles. The average SNR per excitation, in contrast, peaks at higher excitation angles with longer TRs (Figure 4-5). It is important to note that the sequence parameter combinations that result in very low total SNR (Figure 4-4) or average SNR per excitation (Figure 4-5) do not correspond well to regions of high fit error (Figures 4-2 and 4-3) except at the lowest excitation angles.

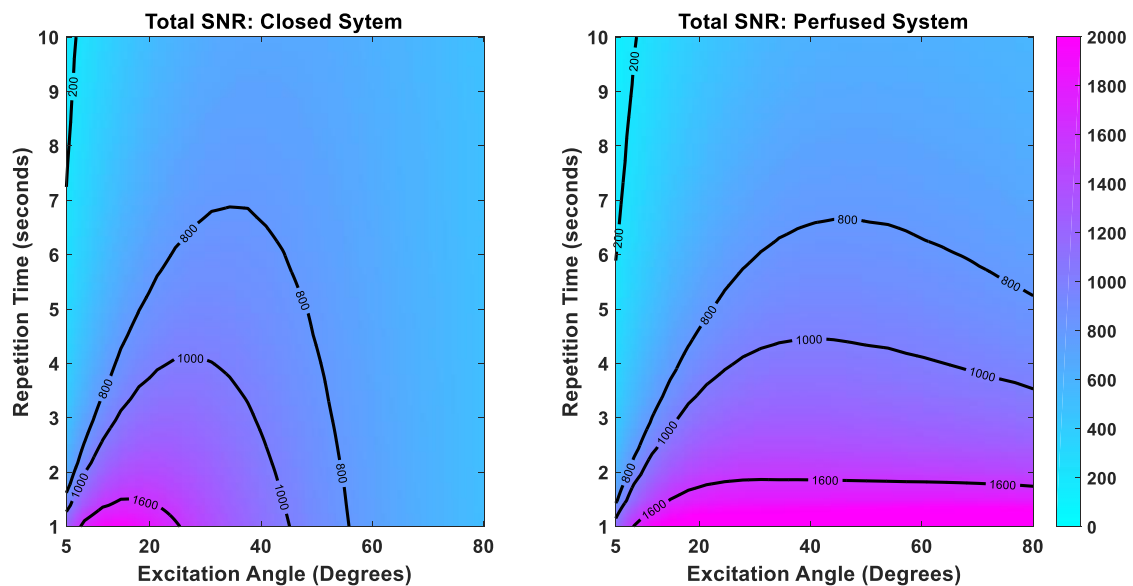


Figure 4-4. Relative total SNR of each study for the high driving exchange rate for both closed and perfused system. The total SNR peaks at a moderate excitation angle and short repetition time for the

closed system as opposed to the perfused system where the total SNR is relatively independent of excitation angle except at the lowest excitation angles. The results are similar for lower driving exchange (data not shown).

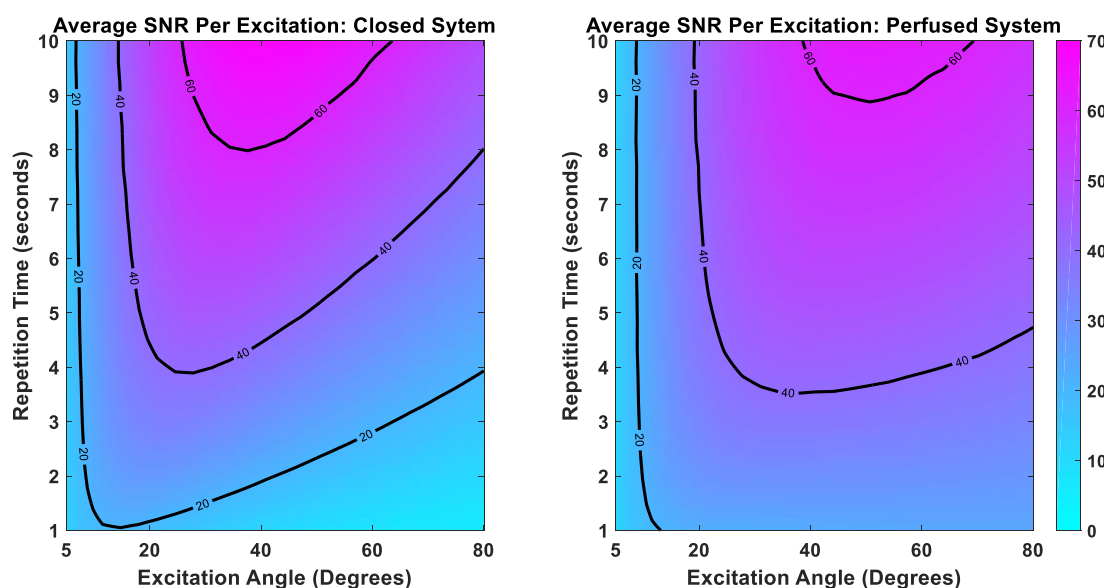


Figure 4-5. Average SNR per excitation for the closed and perfused systems with a high driving exchange rate. The average SNR is greater at higher excitation angles and longer TRs. Additionally, the SNR of the perfused system has a weak dependence on TR for higher excitation angles. Average SNR plots are similar for lower driving exchange (data not shown).

To explore the cause of fitting errors, we considered fit residual as a metric of fitting performance. The normalized square-2 norm of the fits for the high conversion rate (Figures. 4-2a, 4-3a) are shown in Figure 4-6. For the closed system, the norm increases with larger excitation angles with a slight dependence on TR. In contrast, the perfused system shows fairly low and uniform residuals. Higher fit norms (Figure 4-6) do not correlate with parameter combinations that resulted in inaccurate fitting of the exchange rate (Figures 4-2a, 4-3a), which implies that fit quality alone cannot explain inaccurate fitting results.

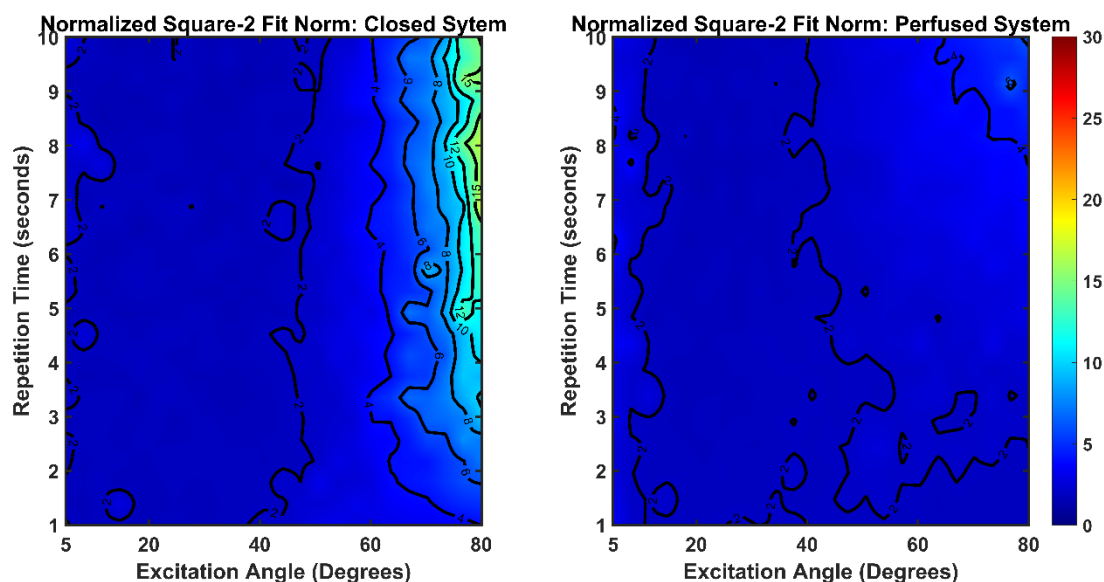


Figure 4-6. Normalized Square-2 norms of the fits for both closed and perfused systems, with a high driving k_{pl} . The norms for the closed system rise rapidly at higher excitation angles. In contrast, the norms for the perfused system are uniformly lower. The norms of both closed and perfused systems have limited dependence on TR. Similar results were observed with a lower driving exchange (data not shown).

Because of a fundamental motivating interest in the detection of changes in metabolism by MRS of hyperpolarized (HP)-pyruvate, we sought to determine which set of sequence parameters would provide the most accurate measurement of differences between high and low driving exchange rates. Maps for the error in the observed differences, or contrast error, for the closed and perfused systems are shown in Figure 4-7. In general, regions of sequence parameter values that result in the most accurate measurement of contrast closely match the corresponding regions for data reflecting the higher driving exchange rates. This is not true at the highest excitation angles, where very large errors in analysis of low driving exchange rate more significantly affect the differences that were observed.

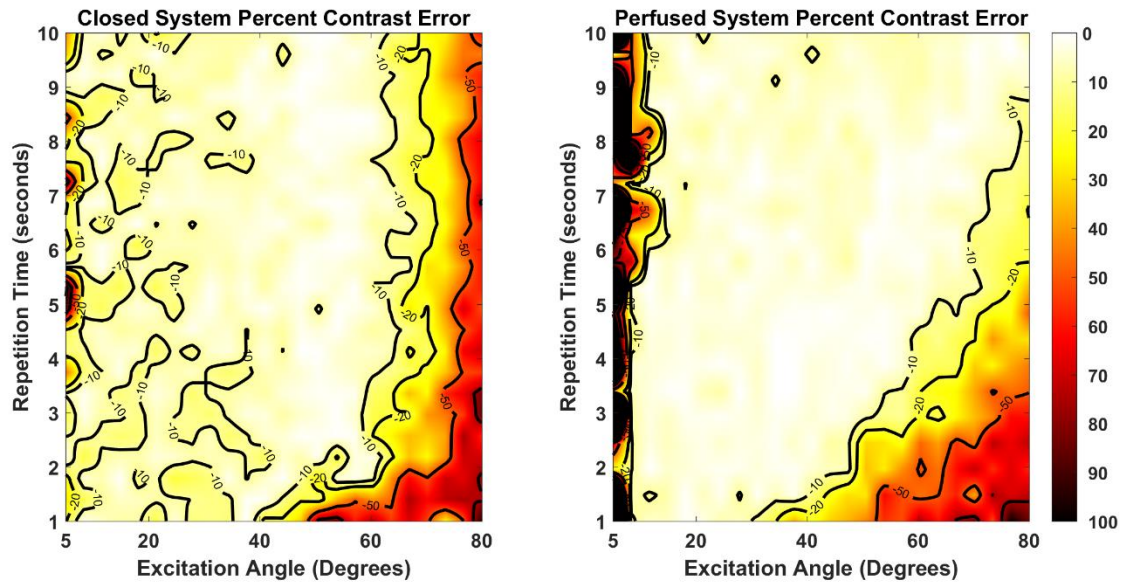


Figure 4-7. Contrast error maps for the closed (left) and perfused (right) system approximations. The errors ranged from 1% to more than 100%. The large discrepancy between the simulation exchange rates in the two systems led to accuracy plots that closely matched the higher exchange rate plots.

Using instantaneous excitation loss modeling in equation (4.6), figures 4-2 and 4-3 were recalculated. The results in figures 4-8 and 4-9 show a slight but meaningful divergence from the results modeled with excitation losses averaged over the entire repetition time. Generally, fitting with the instantaneous excitation losses allowed for more accurate k_{pl} measurement at longer repetition times and larger excitation angles and had little effect on accuracy for smaller excitation angles. This illustrates the limitations of the averaged excitation loss model and highlights the importance of calculating the basic physics modeled by the Bloch simulator. The errors introduced by the modeling assumptions, i.e., the differences between 4-8 and 4-9 vs 4-2 and 4-3, can be found by processing the same simulation data with different modeling assumptions.

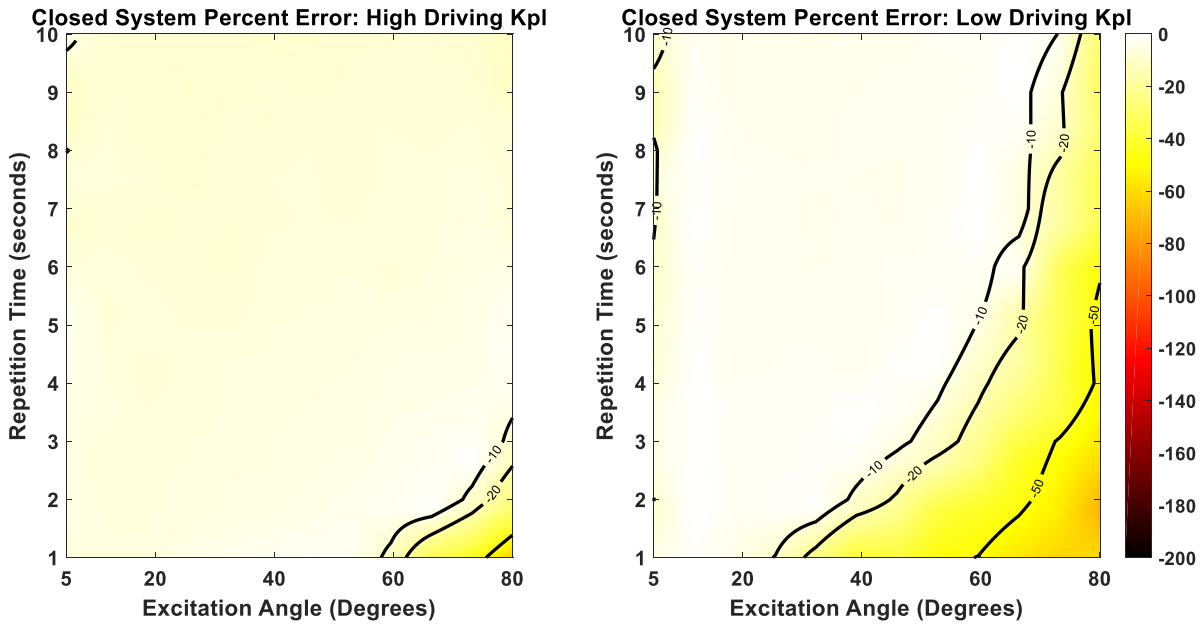


Figure 4-8. Percent error plots of driving versus fit exchange rates for the closed system approximation.

These are similar to Figure 4-2 but are fit with equation (4.6). Left, high simulation exchange rate of 0.1

s^{-1} ; right, low simulation exchange rate of $0.02 s^{-1}$.

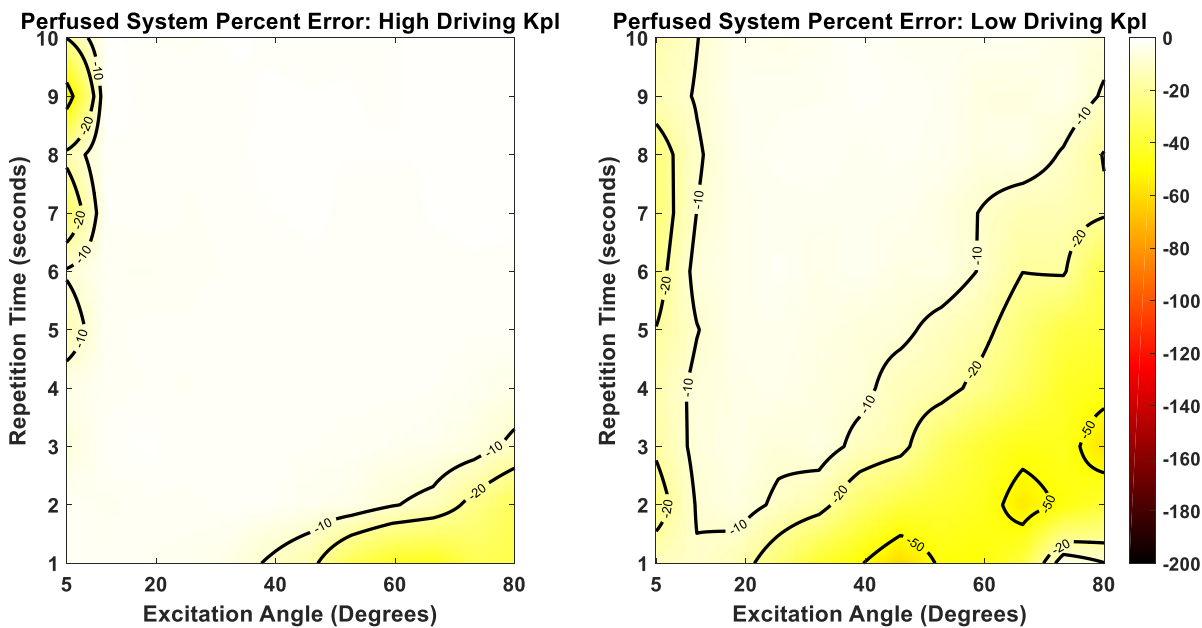


Figure 4-9. Percent error plots of driving versus fit exchange rates for the perfused system. These are similar to Figure 4-3 but are fit with equation (4.6). Left, high simulation exchange rate of 0.1 s^{-1} ; right, low simulation exchange rate of 0.02 s^{-1} .

The results in figure 4-2 demonstrated that in a closed system, sequence parameters have a limited effect on the accuracy of exchange rate measurements, and only become a significant source of error at extreme TR, excitation angles, or lower limits of chemical exchange. The closed system model best represents a phantom environment, but it does not realistically model all the characteristics of biological systems. In a perfused system (Figure 4-3), which applies to *in vivo* studies, sequence parameters can more significantly impact the measured exchange rates. As shown in figure 4-4 and 4-5, errors are unlikely to be a result of poor SNR except under relatively extreme conditions of very low excitation angles where the low total SNR does correspond to a region of inaccurate exchange rate fitting. The quality of the fit is also not a primary source of these errors. If poor fit quality were the dominate cause of inaccurate fitting results, correlation between higher fit residuals and error in k_{pl} would be expected. However, as shown in Figure 4-6, fit residuals are either uniform or do not correspond with sequence combinations that result in large k_{pl} errors in the fit (Figures 4-2 and 4-3).

Section 4.4 Discussion

This work develops the computational structure needed to begin designing and optimizing hyperpolarized acquisition strategies to be simulated. Hyperpolarized magnetic resonance is sensitive to a wide array of parameters, many of which add to its usefulness, such as chemical exchange, while others likely serve as confounders, such as sequence parameter dependence, sensitivity to agent delivery, decay constants' dependence on tissue type, etc. With such a complicated parameterization, as

well as multiple proposed models, the ability to rapidly and meaningfully simulate hyperpolarized studies allows quick and efficient exploration of these parameter spaces.

Using the simulation architecture, sensitivity to acquisition design and modeling assumptions was found for even the simplest dynamic spectroscopy studies of hyperpolarized pyruvate. Sequence parameters will have different effects on the accuracy of the results for perfused versus closed system assumptions. Therefore, optimization of sequences under a particular assumption may not apply under different delivery conditions.

Many physical and biologic processes affect the signal evolution in HP-MRS measurements. Since the acquisition strategy itself perturbs the system and affects subsequent measurements, it is critical that the acquisition strategy is not itself a confounder. If the parameter of interest is chemical exchange, the sampling strategy must sample the most critical information pertaining to the exchange rate. This work shows that properly tuned sequences result in more accurate estimation of the exchange rate than if less relevant data were sampled, such as exhaustive sampling before significant exchange has occurred.

At the extreme ranges of exchange rates, excitation angles, and TRs, the effects on fitting error are exacerbated in the closed system. A single 80° pulse reduces the entire signal of all of the subsequent measurements by 83%. If significant exchange of HP-pyruvate to lactate has yet to take place, then accurate estimation of the exchange rate is unlikely. This is likely to be the source of high error rates in excess of 250% in the situations with the low simulation exchange rates and high excitation angles as shown in Figure 4-3. If the chemical conversion is fast enough, rapid use of the signal from high excitation angles can still result in accurate exchange modeling, as significant lactate buildup will occur during the first few pulses. This explains the increased accuracy of results at high excitation angles and high simulation exchange rates. Perfusion enables fresh HP-pyruvate to flow into the tissue over time, reducing the attenuating effect of high excitation angles on the total SNR (Figure 4-

5), and may account for the reduced severity of the errors at high excitation angles and low simulation exchange rates shown in Figure 4-4. Additionally, all data sets exhibited accurate fit estimates at long TRs. This likely resulted from exact matching of the HP-pyruvate delivery in the analysis and driving models. In practice, pyruvate arrival time will not be known exactly as it is not detectable until after excitation. Very long TRs will then correspond to larger uncertainty in the pyruvate delivery time and will likely drive errors in the analysis. The effect of uncertain delivery could degrade the relatively accurate estimations of fit exchange at the longer TRs.

When attempting to detect a difference in the exchange rate of HP-pyruvate to lactate, investigators must take great care in selecting the sequence parameters, as the biases imposed by their sampling strategies may completely obscure any underlying rate differences. Attempting to find a single best-case sampling strategy for multiple pyruvate-to-lactate exchange rates may not always be possible and some sequence parameter bias could be unavoidable. Additionally, the sequence parameter effects on measurement will need to be accounted for when comparing rate measurements made with different sequence parameter values.

Although the exchange rate constants we considered represent the extremes of realistic metabolism, one of the strengths of using hyperpolarized pyruvate is the relatively large change in exchange rates that can be detected. Therefore, it is not unreasonable to have a study that attempts to detect a change in exchange rate of nearly an order of magnitude, as was simulated in this work. This large difference in exchange rates biased the contrast error to more closely match errors associated with the high simulation exchange rate. This is expected, as an error rate of 10% for an exchange rate of 0.1 will have a greater effect on the contrast than will the same percent error for an exchange rate of 0.02. Sequence parameter combinations that are accurate for the high simulation exchange rate data begin to degrade in terms of contrast accuracy at higher excitation angles for the closed approximation. This is because errors in the low simulation exchange become large enough to approach errors at the

higher exchange rate. Additionally, there were some sequence parameter combinations that resulted in extremely accurate detection of contrast with reduced accuracy for detection of either the high or low exchange rate data (Figures 4-4 and 4-8). This implies that the biases from those sequence parameters offset each other allowing for an accurate difference from two less accurate measurements.

The results of the perfused studies suggested the use of higher excitation angles than generally used. Conservative sampling strategies are used to ensure that the signal is not completely consumed before significant exchange of HP-pyruvate to lactate can progress. If fresh HP-pyruvate is constantly flowing into the voxel or slice over some time frame, such conservative sampling is no longer necessary. If the excitation pulse significantly impacts the bulk of the HP-pyruvate pool, such as in sampling of the heart or whole-body excitation, the assumption that fresh HP-pyruvate is flowing into the voxel would begin to breakdown and conservative sampling would likely be needed. Additionally, higher excitation angles will cause more sensitivity to errors in excitation angle and will require even more careful measurement of excitation profiles and calibration of excitation pulses.

High excitation angle schemes may not be effective for magnetic resonance spectroscopic imaging studies in which many more excitations are needed to encode spatial information. We anticipate that similar simulation-based studies of imaging sequences will highlight opportunities for optimization to improve image quality and quantitative accuracy.

In this study we assumed that every variable used in the analysis model aside from the exchange rate was known exactly. Future studies will be able to determine how sampling strategies affect estimates of pyruvate-to-lactate exchange rates with more unknowns in the analysis model. A critical examination of the propagation of errors for acquisition strategies that include prior information will also be crucial.

Although MRI and MRS of HP-agents have demonstrated amazing promise as a non-invasive clinical probe of metabolism, there are still many challenges ahead. Care must be taken to ensure that

this technique is optimally used as it moves toward clinical use, including a good understanding of circumstances that may lead to bias or error. This work shows that even the most simplistic pulse sequences and modeling strategies can result in estimates of chemical exchange that are dependent on acquisition parameters. Investigators must take great care in acquiring, processing, and comparing results from dynamic studies with HP-agents to ensure that sequence parameter effects are accounted for. Moreover, simulation studies such as these are imperative as increasingly advanced techniques are employed for acquisition, processing, or modeling of MRI and MRS of HP-agents. To that end, the modified Bloch-McConnell equations described herein will serve as powerful tools to characterize the complex relationships among detection methods and quantification of MRI and MRS of HP-agents.

Chapter 5. System Validation

This Chapter based upon

Walker, C. M., Chen, Y., Lai, S. Y. & Bankson, J. A. A novel perfused Bloch-McConnell simulator for analyzing the accuracy of dynamic hyperpolarized MRS. *Med Phys* 43, 854, doi:10.1118/1.4939877 (2016).

Copyright © 2016 American Association of Physicists in Medicine. Reproduced with permission of American Association of Physicists in Medicine,

Walker, C. M., Lee J., Ramirez M.S., Schellingerhout D., Millward S., Bankson J.A. A catalyzing phantom for reproducible dynamic conversion of hyperpolarized [1-(1)(3)C]-pyruvate. *PLoS One* 8, e71274, doi:10.1371/journal.pone.0071274 (2013)

© 2013 Walker et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

This chapter is intended to address Aim 3.

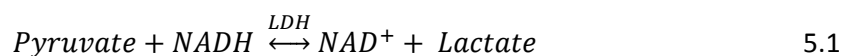
Section 5.1 Introduction and Theory

The simulation results outlined in the previous chapter need to be validated in physical systems. Using *in vivo* models will allow for inherent chemical conversion of pyruvate to lactate as well as delivery via endogenous vasculature. However, *in vivo* models have many practical limitations. Living systems are constantly changing and the assumption that the same study performed on the same living system at a different time point will yield the same measurement value, particularly in murine models of cancer does not generally hold. This is exacerbated by the amount of pyruvate that is generally delivered, which can be quite large in order to have sufficient signal and thus can alter the metabolism after injection.

Additionally, very little pyruvate is converted into lactate in healthy tissue and therefore, tumor models are normally used for hyperpolarized studies. Tumors introduce additional temporal heterogeneity as they are rapidly growing and the assumption that the same cellular and physiologic status remain across multiple days cannot be relied upon. It is also impossible to know the exact biologic and physical parameters in living systems, and therefore the determination of the accuracy of the quantization of said parameters is at best an approximation for living systems. These inherent limitations are coupled with the practical constraints on the use of living systems, such as cost, sensitivity to diet and anesthesia, etc. Initial validation of the simulation system would be simpler using a more controllable and repeatable model than those offered by living systems.

Fortunately, the conversion of pyruvate to lactate is a relatively simple reaction involving a single enzyme catalyst and coenzyme and can be readily performed in solution⁷⁷. This allows fine-tuned control over the rate and extent of the reaction that is more controllable and repeatable than a living system. Performing this reaction in a controlled buffer alters the delivery of pyruvate into the system as compared to living systems which have endogenous vasculature. Therefore, in a dynamic enzyme phantom, pyruvate will be delivered in a nearly instantaneous bolus and these results will be a closer match to the closed system results from the previous chapter.

Pyruvate is specifically converted into lactate by the enzyme lactate dehydrogenase (LDH) and coenzyme nicotinamide adenine dinucleotide (NADH):



This ordered ternary complex is modeled using classical enzyme kinetics^{57,78-80} to derive reaction velocities (Mol/s) of the reaction as a function of constituent concentrations as shown in Appendix A. The enzyme LDH is a relatively stable protein and can be mixed with NADH up to fairly high concentrations in a buffer. This enzyme mixture is then able to convert pyruvate into lactate, and if some of the pyruvate is hyperpolarized then the signal evolution can be measured using dynamic

spectroscopy. In order to measure such conversion, the phantom system needs to be positioned in the sensitive volume of the scanner and therefore some delivery system will be required. Additionally, B_0 homogeneity is critical for high quality magnetic resonance spectroscopy, and therefore interfaces between the buffer system and a substance with a different magnetic susceptibility, such as air or plastic, need to be reduced. Finally, the delivery and mixing of the hyperpolarized pyruvate with the enzyme system must be rapid to ensure that the pyruvate arrives in an instantaneous bolus and then reacts to form lactate as assumed for the closed system simulations. With such a phantom system, it will be possible to run multiple studies with the same conversion rate allowing for reliable measurement of a rate constant under specific sequence conditions. Multiple excitation angles and repetitions times can be used to ensure that the biases predicted by simulation for the closed system are measured in a physical system.

The perfused system assumes delivery via native vasculature that is difficult to emulate in phantom systems. Additionally, cellular metabolism is complex, and multiple processes are involved to maintain the cellular concentration of NADH and pyruvate, which are not replicated in an isolated buffer. Therefore, simulation studies were compared to a set of measurements made in a mouse model of thyroid cancer to demonstrate that the simulation predictions that most closely model living tissue are confirmed *in vivo*.

Section 5.2 Methods

A. Hyperpolarization

13 mg of [1- ^{13}C]-pyruvic acid with 7.5 mM OX063 (GE Healthcare) and 0.375 mM Prohance (Bracco Diagnostics) were hyperpolarized in a HyperSense DNP system (Oxford Instruments) as described previously^{25,81}. The sample was dissolved in 4 mL of buffer consisting of 40 mM 4-(2-hydroxyethyl)-1-piperazineethanesulfonic acid (HEPES), 94 mM NaOH, 30 mM NaCl, and 50 mg/L EDTA

with a pH of 12.5. Once the dissolution process was complete, 0.15 mL of HP [$1\text{-}^{13}\text{C}$]pyruvate (nominally 30% polarization) was drawn into a syringe for injection into the phantom.

B. Dynamic Spectroscopy Repeatability

The enzyme phantom consisted of 2 mL of buffer containing 2 mM hyperpolarized [$1\text{-}^{13}\text{C}$]pyruvate, 40 mM lactate, 3.92U/mL LDH, and 4 mM B-NADH. Phantom concentrations were optimized to reduce reaction rate sensitivity to variabilities in the concentrations of its components, ensure that the reaction had run to completion before the hyperpolarized signal had decayed below the threshold of detectability, and had progressed at a rate consistent with previous *in vivo* observations. Special consideration was given to reducing the sensitivity due to pyruvate concentration and LDH activity, as these were assumed to be the least reproducible characteristics of the phantom system. While LDH is a remarkably stable enzyme, it is still a delicate protein and is sensitive to many environmental conditions such as temperature and pH. Since the injection into the phantom system was performed by hand, the volume of pyruvate, and therefore its delivered concentration, would be more difficult to control than the concentration of any of the other reagents. A custom phantom container was machined out of a cylinder of Ultem resin stock, which matches the susceptibility of water, and fitted with a 1m long, 3.175 mm diameter polyethylene catheter (Coilhose Pneumatics, East Brunswick, NJ) for remote injection into the cavity when it is located at the isocenter of the magnet. The rectangular cavity was 1×1×3 cm with the injection catheter connecting to the front as shown in Figure 5-1. LDH and NADH were thawed from aliquoted solutions that had been stored at -80°C and were mixed in a 5 mL syringe shortly before the pyruvate finished polarizing. NADH was mixed with buffer to a concentration of 5mM and then froze in 200 μL aliquots while LDH aliquots of 200 μL at 250 U/mL in buffer. Once polarization of the pyruvate was complete, the HP pyruvate was injected into the phantom followed by the enzyme substrate mixture to fill the phantom cavity. The nominal final concentrations were 2 mM hyperpolarized ^{13}C -Pyruvate, 40 mM Lactate, 3.92U/mL LDH (Worthington), and 4 mM B-NADH (Sigma Aldrich) in the Tris

buffer (81.3 mM trisma preset crystals pH 7.2, 203.3 mM NaCl) (Sigma Aldrech). The phantom was held at 28°C with a final pH of 7.2 and a 3-mL final volume.

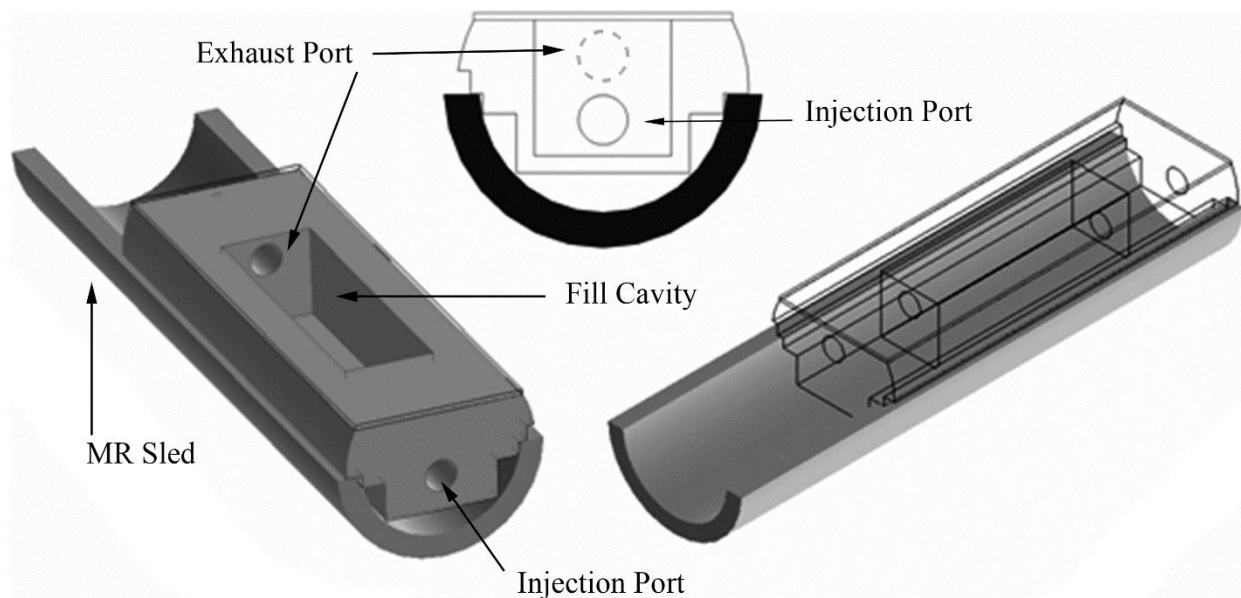


Figure 5-1. A schematic view of the dynamic chemical phantom structure. The injection and discharge ports were fitted with catheters to facilitate rapid mixture of reagents at the isocenter. A thin acrylic sheet was attached to the top to seal the fill cavity. This top could be removed to allow cleaning after injection. The phantom rested on a sled that allowed convenient removal and insertion of the phantom and included warm circulating water to maintain a constant temperature.

Dynamic spectroscopy was performed on a 7-T/30-cm Biospec System (Bruker Biospin Corp., Billerica, MA) using B-GA12 gradient (120-mm inner diameter (ID), $G_{\text{max}} = 400 \text{ mT/m}$) and a dual-tuned $^1\text{H}/^{13}\text{C}$ volume coil (72-mm ID, Bruker Biospin MRI). Dynamic ^{13}C spectra were acquired with a 2.5 kHz bandwidth, 4098 points, 10° excitations, 2-sec TR, with 60 repetitions over a 3-min scan time beginning at dissolution and triggered by the HyperSense system. To evaluate performance and repeatability, the measurement was repeated seven times using an identical reagent concentration.

The signal from each metabolite at each TR was determined by integrating the full-width at half-maximum (FWHM) of each metabolite peak. Signal amplitudes were normalized to account for variations in the amount of polarized pyruvate that is present at the onset of scanning. Two quantitative parameters were used to characterize the reaction rate for each measurement: total lactate signal normalized to the total carbon signal, and k_{PL} for the closed system model described by equations (4.1).

C. Spectroscopic Phantom Imaging

To demonstrate the usefulness of the enzyme phantom for evaluating spatial sequence performance, a 10-mL standard imaging phantom was drained and fitted with the same injection catheter described above. A slightly lower concentration of NADH (2 mM) was used among an otherwise identical mixture due to the increased phantom volume. A custom-built dual-tuned $^1\text{H}/^{13}\text{C}$ linear birdcage coil with a 35 mm ID was used in conjunction with B-G6 gradients (60-mm ID, $G_{\text{max}} = 1000 \text{ mT/m}$, Bruker Biospin Corp.) . The phantom was scanned with a radial echo planar spectral imaging (EPSI) sequence⁸². This was a single image and consume the entire hyperpolarized signal to acquire a single set of spectroscopic imaging data. The acquisition was started ~40 seconds after all components were combined in the phantom, and the data were acquired with a repetition time of 60 ms, an initial echo time of 5.5 ms, and a 1.3 msec echo spacing to form a 32-point echo train. A variable flip angle was used to ensure equal sampling of the longitudinal magnetization⁸³. The spectral bandwidth was 23.8 kHz with a 744 Hz or 9.85 ppm spectral width. Fifty spatial projections were taken with 32 readout points over a 4 cm by 4 cm field of view and a 2 cm slice thickness.

D. Dynamic Spectroscopy Sequence Parameter Dependence

Slightly altered phantom concentrations of 2 mM hyperpolarized $[1-^{13}\text{C}]$ -pyruvate, 40 mM lactate, 4U/mL LDH, and 4 mM B-NADH were used to assess and validate the simulation results for the closed system. A slightly altered phantom enclosure was used where the cavity was 1x1x2 cm is size.

Dynamic spectroscopy was performed on a 7-T/30-cm Biospec System (Bruker Biospin Corp., Billerica, MA) using B-GA12SHP gradient and a dual-tuned $^1\text{H}/^{13}\text{C}$ volume coil (72-mm ID, Bruker Biospin MRI). Simulation results for the closed system slow exchange, figure 4-2, suggested that dynamic spectroscopy acquired at $\text{TR}=2\text{-s}$, $\theta = 20^\circ$ and $\text{TR}=7\text{-s}$, $\theta = 60^\circ$ would not bias the measurement and should result in a similar k'_{pl} measurement while using $\text{TR}=2\text{-s}$, $\theta = 60^\circ$ would result in a significant underestimation of k'_{pl} . Dynamic ^{13}C pulse-acquire spectroscopy was performed at 4096 Hz bandwidth over 2048 spectral points, and three combinations of excitation angle and repetition time ($\text{TR}=2\text{s}$, $\theta = 20^\circ$; $\text{TR}=2\text{s}$, $\theta = 60^\circ$; or $\text{TR} = 7\text{s}$, $\theta = 60^\circ$) were used, with a total scan duration of 3-min beginning 20 seconds before dissolution. Each parameter combination was repeated three times.

The dynamic spectroscopy signal was analyzed using the same process as had been used for the simulated data to generate dynamic curves. The curves were fit with T1s for pyruvate and lactate of 61 and 35 sec respectively as consistent with previous measurements. The exchange rate was fit with the closed system model using the signal values at the time of the peak of the pyruvate signal as an initial condition. The studies were grouped based on the sequence parameters and a two-tailed t-test assuming unequal variances was used to detect any differences between the groups.

E. Dynamic Spectroscopy in Vivo

Nude mice bearing orthotopic xenografts of anaplastic thyroid cancer⁸⁴ were anesthetized and placed prone on an imaging sled. 2% isoflurane in oxygen was delivered through a nose cone under observation using a commercial small-animal physiological monitoring system (Small Animal Instruments, Inc., Stony Brook, New York). 200 μL of HP [$1\text{-}^{13}\text{C}$] pyruvate (nominally 30% polarization) was administered to the animals via a tail-vein catheter. All animal procedures were approved by our Institutional Animal Care and Use Committee, which is accredited by the Association for the Assessment and Accreditation of Laboratory Animal Care International.

Imaging and dynamic spectroscopy was performed on a 7-T/30-cm Biospec System (Bruker Biospin Corp., Billerica, MA) using B-GA12SHP gradient and a dual-tuned $^1\text{H}/^{13}\text{C}$ volume coil (40-mm ID, Bruker Biospin MRI). Simulation of the high exchange rate perfused data, figure 4-3, also predicted a significant underestimation of k'_{pl} at $\text{TR}=2\text{-s}$, $\theta = 60^\circ$ and no bias at $\text{TR}=2\text{-s}$, $\theta = 20^\circ$ and $\text{TR}=7\text{-s}$, $\theta = 60^\circ$. Therefore, slice selective dynamic ^{13}C pulse-acquire spectra were acquired with a 10-mm slice centered over the tumors, a 5 kHz bandwidth over 2048 spectral points, and three combinations of excitation angle and repetition time ($\text{TR}=2\text{s}$, $\theta=20^\circ$; $\text{TR}=2\text{s}$, $\theta=60^\circ$; or $\text{TR}=7\text{s}$, $\theta=60^\circ$), with a total scan duration of 3-min beginning at dissolution and triggered by the HyperSense system.

Signals from *in vivo* dynamic spectroscopy were analyzed using the same process as was used for the simulated data to generate dynamic curves. The unknowns that were determined by analysis of the dynamic curves included k_{pl} , the shape terms for the gamma variate VIF, the injection time, and the excitation angle. The fitting results for the excitation angle never differed by more than 8% from the prescribed excitation angle and mainly served as an internal control. The studies were grouped based on the sequence parameters and a two-tailed t-test assuming unequal variances was used to detect any differences between groups.

Section 5.3 Results

A. Repeatability Studies

The phantom system, shown in Figure 5-1, was assembled and tested (N=7 replicates), demonstrating reproducible conversion of hyperpolarized tracer as summarized in Figure 5-2 and table 5-1. After a brief delay between the start of data acquisition and the injection of the polarized tracer, the pyruvate signal peaked quickly as it filled the chamber. The pyruvate signal decayed due to relaxation, signal excitation, and chemical conversion to undetectable levels in less than two minutes. The lactate signal rose until the growth of the HP lactate pool, from chemical exchange, was reduced below the losses due to relaxation and signal excitation at which point it similarly decayed. The

coefficient of variation for common measures of this reaction, including the ratio of total lactate to total ^{13}C signal and the forward reaction rate (k_{pl}) were 14.5% and 19.0%, respectively, as summarized in Table 4-1. This level of variability is less than the average within-group variation of approximately 28% in 9 animal studies^{16,85-92} that was reported recently in the literature, and is summarized in Table 5-2.

Dynamic Signal Curves

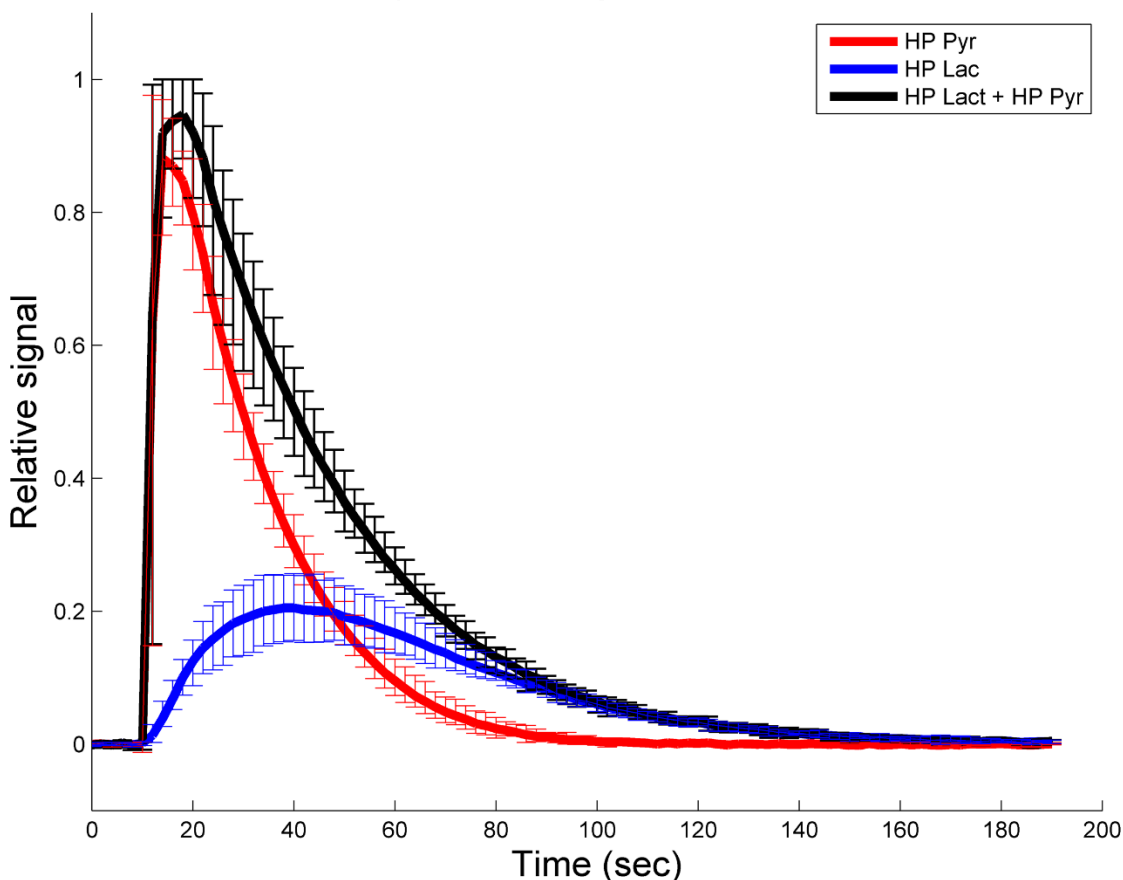


Figure 5-2. Dynamic signal evolution across (seven) injections into enzyme phantom. The mean signal for lactate and pyruvate, normalized to the peak carbon signal for each injection, is displayed with error bars that indicate the minimum and maximum values at each time over all injections. The total HP ^{13}C was estimated by summing the signal from HP ^{13}C Lactate and HP ^{13}C Pyruvate. The average linewidth for pyruvate and lactate peaks were 19 ± 5 Hz and 17 ± 5 Hz, respectively.

Parameter	Mean	Standard Deviation	Coefficient of Variation
Lac/(Lac+Pyr)	0.391	0.048	12.3%
Lac/Pyr	0.652	0.124	19.0%
k_{PL}	0.020	0.0038	19.0%

Table 5-1. The mean, standard deviation and coefficient of variation for all repetitions (N = 7) of the dynamic phantom

Reference	Location	Disease	Parameter	No. Animals	Coeff. of Variation
Albers ⁸⁵	Prostate	Cancer	Metabolite SNR	5,4,3,3,	25%
Day ¹⁶	Subcutaneous	Lymphoma	Kpl	8	17%
Laustsen ⁸⁶	Kidney	Diabetes	Lac/Total 13C Signal	10,6	40%
Thind ⁸⁸	Thorax	Radiation Injury	Lac/Pyr	6,4,5	36%
Bohndiek ⁹¹	Subcutaneous	Colorectal Cancer	Lac/Pyr	N/A	24%
Park ⁹²	Brain	Glioblastoma	Lac/Pyr	7,9	54%
Bohndiek	Subcutaneous	Lymphoma	Kpl	10,7,7,	37%
Matsumoto	Subcutaneous	Squamous Cell Carc.	Lac/Total 13C Signal	5,4	12%
Laustsen ⁸⁶	Heart	Normal	Lac/Total 13C Signal	11,6	28%
Average					29%

Table 5-2. Survey of HP parameter variation in recent animal studies

To test the feasibility for such a reaction to be conducted inside a phantom with the spatial details that are necessary to validate spectroscopic imaging sequences, a standard MRI quality assurance phantom was drained and refilled with a similar catalytic mixture. Snapshot spectroscopic imaging shows a relatively homogeneous mixture of components and distribution of the agent and metabolite 40s after initiation of the reaction. Images of HP pyruvate and lactate, alone and in overlay over reference proton images, can be seen in Figure 5-3. While the resolution of the MRSI sequence is significantly lower than that of the proton image, it is possible to resolve features within the spectroscopic images for both individual metabolites. No significant spatial distortions are seen, but importantly, artifacts, specifically interpolation artifacts can seen as thin black lines on the pyruvate and lactate images. They can be identified and could be characterized through the use of this phantom system and minimized to reduce the likelihood of interference in subsequent measurements made *in vivo*.

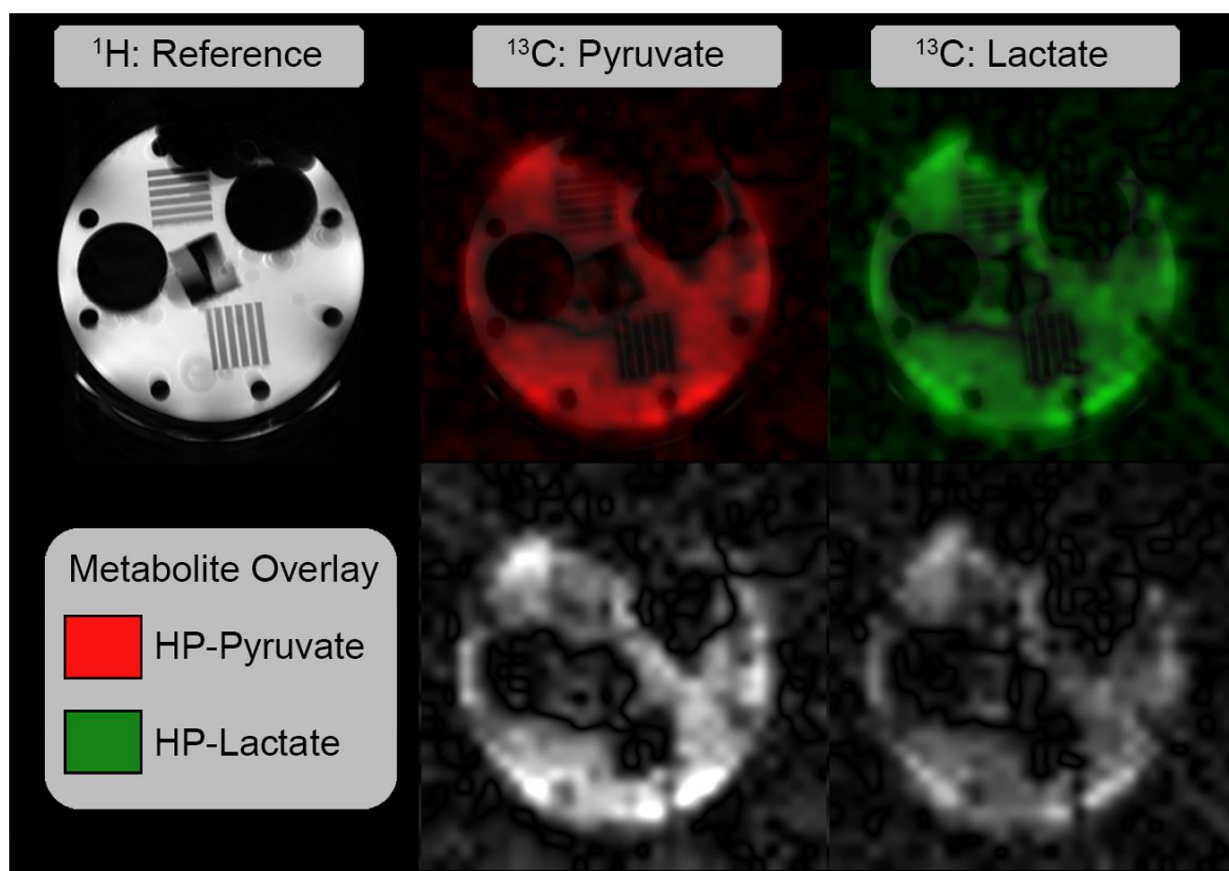


Figure 5-3. Spectroscopic images of the reaction carried out in a standard imaging phantom. Proton imaging (top left) shows the phantom structure in high resolution. Spectroscopic imaging data acquired using a radial EPSI sequence allow metabolite-specific visualization of the agent distribution (bottom row). Spectroscopic data can be intrinsically registered to high-resolution proton images (top center and top right).

B. Closed System Parameter Dependence

Due to poor heating of the second phantom structure, the rate of chemical exchange of the third set of phantoms studies was much lower than those of the prior two studies. This system was much closer to the low conversion rate system seen in figure 4-2. To test for the large error in k'_{pl} at high excitation angles and short repetition time predicted by figure 4-2, three excitation angles were used (N=3

replicates). As seen in figure 5-4, the mean k'_{pl} low excitation angle ($TR=2-s, \theta = 20^\circ$) closely matched those at the longer repetition times ($TR=7-s, \theta = 60^\circ$) and no significant difference was detected, $p=0.737$. When the excitation angle was high and the repetition time was short ($TR=2-s, \theta = 60^\circ$) there was no detectable lactate peak and the only signal detected at 183.1 was likely due to noise or spillover from the sizable pyruvate or pyruvate hydrate peaks. As predicted by simulation, the average k'_{pl} detected at $TR=2-s, \theta = 60^\circ$ was significantly lower than $TR = 2-s, \theta = 20^\circ$ and $TR = 7-s, \theta = 60^\circ$ with $p = 0.002$ and 0.003 respectively. Qualitatively, the predicted and measure signal curves show agreed remarkably well.

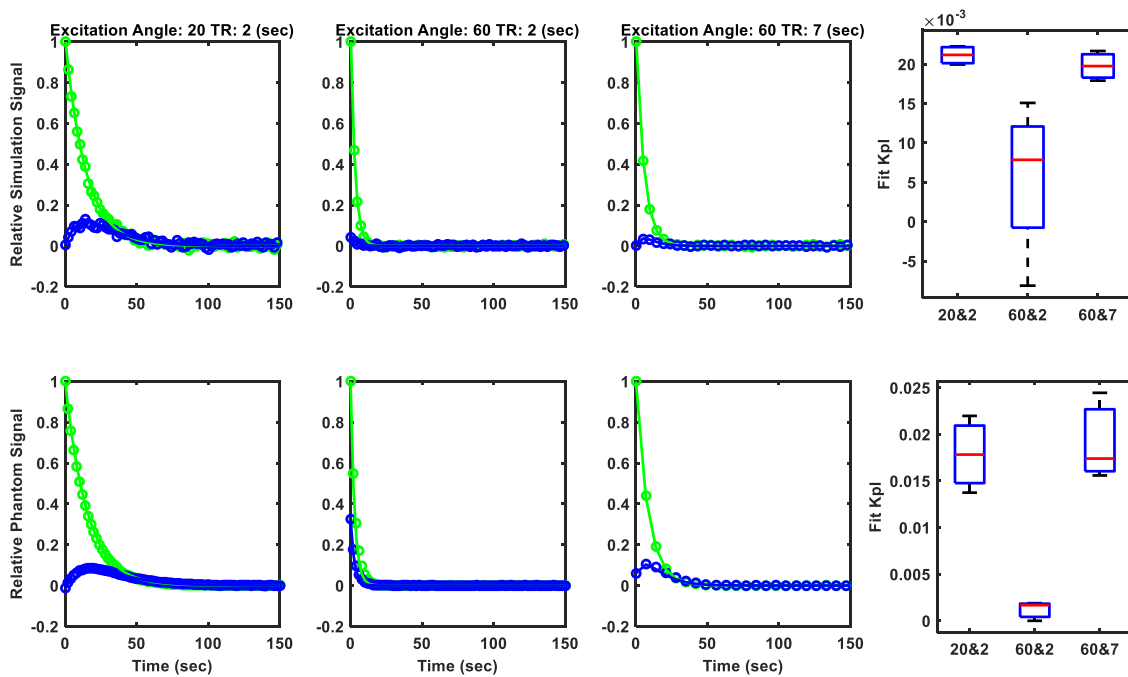


Figure 5-4. Comparing simulated to dynamic phantom data for the closed system. A qualitative comparison of the closed system signal curves predicted by the simulation at various excitation angles and repetition times to the measured signal curves in the dynamic phantom system. Additionally, the fitted k'_{pl} values are compared in both the simulation and the phantom.

B. In Vivo Studies

To examine the correspondence between these simulations and measurements *in vivo*, a cohort of mice were scanned using protocols with parameter combinations that the simulations suggested would introduce varying levels of measurement bias. The results (Fig. 5-5) show that when $TR=2$ -s and $\theta=60^\circ$, exchange is significantly underestimated compared to $TR=2$ -s, $\theta=20^\circ$ ($P=0.035$), where relatively accurate measurements are expected. Interestingly, the exchange measured with $TR=2$ -s and $\theta=60^\circ$ was $\sim 50\%$ lower than the values measured at $TR=2$ -s and $\theta=20^\circ$. This closely matches the predicted bias of $\sim 60\%$ seen in Figure 4-3. Additionally, bias is reduced again with $TR=7$ -s and $\theta=60^\circ$ which also agrees Figure 4-3. Notably the variance is higher under these conditions, which is likely due at least in part to the increased uncertainty in the injection time due to the longer sampling intervals.

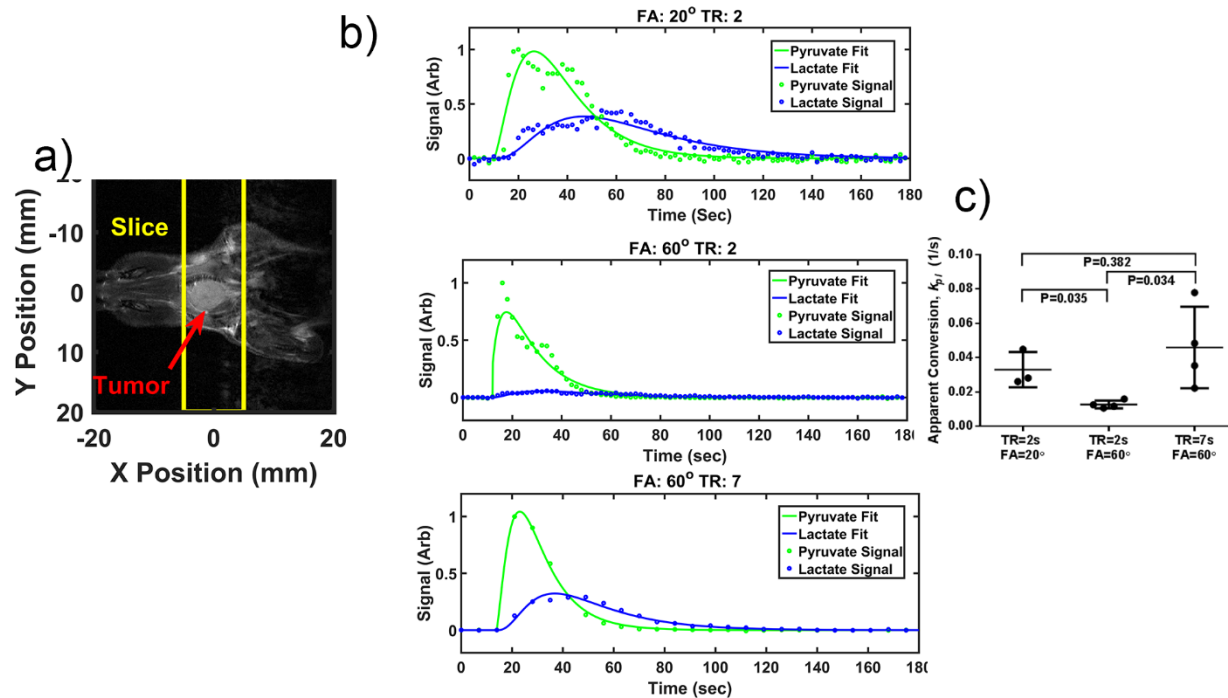


Figure 5-5. Comparison of *in vivo* vs. simulated kinetic data analysis from data acquired using different acquisition parameter combinations. a) Anatomical image of a mouse bearing an anaplastic thyroid tumor and the slice used for dynamic HP spectroscopy. b) Dynamic metabolite curves of the same animal scanned with excitation $TR=2$ -s and $\theta=20^\circ$ (top), $TR=2$ -s and $\theta=60^\circ$ (middle), and $TR=7$ -s and $\theta=60^\circ$

=60° (bottom). c) k_{pl} values from animals scanned with TR=2-s and $\theta = 20^\circ$ (n=3), TR=2-s and $\theta = 60^\circ$ (n=4), and TR=7-s and $\theta = 60^\circ$ (n=4). Data acquired with TR=2-s and $\theta = 60^\circ$ significantly underestimates k_{pl} compared to the other two groups ($P < 0.035$).

Section 5.4 Discussion

These simulation results are of limited use in isolation and require validation in physical systems. However, physical systems for repeated controlled hyperpolarized studies are not yet well developed and some inherent challenges remain. When the study endpoint is the characterization of chemical exchange, a dynamic chemical reaction will be needed. Additionally, the system will need to be able to repeatedly carry out the reaction of interest in some controllable manner. These two requirements make working in living systems practically challenging. In order to move away from living systems, a novel dynamic chemical exchange phantom was developed where exchange rates could be controlled. It demonstrated an improved repeatability over *in vivo* systems. This system was used to validate simulation predictions that did not assume pyruvate delivery by native vasculature.

This phantom system provides new capabilities for experimental development and validation with distinct advantages over single-tracer injections, static multi-compartment thermal equilibrium phantoms, and in-vivo models. The platform provides dynamic evolution of HP tracer signals through chemical exchange in a manner that is consistent with that observed in target biology and can be tuned to mimic different disease conditions. The spatial characteristics of the phantom are known a priori, allowing rigorous evaluation of data encoding, acquisition, and reconstruction algorithms. This is especially important when considering data reduction strategies that are designed to address key limitations in the measurement of hyperpolarized tracers but that blur traditional definitions of spatial and temporal resolution in the observation of dynamic processes. Static phantoms are useful for confirming some functionality, but do not create the dynamic conditions that could lead to artifacts in

reconstruction algorithms that are based to any extent on the assumption of a stationary subject.

Assessment using *in vivo* models is challenging because of biological heterogeneity and the evolution of target processes in diseases such as cancer that can progress rapidly and increase within-group variations even in a matter of days. With this platform, acquisitions can be readily repeated, at arbitrary intervals, to extract statistical measures of image properties. The system has a known distribution of metabolites, and could be designed with multiple compartments⁷³ with reaction rates tuned to simulate different tissues or disease states in parallel. This platform is ideal for exploration of thresholds for detectability of pathologies that may not be evident in ¹H MRI, for early testing of new sequences to ensure preservation of spatial and temporal accuracy, and even for regular quality assurance scans to confirm that similar acquisition, reconstruction, and analysis parameters lead to similar data over time both within and between laboratories and institutions.

Hyperpolarized contrast agents are relatively new, and research into the best practices for signal acquisition, reconstruction, and analysis is ongoing. This dynamic phantom will enable robust, reproducible, and tunable baseline measurements, providing a benchmark through which experimental strategies can be compared and optimized. This system catalyzes the final step in aerobic glycolysis, the conversion of pyruvate into lactate, without the need for animal subjects, human subjects, or cell suspensions that can increase the cost and the variability of technical measurements. The 14.5-19% variation that we observed is a result of many factors. LDH is sensitive to a range of experimental factors⁷⁷; small variations in temperature, pH, or even time from thawing to injection can affect the enzyme activity and therefore the rate of the reaction. To ensure that the reaction progresses to completion, which is truest to *in vivo* studies, NADH has to be in excess and thus the rate of the reaction will depend on pyruvate concentration. In this work, the injection of a small amount of hyperpolarized pyruvate was performed by hand, potentially leading to unnecessarily high variations in the final concentration of

pyruvate. This variability can be reduced by utilizing automated injections that are currently under development.

A crucial step in the translation of powerful new imaging technologies into routine preclinical and clinical use is the establishment of well-defined reference standards⁹³ to provide a common reference against which experimental circumstances can be compared. This reference can be used to ensure comparable results across platforms, laboratories, and institutions, and to aid in study design and execution. This dynamic single enzyme phantom helps fill this critical need. The physical structure of the phantom can be tailored to more closely approximate preclinical or clinical applications, and the rate of the reaction can be controlled through multiple compartments in a spatially-dependent manner to simulate a wide range of disease states. This phantom platform represents a flexible and powerful tool to aid in the development, optimization, validation, and certification of techniques, processes, and instrumentation that are crucial to ensure the successful and efficient clinical translation of powerful new imaging capabilities afforded by MRSI of hyperpolarized tracers such as [1-¹³C]-pyruvate.

Using the phantom system, the simulation prediction from chapter 4, namely that a low rate of conversion, high excitation angles and rapid repetition times would suppress the apparent production of hyperpolarized lactate, was confirmed. Tuning the phantom system to match the low conversion rate used in the simulations showed a remarkable correlation in the expected mean k'_{pl} measured and the signal evolution curves. This shows that the dynamic enzyme phantom system was an ideal model to validate the simulation architecture in the simplest case, where endogenous vasculature delivery is ignored. Additionally, the *in vivo* studies show strong agreement with the simulation predictions demonstrating the validity of the simulation architecture to account for perfusion. In aggregate, these results serve as a strong validation of the simulation architecture and support the dual ideas that simulation of hyperpolarized studies is a useful method for developing and optimizing acquisitions.

Chapter 6. Conclusion and Future Work

Magnetic resonance spectroscopy of hyperpolarized agents, specifically pyruvate, is a powerful tool in characterizing tissue. However, to be fully realized as a clinical biomarker, HP-pyruvate must directly relate to metabolism in a well characterized quantitative manner. To ensure that clinical endpoint is robust enough to be used in clinical decision making, the verification, validation and optimization tools that were outline in this work will be critical. The parameter space associated with hyperpolarized MRI is extensive, with acquisition design, tissue characteristics like perfusion, and cellular processes such as uptake and redox status all playing a role. The Bloch simulator developed in this work overcomes the computational burdens associated with modeling hyperpolarized signal evolution to allow rapid exploration of the parameter space associated with hyperpolarized pyruvate. Additionally, because it is a simulation platform, the underlying values of parameters of interest are known. Therefore, the accuracy and reproducibility of data processing and modeling strategies can be evaluated with a fidelity that is not possible in physical systems.

Using this simulation architecture, it was shown that the excitation angle and repetition time that are used for dynamic spectroscopy can significantly bias the measurement of the exchange rate for hyperpolarized pyruvate. Stated generally, rapidly pulsing with high excitation angles leads to a significant underestimation of the exchange rate while no underestimation was observed for rapid, small excitation or slow, large excitation. This bias was demonstrated across a range of metabolic parameters, perfusion models and data processing strategies. The bias did not correlate with sequences that lead to poor quality fits or to a low SNR suggesting that the bias imposed is inherent in the acquisition and is not caused by poor data quality or modeling. The sequences tested represent the most simplistic hyperpolarized studies, and their inherent dependence on the acquisition parameters stresses the critical need for all hyperpolarized acquisition and processing strategies, especially the more

complicated methods that have been proposed for hyperpolarized imaging, to be thoroughly characterized and validated using systems such as the Bloch simulator developed in this work.

In order to extend the simulation result into a physical system, exchange rates were measured in an isolated phantom that showed superior reproducibility to current *in vivo* work in the field. The phantom system was then used to show that rapid large, excitation schemes do significantly underestimate the measure exchange rate as compared to rapid, low excitation or slow, large excitation schemes. The phantom system designed in this work is more than a tool to validate simulation results; it represents the necessary structure for validation of any hyperpolarized study where exchange measurement is the endpoint. Additionally, the phantom platform will serve as an ideal standard for quality assurance and validation as hyperpolarization moves into routine clinical care.

While the phantom system reproducibly converts HP-pyruvate to HP-lactate, it doesn't fully mimic living systems. Using a mouse model of thyroid cancer, the simulation results for a perfused system were confirmed, showing that even in living systems the acquisition parameters can significantly alter exchange rate quantification. These results show that quantitative measurement of hyperpolarized exchange rates is sensitive to the acquisition parameters. The computational and physical platforms developed in this work are ideal tools for careful validation and optimization of such acquisitions.

B. Future Directions for the Simulation Architecture

The strength of this platform is understood by considering its future directions. The equations presented in this project do not account for spatial dimensions. However, incorporating three-dimensional space is fairly straightforward. The only real complexity comes in the form of the computational burden, as the incorporation of three spatial dimensions greatly increases the number of spins that must be independently calculated. Preliminary spectroscopic imaging sequences⁷⁵ have been developed for this simulation platform. Example images acquired using radial multi-band frequency

encoding are displayed⁷⁵ in Figure 6-1. These images are “snap-shot” images and attempt to characterize the spectral and spatial aspects of a system as a single time point or over some time window at the cost of the entire hyperpolarized signal. These snap shot images, while useful as shown by their use in the first clinical²⁸ trial with hyperpolarized pyruvate, are not the absolute end goal of hyperpolarized studies. The ability to monitor a metabolic process with hyperpolarized pyruvate makes the loss of temporal resolution unacceptable in most cases. However, in order to encode the spectral, spatial, and temporal aspects of a hyperpolarized signal requires either spectrally selective excitation pulses⁹⁴⁻⁹⁷ or advanced reconstructions^{53,75,98-101}. Additionally, if properly carried out, dynamic spectral spatial studies can be processed to yield an exchange rate constant which can then be quantitatively compared to the actual exchange rate used in the imaging voxel. This direct method of comparison to a physical parameter allows straightforward determination of accuracy with methods that are very similar to this work. Finally, spectroscopic imaging tends to require many more excitations than dynamic spectroscopy and a similar, if not exacerbated, dependence on sequence parameters is likely. Both spectral-spatial pulse based imaging and advance constrained reconstructions are presently being added to the simulation architecture to ensure its continued utility to the field of hyperpolarized magnet resonance as it moves from dynamic spectroscopy to dynamic spectral imaging.

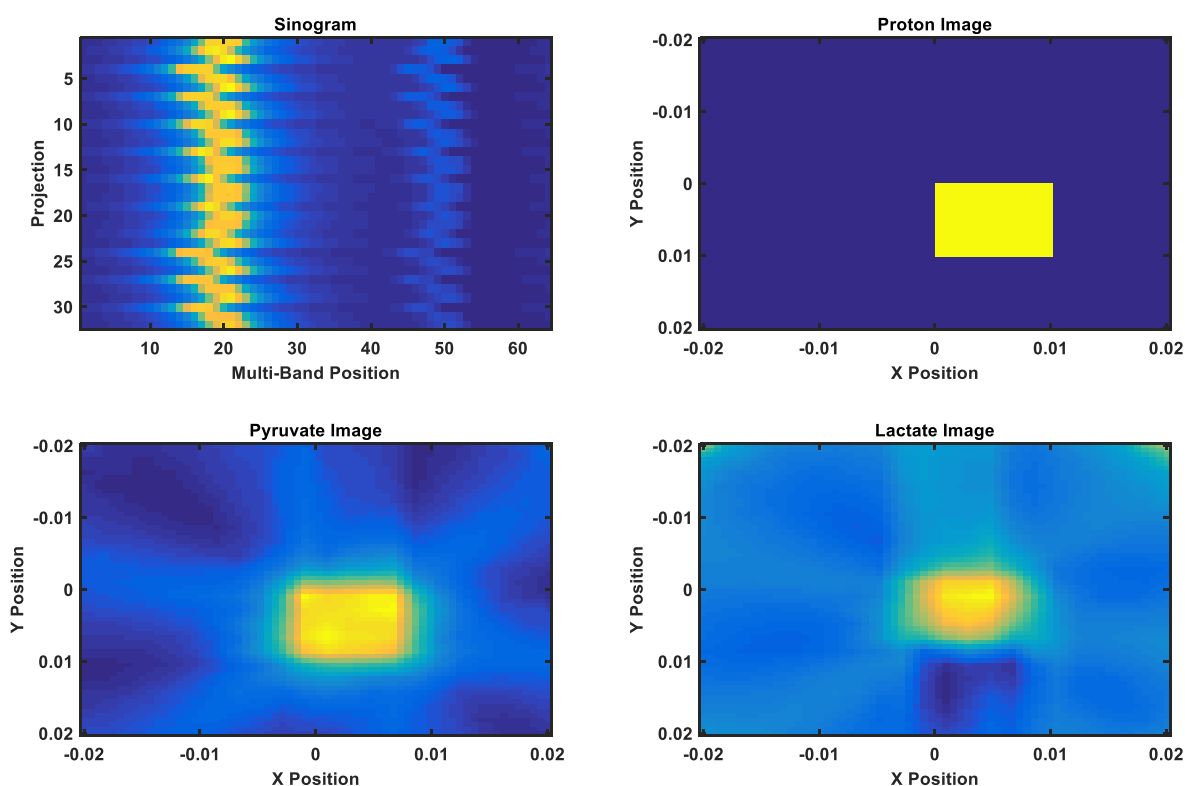


Figure 6-1. Simulated radial multi-band frequency-encoded snap shot image of a square of perfused tissue converting pyruvate to lactate. The images were simulated with 64 voxels (an 8x8 grid for 10x10 mm area) containing identical spins.

C. Future Directions for the Dynamic Phantom

Finally, the dynamic phantom developed to validate these studies represents a powerful paradigm for not only validation of hyperpolarized acquisition and processing but also as a reference standard for quality assurance which will be greatly needed as hyperpolarized imaging moves into routine clinical use. The initial phantoms that we have described have some limitations. Most paramount relates to the use of enzymes that are sensitive to a plethora of reaction conditions as well as to storage and age. We have considered an alternative reaction, requiring only simple chemical compounds. As shown in figure 6-2, when mixed with hydrogen peroxide, pyruvate is broken down into

acetate and CO_2 ¹⁰². This reaction involves no delicate enzymes or coenzyme and can be tuned by the concentration of either pyruvate or hydrogen peroxide. As shown in figure 6-2, this reaction can proceed in solution with rates similar to the lactate-to-pyruvate exchange rate seen *in-vivo*. Therefore, for the simple chemical conversion of pyruvate into some downstream product, the far more robust conversion to acetate is maybe a preferred choice. However, if the specific chemical shifts of the metabolites or the method of exchange are critical to the detection process then the full enzyme system will be needed. This will be true for excitation schemes that are specific to lactate's and pyruvate's resonance frequencies or if the readout band is tuned precisely to a particular set of chemical shifts.

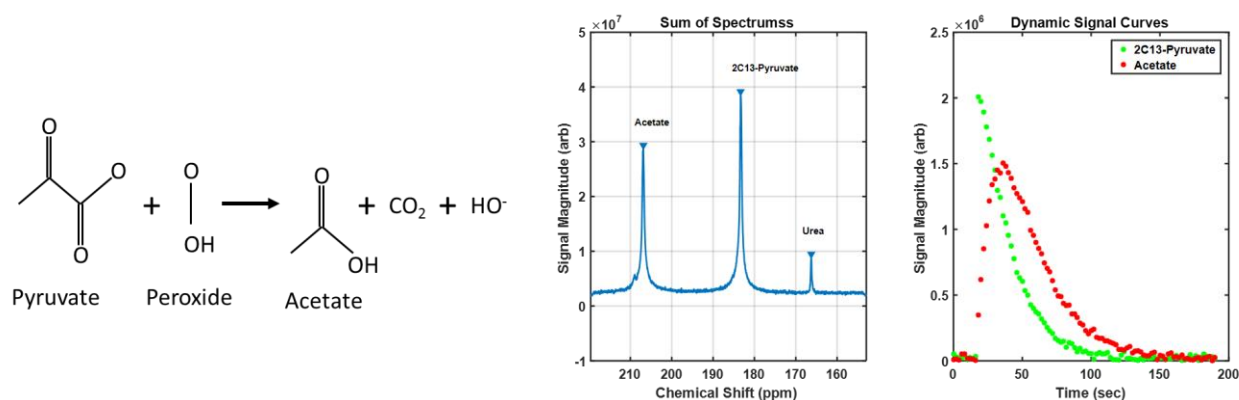


Figure 6-2. The reaction schematic of pyruvate and peroxide (left). The total spectrum of hyperpolarized 2-C13-Pyruvate reacting with peroxide in a phantom monitored with magnetic resonance spectroscopy (center). The dynamic signal curves of hyperpolarized pyruvate and acetate showing the initial arrival of pyruvate and its subsequent reaction to acetate (right).

Additional work is underway to better characterize the enzyme rate constants for LDH so that the heuristic concentrations used in this phantom design can be altered to account for different concentrations, enzyme activity or even temperature. Ideally a set of experimental constraints, the desired reaction rate, and the enzyme kinetics outlined in appendix A would be used to calculate the exact phantom concentrations to be used. Measurement of the physical constants needed for such a

system are currently being investigated. Additionally, the phantom enclosure itself is being further refined. Multiple chambers have been developed to allow for the exchange rate contrast to be explored in a single study⁷³, and the capability to automatically deliver a fixed amount of pyruvate over a repeatable time frame is under development.

In summary, this work aimed at developing a robust simulation platform for hyperpolarized magnetic resonance spectroscopy. The simulation platform was leveraged to show that sequence parameters significantly bias the measured exchange rates. Such bias was validated in a novel phantom system designed to approximate the chemical conversion of pyruvate to lactate observed *in vivo* while offering improved repeatability and practicality. Finally, that sequence parameter imposed bias predicted for perfused tissue was validated in a mouse cohort. These initial validation studies show that sequence parameters will affect the exchange rate quantification for hyperpolarized pyruvate. Additionally, no significant bias in exchange rate measurements was detected using sequence parameters designed by simulation to avoid such bias.

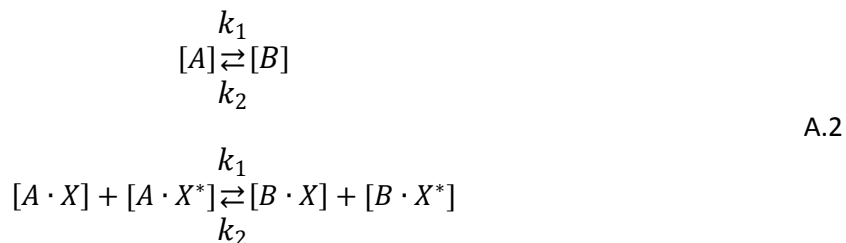
Appendix A: Hyperpolarized Exchange Kinetics

Section A.1 Label Exchange

Practically hyperpolarized nuclei will never make up the entirety of the nuclei for a specific chemical species in a sample. Therefore, hyperpolarized nuclei can be considered as a label that is placed on a fraction of the nuclei in a system⁶⁷. This fractional labeling can be modeled as:

$$[A] = [A \cdot X] + [A \cdot X^*] \quad \text{A.1}$$

where $[A]$ is the total concentration of the chemical species A , $[A \cdot X]$ is the concentration of A without a hyperpolarized nuclei and $[A \cdot X^*]$ is the concentration of A with a hyperpolarized nucleus. If A is exchanging with a separate chemical species B following equation (3.4):



Then the rate of change of the labels will simply be the rate of exchange between the two pools multiplied by the probability that an exchanging compound will be labeled:

$$\begin{aligned} \frac{d[A \cdot X^*]}{dt} &= V_r \frac{[B \cdot X^*]}{[B]} - V_f \frac{[A \cdot X^*]}{[A]} \\ \frac{d[B \cdot X^*]}{dt} &= V_f \frac{[A \cdot X^*]}{[A]} - V_r \frac{[B \cdot X^*]}{[B]} \end{aligned} \quad \text{A.3}$$

where V_f and V_r are the forward and reverse velocity of the reaction respectively. Note the net change in reaction is $V_f - V_r$. Under the conditions outlined in equation A.2 the forward and reverse velocities will be:

$$\begin{aligned} V_f &= k_1[A] \\ V_r &= k_2[B] \end{aligned} \quad \text{A.4}$$

Substituting A.4 into A.3 yields:

$$\frac{d[A \cdot X^*]}{dt} = k_2[B] \frac{[B \cdot X^*]}{[B]} - k_1[A] \frac{[A \cdot X^*]}{[A]} = k_2[B \cdot X^*] - k_1[A \cdot X^*]$$

$$\frac{d[B \cdot X^*]}{dt} = k_1[A] \frac{[A \cdot X^*]}{[A]} - k_2[B] \frac{[B \cdot X^*]}{[B]} = k_1[A \cdot X^*] - k_2[B \cdot X^*]$$
A.5

Therefore, assuming first order kinetics, the rate of change of the labeled compound is determined solely by the concentration of the labeled compounds and the exchange terms. The rate of change is independent of any net exchange of the total compound. The equivalence of equations A.5 and A.3 is shown in figure A-1.

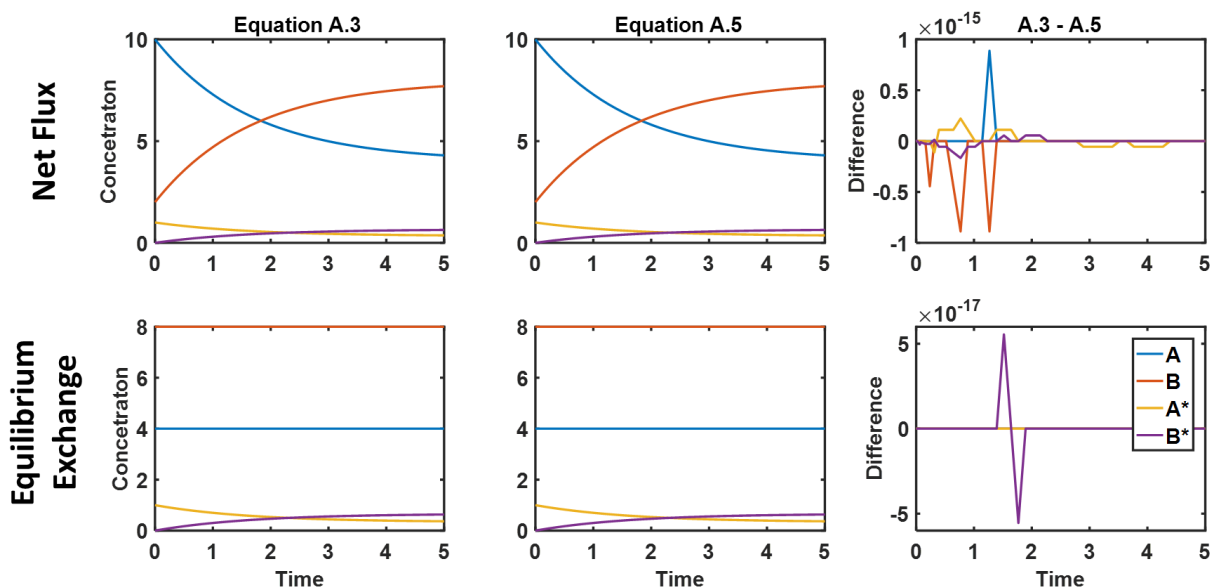
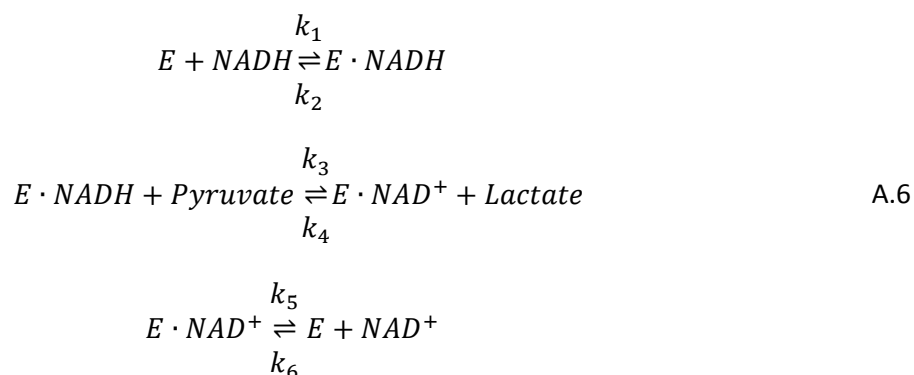


Figure A-1. Comparison of label exchange modeling. A comparison of a full modeling of label exchange, equation (A.3) vs the reduced form equation (A.5) under both net flux conditions (top) and equilibrium exchange (bottom)

Section A.2 Enzyme Flux Kinetics

In the case of hyperpolarized pyruvate, it is converted to lactate via an enzyme catalyzed reaction and equation A.2 might not strictly hold. The conversion of lactate to pyruvate has been shown to follow a Theorell-Chance mechanism¹⁰³ outlined schematically in figure A-2 and is given by⁶⁶:



where E is free enzyme, $E \cdot NADH$ and $E \cdot NAD^+$ is enzyme bound to $NADH$ or NAD^+ respectively, and the k_i terms are the rate constants for the i^{th} reaction step.

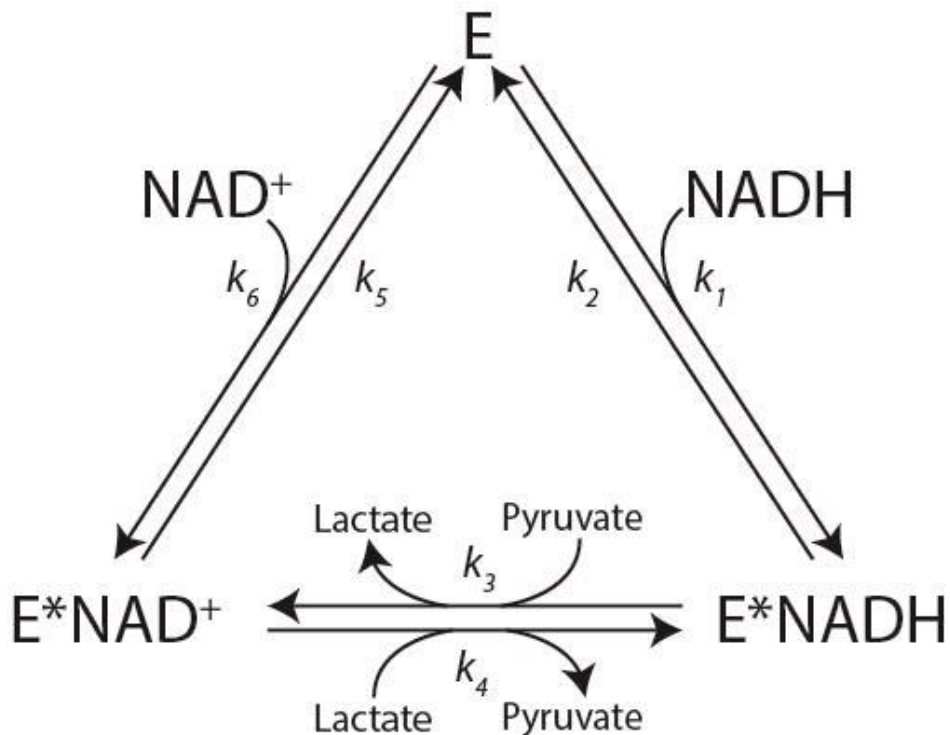


Figure A-2. A schematic of the Theorell-Chance mechanism for lactate dehydrogenase. Free enzyme is bound by either $NADH$ or NAD^+ , and these enzyme complexes are then able to convert pyruvate to lactate. The binding, conversion and disassociation of either pyruvate or lactate are so rapid that they are considered as a single step with a single rate constant pair. Such an approximation is known as a Theorell-Chance mechanism.

Assuming that enzyme concentration is low enough that there is little change in the concentration of the enzyme complexes over the majority of the reaction, known as the steady state assumption⁶⁶:

$$\frac{dE}{dt} = \frac{dE \cdot NAD^+}{dt} = \frac{dE \cdot NADH}{dt} = 0 \quad A.7$$

$$E_T = E + E \cdot NAD^+ + E \cdot NADH$$

where E_T is the total enzyme concentration. Under the steady state assumption, the method outlined by Fromm¹⁰⁴ can be used to determine the forward and reverse velocities of the reaction. Initially the concentrations of the individual enzyme species are determined based on the reagent concentrations. For LDH this would be:

$$\begin{aligned} [E] &= k_2k_5 + k_2k_4[L] + k_5k_3[P] \\ [E \cdot NADH] &= k_1k_4[L][NADH] + k_1k_5[NADH] + k_4k_6[L][NAD^+] \\ [E \cdot NAD^+] &= k_2k_6[NAD^+] + k_3k_6[P][NAD^+] + k_1k_3[P][NADH] \\ [E_T] &= k_2k_5 + k_2k_4[L] + k_5k_3[P] + k_1k_4[L][NADH] + k_1k_5[NADH] \\ &\quad + k_4k_6[L][NAD^+] + k_2k_6[NAD^+] + k_3k_6[P][NAD^+] \\ &\quad + k_1k_3[P][NADH] \end{aligned} \quad A.8$$

Because the concentration of the enzyme species is not changing, the velocity of each step must be the same for each direction. Therefore, the forward velocity will be:

$$V_f = k_3[P][E \cdot NADH] \quad A.9$$

It is difficult to determine the concentration of the particular enzyme complexes. However, the total enzyme concentration is often known. Multiplying equation (A.9) by $\frac{[E_T]}{[E_T]}$ and substituting equation (A.8) for $[E \cdot NADH]$ yields:

$$V_f = \frac{E_T k_3 [P] (k_1 k_4 [L] [NADH] + k_1 k_5 [NADH] + k_4 k_6 [L] [NAD^+])}{denom}$$

$$denom = k_2 k_5 + k_2 k_4 [L] + k_5 k_3 [P] + k_1 k_4 [L] [NADH] + k_1 k_5 [NADH] + k_4 k_6 [L] [NAD^+] + k_2 k_6 [NAD^+] + k_3 k_6 [P] [NAD^+] + k_1 k_3 [P] [NADH]$$
A.10

Similarly, the reverse velocity would be given by:

$$V_r = \frac{E_T k_4 [L] (k_2 k_6 [NAD^+] + k_3 k_6 [P] [NAD^+] + k_1 k_3 [P] [NADH])}{denom}$$
A.11

Therefore, the net change in the reaction would follow:

$$v \equiv \frac{d[L]}{dt} = \frac{-d[P]}{dt} = V_f - V_r$$
A.12

The $E_T k_3 k_4 k_6 [L] [P] [NAD^+]$ and $E_T k_1 k_3 k_4 [L] [P] [NADH]$ appear in both V_f and V_r and will cancel leaving:

$$v = \frac{E_T (k_1 k_3 k_5 [P] [NADH] - k_2 k_4 k_6 [L] [NAD^+])}{denom}$$
A.13

If only the initial rate is considered⁸⁰, then $[L] = [NAD^+] = 0$ and equation (A.13) reduces to:

$$\frac{v}{E_t} = \frac{k_1 k_3 k_5 [P] [NADH]}{k_2 k_5 + k_5 k_3 [P] + k_1 k_5 [NADH] + k_1 k_3 [P] [NADH]}$$
A.14

Multiplying by $\frac{1/k_1 k_3 k_5 [P] [NADH]}{1/k_1 k_3 k_5 [P] [NADH]}$ to cancel out the top terms yields:

$$\frac{v}{E_t} = \frac{1}{k_2 / \frac{1}{k_1 k_3 [P] [NADH]} + \frac{1}{k_1 [NADH]} + \frac{1}{k_3 [P]} + \frac{1}{k_5}}$$
A.15

Inverting results in the initial rate of the reaction, which has been shown to match the rates that were measured in LDH isolated from rabbit muscle^{57,79,80}. Note that the exchange rate diagram used in those

references assume that the conversion of lactate to pyruvate is the forward reaction and the exchange constants, k_i , are in reverse order than this derivation.

$$\frac{E_t}{v} = \frac{k_2}{k_1 k_3 [P][NADH]} + \frac{1}{k_1 [NADH]} + \frac{1}{k_3 [P]} + \frac{1}{k_5} \quad \text{A.16}$$

If the products $[L]$ and $[NAD^+]$ are present, then equation (A.16) needs to be modified following the procedure outlined in ^{4,105}, and accounting for product and substrate inhibition¹⁰⁴ equation (A.16) is modified to:

$$\frac{[E]_t}{v} = \frac{\left(1 + \frac{[P]}{k_i}\right)}{k_5} + \frac{1}{k_1 [NADH]} + \frac{\left(1 + \frac{[L]}{k_{ii}}\right) \left(1 + \frac{k_5 [L]}{k_4}\right)}{k_3 [P]} + \frac{k_2 \left(1 + \frac{k_5 [L]}{k_4}\right)}{k_1 k_3 [P][NADH]} \quad \text{A.17}$$

This matches the initial reaction velocity of pyruvate conversion in lysed lymphoma cells⁵⁷.

Section A.3 Enzyme Exchange Kinetics

If just the forward velocity is to be considered, then only equation (A.9) needs to be considered.

By inspection of figure A-2 the relation:

$$[E \cdot NADH] = \frac{k_1 [E][NADH]}{k_2} \quad \text{A.18}$$

is apparent. Substituting equation (A.16) into (A.9) yields:

$$V_f = \frac{k_3 k_1}{k_2} [P][E][NADH] \quad \text{A.19}$$

In order to remove the $[E]$ terms they are to be replaced with $[E_T]$. Combining equation (A.8) with equation (A.18) and noting relations similar to equation (A.18) by inspection of figure A-2, then:

$$[E \cdot NAD^+] = \frac{k_1 k_3 [E][P][NADH]}{k_2 k_4} \quad \text{A.20}$$

$$[E_T] = [E] \left\{ 1 + \frac{k_3 k_1}{k_2} [P][NADH] + \frac{k_1 k_3 [E][P][NADH]}{k_2 k_4} \right\} \quad \text{A.21}$$

Note that equation (A.20) could have been written as a function of $[NAD^+]$ and $[L]$ but this increases the number of concentrations that need to be considered. Multiplying equation (A.19) by $\frac{E_T}{E_T}$ to remove the $[E]$ term and substituting equation (A.21) leaves:

$$V_f = E_T \frac{\left(\frac{k_3 k_1}{k_2} [P][NADH]\right)}{1 + \frac{k_3 k_1}{k_2} [P][NADH] + \frac{k_1 k_3 [E][P][NADH]}{k_2 k_4}} \quad \text{A.22}$$

Finally, accounting for inhibitory complexes equation (A.22) is expanded to:

$$V_f = E_T \frac{\left(\frac{k_3 k_1}{k_2} [P][NADH]\right)}{1 + \frac{k_3 k_1}{k_2} [P][NADH] + \frac{k_1 k_3 [P][NADH] \left(1 + \frac{[P]}{K_i}\right)}{k_2 k_4}} \quad \text{A.23}$$

Equation (A.23) has been show to model the exchange of hyperpolarized pyruvate to lactate in murine lymphoma cells⁵⁷.

Section A.3 Enzyme Exchange Kinetics of Hyperpolarized Pyruvate

Relating equation (A.23) to equation (A.5) assuming no reverse exchange results in:

$$\begin{aligned} \frac{d[P^*]}{dt} &= E_T \frac{\left(\frac{k_3 k_1}{k_2} [NADH]\right) [P]}{1 + \frac{k_3 k_1}{k_2} [P][NADH] + \frac{k_1 k_3 [P][NADH] \left(1 + \frac{[P]}{K_i}\right)}{k_2 k_4}} \times \frac{[P^*]}{[P]} \\ &= E_T \frac{\left(\frac{k_3 k_1}{k_2} [NADH]\right) [P^*]}{1 + \frac{k_3 k_1}{k_2} [P][NADH] + \frac{k_1 k_3 [P][NADH] \left(1 + \frac{[P]}{K_i}\right)}{k_2 k_4}} \end{aligned} \quad \text{A.24}$$

For the pseudo first-order kinetics that are assumed in equation (3.4), (3.5) and (A.2), all the terms except $[P^*]$ must remain relatively constant. If that condition is met, then:

$$k'_{pl} = E_T \frac{\left(\frac{k_3 k_1}{k_2} [NADH]\right)}{1 + \frac{k_3 k_1}{k_2} [P][NADH] + \frac{k_1 k_3 [P][NADH] \left(1 + \frac{[P]}{K_i}\right)}{k_2 k_4}} \quad \text{A.23}$$

This implies that the apparent exchange rate measured by hyperpolarized pyruvate is simply the rate constant that defines the enzymatic exchange rate as a function of pyruvate concentration. If, however, there was some reverse conversion of lactate to pyruvate, or the substrate concentration were time dependent, then the apparent exchange rate would be a more complicated parameter and unlikely to be constant over time.

Appendix B: Source Code for HypWright

In this section the source code for the perfused Bloch-McConnell simulator is present along with a brief discussion of the functionality of the code. Discussion of particular design decisions will be outlined along with alternative approaches and areas for planned future development

Section B.1 Higher Level Structures

World Class

The World object follows a singleton pattern. Logically, there should never be more than a single world and therefore enforcing a singleton pattern ensures that user errors such as multiple worlds operating simultaneously can be avoided. This may be an issue for batch processing and the removal of the singleton framework may need to be considered for future studies.

The world object is first initialized which clears out any old world properties leaving a fresh world to be populated with the simulation parameters. The world stores the strength of the static magnetic field, which is assumed to be ever present and homogenous. If more complicated B_0 interaction need to be accounted for, a separate scanner object may need to be developed and incorporated into the pulse sequence objects. The pulse sequence object is the logical structure that stores all of the sequence information and is described in section B.2. Aside from the pulse sequence and B_0 the final simulation parameter stored in the world object is an array of voxels. Voxels represent a volume of space and are described after the world object.

The world object additionally stores the value of the last time point at which a solution was calculated. Attempting to evaluate the simulation past this point will result in an error. Finally, there is memory allocated for solutions to be stored on the world level. However, current implementation stores the solution at the voxel level. Lower level storage of the solution structure allows easier access to the

solution values but is vulnerable to user errors such as the addition of a voxel after calculation which will not have a defined solution.

An attempt has been made to parallelize this code with respect to voxels since they are assumed to be independent. However, storing and mixing the magnetization from multiple parallel voxels was non-trivial, and more work will be needed to ensure proper parallelization. Once functional, a similar parallelization algorithm should work for the calculation step, and will likely be simpler as calculation result are stored on the voxel level and do not need to be aggregated.

```
classdef (Sealed) world < handle
    %WORLD: Hello World!
    % storage for global system states, this is a singleton and global
    % PROPERTIES
    % B0 - The main magnetic field of the scanner
    % pulseSequence - The MR pulse sequence
    % Voxels - all of the active voxels
    % init - logical for weather or not the world has been initiated
    % calEndTime - the last timepoint for which a solution has been found
    % METHODS
    % B0 - The main magnetic field of the scanner, default 3T
    % PulseSequence - The MR pulse sequence
    % Voxels - all of the active voxels
    % init - logical for weather or not the world has been initiated
    % Methods
    % setB0(B0) - sets the static B field to B0
    % setPulseSequence(pulseSequence) - sets the stored pulse sequence
    % to the input PulseSequence
    % initworld() initializes the world and sets B0 to its default and
    % initializes an empty pulse sequence
    % initworld(B0) - initializes the world with some input B0 that
    % should be a 3x1 column vector of the form [x;y;z]
    % addVoxel(voxelList) - adds all the voxels in voxelList to the
    % world
    % calculate(times) - calculates the MR signal for all the time points
    % specified by (time)
    % evaluate(time) - returns a M vector for each time point in the
    % time vector t
    % *Note M is only defined from 0 to the end time passed in to
    % calculate, and will only reflect the system state at the last
    % calculate. This function will sort the time vector and remove any
    % points outside of the range [0, calEndTime] as well as removing
    % any redundant time points rounding to the nearest picosecond
    properties
    end
    properties (SetAccess = private)
        B0; % The main magnetic field of the scanner
```

```

pulseSequence; % The MR pulse sequence
voxels; % all of the active voxels
init; % logical for whether or not the world has been initiated
calEndTime % the last timepoint for which a solution has been found
solutions; % a cell of enums that stores the calculated solutions
end
properties (Constant)
    voxelSize = 1e-15; % The voxel size, I am not sure if this is used
end

methods (Access = private)
    function self = world
        % CONSTRUCTOR: Starts the Bloch Simulator. Initializes a pulse
        % sequence
        self.init = false;
    end
end
methods (Static)
    function singleObj = getworld
        persistent localObj
        if isempty(localObj) || ~isvalid(localObj)
            localObj = Hypwright.world;
        end
        singleObj = localObj;
    end
end
methods
    function value = getB0(self), value = self.B0; end
    function setB0(self,B0), self.B0 = B0; end
    function value = getPulseSequence(self), value = self.pulseSequence; end
    function setPulseSequence(self,pulseSequence), self.pulseSequence = pulseSequence; end
    function b = getB(self,x,y,z,t)
        % Gets the combined magnetic field from all sources at a
        % position (x,y,z) and a time t.
        b = repmat(self.B0,1,length(t))+self.pulseSequence.B(x,y,z,t);
    end

    function initworld(self,varargin)
        % INITWORLD: initializes the a new empty world.
        % initworld() initializes the world and sets B0 to its default and
        % initializes an empty pulse sequence and clears out any voxel
        % initworld(B0) - initializes the world with some input B0 that
        % should be a 3x1 column vector of the form [x;y;z]
        p = inputParser;
        p.addParameter('B0',[0;0;3.0],@isnumeric);
        p.parse(varargin{:})
        self.B0 = p.Results.B0;
        self.pulseSequence = [];
        self.solutions = [];
        self.calEndTime = 0;
        self.clearVoxels();
        self.init = true;
    end
end

```

```

function addVoxel(self,voxelList)
    % ADDVOXEL - adds a voxel to the world
    % addVoxel(voxelList) - adds all the voxels in voxelList to the
    % world. They are added in the order they were passed in.
    % Currently there is not great for managing and manipulating
    % multiple voxels, this should probably be addressed if more
    % complicated voxel geometries are to be used, probably a
    % factory object.
    for i = 1:numel(voxelList)
        self.Voxels = [self.Voxels,voxelList(i)];
    end
end
function clearVoxels(self)
    % CLEARVOXELS - removes all voxels from the world
    self.Voxels = [];
end
function calculate(self,timeRange)
    % CALCULATE - calculates the MR signal from all voxels over
    % some time range, assumes a start time of zeros if only one
    % number is passed in

    % Compiles the pulse sequence for efficiency. Need to add a
    % check to make sure the sequence has not already been
    % compiled.
    self.pulseSequence.compile();
    % Generate temporary variables for clarity
    tmpEndTime = timeRange(end);
    tmpPS = self.pulseSequence;
    tmpB0 = self.B0;
    tmpSolutions = cell(numel(self.Voxels),1); % Allocate space for the solutions
    % Let each voxel calculate it's own solution,
    for i = 1:numel(self.Voxels)
        tmpSolutions{i} = self.Voxels(i).calculate(tmpEndTime,tmpPS,tmpB0);
    end
    self.solutions = tmpSolutions; % store the solutions and replace any old solutions
    % stores the end time of the calculated range
    self.calEndTime = tmpEndTime;
end
function [signal, freqAxis, timeAxis,M] = evaluate(self,times,varargin)
    % EVALUATE - returns the complex MR signal for the time points
    % passed in. The world needs to be calculated before it can be
    % evaluated.
    % [signal, freqAxis, timeAxis] = evaluate(times) -
    % evaluate(time) - returns a M vector for each time point in the
    % time vector t
    % *Note M is only defined from 0 to the end time passed in to
    % calculate, and will only reflect the system state at the last
    % calculate. This function will sort the time vector and remove any
    % points outside of the range [0, calEndTime] as well as removing
    % any redundant time points rounding to the nearest picosecond
    times = (unique(round(times.*10e12)))./10e12;
    times = sort(times);
    p = inputParser;

```

```

p.addOptional('ref',0,@isnumeric)
p.addOptional('verbose',0,@islogical)
p.parse(varargin{:})
tmpB0 = self.B0;
tmpRef = p.Results.ref;
tmpM = zeros(numel(self.Voxels),3,length(times));
tmpVoxels = self.Voxels;
for i = 1:numel(self.Voxels)
    tmpM(i, :, :) = tmpVoxels(i).getM(times,tmpRef,tmpB0);
end
% Attempt to parallelize Not working
tic
%
% tmpSolutions = self.solutions;
% tmpM = zeros(numel(self.Voxels),...
%     size(tmpSolutions{i}.functions,2),length(times),3);
% parfor i = 1:numel(self.Voxels)
%     for j = 1:size(tmpSolutions{i}.functions,2)
%         density = tmpSolutions{i}.spinDensity{j};
%         omegaRef = tmpSolutions{i}.frameFreq{j}(tmpB0);
%         for k = 1:length(times)
%             iSolution = find(times(k)<tmpSolutions{i}.pulseTimes,1,'first');
%             if isempty(iSolution)
%                 iSolution = numel(tmpSolutions{i}.pulseTimes);
%             end
%             theta = times(k)*omegaRef;
%             if tmpSolutions{i}.useAnalytical(iSolution,j)
%                 tmpMFrame = tmpSolutions{i}.functions{iSolution}(times(k));
%             else
%                 tmpMFrame = deval(...
%                     tmpSolutions{i}.functions{iSolution},(times(k)));
%             end
%             MSum = [0;0;0];
%             for m = 1:3:size(tmpMFrame,1)
%                 MSum = MSum+density*...
%                     [cos(theta),-sin(theta),0;...
%                      sin(theta),cos(theta),0;...
%                      0,0,1]*tmpMFrame(m:m+2);
%             end
%             tmp1(i) = MSum(1);
%             tmp2(i) = MSum(2);
%             tmp3(i) = MSum(3);
%         end
%     end
% end
% % tmpM = [tmp1;tmp2;tmp3];
% toc
% keyboard
M = squeeze(sum(tmpM,1));
signal = M(1,:)+1i*M(2,:);
signal = signal.';
BW = 1/(times(2)-times(1));
freqAxis = linspace(-BW/2,BW/2,length(times));
timeAxis = times;

```

```
if (p.Results.verbose)
    figure
    subplot(2,1,1),plot(times,real(signal),'r',times,imag(signal),...
        'b')
    xlabel('Time (seconds)')
    FTSig = fftshift(fft(fftshift(signal)));
    subplot(2,1,2),plot(freqAxis,real(FTSig),'r',freqAxis,...
        imag(FTSig),'b',freqAxis,abs(FTSig),'k')
    xlabel('Frequency (HZ)')
end
end
```

```
end
```

```
end
```

Voxel Class

The voxel class is the main working object of the simulation structure. Nearly all calculation and evaluation is done in this class. Logically, it represents some arbitrary volume of space that is best described by a single point in space as the extent of the voxel is currently poorly defined. A more rigorous definition of a voxel may be necessary as more complicated numeric phantoms are considered. However, adding much more complexity to the voxel will start pushing this simulator into finite element methodologies.

The voxel is packed with an arbitrary number of spin objects. The density property that is inherent to each spin controls how much of the signal that spin contributes to the voxels' signal. The solutions are calculated using two methods as outline in Section 3. Spins must store a function that will return their time derivative at a position and time, as well as some initial condition that is assumed to exist at time 0. This is used for the iterative solver that utilizes an adaptive 4th order *Runge-Kutta* method. If applicable, an analytical solution can also be defined. To use the analytical solution, the voxel checks with both the spin object and the pulse sequence to ensure that the analytical solution is valid. If so, the analytical solution is defined by the spin object and then stored in the voxel. This level of abstraction allows for a wide variety of ordinary differential equations to be solved and the solution space stored for subsequent evaluation with arbitrary temporal precision. Additionally, all the implementation details are stored in the spin objects themselves, so as long as the spin objects have a valid interface with the voxel they can be properly solved.

Additionally, it should be noted that the analytical solution is evaluated for each time point independently. This can lead to some redundancy when time integrals are part of the solution. Independent evaluations of such integrals can lead to significant duplication of calculation as the same integral is calculated over very similar time frames. It is likely that a cumulative sum approach that

utilizes the previous evaluation points would remove this overhead and greatly speed up the analytical solution for perfused systems.

```
classdef voxel < handle
    %VOXEL Represents a volume of space
    % A voxel represents a volume of space that contains some set of spins.
    % once the calculate method has been run the voxel stores a solution
    % describing the evolution of the total magnetization vector up to some
    % time. this can be evaluated with the getM method.
    % Properties
    % position - Vector defining the position of the voxel
    % Methods
    % voxel(position) - initializes an empty voxel at the coordinates
    % defined by position
    % voxel(position, spinList)- initializes a voxel at the coordinates
    % defined by position and fills it with any spin groups in the
    % optional variable spinList
    % addSpin(spin) - adds the input spin to the voxel
    % calculate(endTime) - runs the solvers to the specified end time
    % getM(t) - returns the magnetization vector for all time points in
    % the vector t

    properties
        position % Vector defining the position of the voxel
        sol = {}; % list of all the solution structures defining the time evolution of M
        analyticSol = {}; % functions for all the analytic solutions
        debug = false; % switches on debug mode (will save the Mz of the spin)
        solTimes % the time ranges each solution structure spans
    end
    properties (SetAccess = private)
        solution % cell that stores all the data needed to solve for this voxel
    end
    properties (Access = private)
        spinGroups % list of all spins in the voxel
    end
    properties (Constant)
        T2Star = 2e-14; % determines B0 inhomogeneity thus T2 star in this voxel
        numSubSpins = 1^3; % Defines the number of spin groups in this voxel
    end

    methods
        function self = voxel(position,varargin)
            % CONSTRUCTOR - initializes the voxel at some position
            % voxel(position) - initializes an empty voxel at the coordinates
            % defined by position
            % voxel(position, spinList)- initializes a voxel at the coordinates
            % defined by position and fills it with any spin groups in the
            % optional variable spinList
            p = inputParser();
            p.addOptional('spinList',[])
            p.parse(varargin{:})
```

```

self.position = position;
for i = 1:numel(p.Results.spinList)
    self.spinGroups = {p.Results.spinList(i)};
end
end
function addSpin(self,spin)
    % ADDSPIN - adds spins to the Voxel
    % addSpin(spin) - adds the input spin to the voxel
    self.spinGroups{end+1} = spin;
end
function solution = calculate(self,endTime,PS,B0)

```

```

    % CALCULATE - calculates the time dependent magnetization vector for
    % this voxel

```

```

pulseTimes = PS.eventTimes(1,:); % Times when B changes in the Pulse sequence
pulseTimes = pulseTimes(find(pulseTimes>0,1,'first'):end);
pulseTimes = pulseTimes(pulseTimes<=endTime);
pulseTimes = [0,pulseTimes,endTime];
pulseTimes = sort(pulseTimes);
pulseTimes = unique(round(pulseTimes.*1e9))./1e9;
pulseTimes = unique(pulseTimes);
self.solTimes = pulseTimes;
solution.pulseTimes = pulseTimes;
% initialize the storage for the solutions
self.anlyticSol = cell(length(pulseTimes)-1,numel(self.spinGroups));
self.sol = cell(length(pulseTimes)-1,numel(self.spinGroups));
% generate and stor the solutions for each pulse time
for i = 1:length(pulseTimes)-1
    tSpan = [pulseTimes(i),pulseTimes(i+1)];
    x = self.position(1);
    y = self.position(2);
    z = self.position(3);
    t = mean(tSpan);
    tmpSpinGroups = self.spinGroups;
    if(i>1)
        tmpAnlyticSol2 = tmpAnlyticSol;
        tmpSol2 = tmpSol;
    else
        tmpAnlyticSol2 = self.anlyticSol(i,:);
        tmpSol2 = self.sol(i,:);
    end
    tmpAnlyticSol = self.anlyticSol(i,:);
    tmpSol = self.sol(i,:);
    ODEBool = PS.solver(tSpan);
    %tic
    for j = 1:numel(tmpSpinGroups)
        % callculate current M
        if i == 1,
            tmpM = tmpSpinGroups{j}.M;
        else
            if isempty(tmpSol2{j})
                tmpFun = tmpAnlyticSol2{j};
            else
                tmpFun = tmpSol2{j};
            end
        end
    end
end

```

```

        tmpM = tmpFun(tSpan(1));
    else
        tmpFun = tmpSol2{j};
        tmpM = deval(tmpFun,tSpan(1));
    end
end
if(ODEBool || ~tmpSpinGroups{j}.useAnalytical())
    % used ODE solver when PS is changing
    odefun = @(M,t)tmpSpinGroups{j}.dM(x,y,z,M,t,PS,B0);
    %
    %
    figure
    ode45(odefun,tSpan,tmpM)
    tmpSol{j} = ode45(odefun,tSpan,tmpM);
    solution.functions(i,:) = tmpSol;
    solution.useAnalytical(i,:) = false;
else
    B = B0+PS.B(x,y,z,t);
    tmpAnalyticSol{j} = ...
        @(t)tmpSpinGroups{j}.analytical(...
            x,y,z,tSpan(1),tmpM,t,PS,B0,B);
    tmpSol{j} = {};
    solution.functions(i,:) = tmpAnalyticSol;
    solution.useAnalytical(i,:) = true;
    %
    %
    %
    %
    odefun = @(M,t)tmpSpinGroups{j}.dM(x,y,z,M,t,PS,B0);
    figure
    ode45(odefun,tSpan,tmpM)
    pause(waitforbuttonpress)
    %tmpsol{i,j} = ode45(odefun,tspan,tmpM);
end
end
for j = 1:numel(tmpSpinGroups)
    solution.frameFreq{j} = @(B0)...
        self.spinGroups{j}.calculationFrame(B0);
    solution.spinDensity{j} = self.spinGroups{j}.density;
end
self.analyticSol(i,:) = tmpAnalyticSol;
self.sol(i,:) = tmpSol;
%toc
end

end
function M = getM(self,t,ref,B0)
    % GETM - returns the magnetization vector over some time vector
    % getM(t) - returns the magnetization vector for all time points in
    % the vector t
    % set up sub spins within this voxel
    B0inHomogFact = 0;% normrnd(0,self.T2Star,self.numSubSpins,1)*self.T2Star;
    while std(B0inHomogFact) > 1.3*self.T2Star
        B0inHomogFact = normrnd(0,self.T2Star,self.numSubSpins,1);
    end
    M = zeros(3,length(t));
    % Break up the time vector into chunks that mach the diffrent
    % solutions
    start = find(t(1)<self.solTimes,1,'first');

```

```

devalTimes = 1;
if self.debug
tmp = {};
save('tmp','tmp')
end
for j = start:numel(self.solTimes)
    devalTimes(end+1) = find(t<self.solTimes(j),1,'last');
    tmp = find(t>=self.solTimes(j),1,'first');
    if isempty(tmp)
        break;
    else
        devalTimes(end+1) = tmp;
    end
end
devalTimes(end+1) = length(t);
devalTimes = sort(devalTimes);
for n = 1:2:numel(devalTimes)-1
    tDeval = t(unique(devalTimes(n):devalTimes(n+1)));
    % find wich solution to use for this time
    j = find(self.solTimes>tDeval(1),1,'first');
    if isempty(j), j = length(self.solTimes); end
    j = j-1;
    tmpM = zeros(3,numel(tDeval));
    for i = 1:numel(self.spinGroups)
        % calculate M0 for a spin group at the passed in time
        if isempty(self.sol{j,i})
            % Calculat with analytical
            tmp = self.anlyticSol{j,i}(tDeval(1));
            Mframe = self.anlyticSol{j,i}(tDeval);
            if self.debug
                load('tmp.mat')
                tmp{j,i} = Mframe;
                save('tmp','tmp')
            end
        else
            % calculate woth ode
            Mframe = deval(self.sol{j,i},tDeval);
            if self.debug
                load('tmp.mat')
                tmp{j,i} = Mframe;
                save('tmp','tmp')
            end
        end
        % account for sub spins
        for m = 1:self.numSubSpins
            for p = 1:numel(tDeval)
                % rotate to refrence frame
                theta = (self.spinGroups{i}.calculationFrame(B0)+ref)*...
                    (1+B0inHomogFact(m))*tDeval(p);
                %loop over all the spins in a group
                for k = 1:3:size(Mframe,1)
                    tmpM(:,p) = tmpM(:,p)+self.spinGroups{i}.density*...
                        [cos(theta),-sin(theta),0;sin(theta),cos(theta),0;...

```

```

                                0,0,1]*Mframe(k:k+2,p);
                                end
                                end
                                end
                                M(:,unique(devalTimes(n):devalTimes(n+1))) =...
                                tmpM./self.numSubSpins;
                                end
                                end
                                end
                                end
                                end

```

Section B.2: Pulse Sequence

Pulse Sequence Class

The pulse sequence class takes an assembly of gradient and radiofrequency pulses and compiles them into a magnetic vector that is a function of time. A fair amount of logic is used to optimize the sequence before calculation, as the state of the magnetic field at a particular time is called with great frequency during analytical and numerical calculation, and performance profiling has found that interaction with the pulse sequence is a major source of computational burden.

During calculation, multiple solvers can be used, and since the solver to be used can depend on the pulse sequence, any time the pulse sequence is changed an event is stored. Events are merely flags that are used by the pulse sequence compiler and the voxel class as points where the solvers need to re-evaluate which method to use.

Gradient and radiofrequency pulses are sorted and compiled separately. However, they are eventually combined to yield a single magnet field vector as a function of time and position. Pulses are added to the end of the storage array and can be removed by their index in the array. A more robust structure for the addition and removal of pulses would likely be beneficial. With the current implementation it is best to build the correct sequence the first time.

The ADC represents the analog-to-digital converter that can be used to define evaluation points. However, this is not required.

Compilation of gradients currently has logic to account for slew rates. However, this has not been rigorously test and should be used with great care. RF pulses are not allowed to overlap in time and should be combined into a single waveform if overlapping is required.

The display function has been overridden to show a sequence diagram. The methodology for how repetition time and RF pulses are displayed in the sequence diagram should probably be reconsidered and textual information about the sequence should be included.

```

classdef PulseSequence < handle
    %PULSESEQUENCE Class representing a MRI pulse sequence
    % Detailed explanation goes here
    properties
        ADC = []; % A vector storing start, stop time and bandwidth of the ADC
    end
    properties (SetAccess = private)
        rfPulses = {} % Vector of rf Pulses in the sequence
        gradientPulses = {} % Vector of rf Pulses in the sequence
        time = {}
        eventTimes % vector defining when pulses are on or off
        slewRate = 1e9 % maximal gradient slope in T/(m*sec)
        gradientVect % The gradient amplitudes
        RFVect % Grid Containg the Times for each RF Pulse
    end
    properties (Constant)
        maxSize = 1e4; % Maximum number of points used for displaying pulses
    end
    properties (Dependent)
        timeStep % Not sure whatthis does
    end
    methods
        function compile(self)
            if (isempty(self.gradientPulses))
                self.gradientVect = [0,1e9;0,0;0,0;0,0];
            else
                self.compileGrads();
            end
            if (isempty(self.rfPulses))
                self.RFVect = [0,1e9;0,0];
            else
                self.compileRF();
            end
            self.updateTime()
        end
        function val = get.timeStep(self)
            % timeStep: not quite sure whatthis does
            val = 1;
            for i = 1:length(self.eventTimes)-1
                if (self.eventTimes(i+1)-self.eventTimes(i)<val)
                    val = self.eventTimes(i+1)-self.eventTimes(i);
                end
            end
        end
        function b = B(self,x,y,z,t)
            % B: returns the Bfield at some point defined by (x,y,z) at some time
            % t
            b = zeros(3,length(t));
            RFPulses = interp1(self.RFVect(1,:),self.RFVect(2,:),t,'nearest','extrap');
            for i = 1:length(t)
                if RFPulses(i) ~= 0

```

```

        b(:,i) = [real(self.rfPulses{RFPulses(i)}.B(x,y,z,t(i)));...
                 imag(self.rfPulses{RFPulses(i)}.B(x,y,z,t(i)))*0];
    end
end
% Old implementation
% for i = 1:numel(self.rfPulses)
%     if(t>=self.rfPulses{i}.startTime && t<=self.rfPulses{i}.endTime)
%         b = b + [real(self.rfPulses{i}.B(x,y,z,t));...
%                 imag(self.rfPulses{i}.B(x,y,z,t));zeros(1,length(t))];
%     end
% end
if(size(self.gradientVect,2) >1 )
    b = b + self.BGrad(x,y,z,t);
end
end
function b = BGrad(self,x,y,z,t)
    % B: returns the Bfield of just the gradient fields at some time t
    % and some location x,y,z
    b = zeros(3,length(t));
    b(3,:) = interp1(self.gradientVect(1,:),self.gradientVect(2,:),t,...
        'nearest','extrap')*x + ...
        interp1(self.gradientVect(1,:),self.gradientVect(3,:),t,...
        'nearest','extrap')*y + ...
        interp1(self.gradientVect(1,:),self.gradientVect(4,:),t,...
        'nearest','extrap')*z;
    % b = zeros(3,length(t));
    % for i = 1:numel(self.gradientPulses)
    %     b = b + self.gradientPulses{i}.B(x,y,z,t);
    % end
end
function addPulse(self,Pulse)
    % ADDPULSE: adds a pulse to the Pulse Sequence
    RFPulseList = {'Hypwright.SincPulse','Hypwright.BlockPulse',...
        'Hypwright.BlockPulseSpatial','Hypwright.SincPulseSpatial'};
    GradientPulseList = {'Hypwright.GradientPulse',...
        'Hypwright.LinearGradientPulse'};
    if(find(ismember(class(Pulse),RFPulseList)))
        self.rfPulses[length(self.rfPulses)+1] = Pulse;
    else if(find(ismember(class(Pulse),GradientPulseList)))
        self.gradientPulses[length(self.gradientPulses)+1] = Pulse;
    else error(['Pulse passed in not a recognized pulse type.'...
        'Consider updating the pulse lists in the PS object'])
    end
end
function removePulse(self,n)
    % REMOVEPULSE: removes the nth pulse in the pulse sequence
    if(~(length(self.rfPulses) > n || n > 0 || isscalar(n)))
        disp('Error! Pulse index selected to delete not in the sequence')
        return
    end
    self.rfPulses(n) = [];
end

```



```

function addADC(self,startTime,bandwidth,nPoints)
    % ADDADC: adds a virtual ADC to record the MR data.
    % addADC(self,StartTime,Bandwidth,nPoints) (StartTiem), sampling rate (Bandwidth) and
number of
    % points (nPoints). The lenght of time the ADC is on is defined by
    % nPoints/Bandwidth
    endTime = startTime+nPoints*(1/bandwidth);
    self.ADC(end+1,:) = [startTime,endTime,bandwidth];
end
function removeADC(self,n)
    % REMOVEADC - removes an ADC
    % removeADC(n) - removes the nth (n) ADC from the pulse sequence
    self.ADC(n,:) = [];
end
% TODO this seems like an old method and should be removed
function [value,isterminal,direction] = events(self,t)
    % EVENTS: uses the stored pollynomilas to tell the solver when RF
    % pulses turn on and off
    value = polyval(self.eventTimes,t);
    isterminal = 1;
    direction = 0;
end
% TODO: it is not very intuitive to remove pulses based on some arbitray
% number assigned when they were created. need a beeter method to
% identify pulses
function clearRFPulses(self)
    %CLEARPULSES: Clear all pulses from the pulse sequence
    self.rfPulses = [];
    self.updateTime();
end
% TODO: rework the ADC display feature to have ADC be accounted for in
% the sampling time
function display(self,varargin)
    % DISPLAY - displays all the RF,aand gradient pulses as well as when
    % ADCS are on
    % display() - displays in a new figure with thedefault time range )
    % to 100 seconds
    % display([startTime,endTime]) displays in a new figure from the
    % start time to the end time
    % display([startTime,endTime], figure) same as above but plots in
    % the passed in figure
    self.compile();
    p = inputParser();
    p.addOptional('timeRange',[0,self.eventTimes(end)])
    p.addOptional('figure',[])
    p.parse(varargin{:})
    if isempty(p.Results.figure)
        figure('units','normalized','outerposition',[0.25 0 0.5 1]);
    else
        figure(p.Results.figure);
    end
    for i = 1:numel(self.time)
        if self.time(i)> p.Results.timeRange(1) &&...

```

```

        self.time(i) < p.Results.timeRange(2)
dispTime(i) = self.time(i);
end
if self.time(i) > p.Results.timeRange(2)
    break
end
end
if dispTime(1) > self.time(1)
    dispTime = [self.time(1), dispTime];
end
if dispTime(1) > p.Results.timeRange(1)
    dispTime = [p.Results.timeRange(1), dispTime];
end
if dispTime(end) < p.Results.timeRange(2)
    dispTime = [dispTime, p.Results.timeRange(2)];
end
x=0; y=0; z=0;
RFDisp = zeros(size(dispTime));
GXDisp = zeros(size(dispTime));
GYDisp = zeros(size(dispTime));
GZDisp = zeros(size(dispTime));
ADCDisp = zeros(size(dispTime));
for i = 1:numel(dispTime)
    for j = 1:numel(self.rfPulses)
        RFDisp(i) = RFDisp(i) + self.rfPulses{j}.B(x,y,z,dispTime(i));
    end
end
for i = 1:numel(dispTime)
    for j = 1:numel(self.gradientPulses)
        if(dispTime(i) < self.gradientPulses{j}.endTime && ...
            dispTime(i) > self.gradientPulses{j}.startTime)
            tmpGVect = self.gradientPulses{j}.slope;
            GXDisp(i) = GXDisp(i) + tmpGVect(1);
            GYDisp(i) = GYDisp(i) + tmpGVect(2);
            GZDisp(i) = GZDisp(i) + tmpGVect(3);
        end
    end
end
for i = 1:numel(dispTime)
    for j = 1:length(self.ADC)
        if (dispTime(i) > self.ADC(j,1) && ...
            dispTime(i) < self.ADC(j,2))
            ADCDisp(i) = ADCDisp(i) + 1;
        end
    end
end
subplot(6,1,1), plot(dispTime, real(RFDisp), ...
    'b', dispTime, imag(RFDisp), 'r', dispTime, abs(RFDisp), 'k')
xlabel('Time (seconds)'), ylabel('RF Magnitude'), title('RF Pulses')
legend('Real', 'Imaginary', 'Magnitude')
subplot(6,1,2), plot(dispTime, GXDisp, 'k')
xlabel('Time (seconds)'), ylabel('Gradient slope')
title('X Gradient'),

```

```

subplot(6,1,3),plot(dispTime,GYDisp,'k')
xlabel('Time (seconds)'),ylabel('Gradient Slope')
title('Y Gradient')
subplot(6,1,4),plot(dispTime,GZDisp,'k')
xlabel('Time (seconds)'),ylabel('Gradient Slope')
title('Z Gradient')
subplot(6,1,5), plot(dispTime,ADCDisp)
xlabel('Time (seconds)'),ylabel('Number of ADCs on')
title('ADC')
axis([dispTime(1),dispTime(end),min(ADCDisp)-0.5,max(ADCDisp)+0.5])
subplot(6,1,6), plot(self.eventTimes(:,1),self.eventTimes(:,2))
xlabel('Time (seconds)'),ylabel('Using Analytic Solution')
title('Solver Type')
end
function S = solver(self,eventTimes)
    % SOLVER - returns the time dependency in the pulse sequence
    S = false;
    for i = 1:numel(self.rfPulses)
        if (self.rfPulses{i}.startTime - eventTimes(1)) < 1e-8 &&...
            (self.rfPulses{i}.endTime - eventTimes(2)) > -1e-8
            S = S || self.rfPulses{i}.timeDependence;
        end
    end
end
end
methods (Access = private)
function compileGrads(self)
    %A sub function that Converts the Gradient Pulses into a single
    % 4 by n vector that has the valuse for each gradient direction and
    % the times those valuse change
    %Initialize the gradient vector with the first gradient pulse
    % Calculate the slew time
    slewTime = max(abs(0-self.gradientPulses{1}.slope(:)))/self.slewRate;
    % Fill the time vectors for the pulse
    tmpVect(1,1) = self.gradientPulses{1}.startTime;
    tmpVect(1,2) = self.gradientPulses{1}.startTime+slewTime;
    tmpVect(1,3) = self.gradientPulses{1}.endTime-slewTime;
    tmpVect(1,4) = self.gradientPulses{1}.endTime;
    % Fill the gradient slopes for the pulse
    tmpVect(2:4,1) = 0;
    tmpVect(2:4,2) = self.gradientPulses{1}.slope(:);
    tmpVect(2:4,3) = 0;
    tmpVect(2:4,4) = -self.gradientPulses{1}.slope(:);
    % Add the rest of the gradient to the pulse
    for i=2:numel(self.gradientPulses)
        % get the gradient slope valuse at the begining and end of the
        % pulse
        startSlope = [interp1(tmpVect(1,:),tmpVect(2,:),...
            self.gradientPulses{1}.startTime,'nearest','extrap');...
            interp1(tmpVect(1,:),tmpVect(3,:),...
            self.gradientPulses{1}.startTime,'nearest','extrap');...
            interp1(tmpVect(1,:),tmpVect(4,:),...
            self.gradientPulses{1}.startTime,'nearest','extrap')];
    end
end

```

```

endslope = [interp1(tmpVect(1,:),tmpVect(2,:),...
    self.gradientPulses{1}.endTime,'nearest','extrap');...
    interp1(tmpVect(1,:),tmpVect(3,:),...
    self.gradientPulses{1}.endTime,'nearest','extrap');...
    interp1(tmpVect(1,:),tmpVect(4,:),...
    self.gradientPulses{1}.endTime,'nearest','extrap')];
% Calculate the slew times
slewTimeStart = max(abs(startSlope...
    -self.gradientPulses{1}.slope(:)))/self.slewRate;
slewTimeEnd = max(abs(endSlope...
    -self.gradientPulses{1}.slope(:)))/self.slewRate;
j = (i-1)*4+1; % counter for tmpVect
% Fill the time vectors for the pulse
tmpVect(1,j) = self.gradientPulses{i}.startTime;
tmpVect(1,j+1) = self.gradientPulses{i}.startTime+slewTimeStart;
tmpVect(1,j+2) = self.gradientPulses{i}.endTime-slewTimeEnd;
tmpVect(1,j+3) = self.gradientPulses{i}.endTime;
% Fill the gradient slopes for the pulse
tmpVect(2:4,j) = 0;
tmpVect(2:4,j+1) = self.gradientPulses{i}.slope(:);
tmpVect(2:4,j+2) = 0;
tmpVect(2:4,j+3) = -self.gradientPulses{i}.slope(:);
[~,I]=sort(tmpVect(1,:)); % Sort Times
tmpVect = tmpVect(:,I); % match slopes to Times
% Combine duplicates
tmpI = 1;
sumI = 1;
trashI = [];
for k = 2:size(tmpVect,2)
    % grab all the slop changes at a particular time
    if (tmpVect(1,k) == tmpVect(1,tmpI))
        sumI = [sumI,k]; % cant think of a way to pre-allocate this
        % sum all identical slope changes then move to next time point
    else
        tmpVect(2:4,sumI(1)) = sum(tmpVect(2:4,sumI),2);
        trashI = [trashI,sumI(2:end)];
        tmpI = k;
        sumI = k;
    end
end
tmpVect(:,trashI) = [];
end
tmpBSlope = zeros(3,1);
self.gradientVect = zeros(size(tmpVect));
for i=1:size(tmpVect,2)
    self.gradientVect(1,i) = tmpVect(1,i);
    tmpBSlope = tmpBSlope+tmpVect(2:4,i);
    self.gradientVect(2:4,i) = tmpBSlope;
end
% Debug Plotting
figure('Position',[700,200,1100,800])
subplot(3,1,1),plot(self.gradientVect(1,:),self.gradientVect(2,:));
title('Slope X'),xlabel('Time (seconds)'),ylabel('Gradient Slope (T/m)')

```

```

%         subplot(3,1,2),plot(self.gradientVect(1,:),self.gradientVect(3,:));
%         title('Slope Y'),xlabel('Time (seconds)'),ylabel('Gradient Slope (T/m)')
%         subplot(3,1,3),plot(self.gradientVect(1,:),self.gradientVect(4,:));
%         title('Slope Z'),xlabel('Time (seconds)'),ylabel('Gradient Slope (T/m)')

end
function compileRF(self)
    self.RFVect = [-1,1e3;0,0];
    for i = 1:numel(self.rfPulses)
        % Store which pulse is on
        % note a 2 pico second buffer is added to each pulse. this will
        % result in an error if a pulse starts at the exact time one
        % ends
        self.RFVect = [self.RFVect,[self.rfPulses{i}.startTime-1e-12;0]];
        self.RFVect = [self.RFVect,[self.rfPulses{i}.endTime+1e-12;0]];
        % Store 0 for when pulse is off
        self.RFVect = [self.RFVect,[self.rfPulses{i}.startTime;i]];
        self.RFVect = [self.RFVect,[self.rfPulses{i}.endTime;i]];
    end
    [~,I] = sort(self.RFVect(1,:));
    self.RFVect = self.RFVect(:,I);
    for i = 1:2:size(self.RFVect,2)
        if self.RFVect(2,i) ~= self.RFVect(2,i+1)
            error('RF Pulses %d and %d are overlapping with times %d, %d and %d,
%d\n',...

                self.RFVect(i,1),self.RFVect(i+1,1),...
                self.rfPulses{i}.startTime,self.rfPulses{i}.endTime,...
                self.rfPulses{i+1}.startTime,self.rfPulses{i+1}.endTime)
        end
    end
end
function updateTime(self)
    % UPDATETIME: iterates through the pulse sequence and stores the
    % times that pulses are turned on or off for use by the solver
    allPulses = [self.rfPulses,self.gradientPulses];
    pointsPerPulse = self.maxSize/length(allPulses);
    self.time = 0;
    self.eventTimes = [];
    for i = 1:length(allPulses)
        self.time = [self.time,linspace(allPulses{i}.startTime,...
            allPulses{i}.endTime,pointsPerPulse)];
        self.eventTimes(end+1) = allPulses{i}.startTime;
        self.eventTimes(end+1) = allPulses{i}.endTime;
    end
    % stores the on and off times for each pulse as a root to a
    % polynomial
    self.time = sort(self.time);
    self.eventTimes = unique(self.eventTimes);
    self.eventTimes = sort(self.eventTimes);
    %
    self.eventTimes = [];
    %
    for i = 1:size(self.RFVect,2)
    %
        if(self.RFVect(2,i))
    %
            self.eventTimes = [self.eventTimes,[self.RFVect(1,i);1]];
    %
        else

```

```

%             self.eventTimes = [self.eventTimes,[self.RFVect(1,i);0]];
%         end
%     end
%     for i = 1:size(self.gradientVect,2)-1
%         if(sum(self.gradientVect(2:4,i))~=sum(self.gradientVect(2:4,i+1)))
%             self.eventTimes = [self.eventTimes,[self.gradientVect(1,i+1);1]];
%         else
%             self.eventTimes = [self.eventTimes,[self.gradientVect(1,i+1);0]];
%         end
%     end
%     [~,I] = sort(self.eventTimes(1,:));
%     self.eventTimes = self.eventTimes(:,I);
%     I = find(diff(self.eventTimes(1,:))<=0);
%     self.eventTimes(2,I+1) = max([self.eventTimes(2,I);self.eventTimes(2,I+1)]);
%     self.eventTimes(:,I) = [];
%
%     end
%
% end
end

```

RF Pulse Class

The RF class represents an arbitrary radiofrequency pulse and should be mostly used as a parent object, yet full abstraction seemed too extreme. All RF pulses must have a duration and a center time. Memory is allocated for a carrier frequency to mix the RF pulse to near the Larmor frequency. However, this is not required and can be set to zero. For this high level class, the magnetic field is stored as a function of location and time.

The display function has been overridden to show the pulse shape and the mixed waveform. The textual information about each pulse should probably be in the children classes.

```
classdef RFPulse < handle
    %RFPULSE The base Class or Radio frequency Pulses
    % Properties
    % Bfun: function pointer defining the pulse
    % center: the center of the pulse
    % durratation: the length of the pulse (truncates Bfun otherwise)
    % omega: carrier frequency of the pulse
    % Name: a (hopefully) unique name for the pulse
    % startTime: start time of the pulse (truncate Bfun before this point)
    % endTime: end time of the pulse (truncates Bfun after this point)
    % Methods
    % RFPulse(center, durratation, omega, name) initializes center time (center),
    % pulse durratation (durratation), carrier frequency (omega), and name
    % display(self) displays in a new figure
    % display(self,h) displays in a passed in figure h
    % B1(x,y,z,t) returns a 3D vector defining B1 for this pulse at
    % the passed in position (x,y,z) and time (t)
    properties (SetAccess = protected)
        Bfun % function pointer defining the pulse
        center % the center of the pulse
        durratation % the length of the pulse (truncates Bfun otherwise)
        omega % carrier frequency of the pulse
        name % a (hopefully) unique name for the pulse
    end
    properties (Dependent)
        startTime % start time of the pulse
        endTime % end time of the pulse
    end
    properties (Constant)
        timeDependence = true;
    end
    methods (Abstract = true, Access = protected)
        calB(self) % returns a pointer to the function defining the pulse
    end
end
```

methods

```

function setCenter(self,center),self.center = center; end
function setDuration(self,duration), self.duration = duration; end
function setOmega(self,omega), self.omega = omega; end
function val = get.startTime(self),val=self.center-self.duration/2;end
function val = get.endTime(self),val=self.center+self.duration/2;end
function self = RFPulse(center,duration,omega,name)
    % CONSTRUCTOR - Base Constructo for all RFPulse subclasses
    % RFPulse(center, duration,omega, name) initializes center time,
    % duration, carrier frequency omega, and name
    self.center = center;
    self.duration = duration;
    self.omega = omega;
    self.name = name;
end
function display(self,varargin)
    % DISPLAY - Displays the RF pulse envelope, in both frequency and
    % time domaines
    % display(self) displays in a new figure
    % display(self,h) displays in a passed in figure h
    p = inputParser();
    p.addOptional('figure',[])
    p.parse(varargin{:})
    N = 2^10;
    t = linspace(-self.duration/2,self.duration/2,N);
    B1 = zeros(length(t),1);
    x = 0; y = 0; z = 0;
    for i = 1:length(t)
        B1(i) = self.Bfun(x,y,z,t(i));
    end
    FT = fftshift(fft(fftshift(B1)));
    freqAxis = linspace(1/(t(2)-t(1))/length(t),1/(t(2)-t(1)),...
        length(t)); % calculate frequency axis
    if isempty(p.Results.figure)
        figure;
    else
        figure(p.Results.figure);
    end
    subplot(2,1,1),plot(t,real(B1),'k',t,imag(B1),'r',t,abs(B1),'b')
    xlabel('Time (seconds)')
    ylabel('B1 (Tesla)')
    legend('x','y','Magnitude')
    subplot(2,1,2),plot(freqAxis,real(FT),'k',freqAxis,imag(FT),'r',...
        freqAxis,abs(FT),'b')
    xlabel('Frequency (Hz)')
    ylabel('Magnitude (arb)')
    legend('real','Imaginary','Magnitude')
end
function B1 = B(self,x,y,z,t)
    % B1: gives the B1 of this pulse at a time and location
    % B1(x,y,z,t) returns a 3D vector defining B1 for this pulse at
    % the passed in postion (x,y,z) and time (t)
    B1 = zeros(1,length(t));

```



```
        pulseOnTimes = find(self.startTime < t & t < self.endTime);  
        if ~isempty(pulseOnTimes)  
            B1(pulseOnTimes) = self.Bfun(x,y,z,t(pulseOnTimes)-self.center)...  
                .*exp(1i*self.omega*(t(pulseOnTimes)));  
        end  
    end  
end
```

Block Pulse Class

A Block Pulse is a block-shaped waveform that is then mixed up to a particular carrier frequency.

The amplitude should be set to $\frac{\theta}{\gamma \tau}$. The duration of the block pulse will determine the width of the sinc

excitation profile and the carrier frequency omega will determine the center frequency of the sinc

profile. A child class that incorporates spatial variability has been written, but will not be documented in this text.

```
classdef BlockPulse < Hypwright.RFPulse
    %SINCPULS Sinc Enveloped RF Pulse
    % Properties
    % bandwidth: Bandwidth of the sinc pulse
    % amplitude: Amplitude of the Sinc
    % lobes: number of lobes in the sinc envelope default(5)
    % Methods
    % SincPulse(center,bandwidth,amplitude,omega,varargin) - sets the
    % center time (center), the pulse bandwidth (bandwidth), the pulse
    % amplitude (amplitude) and the carrier frequency (omega) will set
    % the name to a random number and the number of lobes to 5
    % SinPulse(...,lobes) - same as above but accepts a positive integer
    % as the number of lobes in this pulse
    % SinPulse(...,name) - same as above but the last argument will be
    % set as the pulses' name, to use the default value (5) for the
    % number of lobes just pass in [] as the 5th argument
    % setDuration(self,duration) - does nothing and warns the user note
    % that for a sinc pulse the duration is a function of the bandwidth
    properties
    end
    properties(SetAccess = private)
        amplitude % Amplitude of the Pulse
    end
    methods
        function self = BlockPulse(center,duration,omega,amplitude,varargin)
            % CONSTRUCTOR - Initializes the Block pulse
            % BlockPulse(center,bandwidth,amplitude,omega,varargin) - sets the
            % center time (center), the pulse duration (duration), the pulse
            % amplitude (amplitude) and the carrier frequency (omega)
            % BlockPulse(...,name) - same as above but the last argument will be
            % set as the pulses' name
            p = inputParser();
            p.addOptional('name',sprintf('Pulse%d',int16(rand(1)*10000)),@isstr)
            p.parse(varargin{:})
            self = self@Hypwright.RFPulse(center,duration,omega,p.Results.name);
            self.amplitude = amplitude;
            self.calB();
        end
    end
end
```

```

function setDuration(self,value)
    % SETDuration - sets the amplitude of the pulse
    self.duration = value;
    self.calB();
end
function setAmplitude(self,value)
    % SETAMPLITUDE - sets the amplitude of the pulse
    self.amplitude = value;
    self.calB();
end
end
methods (Access = protected)
    function calB(self)
        % CALB - re-calculates the function that defines the envelope for
        % this sinc pulse
        self.Bfun = @(x,y,z,t)self.amplitude;
    end
end
end

```

Sinc Pulse Class

The sinc pulse class builds a n-lobed sinc pulse with a set excitation bandwidth, and an amplitude set to $\frac{\theta}{\gamma}$. The carrier frequency omega will determine the center of the excitation band and the center time defines the center of the pulse. The pulse duration and therefore, the start and end times will depend on the pulse bandwidth and on the number of lobes and are properties that can be returned, but not set.

```
classdef SincPulse < Hypwright.RFPulse
    %SINCPULS Sinc Enveloped RF Pulse
    % Properties
    % bandwidth: Bandwidth of the sinc pulse
    % amplitude: Amplitude of the Sinc
    % lobes: number of lobes in the sinc envelope default(5)
    % Methods
    % SincPulse(center,bandwidth,amplitude,omega,varargin) - sets the
    % center time (center), the pulse bandwidth (bandwidth), the pulse
    % amplitude (amplitude) and the carrier frequency (omega) will set
    % the name to a random number and the number of lobes to 5
    % SinPulse(...,lobes) - same as above but accepts a positive integer
    % as the number of lobes in this pulse
    % SinPulse(...,name) - same as above but the last argument will be
    % set as the pulses' name, to use the default value (5) for the
    % number of lobes just pass in [] as the 5th argument
    % setDuration(self,duration) - does nothing and warns the user note
    % that for a sinc pulse the duration is a function of the bandwidth
    properties
    end
    properties(SetAccess = private)
        bandwidth % Bandwidth of the sinc pulse
        amplitude % Amplitude of the Sinc
        lobes % number of lobes in the sinc envelope
    end
    methods
        function self = SincPulse(center,bandwidth,amplitude,omega,varargin)
            % CONSTRUCTOR - Initializes the Sinc pulse
            % SincPulse(center,bandwidth,amplitude,omega,varargin) - sets the
            % center time (center), the pulse bandwidth (bandwidth), the pulse
            % amplitude (amplitude) and the carrier frequency (omega) will set
            % the name to a random number and the number of lobes to 5
            % SinPulse(...,lobes) - same as above but accepts a positive integer
            % as the number of lobes in this pulse
            % SinPulse(...,name) - same as above but the last argument will be
            % set as the pulses' name, to use the default value (5) for the
            % number of lobes just pass in [] as the 5th argument
            function val = lobeTest(x)
```

```

        if isempty(x)
            val = 1;
        else
            val = (mod(x,1) == 0 && x > 0);
        end
    end
    p = inputParser();
    p.addOptional('lobes',5,@lobeTest)
    p.addOptional('name',sprintf('Pulse%d',int16(rand(1)*10000)),@isstr)
    p.parse(varargin{:})
    self = self@Hypwright.RFPulse(center,0,omega,p.Results.name);
    self.amplitude = amplitude;
    self.bandwidth = bandwidth;
    if isempty(p.Results.lobes)
        self.lobes = 5;
    else
        self.lobes = p.Results.lobes;
    end
    self.duration = ((self.lobes))*2/self.bandwidth;
    self.calB();
end
function setDuration(self,duration)
    % SETDURATION - overloaded for a nLobed sincPulse as it should not
    % be changeable. Duration is a function of bandwidth
    % setDuration(self,duration) - does nothing and warns the user
    disp(['the durration of this pulse is a function of bandwidth and'...'
        ' should be altered by changing the bandwidth']);
end
function setAmplitude(self,value)
    % SETAMPLITUDE - sets the amplitude of the pulse
    self.amplitude = value;
    self.calB();
end
function setBW(self,newBW)
    % SETBW - Sets the bandwidth of the pulse
    % SetBW(self, newBW) - sets the bandwidth to some new bandwidth
    % (newBW)
    self.functionBW = newBW;
    self.duration = ((self.lobes)-1)*2/self.bandwidth;
    self.calB();
end
end
methods (Access = protected)
function calB(self)
    % CALB - re-calculates the function that defines the envelope for
    % this sinc pulse
    self.Bfun = @(x,y,z,t)self.amplitude.*self.bandwidth.*...
        sinc(self.bandwidth.*t).*...
        interp1(-self.duration/2:self.duration/100:self.duration/2,...
            blackman(...
                length(-self.duration/2:self.duration/100:self.duration/2))...
            ,t);
end

```

```
end  
end
```

Gradient Pulse Class

A gradient pulse class is very similar to the RF pulse class, as it is mostly intended to be used as a parent class to define an interface, but not so strictly as to be abstracted. A gradient pulse stores a duration and center time from which the start and end times are calculated. Otherwise, it stores an arbitrary function to define the magnetic field vector as a function of position and time. The main distinction is that gradient pulses do not have the logic to use a carrier frequency like RF pulses.

Gradient pulses have overridden the display function to show the gradient as a vector field.

```
classdef GradientPulse < handle
    %GRADIENTPULSE Class to represent a gradient field
    % Detailed explanation goes here
    properties
        center
        durruration
        name
    end
    properties (Abstract, Access = protected)
        bFun
    end
    properties (Dependent)
        startTime
        endTime
    end
    methods
        function self = GradientPulse(center, durruration, name)
            % CONSTRUCTOR - initializes a gradient pulse objects
            % GradientPulse(startTime, endTime, slope, magnitude) initializes a
            % gradient pulse with a start time, end time, magnitude, and slope,
            % will give the pulse a randome name
            % GradientPulse(...,name) - same as above but will give the puse a
            % specified name
            self.center = center;
            self.durruration = durruration;
            self.name = name;
        end
        function val = get.startTime(self),val=self.center-self.durruration/2;end
        function val = get.endTime(self),val=self.center+self.durruration/2;end
        function bout = B(self,x,y,z,t)
            % B: returns the gradient Bfied at some point and time
            % bout = B(x,y,z,t) returns the B-filed(bout) ate somepoint (x,y,z)
            % some time t
            if length(t) == 1
                if ((t > self.startTime)&&(t < self.endTime))
                    bout = self.bFun(x,y,z,t);
                end
            end
        end
    end
end
```

```

        else
            bout = zeros(3,1);
        end
    else
        bout = zeros(3,length(t));
        pulseOnTimes = find(self.startTime -t < 1e-9 & t < self.endTime);
        if ~isempty(pulseOnTimes)
            bout(:,pulseOnTimes) = self.bFun(x,y,z,t(pulseOnTimes));
        end
    end
end

function display(self,varargin)
    p = inputParser();
    p.addOptional('axis',[])
    p.parse(varargin{:})
    if isempty(p.Results.axis)
        figure
        curAxis = gca;
    else
        curAxis = p.Results.axis;
    end
    span = -1:0.3:1;
    [X,Y,Z] = meshgrid(span,span,span);
    U = zeros(size(X));
    V = zeros(size(Y));
    W = zeros(size(Z));
    centerTime = (self.startTime+self.endTime)/2;
    for i = 1:numel(span)
        for j = 1:numel(span)
            for k = 1:numel(span)
                tmpVect = self.B(i,j,k,centerTime);
                U(i,j,k) = tmpVect(1);
                V(i,j,k) = tmpVect(2);
                W(i,j,k) = tmpVect(3);
            end
        end
    end
    quiver3(curAxis,X,Y,Z,U,V,W)
end

end

methods (Access = private)
    function calB(self)
        % CALB: recalculates the function defining the gradient pulse
        self.bFun = @(x,y,z,t)[0,0,0;0,0,0;self.slope]*[x;y;z];
    end
end

end

```


Linear Gradient Pulse

The linear gradient pulse class represents a gradient that has a linear spatial dependence. The slope of the gradient is stored in a vector $slope = [slope_x, slope_y, slope_z]$.

```
classdef LinearGradientPulse < Hypwright.GradientPulse
    %GRADIENTPULSE Class to represent a gradient field
    % Detailed explanation goes here
    properties (SetAccess = private)
        slope
    end
    properties (Access = protected)
        bFun
    end
    properties (Constant)
        timeDependence = false;
    end
    methods
        function self = LinearGradientPulse(center, duration, slope,varargin)
            % CONSTRUCTOR - initializes a gradient pulse objects
            % GradientPulse(startTime, endTime, slope, magnitude) initializes a
            % gradient pulse with a start time, end time, magnitude, and slope,
            % will give the pulse a random name
            % GradientPulse(...,name) - same as above but will give the pulse a
            % specified name
            p = inputParser();
            p.addOptional('name',sprintf('GradPulse%d',int16(rand(1)*10000)),...
                @isstr)
            p.parse(varargin{:})
            self = self@Hypwright.GradientPulse(center,duration,p.Results.name);
            self.slope = slope;
            self.calB();
        end
        function setslope(self,slope)
            self.slope = slope;
            self.calB();
        end
    end
    methods (Access = private)
        function calB(self)
            % CALB: recalculates the function defining the gradient pulse
            self.bFun = @(x,y,z,t)[0,0,0;0,0,0;self.slope]*[x;y;z];
        end
    end
end
```

Section B.3: Spin Groups

SpinGroups represent a set of spins that follows one of the models outlined in chapter 3. They store all of the parameters and logic to define their interactions with a pulse sequence following the interface needed to be calculated and evaluated by a voxel object.

Spin Group Class

The SpinGroup class is an abstract class that defines the interface for spin groups. It is abstract and therefore can never be instantiated. All spin groups should inherit from this class. Its current form requires that a spin group store some function that returns the derivative of the magnetization as a function of position, time, initial magnetization and B_0 . Also the frequency of the calculation is required to ensure proper frame shifting to the rotating frame is performed. Additional functions for the analytical solution should probably be added as they are expected in the voxel class. Also the dM function interface needs to be updated. Fortunately, due to Matlab's inheritance rules, these changes are merely housekeeping measures and will not affect the performance of the other classes.

```
classdef (Abstract) SpinGroup < handle
    %SPINGROUP The parent class for all spin groups
    % defines the interface of a spin group

    properties (Abstract)
    end

    methods (Abstract)
        dm = dM(self,position,M,time,PS,B0)
        % dM(self,position,M,time) - calculates the dm of the spin at some
        % position and time in the calculation frame defined by the spin.
        val = calculationFrame(self,B0)
        % calculationFrame the vector defining the angular momentum of the
        % calculation frame dm is defined in. remember to create a get method
        % for this variable in inherited classes
    end
end
```

Isolated Spin Group Class

The isolated spin group class represents a single magnetization vector with an initial position M , equilibrium position M_0 , T_1 and T_2 values, gyromagnetic ratio γ , chemical shift ppm and a density. The density is simply a scaling factor that will be applied to the magnetization. It allows the signal contribution of each spin group to be controlled. The analytical solution builds the A matrix used in equation (3.7) but for a single spin. The solution is then defined by equation (3.9). Note that equations (3.7) and (3.9) requires M_0 to be zero and as of yet no analytical solution for a spin with a nonzero magnetization has been written into this simulation code.

```
classdef IsolatedSpinGrp < Hypwright.SpinGroup
    %ISOLATEDSPINGRP a class that represents a set of isolated spins
    % Detailed explanation goes here
    % Properties
    % M - magnetization vector
    % M0 - Equilibrium magnetization
    % T1 - T1 decay constant
    % T2 - T2 Decay constant
    % gamma - gyromagnetic ratio
    % ppm - ppm shift of the spin
    % density - relative number of spins in this group
    % Methods
    % IsolatedSpinGrp(M,M0,T1,T2,gamma,density) - initializes the spin
    % group with an initial magnetization (M), equilibrium magnetization
    % (M0), T1 decay (T2), T2 Decay (T2), gyromagnetic ratio (gamma)
    % some chemical shift (ppm) and, spin density (density)
    % calculationFrame() - returns the frequency of the rotating
    % reference frame that dm is calculated in
    % dm(x,y,z,t,M) - returns a dm at some position (x,y,z), some time
    % (t), and some initial magnetization M ([Mx;My;Mz])

    properties
        M % magnetization vector
        M0 % Equilibrium magnetization
        T1 % T1 decay constant
        T2 % T2 Decay constant
        gamma % gyromagnetic ratio
        ppm % ppm shift of the spin
        density % relative number of spins in this group
    end
    methods
        function self = IsolatedSpinGrp(M,M0,T1,T2,gamma,ppm,density)
            % Constructor - initializes the spin group
            % IsolatedSpinGrp(M,M0,T1,T2,gamma,ppm,density) - initializes the spin
```

```

    % group with an initial magnetization (M), equilibrium magnetization
    % (M0), T1 decay (T1), T2 Decay (T2), gyromagnetic ratio (gamma),
    % some chemical shift (ppm) and, spin density (density)
    self.M = M;
    self.M0 = M0;
    self.T1 = T1;
    self.T2 = T2;
    self.gamma = gamma;
    self.ppm = ppm;
    self.density = density;
end
function val = calculationFrame(self,B0)
    % CALCULATIONFRAME - returns the frequency of the rotating
    % reference frame that dm is calculated in
    tmp = [0;0;1].*B0*self.gamma*(1+self.ppm);
    val = tmp(3);
end
function dm = dm(self,x,y,z,t,M,PS,B0)
    % DM: returns the delta m at some time and location and given M
    % dm(x,y,z,t,M) - returns a dm at some position (x,y,z), some time
    % (t), and some initial magnetization M ([Mx;My;Mz])
    dm = self.getA(x,y,z,t,PS,B0)*M+1/self.T1*self.M0;
end
function A = getA(self,x,y,z,t,PS,B0,varargin)
    % GETA - gets the matrix that defines dm
    if ~isempty(varargin) == 1
        B = varargin{1};
    else
        B = repmat(B0,1,length(t))+PS.B(x,y,z,t);
    end
    theta = -self.calculationFrame(B0)*t;
    Beff = [cos(theta),-sin(theta),0;sin(theta),cos(theta),0;0,0,1]*B-...
        [0;0;1].*B0;
    A = self.gamma*...
        [0,-Beff(3),Beff(2);Beff(3),0,-Beff(1);-Beff(2),Beff(1),0] + ...
        [-1/self.T2,0,0;0,-1/self.T2,0;0,0,-1/self.T1];
end

function ret = useAnalytical(self)
    %USEANALYTICAL: determines if the Analytical Solution should be used
    %for the given spin group under the given conditions
    ret = all(self.M0 == 0);
end
function vals = analytical(self,x,y,z,t0,M,t,PS,B0,varargin)
    % ANALYTICAL: return a function handle to the analytical solution
    A = self.getA(x,y,z,t0+1e-9,PS,B0,varargin{:});
    vals = cell2mat(arrayfun(@(t2)expm(A*(t2-t0))*M,...
        t,'UniformOutput',false));
end
end
end
end

```

Two-Site Exchange Group Class

A two-site exchange group class is similar to an isolated spin group. However, it contains terms for chemical exchange between two chemical pools, k_{ab} and k_{ba} . The magnetization vectors will be 6×1 with each three rows representing a spin group. Density and gamma are identical for each spin. The logic for defining default values can probably be compressed.

```
classdef TwoSiteExchangeGroup < Hypwright.SpinGroup
    % TwoSiteExchangeGroup a class that represents a set of isolated spins
    % Detailed explanation goes here
    % Properties
    % M - magnetization vector
    % M0 - Equilibrium magnetization
    % T1 - T1 decay constant
    % T2 - T2 Decay constant
    % gamma - gyromagnetic ratio
    % density - relative number of spins in this group
    % Methods
    % IsolatedSpinGrp(M,M0,T1,T2,gamma,density) - initializes the spin
    % group with an initial magnetization (M), equilibrium magnetization
    % (M0), T1 decay (T2), T2 Decay (T2), gyromagnetic ratio (gamma) and
    % spin density (density)
    % calculationFrame() - returns the frequency of the rotating
    % reference frame that dm is calculated in
    % dm(x,y,z,t,M) - returns a dm at some position (x,y,z), some time
    % (t), and some initial magnetization M ([Mx;My;Mz])

    properties
        M % magnetization vector
        M0 % Equilibrium magnetization
        T1a % T1 decay constant for spin a
        T2a % T2 Decay constant for spin a
        ppma % chemical shift for spin a
        T1b % T1 decay constant for spin b
        T2b % T2 Decay constant for spin b
        ppmb % chemical shift for spin b
        gamma % gyromagnetic ratio
        density % relative number of spins in this group
        kab % a to b exchange rate
        kba % b to a exchange rate
    end

    methods
        function self = TwoSiteExchangeGroup(varargin)
            % Constructor - initializes the spin group
            % IsolatedSpinGrp(M,M0,T1,T2,gamma,density) - initializes the spin
            % group with an initial magnetization (M), equilibrium magnetization
            % (M0), T1 decay (T2), T2 Decay (T2), gyromagnetic ratio (gamma) and
            % spin density (density)
```

```

p = inputParser();
p.addOptional('M',[0;0;1;0;0;1],@isnumeric)
p.addOptional('M0',[0;0;0;0;0;0],@isnumeric)
p.addOptional('T1a',56,@isnumeric)
p.addOptional('T2a',0.02,@isnumeric)
p.addOptional('ppma',171*1e-6,@isnumeric)
p.addOptional('T1b',30,@isnumeric)
p.addOptional('T2b',0.02,@isnumeric)
p.addOptional('ppmb',185*1e-6,@isnumeric)
p.addOptional('gamma',67.262e6,@isnumeric)
p.addOptional('density',1,@isnumeric)
p.addOptional('kab',0.3,@isnumeric)
p.addOptional('kba',0.0,@isnumeric)
p.parse(varargin{:})
if ~isempty(p.Results.M), self.M = p.Results.M;
else self.M = [0;0;1;0;0;1];end
if ~isempty(p.Results.M0),self.M0 = p.Results.M0;
else self.M0 = [0;0;0;0;0;0];end
if ~isempty(p.Results.T1a),self.T1a = p.Results.T1a;
else self.T1a = 56;end
if ~isempty(p.Results.T2a),self.T2a = p.Results.T2a;
else self.T2a = 0.02;end
if ~isempty(p.Results.ppma),self.ppma = p.Results.ppma;
else self.ppma = 171*1e-6;end
if ~isempty(p.Results.T1b),self.T1b = p.Results.T1b;
else self.T1b = 30;end
if ~isempty(p.Results.T2b),self.T2b = p.Results.T2b;
else self.T2b = 0.02;end
if ~isempty(p.Results.ppmb),self.ppmb = p.Results.ppmb;
else self.ppmb = 185*1e-6;end
if ~isempty(p.Results.gamma),self.gamma = p.Results.gamma;
else self.gamma = 67.262e6;end
if ~isempty(p.Results.density),self.density = p.Results.density;
else self.density = 1;end
if ~isempty(p.Results.kab),self.kab = p.Results.kab;
else self.kab = 0.3;end
if ~isempty(p.Results.kba),self.kba = p.Results.kba;
else self.kba = 0.0;end
end
function val = calculationFrame(self,B0)
% CALCULATIONFRAME - returns the frequency of the rotating
% reference frame that dm is calculated in
tmp = [0;0;1].*B0*self.gamma*...
(1+mean([self.ppma,self.ppmb]));
val = tmp(3);
end
function dm = dm(self,x,y,z,t,M,PS,B0)
% DM: returns the delta m at some time and location and given M
% dm(x,y,z,t,M) - returns a dm at some position (x,y,z), some time
% (t), and some initial magnetization M ([Mx;My;Mz])
Recovery(1:3) = 1/self.T1a*self.M0(1:3);
Recovery(4:6) = 1/self.T1b*self.M0(4:6);
dm = self.getA(x,y,z,t,PS,B0)*M+Recovery.';

```

```

end
function A = getA(self,x,y,z,t,PS,B0,varargin)
    if ~isempty(varargin) == 1
        B = varargin{1};
    else
        B = repmat(B0,1,length(t))+PS.B(x,y,z,t);
    end
    theta = -self.calculationFrame(B0)*t;
    Beff(1:3) = [cos(theta),-sin(theta),0;sin(theta),cos(theta),0;...
        0,0,1]*B-[0;0;1].*B0*...
        (1+mean([self.ppma,self.ppm]))-self.ppma);
    Beff(4:6) = [cos(theta),-sin(theta),0;sin(theta),cos(theta),0;...
        0,0,1]*B-[0;0;1].*B0*...
        (1+mean([self.ppma,self.ppm]))-self.ppm);
    A = zeros(6);
    A(1:3,1:3) = self.gamma*...
        [0,-Beff(3),Beff(2);Beff(3),0,-Beff(1);-Beff(2),Beff(1),0]+...
        [-1/self.T2a,0,0;0,-1/self.T2a,0;0,0,-1/self.T1a];
    A(4:6,4:6) = self.gamma*...
        [0,-Beff(6),Beff(5);Beff(6),0,-Beff(4);-Beff(5),Beff(4),0]+...
        [-1/self.T2b,0,0;0,-1/self.T2b,0;0,0,-1/self.T1b];
    A = A+[-self.kab,0,0,self.kba,0,0;...
        0,-self.kab,0,0,self.kba,0;...
        0,0,-self.kab,0,0,self.kba;...
        self.kab,0,0,-self.kba,0,0;...
        0,self.kab,0,0,-self.kba,0;...
        0,0,self.kab,0,0,-self.kba];
end
function ret = useAnalytical(self)
    %USEANALYTICAL: determines if the Analytical Solution should be used
    %for the given spin group under the given conditions
    ret = all(self.M0 == 0);
end
function vals = analytical(self,x,y,z,t0,M,t,PS,B0,varargin)
    % ANALYTICAL: the analytical solution for this spin over some
    % time range (tSpan) and some initial condition (M)
    A = self.getA(x,y,z,t0+1e-9,PS,B0,varargin{:});
    vals = cell2mat(arrayfun(@(t2)expm(A*(t2-t0))*M,t,...
        'UniformOutput',false));
end
end
end

```

Two-Site Perfusion Exchange Group Class

A Child Class of Two site exchange group but with logic to account for perfusion. This class will represent the extravascular space and should be paired with a Bankson spin group class to represent the vascular space. The only additional parameters are the kve (which should be normalized for ve) and the VIF with a variable b, which should return the VIF magnetization as a function of time.

```
classdef TwoSitePerfusionExchangeGroup < Hypwright.TwoSiteExchangeGroup
    % TwoSiteExchangeGroup a class that represents a set of isolated spins
    % Detailed explanation goes here
    % Properties
    % M - magnetization vector
    % M0 - Equilibrium magnetization
    % T1 - T1 decay constant
    % T2 - T2 Decay constant
    % gamma - gyromagnetic ratio
    % density - relative number of spins in this group
    % Methods
    % IsolatedSpinGrp(M,M0,T1,T2,gamma,density) - initializes the spin
    % group with an initial magnetization (M), equilibrium magnetization
    % (M0), T1 decay (T2), T2 Decay (T2), gyromagnetic ratio (gamma) and
    % spin density (density)
    % calculationFrame() - returns the frequency of the rotating
    % reference frame that dm is calculated in
    % dm(x,y,z,t,M) - returns a dm at some position (x,y,z), some time
    % (t), and some initial magnetization M ([Mx;My;Mz])

    properties
        b % input function
        kve % extravasation fraction
    end
    methods
        function self = TwoSitePerfusionExchangeGroup(varargin)
            % Constructor - initializes the spin group
            % IsolatedSpinGrp(M,M0,T1,T2,gamma,density) - initializes the spin
            % group with an initial magnetization (M), equilibrium magnetization
            % (M0), T1 decay (T2), T2 Decay (T2), gyromagnetic ratio (gamma) and
            % spin density (density)
            p = inputParser();
            p.addOptional('M',[0;0;1;0;0;1],@isnumeric)
            p.addOptional('M0',[0;0;0;0;0;0],@isnumeric)
            p.addOptional('T1a',56,@isnumeric)
            p.addOptional('T2a',0.02,@isnumeric)
            p.addOptional('ppma',171*1e-6,@isnumeric)
            p.addOptional('T1b',30,@isnumeric)
            p.addOptional('T2b',0.02,@isnumeric)
            p.addOptional('ppmb',185*1e-6,@isnumeric)
            p.addOptional('gamma',67.262e6,@isnumeric)
        end
    end
end
```



```

p.addOptional('density',1,@isnumeric)
p.addOptional('kab',0.3,@isnumeric)
p.addOptional('kba',0.0,@isnumeric)
p.addOptional('kve',1.0,@isnumeric)
p.addOptional('b', @(t)zeros(6,1))
p.parse(varargin{:})
if ~isempty(p.Results.M), self.M = p.Results.M;
else self.M = [0;0;1;0;0;1];end
if ~isempty(p.Results.M0),self.M0 = p.Results.M0;
else self.M0 = [0;0;0;0;0;0];end
if ~isempty(p.Results.T1a),self.T1a = p.Results.T1a;
else self.T1a = 56;end
if ~isempty(p.Results.T2a),self.T2a = p.Results.T2a;
else self.T2a = 0.02;end
if ~isempty(p.Results.ppma),self.ppma = p.Results.ppma;
else self.ppma = 171*1e-6;end
if ~isempty(p.Results.T1b),self.T1b = p.Results.T1b;
else self.T1b = 30;end
if ~isempty(p.Results.T2b),self.T2b = p.Results.T2b;
else self.T2b = 0.02;end
if ~isempty(p.Results.ppmB),self.ppmB = p.Results.ppmB;
else self.ppmB = 185*1e-6;end
if ~isempty(p.Results.gamma),self.gamma = p.Results.gamma;
else self.gamma = 67.262e6;end
if ~isempty(p.Results.density),self.density = p.Results.density;
else self.density = 1;end
if ~isempty(p.Results.kab),self.kab = p.Results.kab;
else self.kab = 0.3;end
if ~isempty(p.Results.kba),self.kba = p.Results.kba;
else self.kba = 0.0;end
if ~isempty(p.Results.kve),self.kve = p.Results.kve;
else self.kve = 1.0;end
if ~isempty(p.Results.b),self.b = p.Results.b;
else self.b = @(t)zeros(6,1);end
end
function A = getA(self,x,y,z,t,PS,B0,varargin)
if ~isempty(varargin) == 1
    B = varargin{1};
else
    B = repmat(B0,1,length(t))+PS.B(x,y,z,t);
end
theta = -self.calculationFrame(B0)*t;
Beff(1:3) = [cos(theta),-sin(theta),0;sin(theta),cos(theta),0;...
    0,0,1]*B-[0;0;1].*B0*...
    (1+mean([self.ppma,self.ppmB])-self.ppma);
Beff(4:6) = [cos(theta),-sin(theta),0;sin(theta),cos(theta),0;...
    0,0,1]*B-[0;0;1].*B0*...
    (1+mean([self.ppma,self.ppmB])-self.ppmB);
A = zeros(6);
A(1:3,1:3) = self.gamma*...
    [0,-Beff(3),Beff(2);Beff(3),0,-Beff(1);-Beff(2),Beff(1),0]+...
    [-1/self.T2a,0,0;0,-1/self.T2a,0;0,0,-1/self.T1a];
A(4:6,4:6) = self.gamma*...

```

```

        [0,-Beff(6),Beff(5);Beff(6),0,-Beff(4);-Beff(5),Beff(4),0]+...
        [-1/self.T2b,0,0;0,-1/self.T2b,0;0,0,-1/self.T1b];
    A = A+[-self.kab-self.kve,0,0,self.kba,0,0;...
        0,-self.kab-self.kve,0,0,self.kba,0;...
        0,0,-self.kab-self.kve,0,0,self.kba;...
        self.kab,0,0,-self.kba,0,0;...
        0,self.kab,0,0,-self.kba,0;...
        0,0,self.kab,0,0,-self.kba];
end
function dm = dm(self,x,y,z,t,M,PS,B0)
    % DM: returns the delta m at some time and location and given M
    % dm(x,y,z,t,M) - returns a dm at some position (x,y,z), some time
    % (t), and some initial magnetization M ([Mx;My;Mz])
    Recovery(1:3) = 1/self.T1a*self.M0(1:3);
    Recovery(4:6) = 1/self.T1b*self.M0(4:6);
    dm = self.getA(x,y,z,t,PS,B0)*M+Recovery.'+self.kve*self.b(t);
end
function vals = analytical(self,x,y,z,t0,M,t,PS,B0,varargin)
    % ANALYTICAL: return a function handle to the analytical solution
    warning('off','MATLAB:integral:NonFiniteValue')
    warning('off','MATLAB:trapz:NonFiniteValue')
    A = self.getA(x,y,z,t0+1e-9,PS,B0,varargin{:});
    if length(t) == 1
        ForceFunIntegral = integral(@(t2)expm(-A*(t2-t0))*self.kve*self.b(t2),...
            t0,t,'ArrayValued',true);
        ForceFunIntegral(isnan(ForceFunIntegral)) = 0;
        vals = expm(A*(t-t0))*(M+ForceFunIntegral);
    else
        vals = zeros(length(M),length(t));
        tmpT = t0-0.001:0.001:t(end)+0.001;
        tmpVals = cell2mat(arrayfun(@(t2)expm(-A*(t2-t0))*...
            self.kve*self.b(t2),tmpT,'UniformOutput',false));
        tmpIntegral = cumtrapz(tmpT,tmpVals,2);
        ForceFunIntegral = interp1(tmpT,tmpIntegral.',t).';
        ForceFunIntegral(isnan(ForceFunIntegral)) = 0;
        for i = 1:length(t)
            vals(:,i) = expm(A*(t(i)-t0))*(M+ForceFunIntegral(:,i));
        end
    end
    if (any(isnan(vals(:))) || any(isinf(vals(:))))
        fprintf('warning still have a NaN or INF in the solution')
        tmp = self.analytical(self,x,y,z,t0,M,t(1:floor(end/2)),PS,B0,varargin);
        tmp2 =
self.analytical(self,x,y,z,t0,tmp(end),t(floor(end/2))+1:end,PS,B0,varargin);
        vals = [tmp,tmp2];
    %
    %         odefun = @(M,t)self.dm(x,y,z,M,t,PS,B0);
    %         tmpSol = ode45(odefun,[t0,t(end)],M);
    %         vals = deval(tmpSol,t);
    end
    warning('on','MATLAB:integral:NonFiniteValue')
    warning('on','MATLAB:trapz:NonFiniteValue')
end
end

```

end

Bankson Spin Group Class

A Bankson spin group represents the vascular pool of a voxel. It will switch between defining M_z as the VIF or allowing it to follow the Bloch equations when an RF pulse is on. Instead of taking an arbitrary VIF, it assumes a gamma variate and therefore defines the shape terms α and β , t_0 and a scaling factor as properties. Note that the switching for M_z is dependent on the analytical solution with the numerical solution ignoring the VIF. Therefore, if $M_0 \neq 0$ and the analytical solution is never used, then the VIF will never be used.

```
classdef BanksonSpinGrp < Hypwright.SpinGroup

    properties
        M % magnetization vector
        M0 % Equilibrium magnetization
        T1 % T1 decay constant
        T2 % T2 Decay constant
        gamma % gyromagnetic ratio
        ppm % ppm shift of the spin
        density % relative number of spins in this group
        shapeTerms % terms that define the gamma pdf
        t0 %injection delay
    end
    methods
        function self = BanksonSpinGrp(M,M0,T1,T2,gamma,ppm,density,shapeTerms,t0)
            % Constructor - initializes the spin group
            % v(M,M0,T1,T2,gamma,ppm,density) - initializes the spin
            % group with an initial magnetization (M), equilibrium magnetization
            % (M0), T1 decay (T1), T2 Decay (T2), gyromagnetic ratio (gamma),
            % some chemical shift (ppm), spin density (density), and shape terms
            % for the gamma pdf (shapeTerms)
            self.M = M;
            self.M0 = M0;
            self.T1 = T1;
            self.T2 = T2;
            self.gamma = gamma;
            self.ppm = ppm;
            self.density = density;
            self.shapeTerms = shapeTerms;
            self.t0 = t0;
            % war user if the use analytical will always be true, if so
            % this spin group will not account for perfusion
            if(~useAnalytical(self))
                warning('The Analytical Solution of a Bankson Spin group will no be used. Any
                Perfusion in that spin group will be ignored\n')
```

```

end
end
function val = calculationFrame(self,B0)
    % CALCULATIONFRAME - returns the frequency of the rotating
    % reference frame that dm is calculated in;
    tmp = [0;0;1].*B0*self.gamma*(1+self.ppm);
    val = tmp(3);
end
function dm = dM(self,x,y,z,t,M,PS,B0)
    % DM: returns the delta m at some time and location and given M
    % dM(x,y,z,t,M) - returns a dm at some position (x,y,z), some time
    % (t), and some initial magnetization M ([Mx;My;Mz])
    dm = self.getA(x,y,z,t,PS,B0)*M+1/self.T1*self.M0;
end
function A = getA(self,x,y,z,t,PS,B0,varargin)
    % GETA - gets the matrix that defines dm
    if ~isempty(varargin) == 1
        B = varargin{1};
    else
        B = repmat(B0,1,length(t))+PS.B(x,y,z,t);
    end
    theta = -self.calculationFrame(B0)*t;
    Beff = [cos(theta),-sin(theta),0;sin(theta),cos(theta),0;0,0,1]*B-...
        [0;0;1].*B0;
    A = self.gamma*...
        [0,-Beff(3),Beff(2);Beff(3),0,-Beff(1);-Beff(2),Beff(1),0] + ...
        [-1/self.T2,0,0;0,-1/self.T2,0;0,0,-1/self.T1];
end
function ret = useAnalytical(self)
    %USEANALYTICAL: determines if the Analytical Solution should be used
    %for the given spin group under the given conditions
    ret = all(self.M0 == 0);
end
function vals = analytical(self,x,y,z,t0,M,t,PS,B0,varargin)
    % ANALYTICAL: return a function handle to the analytical solution
    A = self.getA(x,y,z,t0+1e-9,PS,B0,varargin{:});
    vals = cell2mat(arrayfun(@(t2)[subsref(expm(A*(t2-t0))*M,...
        struct('type','()', 'subs',{1:2,1}))];...
        self.shapeTerms(3)*gampdf(...
        t2-self.t0,self.shapeTerms(1),self.shapeTerms(2))],t,...
        'UniformOutput',false));
end
end
end
end

```

Section B.4 Signal Curve Modeling and Fitting.

The modeling architecture is slightly more advanced than the spin group logic and many of the features in the modeling classes should be translated to the spin group objects to improve their flexibility. Additionally, the new version of the modeling has not been written for averaged excitation loss which needs to be done. All models are abstract, which means that they cannot be instantiated. Conceptually they represent a model or set of equations that applies to some data. Therefore, each model must have passed in to it all of the parameters that it needs in order to evaluate any results. Finally, there is practically no input validation for parameters, as it was seen as too burdensome for the least squares fitting procedure.

MultiPool Class

The multi-pool model represents any system of N exchanging spins. The exchange process is defined by the ExchangeTerms matrix, which must be NxN, and by the exchange term k_{rc} representing the exchange from chemical species number in the row to the chemical species number in the column. For pyruvate and lactate, with $k_{pl} = 0.1$ and $k_{lp} = 0.001$; $ExchangeTerms = \begin{bmatrix} 0 & 0.1 \\ 0.001 & 0 \end{bmatrix}$. The T_1 values are also passed in for each chemical species in an Nx1 vector. A list of excitation times is also passed in as a vector. Finally, excitation angles (in degrees) for each chemical species at each excitation angle is passed in as an NxNTR matrix. For constant excitation angles only an Nx1 vector is needed and all TRs will be filled in with the same excitation angle. With these parameters defined, an arbitrary system of exchanging spins can be built and will interact with the series of excitation pulses. Parameters are passed in as a structure with name-value pairs for each parameter. A similar structure is used for fitting. Fitting options can be set, as well as the fitting limits for a least squares fitting.

```
classdef MultiPool
    %TWOPOOL A simple chemical exchange model assuming no inpu functions
    % parameters values
```

```

% * ExchangeTerms - A Matrix defining chemical Exchange. Default: 0
% * T1s - A row vector of T1 decay terms. Default: 100
% * FaList - A matrix of excitation angles. Default: 0
% * TRList - A matrix of excitation times. Default: 0
% * fitOptions - A matlab fit option structure. Default: optimset('lsqcurvefit')
% There is NO input validation for the parameters passed in, for more
% detail on the assumed data structur of these parameters use the
% defaults function

properties
end

methods (Static)
function defaults()
    % DEFAULTS explains the default values for each parameter
    names = {'ExchangeTerms','T1s','FaList','TRLList','fitOptions'};
    discriptions = {'A NxN Matrix of Exchange Terms, where N is the number of chemical
pools. The From pools should be along the rRows with the To pool along the Columns. Diagonal
elemets will be set to zero'...
        ' A Row vector of T1 decay times for each chemical pool.'...
        ' A NxM of matrix of flip angles in radians, where N is the number of
excitations and M is the number of chemical Pools'...
        ' A NxM of Excitation Times in seconds, where N is the number of excitations and
M is the number of chemical Pools'...
        ' Matlab FitOptions object'};
    defaultsvals = {'0','100','0','0','optimset('lsqcurvefit')'};
    for i = 1:numel(names)
        fprintf(''%s': %s\n Default Vaule: %s\n',...
            names{i},discriptions{i},defaultsvals{i});
    end
end
function paramsOut = parseParams(paramsIn)
    % parseParams: a function to fill default param values if they are
    % not defined
    default = struct('ExchangeTerms',0,'T1s',100,'FaList',0,...
        'TRLList',0,'fitOptions', optimset('lsqcurvefit'));
    tmpNames = fieldnames(default);
    paramsOut = paramsIn;
    for i = 1:numel(tmpNames)
        if ~isfield(paramsOut,tmpNames{i})
            paramsOut.(tmpNames{i}) = default.(tmpNames{i});
        end
    end
    % Fill all flip angles with a value if only one flip angle is passed in
    if size(paramsOut.FaList,2)==1
        paramsOut.FaList = repmat(paramsOut.FaList(:,1),...
            1,length(paramsOut.TRLList));
    end
    % Assuming input validation will put too much computational burdon on the fitting
    % Hopeing user will supply valid input!
    % Validte input
    % if (size(params.ExchangeTerms,1)~=size(params.ExchangeTerms,1))
    %     error('Exchange matrix not square.')

```

```

%           end
N = size(paramsIn.ExchangeTerms,1);
K = triu(paramsIn.ExchangeTerms)+tril(paramsIn.ExchangeTerms);
T1 = paramsIn.T1s;
A = zeros(N);
for i = 1:N
    for j = 1:N
        if(i == j)
            A(i,i) = -sum(K(i,:))-1/T1(i);
        else
            A(i,j) = K(j,i);
        end
    end
end
end
paramsOut.A = A;
end
function [TRLList,Mxy,Mz] = compile(M0,params)
% EVALUATE: runs the model based on some input parameters
params = Hypwright.Models.MultiPool.parseParams(params);
A = params.A;
FaList = params.FaList;
TRLList = params.TRLList;
[TRLList, Mxy, Mz] = Hypwright.Models.MultiPool.evaluate(...
    TRLList,FaList,M0,A);
end
function [x,resultParams,allParams,resnorm,residual,exitflag,output,lambda,jacobian]...
    = fitData(params,guess,xdata,ydata,varargin)
p = inputParser();
p.addOptional('lb',[])
p.addOptional('ub',[])
p.parse(varargin{:})
xNames = fieldnames(guess);
j = 1;
xIndex = cell(size(xNames));
for i = 1:numel(xNames)
    iFits = ~isnan(guess.(xNames{i}));
    xIndex{i} = find(iFits==1);
    for k = 1:numel(xIndex{i})
        x0(j) = guess.(xNames{i})(xIndex{i}(k));
        j = j+1;
    end
end
end
params = Hypwright.Models.MultiPool.parseParams(params);
Y0 = ydata(:,1)./sin(params.FaList(:,1));
fun = @(x,xdata)Hypwright.Models.MultiPool.fitFunction(...
    params,x,xNames,xIndex,Y0);
opts = params.fitOptions;
[x,resnorm,residual,exitflag,output,lambda,jacobian] = ...
    lsqcurvefit(fun,x0,xdata,ydata,...
    [p.Results.lb],[p.Results.ub],opts);
resultParams = guess;
allParams = params;
j = 1;

```



```

    for i = 1:numel(xNames)
        for k = 1:numel(xIndex{i})
            resultParams.(xNames{i})(xIndex{i}(k)) = x(j);
            allParams.(xNames{i})(xIndex{i}(k)) = x(j);
            j = j+1;
        end
    end
end

function [TRLList, Mxy, Mz] = evaluate(TRLList, FaList, M0, A)
    % EVALUATE: runs the model based on some input parameters
    fun = @(t,y)A*y;
    Mz = zeros(size(FaList));
    Mxy = zeros(size(FaList));
    Mz(:,1) = M0.*cos(FaList(:,1));
    Mxy(:,1) = (M0.*sin(FaList(:,1)));
    for i = 2:length(TRLList)
        [~,Y] = ode45(fun,[TRLList(i-1),TRLList(i)],Mz(:,i-1));
        Mz(:,i) = Y(end,:).';
        Mxy(:,i) = sin(FaList(:,i)).*Mz(:,i);
        Mz(:,i) = cos(FaList(:,i)).*Mz(:,i);
    end
end

function DataCompare(A, params, M0, xdata, ydata)
    M0 = M0./sin(params.FaList(:,1));
    [TRLList, Mxy, ~] = A.compile(M0, params);
    figure
    for i = 1:size(Mxy,1)
        tmpLine = plot(TRLList, Mxy(i,:));
        hold on
        plot(xdata, ydata(i,:), 'o', 'MarkerEdgeColor', tmpLine.Color);
    end
    hold off
    xlabel('Time (sec)')
    ylabel('Signal (arb)')
end

methods (Access = private, Static)
function Y = fitFunction(params, x, xNames, xIndex, Y0)
    % fitFunction packs the parameter in params and x up and evaluates
    % using the evaluate function over some time (tSpan) with some
    % initial value (Y0)
    j = 1;
    for i = 1:numel(xNames)
        for k = 1:numel(xIndex{i})
            params.(xNames{i})(xIndex{i}(k)) = x(j);
            % check if fitting flip angle (there must be a better
            % way to do this
            if strcmp(xNames{i}, 'FaList')
                params.(xNames{i}) = ...
                    repmat(x(j), size(params.(xNames{i})));
            end
            j = j+1;
        end
    end
end

```

```
        end
    end
    [~, Y, ~] = Hypwright.Models.MultiPool.compile(Y0,params);
end
end
end
```

MultiPool Tofts Class

Similar to the Multipool function, but allows for perfusion using the two physical pool model.

Signals from the vascular and extravascular spaces are combined automatically. The exchange terms

need to be an Nx1 matrix representing k_{ve} for each chemical species. Finally, the VIF is an abstract

function that should return an Nx1 matrix for the M_z of the VIF. Note that the abstract nature of the VIF

does not allow it to be fit.

```
classdef MultiPoolTofts
    %TWOPOOLTOFTS Summary of this class goes here
    % Detailed explanation goes here

    properties
    end

    methods (Static)
        function defaults()
            % DEFAULTS explains the default values for each parameter
            names = {'ExchangeTerms', 'T1s', 'FaList', 'TRLList', ...
                'PerfusionTerms', 'volumeFractions', 'VIF', 'fitOptions'};
            discriptions = {'A NxN Matrix of Exchange Terms, where N is the number of chemical
                pools. The From pools should be along the rRows With the To pool along the Columns. Diagonal
                elemets will be set to zero'...
                ' A Row vector of T1 decay times for each chemical pool.'...
                ' A NxM of matrix of flip angles in radians, where N is the number of
                excitations and M is the number of chemical Pools'...
                ' A NxM of Excitation Times in seconds, where N is the number of excitations and
                M is the number of chemical Pools'...
                ' A Row Vector of perfusion Exchange Constrnats for each chemical pool.'...
                ' A Row Vector of volme fraction for each chemical pool. Only one value can be
                use if all pools have the same volume fraction.'...
                ' A function of a time variable (t) in seconds that returns a Row vector for the
                VIF of each chemical pool at the time t.'...
                ' Matlab FitOptions object'};
            defaultsvals = {'0', '100', '0', '0', '0', '1', '@(t)0', 'optimset(''lsqcurvefit'')'};
            fprintf('*Note* all terms must be a vector of size 1 x N where N is the number of
                chemical Pools\n')
            for i = 1:numel(names)
                fprintf(''%s': %s\n Default Vaule: %s\n', ...
                    names{i}, discriptions{i}, defaultsvals{i});
            end
        end
        function paramsOut = parseParams(paramsIn)
            % parseParams: a function to fill default param values if they are
            % not defined
            default = struct('ExchangeTerms', 0, 'T1s', 100, 'FaList', 0, ...
                'TRLList', 0, 'PerfusionTerms', 0, 'volumeFractions', 1, 'VIF', @(t)0, ...
```

```

        'fitOptions', optimset('lsqcurvefit'));
    tmpNames = fieldnames(default);
    paramsOut = paramsIn;
    for i = 1:numel(tmpNames)
        if ~isfield(paramsOut,tmpNames{i})
            paramsOut.(tmpNames{i}) = default.(tmpNames{i});
        end
    end
    % Fill all flip angles with a value if only one flip angle is passed in
    if size(paramsOut.FaList,2)==1
        paramsOut.FaList = repmat(paramsOut.FaList(:,1),...
            1,length(paramsOut.TRLList));
    end
    % Assuming input validation will put too much computational burden on the fitting
    % Hoping user will supply valid input!
    % Validte input
    if (size(params.ExchangeTerms,1)~=size(params.ExchangeTerms,1))
        error('Exchange matrix not square.')
    end
    N = size(paramsIn.ExchangeTerms,1);
    K = triu(paramsIn.ExchangeTerms)+tril(paramsIn.ExchangeTerms);
    T1 = paramsIn.T1s;
    kve = paramsIn.PerfusionTerms;
    if(length(paramsIn.volumeFractions)==1)
        ve = zeros(N,1)+paramsIn.volumeFractions;
    else
        ve = paramsIn.volumeFractions;
    end
    A = zeros(N);
    for i = 1:N
        for j = 1:N
            if(i == j)
                A(i,i) = -sum(K(i,:))-1/T1(i)-kve(i)/ve(i);
            else
                A(i,j) = K(j,i);
            end
        end
    end
    paramsOut.A = A;
    paramsOut.b = paramsIn.VIF;
    paramsOut.kve = paramsIn.PerfusionTerms;
    paramsOut.ve = paramsIn.volumeFractions;
end
function [TRLList,Mxy,Mz] = compile(M0,params)
    % EVALUATE: runs the model based on some input parameters
    params = Hypwright.Models.MultiPoolToffts.parseParams(params);
    A = params.A;
    b = params.b;
    FaList = params.FaList;
    TRLList = params.TRLList;
    [TRLList, Mxy, Mz] = Hypwright.Models.MultiPoolToffts.evaluate(...
        TRLList,FaList,M0,A,b,params);
end

```

```

function [TRLlist, Mxy, MZ] = evaluate(TRLlist, FaList, M0, A, b, params)
    % EVALUATE: runs the model based on some input parameters
    kve = params.kve;
    ve = params.ve;
    fun = @(t,y)A*y+(kve/ve).'*b(t);
    MZ = zeros(size(FaList));
    Mxy = zeros(size(FaList));
    MZ(:,1) = M0.*cos(FaList(:,1));
    Mxy(:,1) = (params.ve*M0+(1-params.ve)*params.b(TRLlist(1))).*sin(FaList(:,1));
    for i = 2:length(TRLlist)
        [~,Y] = ode45(fun,[TRLlist(i-1),TRLlist(i)],MZ(:,i-1));
        MZ(:,i) = Y(end,:).';
        Mxy(:,i) = sin(FaList(:,i)).*(params.ve.*MZ(:,i)+...
            (1-params.ve).*b(TRLlist(i)));
        MZ(:,i) = cos(FaList(:,i)).*MZ(:,i);
    end
end
function DataCompare(A,params,M0,xdata,ydata)
    M0 = M0./sin(params.FaList(:,1));
    [TRLlist,Mxy,~] = A.compile(M0,params);
    figure
    for i = 1:size(Mxy,1)
        tmpLine = plot(TRLlist,Mxy(i,:));
        hold on
        plot(xdata,ydata(i,:), 'o', 'MarkerEdgeColor', tmpLine.Color);
    end
    hold off
    xlabel('Time (sec)')
    ylabel('Signal (arb)')
end
end
end

```

MultiPool Tofts Gamma VIF Class

A child class MultiPool Tofts Class that simply defines the VIF as a gamma variate. It allows the VIF amplitude and the shape terms to be fit. In order to zero out the VIF for a particular species, it is best to zero out its VIF scale factor. Zeroing the shape terms is a risky choice.

```
classdef MultiPoolToftsGammaVIF < Hypwright.Models.MultiPoolTofts
    %MULTIPOOLTOFTSGAMMAVIF A chemical exchange model assuming two pooled
    %Tofts model of perfusion
    % parameters values
    %* ExchangeTerms - A Matrix defining chemical Exchange. Default: 0
    %* T1s - A row vector of T1 decay terms. Default: 100
    %* FaList - A matrix of excitation angles. Default: 0
    %* TRLIST - A matrix of excitation times. Default: 0
    %* t0 - A row vector for delivery delay of each metabolite. Default: 0
    %* gammaPdfA - A row vector for shape term alpha of each metabolite. Default: 2.8
    %* gammaPdfB - A row vector for shape term beta of each metabolite. Default: 4.5
    %* ScaleFactor - A row vector for each metabolite's VIF scale factor. Default: 1
    %* fitOptions - A matlab fit option structure. Default: optimset('lsqcurvefit')
    %* PerfusionTerms - A row vector for each metabolite's extravasation rate. Default: 0
    %* volumeFractions - A row vector for each metabolite's volume fraction. Default: 1
    % There is NO input validation for the parameters passed in, for more
    % detail on the assumed data structur of these parameters use the
    % defaults function
    properties
    end
    methods (Static)
        function defaults()
            % DEFAULTS explains the default values for each parameter
            names = {'t0','gammaPdfA','gammaPdfB','scaleFactor'};
            discriptions = {'A Row vector of time delays for each metabolite'...
                'A Row vector of shape term Alpha, set this to zero to have no VIF for a
chemical pool'...
                'A Row vector of shape term Beta, this cannot be zero and will be set to 1e-40
if zero is used'...
                'A Row vector of Scale Factor to be applied to the VIF'};
            defaultsvals = {'0','2.8','4.5','1'};
            fprintf('*Note* all terms must be a vector of size 1 x N where N is the number of
chemical Pools\n')
            for i = 1:numel(names)
                fprintf(''%s': %s\n Default Vaule: %s\n',...
                    names{i},discriptions{i},defaultsvals{i});
            end
            defaults@Hypwright.Models.MultiPoolTofts();
        end
        function paramsOut = parseParams(paramsIn)
            % parseParams: Parses the input shape terms of a gamma variate
            % for the VIF, each term should be a vector with shape terms
            % for each chemical species
        end
    end
end
```

```

% Fill Default Values
default = struct('t0',0,'gammaPdfA',2.8,...
    'gammaPdfB',4.5,'scaleFactor',1);
tmpNames = fieldnames(default);
paramsOut = paramsIn;
for i = 1:numel(tmpNames)
    if ~isfield(paramsOut,tmpNames{i})
        paramsOut.(tmpNames{i}) = default.(tmpNames{i});
    end
end
% Build VIF
paramsOut.VIF = @(t)paramsIn.scaleFactor.*...
    gampdf(t-paramsIn.t0,paramsIn.gammaPdfA,paramsIn.gammaPdfB);
% Fill in parent Class defaults
paramsOut = parseParams@Hypwright.Models.MultiPoolToffts(paramsOut);
end
function [TRLlist,Mxy,Mz] = compile(M0,params)
% EVALUATE: runs the model based on some input parameters
params = Hypwright.Models.MultiPoolTofftsGammaVIF.parseParams(params);
[TRLlist,Mxy,Mz] = compile@Hypwright.Models.MultiPoolToffts(M0,params);
end
function [x,resultParams,allParams,resnorm,residual,exitflag,output,lambda,jacobian]...
    = fitData(params,guess,xdata,ydata,varargin)
p = inputParser();
p.addOptional('lb',[])
p.addOptional('ub',[])
p.parse(varargin{:})
xNames = fieldnames(guess);
j = 1;
xIndex = cell(size(xNames));
for i = 1:numel(xNames)
    iFits = ~isnan(guess.(xNames{i}));
    xIndex{i} = find(iFits==1);
    for k = 1:numel(xIndex{i})
        x0(j) = guess.(xNames{i})(xIndex{i}(k));
        j = j+1;
    end
end
end
params = Hypwright.Models.MultiPoolTofftsGammaVIF.parseParams(params);
Y0 = ydata(:,1)./sin(params.FaList(:,1));
fun = @(x,xdata)Hypwright.Models.MultiPoolTofftsGammaVIF.fitFunction(...
    params,x,xNames,xIndex,xdata,Y0);
opts = params.fitOptions;
[x,resnorm,residual,exitflag,output,lambda,jacobian] = ...
    lsqcurvefit(fun,x0,xdata,ydata,...
    [p.Results.lb],[p.Results.ub],opts);
resultParams = guess;
allParams = params;
j = 1;
for i = 1:numel(xNames)
    for k = 1:numel(xIndex{i})
        resultParams.(xNames{i})(xIndex{i}(k)) = x(j);
    end
end

```

```

        allParams.(xNames{i})(xIndex{i}(k)) = x(j);
        j = j+1;
    end
end
end
end
methods (Access = private, Static)
    function Y = fitFunction(params,x,xNames,xIndex,tSpan,Y0)
        % fitFunction packs the parameter in params and x up and evaluates
        % using the evaluate funnction over some time (tSpan) with some
        % initial value (Y0)
        j = 1;
        for i = 1:numel(xNames)
            for k = 1:numel(xIndex{i})
                params.(xNames{i})(xIndex{i}(k)) = x(j);
                j = j+1;
            end
        end
        params.TRLlist = tSpan;
        [~, Y, ~] = Hypwright.Models.MultiPoolTofftsGammaVIF.compile(Y0,params);
    end
end
end
end

```


Gamma Bankson Model Class

The Gamma Bankson Model Class is a sample of the old modeling system that averages excitation losses over the repetition time. It behaves similarly to the abstract modeling above. However, the number of chemical species is defined to be two and therefore no matrices are used as parameters.

```
classdef GammaBanksonModel < Hypwright.Models.BanksonModel
    %BANKSONMODEL model for a two site exchange system with perfused by a
    %vascular pool. b defines the vascular input function wich is assumed to be
    %uneffected.
    % This is the basic two site exchange model. This model has a linear flip
    % angle correction and will not work for non-linear sampling.
    % The parameters for this model follow
    % Kab - exchange reate from pool a to b: default 0
    % Kba - exchange reate from pool b to a: default 0
    % T1a - T1 decay constant for pool a: default 56
    % T1b - T1 decay constant for pool b: default 31
    % flipAngle - excitation angle in radians: default 0
    % TR - repetition time (again this is a linear TR model): default 1
    % kve - vascular extraction fraction: default 0.122
    % ve - vascular volume fraction: default 0.91
    % b - input function, some function of time that returns a change in
    % pool a and b must return a 2 row vector: default [0;0]
    properties (Access = private)
    end
    methods (Static)
        function [Y,T,sol] = evaluate(params,tSpan,Y0)
            % EVALUATE: solves this model over some time span (tSpan), with an
            % initial Y (Y0) and some parameters (params).
            % Params is a struct with the values
            % Kab - exchange reate from pool a to b: default 0
            % Kba - exchange reate from pool b to a: default 0
            % T1a - T1 decay constant for pool a: default 56
            % T1b - T1 decay constant for pool b: default 31
            % flipAngle - excitation angle in radians: default 0
            % TR - repetition time (again this is a linear TR model): default 1
            % kve - vascular extraction fraction: default 0.122
            % ve - vascular volume fraction: default 0.91
            % b - input function, some function of time that returns a change in
            % pool a and b must return a 2 row vector: default [0;0]
            params = Hypwright.Models.GammaBanksonModel.parseParams(params);
            A = [-(params.kve/params.ve+params.Kab+...
                1/params.T1a+((1-cos(params.flipAngle))/params.TR)),...
                params.Kba; params.Kab,...
                -(params.Kba+1/params.T1b+((1-cos(params.flipAngle))/params.TR))];
            % Analytic Solution (VERY SLOW!)
            fun = @(t)expm(A*(t-tSpan(1)))*Y0+integral(...
                @(t2)expm(A*(t-t2))*params.kve/params.ve*params.b(t2),...
                tSpan(1),t,'ArrayValued',true);
```

```

%         Y = zeros(length(Y0),length(tSpan));
%         for i = 1:length(tSpan)
%             Y(:,i) = (1-params.ve)*fun(tSpan(i))+params.ve*params.b(tSpan(i));
%         end
T = tSpan;
bfit = @(t)params.scaleFactor*padarray(gampdf(...
    t,params.gammaPdfA,params.gammaPdfB),1,'post');
fun = @(t,y)A*y+params.kve/params.ve*bfit(t-params.t0);
sol = ode45(fun,tSpan,Y0);
if length(T) == 2, T = sol.x; end
Y = params.ve*deval(sol,T)+(1-params.ve)*bfit(T-params.t0);
end
function [x,resultParams,resnorm,residual,exitflag,output,lambda,jacobian] =
fitData(params,guess,...
    xdata,ydata,varargin)
% FITDATA: fits some data set (xdata, ydata) with some constant
% parameters (params) and variable parameters (guess) using the
% perfused two site exchange model. Which ever parameters are in the
% guess struct will be fit, any parameters in the params struct will
% be held constant, any parameters not specified will be set to their
% defaults and held constant. The fit will return one more argument
% than the number of guesses. The last number is a scaling factor
% applied to the fit data, to better match the magnitude of the
% input
% Params is a struct with the values
% Kab - exchange reate from pool a to b: default 0
% Kba - exchange reate from pool b to a: default 0
% T1a - T1 decay constant for pool a: default 56
% T1b - T1 decay constant for pool b: default 31
% flipAngle - excitation angle in radians: default 0
% TR - repetition time (again this is a linear TR model): default 1
% kve - vascular extraction fraction: default 0.122
% ve - vascular volume fraction: default 0.91
% b - input function, some function of time that returns a change in
% pool a and b must return a 2 row vector: default [0;0]
p = inputParser();
p.addOptional('lb',[])
p.addOptional('ub',[])
p.parse(varargin{:})
xNames = fieldnames(guess);
x0 = zeros(numel(xNames),1);
for i = 1:numel(xNames)
    params.(xNames{i}) = guess.(xNames{i});
    x0(i) = guess.(xNames{i});
end
params = Hypwright.Models.GammaBanksonModel.parseParams(params);
Y0 = ydata(:,1);
fun = @(x,xdata)Hypwright.Models.GammaBanksonModel.fitFunction(...
    params,x,xNames,xdata,Y0);
opts = params.fitOptions;
[x,resnorm,residual,exitflag,output,lambda,jacobian] = ...
    lsqcurvefit(fun,x0,xdata,ydata,...
    [p.Results.lb],[p.Results.ub],opts);

```

```

        resultParams = params;
    for i = 1:numel(xNames)
        resultParams.(xNames{i}) = x(i);
    end
end
function dataCompare(params,xdata,ydata,varargin)
    % DATACOMPARE: displays the model with the parameters parameters
    % (params) against the data (xdata, ydata). optiona 4th argument for
    % a figure axis in which to draw the plot
    % Params is a struct with the values
    % Kab - exchange reate from pool a to b: default 0
    % Kba - exchange reate from pool b to a: default 0
    % T1a - T1 decay constant for pool a: default 56
    % T1b - T1 decay constant for pool b: default 31
    % flipAngle - excitation angle in radians: default 0
    % TR - repetition time (again this is a linear TR model): default 1
    % kve - vascular extraction fraction: default 0.122
    % ve - vascular volume fraction: default 0.91
    % b - input function, some function of time that returns a change in
    % pool a and b must return a 2 row vector: default [0;0]
    % axis - axis handle to plot data
    p = inputParser();
    p.addOptional('axis',[])
    p.parse(varargin{:})
    Y0 = ydata(:,1);
    Y = Hypwright.Models.GammaBanksonModel.evaluate(params,xdata,Y0);
    resNorm = sum(sum((Y-ydata).^2));
    if isempty(p.Results.axis)
        figure;
        curAxis = gca;
    else
        curAxis = p.Results.axis;
    end
    plot(curAxis,xdata,Y(1,:),'g',xdata,Y(2,:),'b',...
        xdata,ydata(1,:),'go',xdata,ydata(2,:),'bo')
    xlabel('Time')
    ylabel('Signal Intensity')
    legend('Model Pool A','Model Pool B')
    title('Comparison of data with two site exchange model')
    fprintf('The norm of the residual is: %d\n',resNorm)
end
end
methods (Static, Access = protected)
function paramsOut = parseParams(paramsIn)
    % parseParams: a function to fill default param values if they are
    % not defined
    default = struct('Kab',0,'Kba',0,'T1a',56,'T1b',31,'flipAngle',0,...
        'TR',1,'kve',0.02,'ve',0.91,'t0',0,'gammaPdfA',2.8,...
        'gammaPdfB',4.5,'scaleFactor',1,'fitOptions', optimset('lsqcurvefit'));
    tmpNames = fieldnames(default);
    paramsOut = paramsIn;
    for i = 1:numel(tmpNames)
        if ~isfield(paramsOut,tmpNames{i})

```

```

        paramsOut.(tmpNames{i}) = default.(tmpNames{i});
    end
    end
    paramsOut.b = @(t)paramsOut.scaleFactor*[...
        gampdf(t-paramsOut.t0,paramsOut.gammaPdfA,paramsOut.gammaPdfB);0];
end
function Y = fitFunction(params,x,xNames,tSpan,Y0)
    % fitFunction packs the parameter in params and x up and evaluates
    % using the evaluate function over some time (tSpan) with some
    % initial value (Y0)
    for i = 1:numel(xNames)
        params.(xNames{i}) = x(i);
    end
    % Uses the last value of x as a scaling factor.
    Y = Hypwright.Models.GammaBanksonModel.evaluate(params,tSpan,Y0);
end
end
end

```

Section B.5 Example Scripts

Example Script for Simulating and Processing single pulse dynamic spectroscopy

```
function [raw,t,freqAx] = PerfusedCalc(base)
```

```
import Hypwright.*
import Hypwright.Models.*
```

Initialize variable

```
if ~exist('base','var')
    base = struct();
end
% Default values, empty fields are only used in the fitting and not need for
% this function, however then still are checked when looking for
% superfluous variables
Default = struct('gamma', 67.262e6, 'readBandwidth', 4096, 'rfBandwidth', 5000,...
    'nPoints', 2048, 't0', 0, 'endTime', 100, 'T1a', 56, 'T2a', 0.02, 'T1b', 30,...
    'T2b', 0.02, 'kve', 0.02, 'vb', 0.09, 've', .91, 'ppma', -7e-6,...
    'ppmb', 7e-6, 'gammaPdfA', 2.8, 'gammaPdfB', 4.5, 'scaleFactor', 1,...
    'Kab', 0.1, 'flipAngle', 20, 'TR', 2, 'verbose', false,...
    'FWHMRange', [], 'A', [], 'noiseLevel', [],...
    'nAverages', [], 'lb', [], 'ub', [], 'centers', [], 'fitOptions', [],...
    'B0', 3.0);
% Check that there are no unused variables in base;
tmpNames = fieldnames(base);
for i = 1:numel(tmpNames)
    if ~isfield(Default, tmpNames{i})
        warning('WARNING! the field "%s" was passed in but does not match any of the default
names.\n', tmpNames{i});
        warning('This variable wont be used!\n')
    end
end
tmpNames = fieldnames(Default);
for i = 1:numel(tmpNames)
    if ~isfield(base, tmpNames{i})
        base.(tmpNames{i}) = Default.(tmpNames{i});
    end
end
base.flipAngle = base.flipAngle*pi/180;
gamma = base.gamma;
readBandwidth = base.readBandwidth;
rfBandwidth = base.rfBandwidth;
nPoints = base.nPoints;
endTime = base.endTime;
t0 = base.t0;
T1a = base.T1a;
T2a = base.T2a;
```

```

T1b = base.T1b;
kve = base.kve;
vb = base.vb;
ve = base.ve;
ppma = base.ppma;
ppmb = base.ppmb;
b = @(t)base.scaleFactor*padarray(padarray(...
    gampdf(t-t0,base.gammaPdfA,base.gammaPdfB),2),1,'post');
Kab = base.Kab;
flipAngle = base.flipAngle;
TR = base.TR;
verbose = base.verbose;
Mz = [];
B0 = base.B0;
%TODO add input validation;

```

Initialize World

```

world = Hypwright.world.getWorld();
world.initworld()
world.setB0([0;0;B0])
Spin = TwoSitePerfusionExchangeGroup([0;0;0;0;0;0],[0;0;0;0;0;0],...
    T1a,T2a,ppma,T1b,T2a,ppmb,gamma,ve,Kab,[],kve/ve,b);
Spin2 = BanksonSpinGrp([0;0;0],[0;0;0],T1a,T2a,gamma,ppma,vb,...
    [base.gammaPdfA,base.gammaPdfB,base.scaleFactor],t0);
V = voxel([0;0;0],Spin);
V.addSpin(Spin2);
if (verbose)
V.debug = true;
end
world.addVoxel(V);

```

Build Pulse Sequence

```

PS = PulseSequence;
t = 0:TR:endTime;
ADC = zeros(nPoints,length(t));
for i = 1:length(t)
    Pulse = SincPulse(t(i),rfBandwidth,flipAngle/(gamma),gamma*B0,[],...
        sprintf('Excitation%d',1));
    PS.addPulse(Pulse)
    ADC(:,i) = Pulse.endTime:1/readBandwidth:Pulse.endTime+(nPoints-1)/readBandwidth;
end
world.setPulseSequence(PS)

```

Calculate

```

world.calculate(t(end)+10);
if (verbose)

```

```

V.debug = true;
tMz = 0:0.1:endTime;
world.evaluate(tMz);
load('tmp')
evSpace = [];
vSpace = [];
for i = 1:size(tmp,1)
evSpace = [evSpace,tmp{i,1}];
vSpace = [vSpace,tmp{i,2}];
end
M(1:3,:) = ve*evSpace(1:3,:)+vb*vSpace;
M(4:6,:) = ve*evSpace(4:6,:);
MzPyr = M(3,:);
MzLac = M(6,:);
V.debug = false;
end
FID = zeros(nPoints,length(t));
for i = 1:length(t)
    [FID(:,i), freqAx] = world.evaluate(ADC(:,i).',-gamma*B0);
end
raw = FID;
t = linspace(ADC(floor(end/2),1),ADC(floor(end/2),end),size(ADC,2));
if (verbose)
    figure
    surf(t,freqAx,abs(fftshift(fft(FID,[],1),1)));
    drawnow
    figure('Name',sprintf('Kab: %.4f Flip Angle %2f Repetition Time %.4f',...
Kab,flipAngle,TR),'NumberTitle','off','Position',[660 50 1040 400])
    plot(tMz,MzPyr,'go',tMz,MzLac,'bo')
    legend('Simulated Pyruvate Mz','Simulated Lactate Mz');
    % Display model and Mz
end
end

function [ fits, fitErr, SNR, exitflag ] = PerfusedFit( base,fitParams,raw,t,freqAxis)

import Hypwright.*
import Hypwright.Models.*
import HypwrightRunners.*

```

Initialize variable

```

if isempty(base)
    base = struct();
end
if isempty(fitParams)
    error('No initial guesses were passed in for any fit parameters');
end
Default = struct('gamma', 67.262e6, 'readBandwidth', 4096, 'rfBandwidth', 5000,...
    'nPoints', 2048, 't0', 0, 'endTime', 100, 'T1a', 56, 'T2a', 0.02, 'T1b', 30,...
    'T2b', 0.02, 'kve', 0.02, 'vb', 0.09, 've', .91, 'ppma', -7e-6,...

```

```

'ppmb', 7e-6, 'gammaPdfA', 2.8, 'gammaPdfB', 4.5, 'scaleFactor', 1, ...
'Kab', 0.1, 'flipAngle', 20, 'TR', 2, 'verbose', false, ...
'FWHMRange', [], 'A', MultiPoolTofftsGammaVIF(), 'noiseLevel', 0, ...
'nAverages', 1, 'lb', 0, 'ub', 100, 'centers', [], 'fitOptions', ...
optimset('lsqcurvefit'), 'B0', 7, 'autoVIFNorm', false);
tmpNames = fieldnames(base);
for i = 1:numel(tmpNames)
    if ~isfield(Default, tmpNames{i})
        fprintf('WARNING! the field %s was passed in but does not match any of the default
names. This variable wont be used!\n', tmpNames{i})
    end
end
% Fill with defaults
tmpNames = fieldnames(Default);
for i = 1:numel(tmpNames)
    if ~isfield(base, tmpNames{i})
        base.(tmpNames{i}) = Default.(tmpNames{i});
    end
end
% calc FWHM range if needed
if isempty(base.FWHMRange) || isempty(base.centers)
    [I, ~, peakI] = FWHMRange(freqAxis, sum(abs(fftshift(fft(raw, [], 1), 1)), 2));
    base.FWHMRange = I;
    base.centers = peakI;
end
IFWHM = base.FWHMRange;
A = base.A;
bfit = @(t) padarray(gampdf(t, base.gammaPdfA, base.gammaPdfB), 1, 'post');
base.b = bfit;
Kab = base.Kab;
noiseLevel = base.noiseLevel;
nAverages = base.nAverages;
lb = base.lb;
ub = base.ub;
centers = base.centers;
verbose = base.verbose;
% Correct for VIF normilization if needed
if base.autoVIFNorm
    base.scaleFactor = 0.021614001850489*base.flipAngle+1.312872065860338e+03;
end

```

Fit data

```

fits = zeros(nAverages, length(fieldnames(fitParams)));
fitErr = zeros(nAverages, 1);
for j = 1:nAverages
    if(noiseLevel ~= 0)
        [noisyData, SNR] = ApplyNoise(raw, noiseLevel, freqAxis, ...
            IFWHM, centers);
        FTDData = fftshift(fft(noisyData, [], 1), 1);
    else

```



```

        FTData = fftshift(fft(raw,[],1),1);
        SNR = inf;
    end
    peakMax = zeros(size(centers));
    phases = zeros(size(centers));
    signals = zeros(size(raw,2),size(centers,2));
    PhaseData = zeros([size(FTData),length(centers)]);
    % Phase correct and FWHM integrate each peak
    %for m = 1:length(centers)
    for m = 1:length(centers)
        if(centers(m) == 0)
            continue
        end
        [~,peakMax(m)] = max(abs(FTData(centers(m),:))); % find the maximal peak location
        phases(m) = angle(FTData(centers(m),peakMax(m))); % find the phase at the above point
        for n = 1:length(t)
            % FWHM integrate the Phased signal
            PhaseData(:,n,m) = real(exp(-1i*(phases(m)))*FTData(:,n));
        end
        [signals(:,m)] = SignalIntegration(freqAxis, squeeze(PhaseData(:, :, m)), IFWHM(m, :));
    end
    nLac = sum(abs(signals(:,2)))/sum(sum(abs(signals)));
    flipAngles(1,:) = base.flipAngle*pi/180;%zeros(1,size(raw,2))+base.flipAngle*pi/180;
    flipAngles(2,:) = base.flipAngle*pi/180;%zeros(1,size(raw,2))+base.flipAngle*pi/180;

```

Model Results

```

tmpNames = fieldnames(fitParams);
fitConstants = base;
for i = 1:numel(tmpNames)
    if isfield(fitConstants,tmpNames{i})
        fitConstants = rmfield(fitConstants,tmpNames{i});
    end
end
params = struct('ExchangeTerms',[0,base.Kab;0,0],'T1s',[base.T1a,base.T1b],...
    'TRLList',t,'FaList',flipAngles,'PerfusionTerms',[base.kve,0],...
    'volumeFractions',[base.ve],'t0',[0;0],'gammaPdfA',[base.gammaPdfA;1],...
    'gammaPdfB',[base.gammaPdfB;1],'scaleFactor',[base.scaleFactor;0],...
    'fitOptions',base.fitOptions);
[fits(j,:),resultParams,allParams,resnorm(j),residual,exitflag,output,lambda,jacobian]...
    = A.fitData(params,fitParams,t,signals.',lb,ub);

end
if(verbose)
    % Display Model accuracy
    A.DataCompare(A,allParams,signals(1,:).',t,signals.')
    drawnow
end
end

```

Example Script for a Multiband Frequency Encode Snapshot

```

clear all
close all
clear classes
clear imports
clc
import Hypwright.*
% Init World
world = Hypwright.world.getWorld();
world.initWorld()
% init Variable
%Change These
FOV = 0.04;
nSpins = 4^2;
nBands = 2;
nProj = 40;
Tr = 0.1;
flipAngle = 10;
nSamp = 64;
xStart = 0.0;
xEnd = 0.01;
yStart = 0.0;
yEnd = 0.01;
protonResolution = 0.001;

%Maybe dont change these
gamma = 67.262e6;
sinoShift = -2;
deltaPPM = 15e-6;
singleBandwidth = gamma*world.B0(3)*deltaPPM;
maxSlope = 0.5*singleBandwidth/(gamma*FOV);
rewindTime = 0.001;
totalBW = singleBandwidth*nBands/(2*pi);
samplingTime = 1/totalBW*nSamp;
resolution = FOV/nSamp;
goldenAngle = pi*(3-sqrt(5));
projAngles = 0:goldenAngle:(nProj-1)*goldenAngle;
projAngles = mod(projAngles,pi);
xPos = linspace(xStart,xEnd,sqrt(nSpins));
yPos = linspace(yStart,yEnd,sqrt(nSpins));

% Build Phantom
for i = 1:length(xPos)
    for j = 1:length(yPos)
        Spin = TwoSiteExchangeGroup([0;0;1;0;0;0.5],...
            [],[],[],-7e-6,[],[],7e-6,gamma,[],0,[]);
        v = Voxel([xPos(i);yPos(j);0],Spin);
        world.addVoxel(v);
    end
end

```

```

end
% Build Pulse Sequence
FID = zeros(nProj,nSamp); %init memory for FID
readGrad = LinearGradientPulse.empty(nProj,0); %init memory for readGrad
PS = PulseSequence;
for i = 1:nProj
S = SincPulse(Tr*(i-1),5000,flipAngle*pi/180/gamma,gamma*3.0,[],sprintf('RF90%d',1));
PS.addPulse(S)
xSlope = cos(projAngles(i))*maxSlope;
ySlope = sin(projAngles(i))*maxSlope;
xRewind = -xSlope/2;
yRewind = -ySlope/2;
rewindGrad = LinearGradientPulse(S.endTime+rewindTime/2,rewindTime,[xRewind,yRewind,0]...
    ,sprintf('RewindGrad%d',0));
PS.addPulse(rewindGrad)
readGrad(i) =
LinearGradientPulse(rewindGrad.endTime+samplingTime/2,samplingTime,[xSlope,ySlope,0]...
    ,sprintf('ReadGrad%d',0));
PS.addPulse(readGrad(i))
end
world.setPulseSequence(PS)
% Calculate
disp('Calculation Time')
tic
world.calculate(readGrad(end).endTime);
toc
% Evaluate
for i = 1:nProj
tic
FID(i,:) = world.evaluate(linspace(readGrad(i).startTime,readGrad(i).endTime,nSamp),-gamma*3.0);
toc
end
% Recon Image
FTData = fftshift(fft(fftshift(FID,2),[],2),2);
pyrBand = circshift(FTData,sinoShift,2);
pyrBand = pyrBand(:,1:32);
lacBand = circshift(FTData,sinoShift+33,2);
lacBand = lacBand(:,1:32);
figure('name','Sinogram')
imagesc(abs(FTData));
figure('Position',[200,350,1250,500],'name','C13 Images')
im = iradon(abs(pyrBand).',projAngles*180/pi);
im = flipud(im);
xRes = linspace(-FOV/2,FOV/2,size(im,1));
yRes = linspace(-FOV/2,FOV/2,size(im,2));
subplot(1,2,1),imagesc(xRes,yRes,im);
set(gca,'YDir','reverse');
title('Pyruvate')
im = iradon(abs(lacBand).',projAngles*180/pi);
im = flipud(im);
subplot(1,2,2),imagesc(xRes,yRes,im);
title('Lactate')
% Make 1H image

```

```
[X,Y] = meshgrid(-FOV/2:protonResolution:FOV/2,-FOV/2:protonResolution:FOV/2);  
protonImage = zeros(size(X));  
I = X>=xStart&X<=xEnd&Y>=yStart&Y<=yEnd;  
protonImage(I) = 1;  
figure('name','Proton Image');  
imagesc(X(1,:),Y(:,1),protonImage);  
colormap gray
```

References

- 1 Weinberg, R. A. *The biology of cancer*. Second edition. (Garland Science, Taylor & Francis Group, 2014).
- 2 Hanahan, D. & Weinberg, R. A. Hallmarks of cancer: the next generation. *Cell* **144**, 646-674, doi:10.1016/j.cell.2011.02.013 (2011).
- 3 Gerlinger, M., Rowan, A. J., Horswell, S., Larkin, J., Endesfelder, D., Gronroos, E., Martinez, P., Matthews, N., Stewart, A., Tarpey, P., Varela, I., Phillimore, B., Begum, S., McDonald, N. Q., Butler, A., Jones, D., Raine, K., Latimer, C., Santos, C. R., Nohadani, M., Eklund, A. C., Spencer-Dene, B., Clark, G., Pickering, L., Stamp, G., Gore, M., Szallasi, Z., Downward, J., Futreal, P. A. & Swanton, C. Intratumor heterogeneity and branched evolution revealed by multiregion sequencing. *N Engl J Med* **366**, 883-892, doi:10.1056/NEJMoa1113205 (2012).
- 4 Siegel, R. L., Miller, K. D. & Jemal, A. Cancer statistics, 2016. *CA Cancer J Clin* **66**, 7-30, doi:10.3322/caac.21332 (2016).
- 5 Prescott, J. W. Quantitative imaging biomarkers: the application of advanced image processing and analysis to clinical and preclinical decision making. *J Digit Imaging* **26**, 97-108, doi:10.1007/s10278-012-9465-7 (2013).
- 6 Stephen, R. M. & Gillies, R. J. Promise and progress for functional and molecular imaging of response to targeted therapies. *Pharm Res* **24**, 1172-1185, doi:10.1007/s11095-007-9250-3 (2007).
- 7 Pantaleo, M. A., Nannini, M., Lopci, E., Castellucci, P., Maleddu, A., Lodi, F., Nanni, C., Allegri, V., Astorino, M., Brandi, G., Di Battista, M., Boschi, S., Fanti, S. & Biasco, G. Molecular imaging and targeted therapies in oncology: new concepts in treatment response assessment. a collection of cases. *Int J Oncol* **33**, 443-452 (2008).

- 8 Morse, D. L. & Gillies, R. J. Molecular imaging and targeted therapies. *Biochem Pharmacol* **80**, 731-738, doi:10.1016/j.bcp.2010.04.011 (2010).
- 9 Mammatas, L. H., Verheul, H. M., Hendrikse, N. H., Yaqub, M., Lammertsma, A. A. & Menke-van der Houven van Oordt, C. W. Molecular imaging of targeted therapies with positron emission tomography: the visualization of personalized cancer care. *Cell Oncol (Dordr)* **38**, 49-64, doi:10.1007/s13402-014-0194-4 (2015).
- 10 Kessler, L. G., Barnhart, H. X., Buckler, A. J., Choudhury, K. R., Kondratovich, M. V., Toledano, A., Guimaraes, A. R., Filice, R., Zhang, Z., Sullivan, D. C. & Group, Q. T. W. The emerging science of quantitative imaging biomarkers: terminology and definitions for scientific studies and regulatory submissions. *Stat Methods Med Res* **24**, 9-26, doi:10.1177/0962280214537333 (2015).
- 11 Lehninger, A. L., Nelson, D. L. & Cox, M. M. *Lehninger principles of biochemistry*. 6th edn, (W.H. Freeman, 2013).
- 12 Vander Heiden, M. G., Cantley, L. C. & Thompson, C. B. Understanding the Warburg effect: the metabolic requirements of cell proliferation. *Science* **324**, 1029-1033, doi:10.1126/science.1160809 (2009).
- 13 Warburg, O. On respiratory impairment in cancer cells. *Science* **124**, 269-270 (1956).
- 14 Hsu, P. P. & Sabatini, D. M. Cancer cell metabolism: Warburg and beyond. *Cell* **134**, 703-707, doi:10.1016/j.cell.2008.08.021 (2008).
- 15 DeBerardinis, R. J., Mancuso, A., Daikhin, E., Nissim, I., Yudkoff, M., Wehrli, S. & Thompson, C. B. Beyond aerobic glycolysis: transformed cells can engage in glutamine metabolism that exceeds the requirement for protein and nucleotide synthesis. *Proc Natl Acad Sci U S A* **104**, 19345-19350, doi:10.1073/pnas.0709747104 (2007).

- 16 Day, S. E., Kettunen, M. I., Gallagher, F. A., Hu, D. E., Lerche, M., Wolber, J., Golman, K., Ardenkjaer-Larsen, J. H. & Brindle, K. M. Detecting tumor response to treatment using hyperpolarized ^{13}C magnetic resonance imaging and spectroscopy. *Nat Med* **13**, 1382-1387, doi:10.1038/nm1650 (2007).
- 17 Witney, T. H., Kettunen, M. I., Day, S. E., Hu, D. E., Neves, A. A., Gallagher, F. A., Fulton, S. M. & Brindle, K. M. A comparison between radiolabeled fluorodeoxyglucose uptake and hyperpolarized (^{13}C) -labeled pyruvate utilization as methods for detecting tumor response to treatment. *Neoplasia* **11**, 574-582, 571 p following 582 (2009).
- 18 Juweid, M. E. & Cheson, B. D. Positron-emission tomography and assessment of cancer therapy. *N Engl J Med* **354**, 496-507, doi:10.1056/NEJMra050276 (2006).
- 19 Sandulache, V. C., Chen, Y., Lee, J., Rubinstein, A., Ramirez, M. S., Skinner, H. D., Walker, C. M., Williams, M. D., Taylor, R., Court, L. E., Bankson, J. A. & Lai, S. Y. Evaluation of hyperpolarized $[1-(^{13}\text{C})_3]$ -pyruvate by magnetic resonance to detect ionizing radiation effects in real time. *PLoS One* **9**, e87031, doi:10.1371/journal.pone.0087031 (2014).
- 20 Puzio-Kuter, A. M. The role of p53 in metabolic regulation. *Genes Cancer* **2**, 385-391, doi:10.1177/1947601911409738 (2011).
- 21 Chaumeil, M. M., Larson, P. E., Yoshihara, H. A., Danforth, O. M., Vigneron, D. B., Nelson, S. J., Pieper, R. O., Phillips, J. J. & Ronen, S. M. Non-invasive in vivo assessment of IDH1 mutational status in glioma. *Nat Commun* **4**, 2429, doi:10.1038/ncomms3429 (2013).
- 22 Zhang, H. The potential of hyperpolarized (^{13}C) MRI in assessing signaling pathways in cancer. *Acad Radiol* **21**, 215-222, doi:10.1016/j.acra.2013.11.015 (2014).
- 23 Galluzzi, L., Kepp, O., Vander Heiden, M. G. & Kroemer, G. Metabolic targets for cancer therapy. *Nat Rev Drug Discov* **12**, 829-846, doi:10.1038/nrd4145 (2013).

- 24 Halestrap, A. P. & Wilson, M. C. The monocarboxylate transporter family--role and regulation. *IUBMB Life* **64**, 109-119, doi:10.1002/iub.572 (2012).
- 25 Ardenkjaer-Larsen, J. H., Fridlund, B., Gram, A., Hansson, G., Hansson, L., Lerche, M. H., Servin, R., Thaning, M. & Golman, K. Increase in signal-to-noise ratio of > 10,000 times in liquid-state NMR. *Proc Natl Acad Sci U S A* **100**, 10158-10163, doi:10.1073/pnas.1733835100 (2003).
- 26 Kurhanewicz, J., Vigneron, D. B., Brindle, K., Chekmenev, E. Y., Comment, A., Cunningham, C. H., Deberardinis, R. J., Green, G. G., Leach, M. O., Rajan, S. S., Rizi, R. R., Ross, B. D., Warren, W. S. & Malloy, C. R. Analysis of cancer metabolism by imaging hyperpolarized nuclei: prospects for translation to clinical research. *Neoplasia* **13**, 81-97 (2011).
- 27 Golman, K., Ardenkjaer-Larsen, J. H., Petersson, J. S., Mansson, S. & Leunbach, I. Molecular imaging with endogenous substances. *Proc Natl Acad Sci U S A* **100**, 10435-10439, doi:10.1073/pnas.1733836100 (2003).
- 28 Nelson, S. J., Kurhanewicz, J., Vigneron, D. B., Larson, P. E., Harzstark, A. L., Ferrone, M., van Criekinge, M., Chang, J. W., Bok, R., Park, I., Reed, G., Carvajal, L., Small, E. J., Munster, P., Weinberg, V. K., Ardenkjaer-Larsen, J. H., Chen, A. P., Hurd, R. E., Odegardstuen, L. I., Robb, F. J., Tropp, J. & Murray, J. A. Metabolic imaging of patients with prostate cancer using hyperpolarized [1-(1)(3)C]pyruvate. *Sci Transl Med* **5**, 198ra108, doi:10.1126/scitranslmed.3006070 (2013).
- 29 Merritt, M. E., Harrison, C., Storey, C., Jeffrey, F. M., Sherry, A. D. & Malloy, C. R. Hyperpolarized ¹³C allows a direct measure of flux through a single enzyme-catalyzed step by NMR. *Proc Natl Acad Sci U S A* **104**, 19773-19777, doi:10.1073/pnas.0706235104 (2007).
- 30 Golman, K., Zandt, R. I., Lerche, M., Pehrson, R. & Ardenkjaer-Larsen, J. H. Metabolic imaging by hyperpolarized ¹³C magnetic resonance imaging for in vivo tumor diagnosis. *Cancer Res* **66**, 10855-10860, doi:10.1158/0008-5472.CAN-06-2564 (2006).

- 31 Haacke, E. M., Thompson, M.R., Venkatesan, R., Brown, R.W., & Cheng, Y.N. *Magnetic resonance imaging : physical principles and sequence design*. (Wiley, 1999).
- 32 Bloch, F. Nuclear induction. *Physical Review* **70**, 460-474, doi:Doi 10.1103/Physrev.70.460 (1946).
- 33 Griffiths, D. J. *Introduction to elementary particles*. (Harper & Row, 1987).
- 34 Griffiths, D. J. *Introduction to electrodynamics*. Fourth edition. edn, (Pearson, 2013).
- 35 Thornton, S. T. & Marion, J. B. *Classical dynamics of particles and systems*. 5th edn, (Brooks/Cole, 2004).
- 36 Griffiths, D. J. *Introduction to quantum mechanics*. 2nd edn, (Pearson Prentice Hall, 2005).
- 37 Abragam, A. *The principles of nuclear magnetism*. (Clarendon Press, 1961).
- 38 Abragam, A. & Goldman, M. Principles of dynamic nuclear-polarization. *Reports on Progress in Physics* **41**, 395-467, doi:Doi 10.1088/0034-4885/41/3/002 (1978).
- 39 Carver, T. R. & Slichter, C. P. Polarization of nuclear spins in metals. *Physical Review* **92**, 212-213, doi:DOI 10.1103/PhysRev.92.212.2 (1953).
- 40 Overhauser, A. W. Polarization of nuclei in metals. *Physical Review* **92**, 411-415, doi:DOI 10.1103/PhysRev.92.411 (1953).
- 41 Abragam, A. & Goldman, M. *Nuclear magnetism : order and disorder*. (Clarendon Press ; Oxford University Press, 1982).
- 42 Ardenkjaer-Larsen, J. H., Macholl, S. & Johannesson, H. Dynamic nuclear polarization with trityls at 1.2 K. *Applied Magnetic Resonance* **34**, 509-522, doi:10.1007/s00723-008-0134-4 (2008).
- 43 Wolber, J., Ellner, F., Fridlund, B., Gram, A., Johannesson, H., Hansson, G., Hansson, L. H., Lerche, M. H., Mansson, S., Servin, R., Thaning, M., Golman, K. & Ardenkjaer-Larsen, J. H. Generating highly polarized nuclear spins in solution using dynamic nuclear polarization. *Nuclear*

- Instruments & Methods in Physics Research Section a-Accelerators Spectrometers Detectors and Associated Equipment* **526**, 173-181, doi:10.1016/j.nima.2004.03.171 (2004).
- 44 Wenkebach, W. T. The solid effect. *Applied Magnetic Resonance* **34**, 227-235, doi:10.1007/s00723-008-0121-9 (2008).
- 45 Hovav, Y., Feintuch, A. & Vega, S. Theoretical aspects of dynamic nuclear polarization in the solid state - the solid effect. *J Magn Reson* **207**, 176-189, doi:10.1016/j.jmr.2010.10.016 (2010).
- 46 Goldman, M. Overview of spin temperature, thermal mixing and dynamic nuclear polarization. *Applied Magnetic Resonance* **34**, 219-226, doi:10.1007/s00723-008-0114-8 (2008).
- 47 Hovav, Y., Feintuch, A. & Vega, S. Theoretical aspects of dynamic nuclear polarization in the solid state--spin temperature and thermal mixing. *Phys Chem Chem Phys* **15**, 188-203, doi:10.1039/c2cp42897k (2013).
- 48 Zhang, W. X., Hu, J. L., Zhuang, J., You, J. Q. & Liu, R. B. Protection of center-spin coherence by a dynamically polarized nuclear spin core. *Physical Review B* **82**, doi:ARTN 045314 10.1103/PhysRevB.82.045314 (2010).
- 49 Redfield, A. G. Nuclear magnetic resonance saturation and rotary saturation in solids. *Physical Review* **98**, 1787-1809, doi:DOI 10.1103/PhysRev.98.1787 (1955).
- 50 Provotorov, B. N. Nuclear magnetic resonance in solids. *Optika I Spektroskopiya* **11**, 123-125 (1961).
- 51 Lumata, L., Merritt, M. E., Malloy, C. R., Sherry, A. D. & Kovacs, Z. Impact of Gd³⁺ on DNP of [1-C-13]pyruvatedoped with trityl OX063, BDPA, or 4-Oxo-TEMPO. *Journal of Physical Chemistry A* **116**, 5129-5138, doi:10.1021/jp302399f (2012).
- 52 Johanneson, H., Macholl, S. & Ardenkjaer-Larsen, J. H. Dynamic nuclear polarization of [1-C-13]pyruvic acid at 4.6 tesla. *Journal of Magnetic Resonance* **197**, 167-175, doi:10.1016/j.jmr.2008.12.016 (2009).

- 53 Bankson, J. A., Walker, C. M., Ramirez, M. S., Stefan, W., Fuentes, D., Merritt, M. E., Lee, J., Sandulache, V. C., Chen, Y., Phan, L., Chou, P. C., Rao, A., Yeung, S. C., Lee, M. H., Schellingerhout, D., Conrad, C. A., Malloy, C., Sherry, A. D., Lai, S. Y. & Hazle, J. D. Kinetic modeling and constrained reconstruction of hyperpolarized [1-¹³C]-pyruvate offers improved metabolic imaging of tumors. *Cancer Res* **75**, 4708-4717, doi:10.1158/0008-5472.CAN-15-0171 (2015).
- 54 Harris, T., Eliyahu, G., Frydman, L. & Degani, H. Kinetics of hyperpolarized ¹³C1-pyruvate transport and metabolism in living human breast cancer cells. *Proc Natl Acad Sci U S A* **106**, 18131-18136, doi:10.1073/pnas.0909049106 (2009).
- 55 Li, L. Z., Kadlenceck, S., Xu, H. N., Daye, D., Pullinger, B., Profka, H., Chodosh, L. & Rizi, R. Ratiometric analysis in hyperpolarized NMR (I): test of the two-site exchange model and the quantification of reaction rate constants. *NMR Biomed* **26**, 1308-1320, doi:10.1002/nbm.2953 (2013).
- 56 Bahrami, N., Swisher, C. L., Von Morze, C., Vigneron, D. B. & Larson, P. E. Kinetic and perfusion modeling of hyperpolarized (¹³C) pyruvate and urea in cancer with arbitrary RF flip angles. *Quant Imaging Med Surg* **4**, 24-32, doi:10.3978/j.issn.2223-4292.2014.02.02 (2014).
- 57 Witney, T. H., Kettunen, M. I. & Brindle, K. M. Kinetic modeling of hyperpolarized ¹³C label exchange between pyruvate and lactate in tumor cells. *J Biol Chem* **286**, 24572-24580, doi:10.1074/jbc.M111.237727 (2011).
- 58 Harrison, C., Yang, C., Jindal, A., DeBerardinis, R. J., Hooshyar, M. A., Merritt, M., Sherry, D.A. & Malloy, C. R. Comparison of kinetic models for analysis of pyruvate-to-lactate exchange by hyperpolarized ¹³C NMR. *NMR Biomed* **25**, 1286-1294, doi:10.1002/nbm.2801 (2012).

- 59 Pages, G. & Kuchel, P. W. Mathematical modeling and data analysis of NMR experiments using hyperpolarized (13)C metabolites. *Magn Reson Insights* **6**, 13-21, doi:10.4137/MRI.S11084 (2013).
- 60 Zierhut, M. L., Yen, Y. F., Chen, A. P., Bok, R., Albers, M. J., Zhang, V., Tropp, J., Park, I., Vigneron, D. B., Kurhanewicz, J., Hurd, R. E. & Nelson, S. J. Kinetic modeling of hyperpolarized 13C1-pyruvate metabolism in normal rats and TRAMP mice. *J Magn Reson* **202**, 85-92, doi:10.1016/j.jmr.2009.10.003 (2010).
- 61 Xing, Y., Reed, G. D., Pauly, J. M., Kerr, A. B. & Larson, P. E. Optimal variable flip angle schemes for dynamic acquisition of exchanging hyperpolarized substrates. *J Magn Reson* **234**, 75-81, doi:10.1016/j.jmr.2013.06.003 (2013).
- 62 Walker, C. M., Chen, Y., Lai, S. Y. & Bankson, J. A. A novel perfused Bloch-McConnell simulator for analyzing the accuracy of dynamic hyperpolarized MRS. *Med Phys* **43**, 854, doi:10.1118/1.4939877 (2016).
- 63 McConnell, H. M. Reaction rates by nuclear magnetic resonance. *Journal of Chemical Physics* **28**, 430-431, doi:Doi 10.1063/1.1744152 (1958).
- 64 Tofts, P. S. Modeling tracer kinetics in dynamic Gd-DTPA MR imaging. *J Magn Reson Imaging* **7**, 91-101 (1997).
- 65 Tofts, P. S., Brix, G., Buckley, D. L., Evelhoch, J. L., Henderson, E., Knopp, M. V., Larsson, H. B., Lee, T. Y., Mayr, N. A., Parker, G. J., Port, R. E., Taylor, J. & Weisskoff, R. M. Estimating kinetic parameters from dynamic contrast-enhanced T(1)-weighted MRI of a diffusable tracer: standardized quantities and symbols. *J Magn Reson Imaging* **10**, 223-232 (1999).
- 66 Segel, I. H. *Enzyme kinetics : behavior and analysis of rapid equilibrium and steady state enzyme systems*. (Wiley, 1975).

- 67 Duffield, R. B. & Calvin, M. The stability of chelate compounds .3. Exchange reactions of copper chelate compounds. *Journal of the American Chemical Society* **68**, 557-561, doi:DOI 10.1021/ja01208a007 (1946).
- 68 Walker, C. M., Lee, J., Ramirez, M. S., Schellingerhout, D., Millward, S. & Bankson, J. A. A catalyzing phantom for reproducible dynamic conversion of hyperpolarized [1-(1)(3)C]-pyruvate. *PLoS One* **8**, e71274, doi:10.1371/journal.pone.0071274 (2013).
- 69 Kettunen, M. I., Hu, D. E., Witney, T. H., McLaughlin, R., Gallagher, F. A., Bohndiek, S. E., Day, S. E. & Brindle, K. M. Magnetization transfer measurements of exchange between hyperpolarized [1-13C]pyruvate and [1-13C]lactate in a murine lymphoma. *Magn Reson Med* **63**, 872-880, doi:10.1002/mrm.22276 (2010).
- 70 Press, W. H., Teukolsky, S.A., Vetterling, W.T., & Flannery, B.P. *Numerical recipes in C++ : the art of scientific computing*. 2nd edn, (Cambridge University Press, 2002).
- 71 Davenport, R. The derivation of the gamma-variate relationship for tracer dilution curves. *J Nucl Med* **24**, 945-948 (1983).
- 72 Gamma, E. *Design patterns : elements of reusable object-oriented software*. (Addison-Wesley, 1995).
- 73 Walker, C. M., Merritt, M., Wang, J. X. & Bankson, J. A. Use of a multi-compartment dynamic single enzyme phantom for studies of hyperpolarized magnetic resonance agents. *J Vis Exp*, doi:10.3791/53607 (2016).
- 74 Kazan, S. M., Reynolds, S., Kennerley, A., Wholey, E., Bluff, J. E., Berwick, J., Cunningham, V. J., Paley, M. N. & Tozer, G. M. Kinetic modeling of hyperpolarized (13)C pyruvate metabolism in tumors using a measured arterial input function. *Magn Reson Med* **70**, 943-953, doi:10.1002/mrm.24546 (2013).

- 75 Ramirez, M. S., Lee, J., Walker, C. M., Sandulache, V. C., Hennel, F., Lai, S. Y. & Bankson, J. A. Radial spectroscopic MRI of hyperpolarized [1-(13) C] pyruvate at 7 tesla. *Magn Reson Med* **72**, 986-995, doi:10.1002/mrm.25004 (2014).
- 76 Fletcher, J. W., Logan, T. F., Eitel, J. A., Mathias, C. J., Ng, Y., Lacy, J. L., Hutchins, G. D. & Green, M. A. Whole-body PET/CT evaluation of tumor perfusion using generator-based ⁶²Cu-ethylglyoxal bis(thiosemicarbazonato)copper(II): validation by direct comparison to ¹⁵O-water in metastatic renal cell carcinoma. *J Nucl Med* **56**, 56-62, doi:10.2967/jnumed.114.148106 (2015).
- 77 Bergmeyer, H.U., & Gawehn, K. *Methods of enzymatic analysis*. 3 edn, Vol. 3 (Verlag Chemie, 1983).
- 78 Yagil, G. & Hoberman, H. D. Rate of isotope exchange in enzyme-catalyzed reactions. *Biochemistry* **8**, 352-360 (1969).
- 79 Zewe, V. & Fromm, H. J. Kinetic studies of rabbit muscle lactate dehydrogenase. *J Biol Chem* **237**, 1668-1675 (1962).
- 80 Zewe, V. & Fromm, H. J. Kinetic studies of rabbit muscle lactate dehydrogenase. II. Mechanism of the reaction. *Biochemistry* **4**, 782-792 (1965).
- 81 Sandulache, V. C., Skinner, H. D., Wang, Y., Chen, Y., Dodge, C. T., Ow, T. J., Bankson, J. A., Myers, J. N. & Lai, S. Y. Glycolytic inhibition alters anaplastic thyroid carcinoma tumor metabolism and improves response to conventional chemotherapy and radiation. *Mol Cancer Ther* **11**, 1373-1380, doi:10.1158/1535-7163.MCT-12-0041 (2012).
- 82 Ramirez, M., Lee, J. & Bankson, J. in *Innovations in Cancer Prevention and Research Conference. Austin Texas 2012* (Cancer Prevention and Research Institute of Texas).

- 83 Zhao, L., Mulkern, R., Tseng, C. H., Williamson, D., Patz, S., Kraft, R., Walsworth, R. L., Jolesz, F. A. & Albert, M. S. Gradient-echo imaging considerations for hyperpolarized ^{129}Xe MR. *J Magn Reson B* **113**, 179-183 (1996).
- 84 Nucera, C., Nehs, M. A., Mekel, M., Zhang, X., Hodin, R., Lawler, J., Nose, V. & Parangi, S. A novel orthotopic mouse model of human anaplastic thyroid carcinoma. *Thyroid* **19**, 1077-1084, doi:10.1089/thy.2009.0055 (2009).
- 85 Albers, M. J., Bok, R., Chen, A. P., Cunningham, C. H., Zierhut, M. L., Zhang, V. Y., Kohler, S. J., Tropp, J., Hurd, R. E., Yen, Y. F., Nelson, S. J., Vigneron, D. B. & Kurhanewicz, J. Hyperpolarized ^{13}C lactate, pyruvate, and alanine: noninvasive biomarkers for prostate cancer detection and grading. *Cancer Res* **68**, 8607-8615, doi:10.1158/0008-5472.CAN-08-0749 (2008).
- 86 Laustsen, C., Ostergaard, J. A., Lauritzen, M. H., Norregaard, R., Bowen, S., Sogaard, L. V., Flyvbjerg, A., Pedersen, M. & Ardenkjaer-Larsen, J. H. Assessment of early diabetic renal changes with hyperpolarized $[1-(^{13}\text{C})\text{pyruvate}]$. *Diabetes Metab Res Rev* **29**, 125-129 (2013).
- 87 Schroeder, M. A., Lau, A. Z., Chen, A. P., Gu, Y., Nagendran, J., Barry, J., Hu, X., Dyck, J. R., Tyler, D. J., Clarke, K., Connelly, K. A., Wright, G. A. & Cunningham, C. H. Hyperpolarized (^{13}C) magnetic resonance reveals early- and late-onset changes to in vivo pyruvate metabolism in the failing heart. *Eur J Heart Fail* **15**, 130-140 (2013).
- 88 Thind, K., Chen, A., Friesen-Waldner, L., Ouriadov, A., Scholl, T. J., Fox, M., Wong, E., Vandyk, J., Hope, A. & Santyr, G. Detection of radiation-induced lung injury using hyperpolarized (^{13}C) magnetic resonance spectroscopy and imaging. *Magn Reson Med* **16**, 24525 (2012).
- 89 Lodi, A., Woods, S. M. & Ronen, S. M. Treatment with the MEK inhibitor U0126 induces decreased hyperpolarized pyruvate to lactate conversion in breast, but not prostate, cancer cells. *NMR Biomed* **26**, 299-306 (2013).

- 90 Clatworthy, M. R., Kettunen, M. I., Hu, D. E., Mathews, R. J., Witney, T. H., Kennedy, B. W., Bohndiek, S. E., Gallagher, F. A., Jarvis, L. B., Smith, K. G. & Brindle, K. M. Magnetic resonance imaging with hyperpolarized [1,4-(13)C2]fumarate allows detection of early renal acute tubular necrosis. *Proc Natl Acad Sci U S A* **109**, 13374-13379 (2012).
- 91 Bohndiek, S. E., Kettunen, M. I., Hu, D. E. & Brindle, K. M. Hyperpolarized (13)C spectroscopy detects early changes in tumor vasculature and metabolism after VEGF neutralization. *Cancer Res* **72**, 854-864 (2012).
- 92 Park, I., Bok, R., Ozawa, T., Phillips, J. J., James, C. D., Vigneron, D. B., Ronen, S. M. & Nelson, S. J. Detection of early response to temozolomide treatment in brain tumors using hyperpolarized 13C MR metabolic imaging. *J Magn Reson Imaging* **33**, 1284-1290 (2011).
- 93 Buckler, A. J., Bresolin, L., Dunnick, N. R., Sullivan, D. C., Aerts, H. J., Bendriem, B., Bendtsen, C., Boellaard, R., Boone, J. M., Cole, P. E., Conklin, J. J., Dorfman, G. S., Douglas, P. S., Eidsaunet, W., Elsinger, C., Frank, R. A., Gatsonis, C., Giger, M. L., Gupta, S. N., Gustafson, D., Hoekstra, O. S., Jackson, E. F., Karam, L., Kelloff, G. J., Kinahan, P. E., McLennan, G., Miller, C. G., Mozley, P. D., Muller, K. E., Patt, R., Raunig, D., Rosen, M., Rupani, H., Schwartz, L. H., Siegel, B. A., Sorensen, A. G., Wahl, R. L., Waterton, J. C., Wolf, W., Zahlmann, G. & Zimmerman, B. Quantitative imaging test approval and biomarker qualification: interrelated but distinct activities. *Radiology* **259**, 875-884, doi:10.1148/radiol.10100800 (2011).
- 94 von Morze, C., Bok, R. A., Reed, G. D., Ardenkjaer-Larsen, J. H., Kurhanewicz, J. & Vigneron, D. B. Simultaneous multiagent hyperpolarized (13)C perfusion imaging. *Magn Reson Med* **72**, 1599-1609, doi:10.1002/mrm.25071 (2014).
- 95 Yen, Y. F., Kohler, S. J., Chen, A. P., Tropp, J., Bok, R., Wolber, J., Albers, M. J., Gram, K. A., Zierhut, M. L., Park, I., Zhang, V., Hu, S., Nelson, S. J., Vigneron, D. B., Kurhanewicz, J., Dirven, H.

- A. & Hurd, R. E. Imaging considerations for in vivo ^{13}C metabolic mapping using hyperpolarized ^{13}C -pyruvate. *Magn Reson Med* **62**, 1-10, doi:10.1002/mrm.21987 (2009).
- 96 Gordon, J. W., Vigneron, D. B. & Larson, P. E. Development of a symmetric echo planar imaging framework for clinical translation of rapid dynamic hyperpolarized ^{13}C imaging. *Magn Reson Med*, doi:10.1002/mrm.26123 (2016).
- 97 Jiang, W., Lustig, M. & Larson, P. E. Concentric rings K-space trajectory for hyperpolarized (^{13}C) MR spectroscopic imaging. *Magn Reson Med* **75**, 19-31, doi:10.1002/mrm.25577 (2016).
- 98 Hu, S., Lustig, M., Chen, A. P., Crane, J., Kerr, A., Kelley, D. A., Hurd, R., Kurhanewicz, J., Nelson, S. J., Pauly, J. M. & Vigneron, D. B. Compressed sensing for resolution enhancement of hyperpolarized ^{13}C flyback 3D-MRSI. *J Magn Reson* **192**, 258-264, doi:10.1016/j.jmr.2008.03.003 (2008).
- 99 Brindle, K. M., Bohndiek, S. E., Gallagher, F. A. & Kettunen, M. I. Tumor imaging using hyperpolarized ^{13}C magnetic resonance spectroscopy. *Magn Reson Med* **66**, 505-519, doi:10.1002/mrm.22999 (2011).
- 100 Cunningham, C. H., Chen, A. P., Albers, M. J., Kurhanewicz, J., Hurd, R. E., Yen, Y. F., Pauly, J. M., Nelson, S. J. & Vigneron, D. B. Double spin-echo sequence for rapid spectroscopic imaging of hyperpolarized ^{13}C . *J Magn Reson* **187**, 357-362, doi:10.1016/j.jmr.2007.05.014 (2007).
- 101 Wiesinger, F., Weidl, E., Menzel, M. I., Janich, M. A., Khagai, O., Glaser, S. J., Haase, A., Schwaiger, M. & Schulte, R. F. IDEAL spiral CSI for dynamic metabolic MR imaging of hyperpolarized $[1-^{13}\text{C}]$ pyruvate. *Magn Reson Med* **68**, 8-16, doi:10.1002/mrm.23212 (2012).
- 102 Lee, Y., Zacharias, N. M., Piwnica-Worms, D. & Bhattacharya, P. K. Chemical reaction-induced multi-molecular polarization (CRIMP). *Chem Commun (Camb)* **50**, 13030-13033, doi:10.1039/c4cc06199c (2014).

- 103 Theorell, H. & Bonnichsen, R. Studies on liver alcohol dehydrogenase .1. Equilibria and initial reaction velocities. *Acta Chem Scand* **5**, 1105-1126, doi:DOI 10.3891/acta.chem.scand.05-1105 (1951).
- 104 Fromm, H. J. & Nelson, D. R. Ribitol dehydrogenase .3. Kinetic studies with product inhibition. *Journal of Biological Chemistry* **237**, 215-& (1962).
- 105 King, E. L. & Altman, C. A Schematic method of deriving the ratelaws for enzyme-catalyzed reactions. *J Phys Chem-US* **60**, 1375-1378, doi:DOI 10.1021/j150544a010 (1956).

Vita

Christopher Michael Walker Was Born on August 22nd 1987 the son of Mike and Cheryl Walker in Raleigh, North Carolina. He graduated from Westlake High School in 2005 in Austin, Texas and matriculated into Trinity University in San Antonio, Texas. He graduated from Trinity University in 2010 with a bachelor's of science in physics with computation as a second major. He began his graduate studies The University of Texas Graduate School of Biomedical Sciences at Houston in Houston, Texas September of 2010 and conducted his research in the magnetic resonance system and instrumentation laboratory of the university of Texas M.D. Anderson Cancer Center.

Permanent address:

5920 Cape Coral Dr.

Austin, TX 78746