OPTIMIZATION OF HARDWARE AND SOFTWARE FOR SOLID STATE NUCLEAR MAGNETIC RESONANCE AT HIGH MAGNETIC FIELDS

By

SETH ALAN MCNEILL

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2009

To God for giving me my talents
To my parents for growing my talents
To my fiancée for supporting my talents

ACKNOWLEDGMENTS

I take this opportunity to express sincere gratitude to my advisors, Dr. Joanna R. Long and Dr. A. Antonio Arroyo for providing me with a unique interdisciplinary academic situation for acquiring this degree. Dr. Long's uncanny knowledge of nearly everything is both disturbing and inspiring to say the least, and Dr. Arroyo's mentoring has made an indelible impact on me throughout my tenure at UF. It was not always easy to coordinate between schools, but these great individuals have always managed to work things out. I am also thankful to my other committee members, Dr. Haniph A. Latchman, Dr. J. Cole Smith, and Dr. Eric Schwartz for being very supportive over the years.

My research was not only financially supported in part by the National High Magnetic Field Laboratory in Tallahassee, FL, but many individuals there contributed greatly to my research. In particular Dr. Peter L. Gor'kov, who makes the most beautiful NMR probes in the world, has taught me some excellent techniques. His ability to design and his desire for perfection were a great inspiration to me. Warm thanks are extended to Dr. William W. Brey who provided much training and insight into RF testing and with whom I had many good discussions of science and engineering. Special thanks to Kiran Shetty, Jason Kitchen and Ashley Blue for providing tips on construction and help for finding the parts that I needed, and also for helping with NMR experiments in Tallahassee.

Here in Gainesville, I thank the AMRIS staff for keeping the spectrometers going and providing a place for my research. I thank Dr. Daniel Plant in particular for trusting me to reconfigure the magnets every time I get on, and for helping me get good data from a marginally installed system. I also thank James R. Rocca for always taking the time to teach me when I had questions.

Throughout my career in the McKnight Brain Institute, the Computer and Information Technology Services guys have always been helpful, even if the service request was ludicrous. The crew at UF's High-Performance Computing Center was always helpful whenever I did calculations over there, especially Jon Akers who managed to provide help both at the MBI and HPC. A great deal of thanks to Shannon Chillingworth, the ECE Academic Services Coordinator, who did an amazing job keeping everything straight for me from my late application through many winding paths until I have reached this point. In the Biochemistry and Molecular Biology office, I thank Regina Corns for managing to keep me paid and my name in all the right spaces for everything to work out even though things (including me) were not always easy to work with or simple. Thanks also go to Denise Mesa and Bradley Moore for doing a wonderful job of keeping all the odds and ends of me being here in line. The combined Long and Edison labs have provided a very stimulating and wonderfully distracting environment for me to work in throughout the years.

My church family at the Gainesville Seventh-day Adventist church has provided a tremendous nurturing environment during my tenure here. The people there have taken a keen interest in me and my academic journey, which has kept me on the straight and narrow. Pastor Dan Graham has been a great pastor, mentor, friend, and fellow struggler on the path to a PhD. Dr. OJ Ganesh has provided much insight into science, computing, and a host of other things since I see him at lab and church and social functions! Hopefully Katia (his wife) will forgive me for having seen more of him than she did during his tenure in the lab.

Thanks go to Dr. Joel Schipper and soon to be Dr. Lavi Zamstein for providing the best rooming situation at the best price in Gainesville for 3 years. It has been a great journey as

Arroyo's trio of PhD students. Thanks also go to Dr. Brian Roth who commiserated with and encouraged me from the other side of the country as we pursued our degrees.

Deep and heartfelt thanks go to my family who have been very encouraging and helpful throughout my life. My parents instilled curiosity and a belief that I could accomplish anything and go anywhere, which when mediated by a strong belief in God has led me to all kinds of interesting places, including here. Two of my brothers and I constantly commiserate as we are all pursuing PhDs. Our oldest brother and his family provide stability and encouragement from the real world, as well as fun discussions and great nieces. My future in-laws have adopted me in early and been extremely supportive. In particular, "stern" words of encouragement my fiancée's mom has helped to keep me going.

My wonderful and beautiful fiancée, Corraine, has been through quite a bit with this degree and is definitely looking forward to its completion. She has really kept me on track, focused on what needed doing, and encouraged me when the going got tough. Thank you so very much!

Most of all, I thank God for giving me life, purpose, and passion throughout everything.

TABLE OF CONTENTS

page

ACKNOWLEDGMENTS 4

LIST OF TABLES 11

LIST OF FIGURES 12

ABSTRACT 15

CHAPTER

1 INTRODUCTION 17

Nuclear Magnetic Resonance (NMR) 17
Solid State NMR 20
Challenges in NMR 21
Sensitivity 21
Stronger static magnetic fields 21
Concentrated samples 23
Smaller detection coils 23
Off-Resonance 23
Inhomogeneity 24
Biological Samples 25
Goals of This Research 26

2 HARDWARE OPTIMIZATION 30

Improvements and Challenges at Higher Fields 30
The Challenge of Radio Frequency Coils at High Frequencies 31
The Low-E Solution 34
750 MHz Probe Assembly 35
Sample Coil Assembly and Integration into a MAS Stator 35
RF Matching Network 37
Probe Body Construction 37
Testing and Specifications 39
Power Efficiency and Homogeneity of RF Fields 39
Power Handling and Stability 41
Shimming 42
RF-Induced Heating 43
Frictional Heating 46
Sensitivity Measurements 46
Cross Polarization Measurements 48
Performance with a Biologically Relevant Sample 49
Performance Using Multipulse Windowless RF Sequences 49

3 RF EXCITATION PROFILES BACKGROUND AND PRIOR ART FOR OPTIMIZATION 74

Introduction 74
Mathematical Representations of NMR Experiments 76
Pauli Method 77
3D Vector Method 80
Off-Resonance and Non-Ideal Rotations 81
Solids 81
Simulators 82
Minimization 86
Clustered Computing 88
Liquids Examples of Compensation by Composite Pulses 89
Solids Examples 91

4 OPTIMIZATION OF RF EXCITATION PROFILES 109

Optimizing Inversion Pulse Sequences 109
Introduction 109
Methods 111
Results 114
Optimizing Refocusing Pulse Sequences 115
Introduction 115
Methods 117
Results 118
Optimizing the DRAWS Pulse Sequence 120
Introduction 120
Methods 121
Results 122

5 CONCLUSIONS AND FUTURE DIRECTIONS 150

Conclusions 150
Future Directions 151
Probe Development 151
Pulse Sequence Optimization 152

APPENDIX

A DATA PROCESSING TECHNIQUES 155

Visualization Aids 155
Nutation Rate Visualization Script 155
Pauli Definitions Figure Script 156
Pulse Path Calculation and Visualization 158
Pulse path calculation 158
Pulse path visualization 159

Trajectory Endpoint Visualization 160
Data Preparation 161
Long2ndnmr.m 161
Bruker_data.m 162
Simulating Experiments with SPINEVOLUTION 165
inv.par 165
adam.ss 166
load_par.m 166
load_ss.m 167
Matlab settings structure 168
write_spinev_inv.m 168
spinev_eval_inv_dly.m 170
write_spinev_pp.m 171
Simulating Experiments and Comparing to Experimental Data 172
Settings for 38-111 simulation 174
Fit over frequency offset and power offset 175
Liquids and Solids Simulation Comparison 176
Refocusing Power Level Comparison Code 178
Refocusing MAS Rate Comparison Code 179
Refocusing Experimental Comparison to Simulation 181
DRAWS Experimental to Simulation Comparison 183
Matlab Script 183
Parameter File 185
CP Simulation Code 185
Static Square CP 185
Spin system 185
SIMPSON input file 186
Parameter file 187
Static Ramped CP 188
SIMPSON input file 188
Parameter file 189
MAS Square CP 190
Spin system file 190
SIMPSON input file 190
Parameter file 191
MAS Ramped CP 192
SIMPSON input file 192
Parameter file 193

B OPTIMIZATION SCRIPTS 195

Inversion Optimization 195
create_run.sh 195
Typical Inversion Optimization Parameter File 197
Compiling Matlab 197
Inversion Optimization Matlab Script 198
Maximum Filename Check 200

Minimum Filename Check 201
Refocusing Optimization 201
Refocusing Matlab Script 201
Refocusing Parameter File 203

C PULSE SEQUENCES 205

Nutation Figure 2-9 205
zg_2d.sam 205
zgig_2d.sam 205
TmDOTP Heating Pulse Sequence Figure 2-11 zgps_p18.sam 206
Inversion Pulse Sequences Figure 4-11 207
inv_ctl.sam 207
inv_38111_wctl.sam 208
inv0.911.sam 210
Refocusing Pulse Sequences Figures 4-20 to 4-22 211
refc_180.sam 211
refc_151342.sam 213
ref2w0.8262.sam 214
DRAWS Pulse Sequences Figure 4-28 216
draws2d_uneqCP.2.1.b.sam 216
spinev2d_755a.2.1.b.sam 219

D FITTING PHI AND PSI 223

Running the Simulations 223
run_sims.sh 223
chimap_clust.m 224
chimap_mfile_compiled.m 226
chimap_cleanup_compiled.m 227
write_chimap_qsub_script_compiled.m 228
GGV TP File 229
Miscellaneous Functions 233
2Dfittingnotes.m 233
do_fits_script.m 233
dqDRAWSfitsims.m 235

LIST OF REFERENCES 237

BIOGRAPHICAL SKETCH 243

LIST OF TABLES

Table page

2-1. Calculation of heat dissipation rates, q_{heat}, per kHz^2 or RF field. 51

LIST OF FIGURES

Figure page

1-1. NMR through space measurement. 27

1-2. NMR through bond measurement. 27

1-3. Comparison of a liquid signal to a powder pattern via simulation. 28

1-4. High power versus low power excitation. 29

2-1. Spectra of ~25-30 mg of *G*GV peptide spinning at 5 kHz MAS at 500 MHz and 750 MHz. 52

2-2. A double quantum recoupling experiment (DRAWS) simulated performance as a function of ^{13}C RF homogeneity at 750 MHz. 53

2-3. Relative sensitivity as a function of rotor diameter. 54

2-4. The dimensions of the coil assembly are in millimeters. 55

2-5. The probe schematic. 56

2-6. Pictures from assembling the probe. 57

2-7. Labeled picture of the probe head. 59

2-8. Homogeneity plot for both channels. 60

2-9. Nutation profiles for both channels. 61

2-10. Probe arcing. 62

2-11. Example TmDOTP peaks with increasing RF input. 63

2-12. RF heating. 64

2-13. MAS heating spectra. 65

2-14. MAS heating. 66

2-15. Glycine spectrum for measuring S/N. 67

2-16. Signal-to-noise for various sample lengths. 68

2-17. CPMAS matching condition profile for adamantane at 10 kHz MAS using a square spinlock pulse on both RF channels. 69

2-18. The ^{13}C CPMAS spectrum of nanocrystalline natural abundance lysozyme demonstrates the probe's capabilities on a crystalline sample. 70

2-19. A 2D DQ-CSA spectrum of *G*AV demonstrates a dipolar recoupling experiment. 71

2-20. An example showing the improvements in signal quality in going to higher field. 72

3-1. On resonance rotation. 96

3-2. Off resonance rotation. 96

3-3. Pauli angle definitions. 97

3-4. Vector rotation definitions. 98

3-5. Orientation dependence of solid state NMR. 99

3-6. Solid state powder pattern. 100

3-7. Spectrometer in the loop optimization. 101

3-8. Simulation based optimization. 102

3-9. Minimization illustration. 103

3-10. The 5 steps that the Nelder-Mead simplex method can take are shown for N = 2. 104

3-11. Off resonance trajectories for a 180 degree pulse. 105

3-12. Endpoints of off resonance 180 degree pulses. 106

3-13. Trajectories for 90-180-90 composite pulse. 107

3-14. Endpoints for 90-180-90 composite pulse. 108

4-1. Inversion pulse sequence. 124

4-2. Inversion trajectories for 180 degree pulse. 125

4-3. Endpoints for 180 degree pulse with varying frequecy offset. 126

4-4. Simulations using SPINEVOLUTION of various liquids composite inversion pulses simulating an adamantane-like solid spinning at 10 kHz. 127

4-5. Finding a useful recycle delay. 128

4-6. Comparison of experimental and simulation data (SPINEVOLUTION) for a single 180° inversion pulse. 129

4-7. Composite inversion 180° pulse showing the lengths and phases of the pulses. 130

4-8. Comparison of experimental and simulation data for a composite inversion pulse. 131

4-9. Optimization undersampling. 132

4-10. An optimized composite inversion 180° pulse showing the lengths and phases of the pulses. 133

4-11. Comparison of various inversion composit pulses. 134

4-12. Inversion homogeneity match to simulation. 135

4-13. Refocusing pulse sequence. 136

4-14. Refocusing explaination. 136

4-15. Refocusing comparison between liquids and solids. 137

4-16. Solids refocusing power response. 138

4-17. Solids refocusing MAS rate response. 139

4-18. A typical spinev input file for simulating a refocusing experiment. 140

4-19. spinev input pp file and commandline call. 141

4-20. Simulation and experiment for a single 180 refocusing pulse. 142

4-21. The refocusing data and simulation for the composite pulse by Bai et al. 143

4-22. The simulation and experimental data optimized in this research. 144

4-23. A comparison of all three refocusing pulses. The bandwidth of both composite pulses is better than that of a single 180 145

4-24. Homogeneity and refocusing pulse simulations. 146

4-25. The DRAWS pulse sequence. 147

4-26. The original DRAWS R group pulse lengths and phases 148

4-27. The optimized DRAWS R group pulse lengths and phases. 148

4-28. DRAWS experimental and simulation data. 149

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

OPTIMIZATION OF HARDWARE AND SOFTWARE FOR SOLID STATE NUCLEAR MAGNETIC RESONANCE AT HIGH MAGNETIC FIELDS

By

Seth Alan McNeill

May 2009

Chair: A. Antonio Arroyo
Cochair: Joanna R. Long
Major: Electrical and Computer Engineering

This research presents hardware and software solutions to many of the problems facing biological solid state nuclear magnetic resonance (ssNMR) spectroscopy at high fields. The low-E 750 MHz magic angle spinning (MAS) probe was designed, constructed, and thoroughly characterized. Under normal operating conditions, a proton (hydrogen, isotope weight 1) RF field nutation rate of 93 kHz and homogeneity (810 degrees/90 degrees) of 93% can be obtained with a sample length of 8.4 mm corresponding to a volume of 80 uL. With a higher power amplifier, we should be able to exceed 110 kHz decoupling fields based on bench measurements. Carbon (isotope weight 13) RF field nutation rates greater than 70 kHz with a homogeneity (810 degrees/90 degrees) of 70% are routinely observed for this sample length; the carbon RF homogeneity can be increased to 89% with a 6.7 mm sample length. Under full proton decoupling for long periods of time, sample heating due to the high RF field is minimal even for samples containing physiological levels of salt. We have not noticed any sample degradation in heat sensitive samples after extensive experimentation. The power handling characteristics, RF fields, and homogeneities make this an ideal probe for applying the full range of MAS solid state

NMR experiments, including sequences which use extended periods of continuous RF pulsing on both channels, to biological samples which are inherently dilute.

A system for optimizing pulse sequences for ssNMR was also developed, demonstrated, and is running. This system was demonstrated on the two standard pulse sequences used to test pulse optimization systems: the inversion experiment and the refocusing experiment. In both cases, pulse sequences were derived which had a wider bandwidth than existing pulse sequences and had extremely good agreement between experiment and simulation. These pulse sequences should be useful in maintaining high signal strength and phase coherence in future research. The methods of optimization and verification allow them to be easily extended to more complex situations in future research..

The combination of the new probe and the method for optimizing pulse sequences for use at higher fields opens many opportunities for new research on biological solids.

CHAPTER 1
INTRODUCTION

Nuclear Magnetic Resonance (NMR)

Nuclear magnetic resonance (NMR) spectroscopy is a method for determining details of molecular structure and dynamics by exploiting the intrinsic nuclear angular momentum referred to as spin. NMR can be used to investigate the distance between atoms both through space at an atomic (angstrom) scale using dipolar couplings, Figure 1-1, and through chemical bonds using scalar couplings, Figure 1-2. Structural NMR experiments measure the distance between atoms and/or the chemical connectivity between the atoms in order to derive the 3D structures of molecules, particularly biomolecules such as proteins. When the density of signal is measured spatially, it is called nuclear magnetic resonance imaging or MRI. NMR and X-ray crystallography are currently the standard methods for determining the 3D structure of biomolecules.

All nuclei have an intrinsic property known as their spin quantum number. Spin quantum numbers can be positive or negative in multiples of ½ and are usually designated by the letter I. Nuclei with a spin number of zero, I = 0, are not NMR active. However, nuclei with I not equal to zero are NMR active since spin angular momentum leads to spin polarization in a magnetic field. The two nuclei of particular interest in this research are ^{1}H and ^{13}C, both of which have a spin number of ½. In the presence of a static magnetic field, designated as B_0, the magnetic moments lead to energy level differences for the different spin states. For a spin number I, there are 2I + 1 states, so for the spin-½ nuclei we are interested in there are 2 energy levels, usually referred to as α and β. The energy difference between these two states is proportional to B_0 with α traditionally being the higher energy state and β being the lower energy state. Using classical mechanics to describe the effect B_0 has on the magnetic moment gives rise to the moment

precessing around the applied magnetic field, B_0, like a gyroscope in a gravitational field. The rate of this precession is shown in Equation 1-1 with units of rad/s and is known as the Larmor frequency of the nucleus.

$$\omega = -\gamma B_0 \tag{1-1}$$

The quantity, γ, is known as the gyromagnetic ratio or the magnetogyric ratio and is specific to a type of nucleus. The gyromagnetic ratio of ^{1}H is 267.552 x 10^6 rad/(s T) and for ^{13}C is 67.283 x 10^6 rad/(s T) [1]. For a magnetic field of 17.6 T, this gives a Larmor frequency ($\omega/2\pi$) of 750 MHz for ^{1}H and 188 MHz for ^{13}C.

This can be related to the energy difference between the two quantum states, α and β, as given in Equation 1-2.

$$\Delta E = \frac{h\omega}{2\pi} = \frac{h\gamma B_0}{2\pi} \tag{1-2}$$

The constant h is Plank's constant. We can use this energy difference to calculate the proportion of spins in the two states at a given field using the Boltzmann distribution, Equation 1-3, where $N_{\alpha,\beta}$ are the number of spins in each state, k_B is Boltzmann's constant, and T is absolute temperature.

$$\frac{N_\beta}{N_\alpha} = e^{\frac{\Delta E}{k_B T}} \tag{1-3}$$

Due to the β state being lower energy, there will be an excess of spins in that state. To get an idea of the magnitude of the signal we are looking at, the proportion of low energy to high energy spins at 17.6 T is 1.00012 for ^{1}H and 1.00003 for ^{13}C. Thus NMR is traditionally viewed as a low sensitivity technique relative to X-ray or UV spectroscopy.

The ratio of excess spins can be manipulated by irradiating the sample at its resonant frequency, the Larmor frequency. Magnetic fields applied at this frequency can excite the spins

to switch state. When the application is stopped, the spins relax back to their steady state, emitting RF energy in the process. This energy is the signal received in NMR spectroscopy. The RF energy applied to the sample can be pulsed to gain more control of the excitation process and therefore the signal received as the spins relax back to their steady state. These pulses are referred to as the pulse sequence, which is the topic of Chapters 3 and 4.

The Larmor frequency of a spin is dependent on the magnetic field experienced by a spin. The static magnetic field induces currents in the electrons around the spin, which creates an opposing magnetic field. The induced current may not produce an exactly opposing magnetic field due to the physical constraints of the structure of the molecule. This is referred to as chemical shielding. The induced magnetic field shifts the Larmor frequency of nearby spins by a small amount, and this is commonly referred to as chemical shift. The chemical shift is dependent on the bonding arrangement within the molecule, which means it can be useful in determining the chemical environment of an atom and how it is bonded to other atoms. Since the induced magnetic field is dependent on the static magnetic field, the chemical shift is linearly dependent on the strength of the static magnetic field, B_0. To allow direct comparison of spectra taken at two different magnetic fields, the chemical shift is usually divided by the Larmor frequency of the magnet in MHz, giving units of parts per million (ppm) rather than Hertz. The chemical shift of ^{13}C can span up to 200 ppm for biological samples, which translates to 25 kHz for a 500 MHz magnet and 37.5 kHz for a 750 MHz magnet.

The chemical shift is also dependent on the orientation of the molecule relative to B_0. These spatial dependencies broaden the signals of interest due to multiple molecular orientations in a given sample, reducing resolution and lowering signal sensitivity. The spread of chemical shifts due to this spatial dependency is referred to as chemical shift anisotropy (CSA). However,

in liquids, the isotropic tumbling of the molecules faster than the NMR time scale averages the different orientations to produce a single sharp peak at the average chemical shift of a spin in a molecule. Thus liquids show no CSA.

Equation 1-1 gives the Larmor frequency of a spin due to an external static magnetic field. Each nucleus has a magnetic moment. This moment not only reacts to the static magnetic field, B_0, but also is itself a magnetic dipole. The magnetic dipoles from different spins interact and affect each other's magnetic environment, shifting each other's Larmor frequency a little. Therefore, the Larmor frequency of a given species (^{13}C for instance) can shift depending on what other spins are nearby and how far away they are. The dipolar coupling constant, b_{jk}, between spins I_j and I_k is given in Equation 1-4.

$$b_{jk} = -\frac{\mu_0}{4\pi}\frac{\gamma_j \gamma_k \hbar}{r_{jk}^3} \tag{1-4}$$

In Equation 1-4, μ_0 is the magnetic permeability of free space (4π x 10^{-7} H/m), h-bar is Planck's constant divided by 2π, and r_{jk} is the distance between the two spins. Only known constants and the distance between the two spins determine the dipolar coupling. Therefore, if the dipolar coupling constant can be determined, the distance between two atoms in a molecule can be determined. It is useful to note that the dipolar coupling constant is not dependent on the magnetic field strength, B_0.

Solid State NMR

Conventional NMR is done on molecules dissolved in a liquid. However, there is increasing interest in doing NMR spectroscopy on biological (and other) molecules that are not readily solubilized. NMR spectroscopy on such samples is referred to as solid-state NMR (ssNMR). ssNMR is considerably more complicated than liquid state NMR since the molecules are no longer tumbling isotropically on the timescales inherent to the experiments; therefore, the

orientation dependent components of the CSA and dipolar interactions are no longer averaged. These spatial dependencies broaden the signals of interest reducing resolution and lowering signal sensitivity, Figure 1-3.

A goal of ssNMR research is to develop techniques to recover the resolution while taking advantage of the added information available from the orientation dependent interactions. One technique frequently employed is to mechanically rotate the samples at very high speeds (5-70 kHz) at a very carefully set angle (the magic angle) with respect to the static magnetic field to mimic isotropic averaging. This technique takes advantage of the fact that the spatial component of many NMR interactions have the form of a second order Legendre polynomial, namely $(3\cos^2\theta - 1)$ where θ is the angle relative to the static magnetic field. This goes to 0 at arctan(sqrt(2)) which has solutions at ~54.74° and ~125.26°. Most magic angle spinning (MAS) NMR is done at the angle ~54.74°.

Challenges in NMR

There are several challenges in NMR: sensitivity, off-resonance, inhomogeneity, and biological samples.

Sensitivity

As was mentioned earlier, NMR is not a sensitive spectrographic method. There are several methods which attempt to improve the sensitivity. The first method is to use stronger static magnetic fields. Equations 1-2 and 1-3 show that the proportion of polarized spins increases as the magnetic field increases. The second method is to use more concentrated samples, and the third method is to use smaller detection coils.

Stronger static magnetic fields

Higher static magnetic fields are becoming more and more readily available. The push for higher magnetic fields is driven by the fact that polarization enhancement goes up as $B_0^{7/4}$,

thereby decreasing acquisition times by a factor of $B_0^{7/2}$ allowing for either shorter acquisition times or detection of more dilute samples. Experiment time reduction allows for more experiments to be done per unit time and for less magnet time that has to be paid for. Detection of more dilute samples is very important in biological studies since many samples of interest are natively in a dilute setting, difficult to isolate, or difficult to acquire.

For ssNMR, increasing the static magnetic field increases the anisotropic chemical shifts (CSAs). These contain important information about the spins of interest. Wider CSAs make differentiating and measuring the CSAs easier, therefore, leading to higher quality measurements.

These benefits come with challenges. Higher field magnets are typically less stable than lower field magnets, particularly when they are also wide bore. This implies that every time an experiment is started, all the parameters for the experiment need to be reoptimized. It is also important to check that the transmitter is still on resonance since magnet frequency (or field strength) drift is much higher on high field magnets.

Higher field magnets cost more and so are often shared between different users. Scheduling and maintenance can become problems as well. Switching between imaging, liquids, and ssNMR provides more opportunity for equipment failures.

The higher frequencies that stronger magnets provide also prove to be more challenging from an RF standpoint on several fronts. Higher fields require much more care in RF design since small changes in size and/or placement of parts can be a significant part of a wavelength different, causing unexpected/unplanned changes in the RF path. In building the coil system in a probe, one has to be careful of the fact that the length of wire in a coil approaches significant portions of a wavelength. This can lead to standing waves which create very bad homogeneity.

Concentrated samples

A seemingly logical solution to a lack of signal is to pack more sample into the spectrometer by increasing the sample concentration. This works well for one subset of samples. Some samples are not conducive to high concentrations and some experiments require dilute samples to be relevant.

Smaller detection coils

The sensitivity per unit volume for a solenoidal coil varies as $1/d$, where d is the diameter of the coil [2,3]. Thus, sensitivity is improved for smaller diameter coils. Therefore, for samples that can be concentrated, the coil size should be minimized.

Off-Resonance

In a sample with only one spin, ^{1}H in water for instance, one would not have to worry too much about off-resonance effects since it could be excited exactly on resonance. Most biological samples of interest, particularly when looking at the ^{13}C spectra, have multiple spins that differ in chemical shift by up to 200 ppm as mentioned earlier. Chemical shift is proportional to B_0, so this translates to 25 kHz for a 500 MHz magnet and 37.5 kHz for a 750 MHz magnet. For ^{19}F or inorganic compounds, this width can be substantially larger. Most NMR pulse sequences assume that the RF pulses are transmitted at the resonant frequency of the spin of interest. As can be seen, this means that many kHz of bandwidth has to be uniformly excited if the whole spectrum is to be observed. A real pulse does not evenly excite at all frequencies. This means that spins further from resonance experience a different pulse from those on resonance, which can make comparisons across a wide spectrum difficult and prone to problems. Higher power pulses can be used to excite spins more uniformly across frequency offsets, but physical constraints limit the amount of power that can be applied, Figure 1-4.

The best methods to fix this challenge seem to be carefully designed pulse sequences that use composite pulses to bring all spins to the desired polarization at the end. Unfortunately, these tend to be longer than simple hard pulses. This extra irradiation time can be problematic when sample overheating is a concern. This extra time can also be a problem when working with MAS ssNMR. If irradiation is too long, MAS synchronization is lost, thereby jeopardizing the selective reinstatement of desired signals due to the many variables involved and non-analytic solutions. This area of research can benefit from the use of computational optimization of pulse sequences to find pulse sequences, which take less time and/or less power to achieve wide bandwidth excitation. This topic is the subject of Chapters 3 and 4.

Inhomogeneity

Inhomogeneity of the applied RF signal, B_1, is a major challenge in ssNMR. B_1 homogeneity is determined by the relationship of the sample to the RF coils through which the excitation magnetization is introduced. The geometry of the sample affects the B_1 homogeneity. For a solenoidal coil, the highest homogeneity is in the center of the coil, with the field strength diminishing away from the center. Maximum homogeneity of the B_1 field is achieved if the sample is restricted to the very center of the coil. This is usually achieved by placing spacers on either side of the sample in the rotor (for ssNMR).

The coil geometry also affects the B_1 field homogeneity. There has been much research on this topic lately and much improvement . This topic will be covered in greater detail in Chapter 2 on hardware development. A simple example is to think about a solenoid. If a solenoid is infinitely long, the magnetic field inside is completely homogeneous down the length of the solenoid. However, due to physical constraints, the coils in probes are of finite lengths. This means that the field strength near the ends of the coil is less than that in the center. If a sample is inside a longer coil, the homogeneity will be better than if the sample is inside a shorter coil.

There are other geometries that play tricks with this idea to improve the homogeneity that will be covered in Chapter 2.

The length of wire used in a coil can also affect homogeneity. As the electrical length of a coil approaches the wavelength of the signal being used, standing waves are setup in the coil, creating nodes of zero current where there is no magnetic field created to excite the sample.

Inhomogeneity affects experiments in several ways. The first way is that different parts of the sample experience different RF fields. This makes them nutate at different rates, which reduces the efficiency of experiments, particularly long, windowless, double-quantum recoupling experiments that rely on long trains of pulses. If all the pulses vary by a small amount in parts of the sample, destructive interference sets up and signal is lost. A second way that inhomogeneity affects experiments is in received signal strength. The receiver sensitivity is proportional to the homogeneity of the field [4].

Biological Samples

Biological ssNMR provides its own set of challenges. For samples that are not concentration limited, the maximum S/N is achieved by using the smallest sample and smallest RF coil as possible. Unfortunately, samples like this are rarely at physiologically relevant conditions. To be physiologically relevant, many proteins (membrane bound ones for example) are concentration limited because have to be in a very dilute environment. For concentration-limited cases, a larger overall sample volume gives better S/N [2].

Biological ssNMR also has the challenge of sample heating. In order to be biologically relevant, samples need to be kept at ambient temperatures. Sample temperatures increase due to RF heating, which can degrade a sample. Another form of sample heating is from the magic angle spinning. The air friction on the outside of the rotor (the surface speeds of which can

approach the speed of sound in some cases) can heat the sample by up to 40 °C, enough to denature a sample.

Goals of This Research

There are two main goals for this research. The first goal is to build and characterize ssNMR probe for high static magnetic field optimized for biological samples by having high homogeneity on both RF channels, high sensitivity for dilute samples through a larger sample volume and low RF heating via a unique probe design. The second goal is to optimize pulse sequences to mitigate the issues associated with the expanded chemical shift and CSA experienced at high fields.

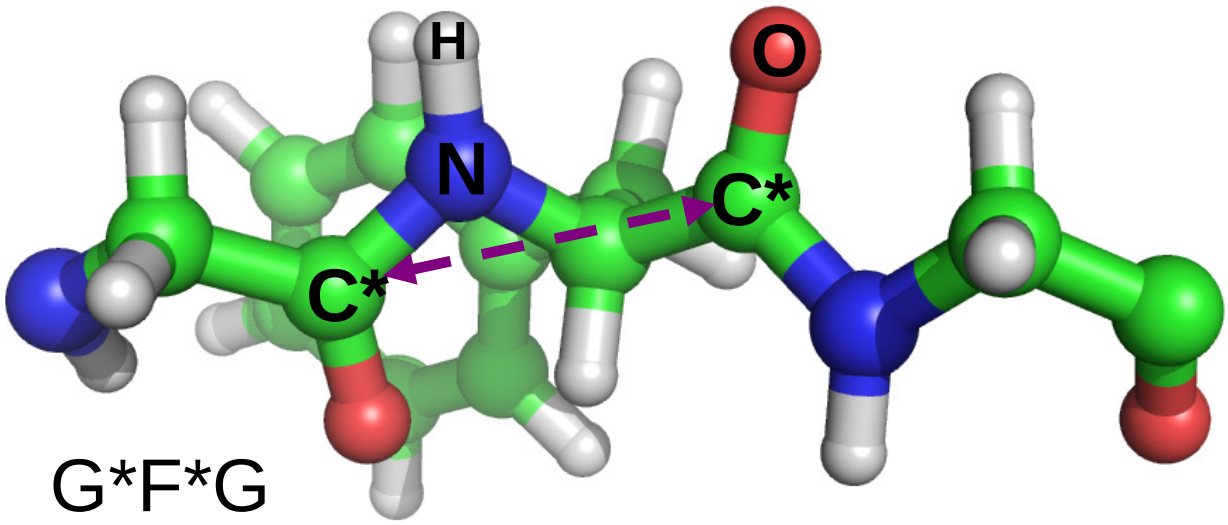

Figure 1-1. NMR through space measurement. NMR can be used to measure the through space distance between atoms in a molecule via dipolar interactions.

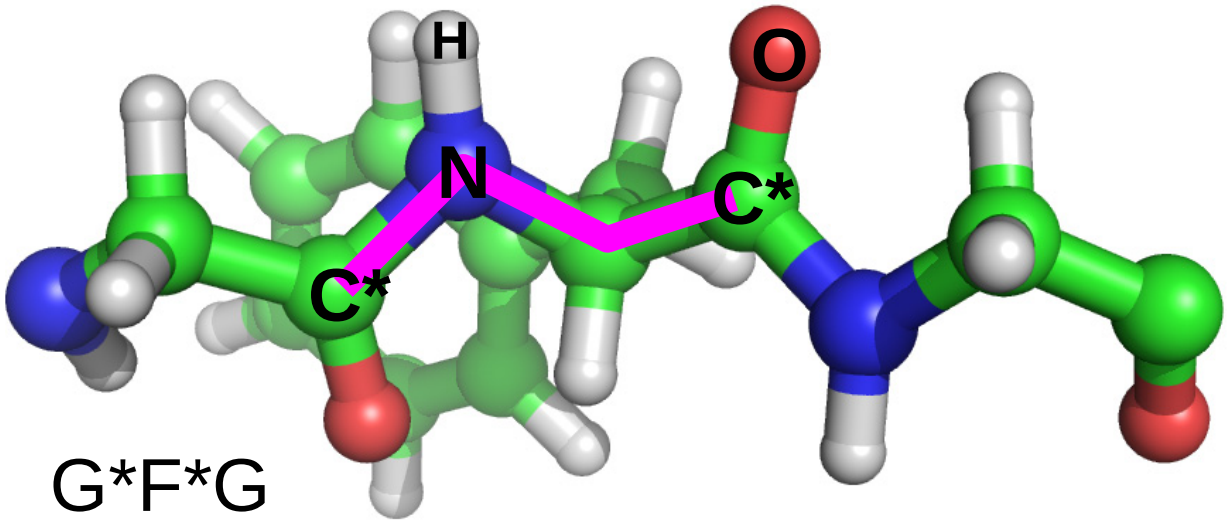

Figure 1-2. NMR through bond measurement. NMR can measure the connectivity of atoms in a molecule via through bond scalar couplings.

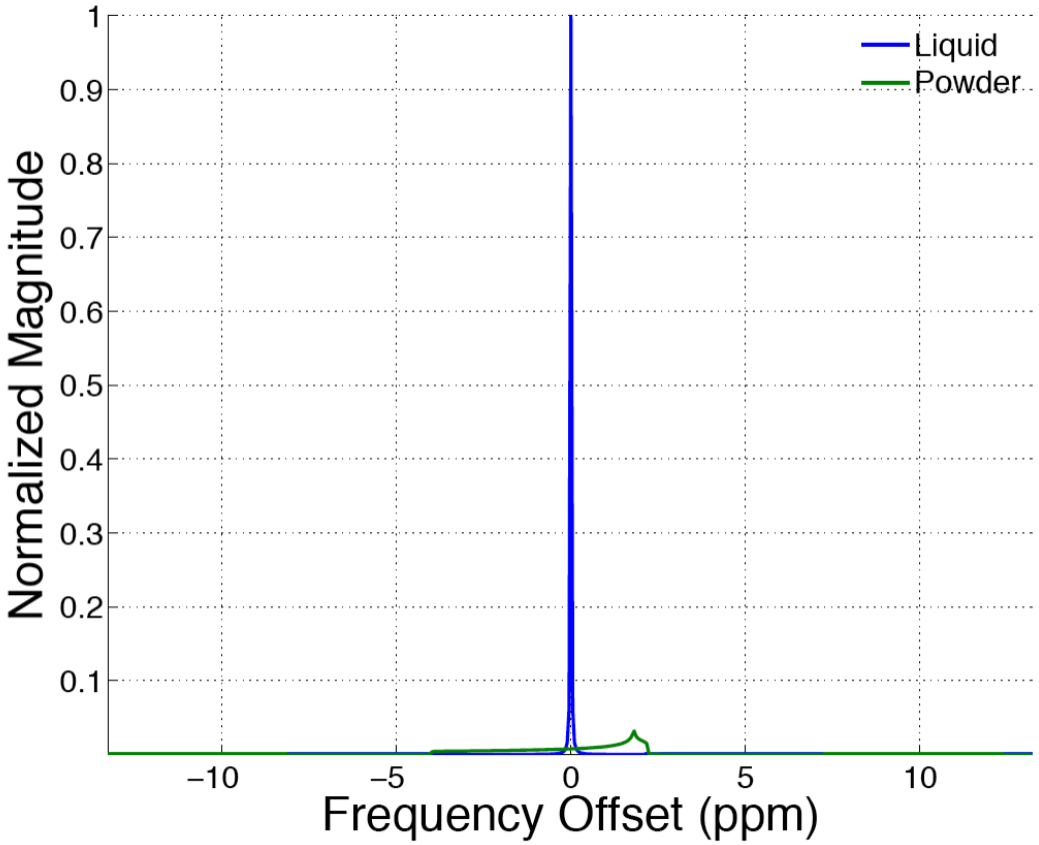

Figure 1-3. Comparison of a liquid signal to a powder pattern via simulation. The liquid peak is the average of all the orientations while the powder pattern shows the orientation dependent chemical shift anisotropy.

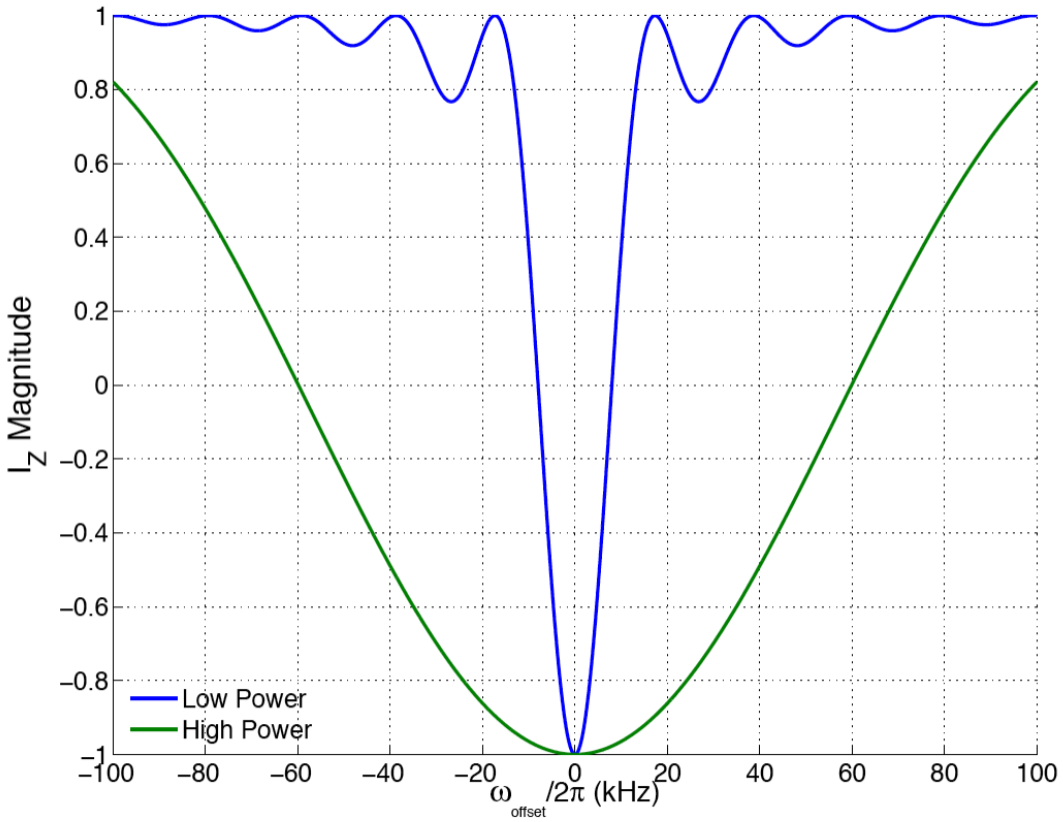

Figure 1-4. High power versus low power excitation. A low power pulse excites spins over a narrow range of frequencies, while a high power pulse excites spins over a wide range of frequencies.

CHAPTER 2
HARDWARE OPTIMIZATION

Improvements and Challenges at Higher Fields

The hardware used in NMR spectroscopy has improved greatly since NMR signals were first observed in the 1940s. In particular, the strength, homogeneity, and stability of magnetic fields has greatly increased, thereby increasing the sensitivity of NMR. This increase in sensitivity broadens the applicability of NMR to include more complex biological systems, particularly for ssNMR techniques. This is important because at a fundamental level, NMR is a very insensitive method of spectroscopy since it is detecting very small changes in the sample. Polarization enhancement estimates suggest the increase in signal-to-noise (S/N) should be on the order of $B_0^{7/4}$, decreasing acquisition times by a factor of $B_0^{7/2}$, which means that increasing the magnetic field from 11.7 T (500 MHz) to 17.6 T (750 MHz) can increase the S/N by about 2.2 times and reduce the amount of experiment time by half or more. The reduction in experiment time means that samples which are less stable or that have less signal can be studied. Performing NMR experiments at high magnetic fields, therefore, increases their applicability to systems that are sensitivity limited.

It also aids the study of samples that are resolution limited. With the advent of stable high field instruments with proton frequencies of 700 to over 900 MHz, the application of solid state NMR (ssNMR) spectroscopy to a wide variety of biomolecular systems has become increasingly feasible. However, realizing these sensitivity gains for a variety of samples, nuclei, and pulsed experiments is not straightforward since the increase in signal is accompanied by an increase in the isotropic and anisotropic chemical shifts, Figure 2-1. This results in increased spectral

This chapter is largely adapted from: Journal of Magnetic Resonance, doi:10.1016/j.jmr.2008.12.008, Seth A. McNeill, Peter L. Gor'kov, Kiran Shetty, William W. Brey and Joanna R. Long, A low-E magic angle spinning probe for biological solid state NMR at 750 MHz, 2008, with permission from Elsevier License Number 2110240542593

widths, requiring the generation of more powerful B_1 fields to excite all frequencies of interest. Additionally, as at lower magnetic fields, many ssNMR experiments require proton decoupling with RF field strengths above $(\omega_1/2\pi) = 100$ kHz on the proton channel for optimal resolution.

The Challenge of Radio Frequency Coils at High Frequencies

With traditional ssNMR probe circuits utilizing multiply-resonant solenoidal coils, achieving efficient and homogeneous B_1 fields is complicated by the electrical length of the sample coil at high proton frequency approaching the quarter wavelength limit [5]. Additionally, the study of biomolecules under physiologically relevant conditions substantially alters the probe performance by loading the coil with samples containing high salt concentrations. Creating stronger B_1 fields also creates stronger electric (E) fields within the sample leading to more heating and, ultimately, sample degradation. The generation of E fields and their contributions to sample heating in ssNMR spectroscopy have been extensively studied in recent years [6-18]. This heating can be overcome by cooling samples down to where the heating does not disrupt the system [6], but often this means cooling the samples well below biologically relevant temperatures to where protein conformations are altered or the molecular dynamics of interest are removed. Lowering the conductivity of the samples is another method of reducing heating [7]. This is possible for some samples, but again may lead away from biologically relevant conditions. Common spectroscopic approaches to minimizing sample heating include the use of very low duty cycles and utilizing very small coils, with a subsequent reduction in scans per unit time and sample volume, respectively, leading to poorer S/N and/or longer acquisition times, particularly for concentration limited samples.

More recently, probe design efforts have focused on the more fundamental issue of modifying the coil design to reduce sample heating by reducing inductance and adding shielding.

Traditional multinuclear ssNMR probe designs employ a single, multiply resonant solenoid as this maximizes the filling factor for the various frequencies and (when wavelength effects can be neglected) helps ensure RF overlap. Several clever modifications to the solenoid have been proposed to reduce heating while preserving as much as possible the efficiency of both high and low frequency channels. These designs also allow commercial probe suppliers to continue to use their well-developed multichannel matching networks. Scroll coils [19] (also known as "Swiss Rolls") offer a robust solution to the problem of sample heating as they have a lower inductance than solenoids and their geometry creates a built-in Faraday shield for the E field since the inner turns shield the sample from the E field generated by the outer turns [8,19]. Both these factors reduce sample heating and improve stability and performance on the proton channel. The ^{1}H efficiency of scroll coils with lossy samples can surpass that of solenoids. However, scroll coils are less efficient than solenoidal coils at lower frequencies due to their low inductance and low Q [8]. Scroll coils also present challenges due to temperature dependent tuning changes inherent to the large capacitance between turns [9,10] this capacitance also limits the available sample volume [18]. The Z-coil, consisting of a central loop with two spiral coils on the ends of the loop [11] lowers sample heating by more than an order of magnitude relative to a solenoidal coil as well as having an RF efficiency that is independent of sample conductivity. However, unlike the scroll, the RF efficiency of the Z-coil with a lossy sample is just comparable to that of the solenoid, and there is also a penalty in sensitivity and efficiency at the lower frequencies. Most recently, Krahn and co-workers have shown that inserting a conductive shield between the sample and the solenoid can reduce the heating [12]. Precise manufacturing of the shield led to an effective decrease in heating at a modest cost in sensitivity due to the decrease in sample filling factor compared to an unshielded solenoid. However, the close proximity of the shield to

the sample coil can be expected to limit the voltages, and hence the achievable B_1, for larger samples. These three single-coil alternatives to the solenoid have been shown to reduce heating at some cost to RF efficiency. While the gains in RF and temperature stability certainly outweigh the loss in sensitivity for lossy samples, an alternative approach which does not compromise the sensitivity and efficiency of the lower frequencies would be attractive, particularly since the bulk of biological ssNMR experiments rely on direct detection of low gamma nuclei due to proton resolution limitations at slow to intermediate magic angle spinning speeds.

One such solution to improve RF performance at the proton frequency and simultaneously reduce sample heating is to use separate coils for the low and high frequencies. There are several benefits to this design: using two coils allows the individual circuits and coils to be optimized for each frequency range; having one coil inside the other allows the inner coil to act as a partial Faraday shield for the outer coil; the RF fields generated by the two coils can be designed to be orthogonal, which increases channel isolation and therefore efficiency; and, when the coil assembly is rotated for magic angle applications, the use of orthogonal coils results in a compromised RF field on only one coil.

An advantageous approach for using crossed coils in MAS probes is to place a low-inductance, segmented ^{1}H saddle coil inside a solenoid tuned for the lower frequency channel [9]. With this configuration, the ^{1}H coil shields the sample from some of the E fields created by the inductance of the solenoid, improving the low-frequency efficiency in the presence of lossy samples. The inverse configuration, in which the low frequency sensitivity is improved by placing a solenoid within a loop gap resonator (LGR), has been used effectively to reduce heating in large volume static probes [18] and is the focus of this project.

The Low-E Solution

The LGR is a coil geometry which works well at proton frequencies due to its low inductance, lower E fields, and short electrical length; LGRs have been used extensively in EPR for high frequency applications [20] as well as in MRI [21]. Previously, we have shown that they also work well for high field static ssNMR applications when combined with an orthogonal solenoid for the lower gamma nuclei [18], a design christened "low-E" due to its favorable mitigation of E fields within the sample space. In such designs, the sample is placed within a solenoidal coil to maximize sensitivity and homogeneity for the low frequency channel; an LGR optimized for the proton channel is orthogonal to and surrounds the solenoid. In this configuration, the solenoid further lowers the E field by acting as a partial Faraday shield between the sample and the LGR. The outer ^{1}H resonator is slit strategically to cancel low-frequency eddy currents, which would otherwise reduce the efficiency of the inner coil. The loss of filling factor for the proton channel in a crossed coil setup like this is largely made up for by the improved efficiency of the single-frequency ^{1}H matching network. Since the solenoid is not called upon to produce a ^{1}H field, its length and number of turns can be increased to improve sensitivity. For a MAS probe, the fact that the field of the LGR can be made orthogonal to the polarizing magnetic field, B_0, further improves ^{1}H efficiency relative to a multiply tuned solenoid.

An additional benefit of the LGR ^{1}H coil is that its homogeneity is excellent. The high B_1 homogeneity on both channels of the ^{1}H LGR/^{13}C solenoid configuration is of critical importance for the application of cross-polarization (CP) and multipulse experiments to samples that are concentration limited. In multipulse recoupling experiments, especially long windowless experiments, the accumulation of phase errors from different parts of the sample nutating at different rates leads to reduced excitation efficiencies and phase errors in the resulting signals,

Figure 2-2. The increased homogeneity of the LGR at the proton frequency and the solenoidal coil at lower frequencies can increase the efficiency and final signal strength of multipulse experiments, particularly experiments that utilize double quantum filtering.

In considering concentration-limited samples, the ^{1}H LGR/^{13}C solenoid configuration allows reasonably straightforward scaling of the sample volume even at high frequencies. For samples that are inherently dilute (i.e. membrane proteins in lipid vesicles or proteins adsorbed on to solid substrates), it is often preferable to use larger sample volumes. This is because sensitivity per unit volume scales as ~(1/d) with respect to the rotor diameter while full rotor sensitivity scales as d^2, so a larger sample diameter presents significant advantages for concentration-limited samples if sufficient RF strength and homogeneity can be achieved, Figure 2-3 [2].

In this chapter I present the design and characterization of a ssNMR magic angle spinning (MAS) probe that utilizes a ^{1}H LGR placed orthogonally to a ^{13}C solenoid implemented on an NMR system with a 17.6 T magnet (750 MHz ^{1}H frequency). The design was optimized for intrinsically dilute samples by utilizing a 4 mm rotor. The use of two separate coils allowed us to significantly increase both the length and number of turns in the solenoidal coil, making highly homogeneous B_1 fields achievable even with the increased volume of the coil.

750 MHz Probe Assembly

Sample Coil Assembly and Integration into a MAS Stator

The probe design described and characterized in this chapter is an adaptation of a previously reported static low-E probe [18]. The coil assembly consists of two RF coils which are orthogonal to each other. The outer coil is a rectangular LGR tuned for the ^{1}H circuit and the inner coil is a solenoidal coil for the low gamma nucleus. In previous work this assembly was optimized for PISEMA experiments on static, oriented membrane-embedded protein samples.

For the present MAS application, the coil assembly, Figure 2-4, was modified so that it could be integrated into a 4 mm MAS stator (model AMP4023-001, Revolution NMR, Inc., Fort Collins, CO) with a top spinning speed of 18 kHz. In MAS implementation of low-E coils, the sensitivity of the detection channel benefits from a stator design where MAS bearings are placed further apart as this provides space for additional turns in the observe solenoid. The coil cavity available in the stator measures 11 x 12 mm in cross-section and 20 mm in length, which is substantially longer than the 12.7 mm length of our coil assembly. This stator is compatible with standard Varian 4.0 mm Pencil style rotors (Revolution NMR p/n AMP4088-001 or Varian p/n MSPA003006). The exact physical dimensions for both coils are provided in Figure 2-4. Regulation of sample temperature is accomplished by VT gas delivered through the side of the stator.

The loop gap resonator was fabricated by forming a 0.25 mm thick, 9.0 mm wide copper strip around a 12.2 x 8 mm rectangular block. The ends of the strip were terminated with non-magnetic chip capacitors (100B series, American Technical Ceramics) to complete the LGR. The resonator was attached to the ^{1}H matching network using low-inductance leads threaded through a Teflon platform that centers the coil assembly in the stator housing, Figure 2-6(a, b). The inner, low gamma coil is an 8-turn 4.6 mm ID x 8.3 mm long cylindrical solenoid. The solenoid was made from 0.6 mm round copper wire (American Wire Gauge #22). Locating the low frequency coil closest to the sample maximizes sensitivity for direct detection. The homogeneity of the B_1 field was improved by using variable spacing between the coil windings. The solenoid leads were also threaded through the Teflon platform which centers the coil with respect to the LGR and the MAS stator.

RF Matching Network

The double-tuned X–^{1}H matching network implemented in our MAS probe is shown schematically in Figure 2-5. The design and performance of this RF circuit has been thoroughly described [18]. For the purposes of the applications described here, the detection channel was tuned to ^{13}C. It can be re-tuned to any other low gamma nuclei by exchanging capacitors C_{7A} and C_8. Variable capacitors C_5, C_6, and C_7 in the low frequency circuit are 1 to 10 pF trimmers (NMNT10-6E, Voltronics Corp., Denville, NJ). In the proton channel, C_1 is a 1 to 6 pF trimmer (Voltronics NMQM6G), C_2 and C_3 are 0.3 to 3 pF trimmers (RP-VC3-6, Polyflon Co. Norwalk, CT). Non-magnetic fixed capacitors employed in the proton channel are Voltronics 11 series chips. In the low frequency channel, we used non-magnetic 100C series chips from American Technical Ceramics, Huntington Station, NY.

The chip capacitor values for the proton LGR (L_0–C_0) had to be chosen to resonate it slightly above the Larmor frequency. This self-resonance frequency is affected by the dielectric material of the stator and the coil platform, which are in close proximity. A small loop was inserted through a hole in the stator to pickup the resonant frequency, f_0, of the entire stator assembly. Chip values (C_0) were chosen to place f_0 between 780 and 790 MHz.

Probe Body Construction

The probe body was machined and assembled at the NHMFL. The outer structure is made of anodized aluminum. The upper plug at the top of the body tube is brass, Figure 2-6c. The RF platform is copper, Figure 2-6f, and the top stator platform is PEEK (Polyetheretherketone). The end caps of the main aluminum body tube caused problems during assembly because they were machined with very close tolerances. Assembly requires putting them in and taking them out several times. They began galling during the assembly process from this putting in and taking out. If only one is stuck, a dowel of suitable material (Teflon was handy) and diameter can be

used as a hammer to knock the plug out from the inside. With both stuck, the challenge is much greater. Subsequent probes have greater clearance planned on these parts. The copper parts are soaked in sodium bisulfate, washed with soap and water, and then quickly dried to remove any oils or oxidation. Gloves are worn during assembly to reduce reintroduction of contaminants and oxidizers.

VT dewars were custom made by James Finley at GlassWorks (www.glassworks.com). Heaters were bought from Bruker for ease of interoperability and durability. Their heaters are better quality than others. Thermocouples are of type T (copper-constantan) from Omega Engineering, Inc. (www.omega.com). They come with a standard 2-prong thermocouple plug with the thermocouple installed in a stainless steel housing. The plug and housing are removed to reduce the space required for the thermocouple. A jack for Bruker style probe thermocouples is installed to maintain interoperability with the Bruker spectrometer where this probe is installed.

Tuning rods are fiberglass rods. The upper ends are turned down to accept a brass screwdriver type head that fits into the bottom of the variable capacitors. This screwdriver head and the knob at the base are glued to the fiberglass rod using 5-minute epoxy, Figure 2-6(d, e). A label maker is used to make labels for the knobs. After applying the labels, a loop of shrink-wrap is put around the label to keep it in place. The ^{1}H channel knobs are aluminum and the ^{13}C channel knobs are brass colored. The tuning knobs are larger than the match knobs. This approach makes differentiating which knob does what much easier during normal use.

This probe was the first to use a very narrow diameter, high pressure tubing for the MAS bearing and drive gases, Figure 2-6g. This tubing greatly reduces the space required for transporting air to the probe head. The unfortunate side effect of the narrow tubing is that higher pressures are required to get sufficient flow for higher MAS rates.

Brass tubing (~1/8" diameter) was used as guides for the thermocouple and body/electronics air between the bottom and top of the probe. The VT air dewar had a stainless steel guide between the top and bottom of the probe. Tape is added to the outside of the dewar prior to installation to prevent it from rattling in the guide. The spring mechanism which holds the heater into the dewar also holds the dewar into the probe.

The RF signals are transported from the base to the RF section via semi-rigid coax cables, Figure 2-6g. The ^{1}H channel has an N-type connector and the X channel has a BNC type connector to prevent accidental attachment of the wrong RF cable to either channel. The final probe head is shown in Figure 2-7

Testing and Specifications

Power Efficiency and Homogeneity of RF Fields

The low-E resonator MAS probe was fully characterized via NMR experiments using a 750 MHz Bruker AV2 system with an 89 mm bore magnet and a CPC MRI Plus model 19T300 ^{1}H amplifier. The power going into the probe was measured using a directional coupler and an RF power meter (Agilent E4416A meter with an E9323A power sensor). Adamantane (Acros Organics) was used for direct observation of the ^{13}C and ^{1}H resonances for B_1 field and homogeneity measurements since it is a low-loss material and its dipolar couplings are inherently small due to molecular motion and can be removed to first order by MAS at moderate rates. It was also used for calibrating chemical shifts and lineshape measurements. For restricted volume measurements, Kel-F spacers were used to reduce the sample length and to center the sample within the coils. ^{1}H and ^{13}C nutation experiments utilized a single, variable length pulse on the observe channel and, for ^{13}C experiments, CW proton decoupling (83 kHz) was applied during acquisition. ^{13}C homogeneity was determined by irradiating and monitoring the carbon resonance

at 38.48 ppm; ^{1}H homogeneity was determined by irradiating and monitoring the unresolved proton resonances at an average position of 2.6 ppm.

The maximum ^{1}H power available on the spectrometer is 220 Watts, which is below the power limit of the probe. With our limited available power, the maximum ^{1}H decoupling field is achieved by bypassing the duplexer and connecting the amplifier output directly to the probe; this is the setup we typically use in ssNMR experiments. To measure maximum ^{1}H nutation rates achievable by this setup using NMR, we prepared a sample of chloroform sealed in a 1 mm capillary with 5-minute epoxy. The capillary was inserted inside a thick-walled rotor along with ground KBr to stabilize the spinning at ~1 kHz. The ^{1}H nutation rate was then measured via indirect detection [22].

B_1 homogeneity measurements as a function of adamantane sample length are shown in Figure 2-8. Homogeneity is reported as the ratio of the signal intensities after 810° and 90° pulses (810°/90°). RF field strengths ($\omega_1/2\pi$) were measured to be 104 kHz at 220 W of input power for ^{1}H (resonant frequency of 750.2 MHz) and 72 kHz at 75 W of input power for ^{13}C (resonant frequency of 118.6 MHz). As expected, the LGR coil for the proton channel, the increased number of turns in the solenoidal coil, and restricting the sample length to within the coils led to enhanced B_1 homogeneity at both low and high frequencies. Example nutation profiles for both ^{1}H and ^{13}C can be seen in Figure 2-9. The length of sample in Figure 2-9 is 81% of the length of the solenoid. Isolation between probe channels was measured using a HP8752C Vector Network Analyzer (Hewlett Packard). Without external filters, the isolation achieved between the ^{13}C and ^{1}H ports is 45 dB at the ^{1}H frequency and 24 dB at the ^{13}C frequency.

Placement of the ^{1}H loop gap resonator as the outer coil leads in principle to less efficient performance on the high frequency channel, but this compromise is offset by orienting the

resonator orthogonal to B_0 and by the high RF homogeneity of the LGR. By choosing this geometry, the ^{1}H B_1 field in the x-y plane is not attenuated by rotation of the coil assembly from a static orientation orthogonal to the external magnetic field to an orientation in which the solenoidal coil axis is at the magic angle. More importantly, the placement of the solenoidal coil inside the assembly improves the filling factor on the observe channel. This helps to offset the loss of B_1 field in the solenoid due to its magic angle orientation. Because the length of the solenoid is not limited by ^{1}H wavelength effects, we are able to utilize an 8-turn solenoid, which further improves the performance of the ^{13}C channel relative to multiply-tuned solenoids containing fewer turns.

Power Handling and Stability

The probe's power handling capabilities were bench tested to determine if long, high-power pulses led to either arcing or detuning of the resonant circuits. No arcing was observed during 80 ms long pulses in the ^{1}H channel at powers exceeding 280 W, which corresponds to $(\omega_1/2\pi) \approx 117$ kHz. However, the first implementation of the probe exhibited detuning of the ^{1}H resonance by as much as 0.9 MHz once the decoupling pulse length exceeded 20 ms. The chip capacitors in the ^{1}H LGR are heated up by the high current needed to produce strong decoupling fields, and this can lead to small changes in capacitor values. This problem was narrowed down to a lack of cooling mechanism for these chips. To correct it, a channel was cut in the Teflon platform underneath the chip capacitors, allowing the gases circulating inside the sample compartment to flow around the chips and cool them on all four sides. This measure significantly decreased detuning of the ^{1}H channel to a much smaller, comfortable level, which does not require tuning adjustments during NMR experiments. A subsequent test has shown that high power detuning in the ^{1}H channel can be eliminated if the 100B series chip capacitors in the

LGR are replaced by their temperature-compensated NP0 counterparts, such as non-magnetic version of 700B series. The ^{13}C channel was stable under high power conditions with pulses up to 20 ms in length at powers exceeding 75 W ($\omega_1/2\pi \approx 72$ kHz) and 5 ms long pulses at 117 W ($\omega_1/2\pi \approx 90$ kHz).

One of the first versions of the coil assembly did arc at lower power levels. The arcing took out a ^{1}H HPPR/2 preamplifier slice at AMRIS. The probe was transported back to the NHMFL in Tallahassee for further testing and repair. This provided some good pictures of what arcing looks like, Figure 2-10. Hard arcing like we were experiencing makes an audible click too. Arc testing was done in a lab with an older high power amplifier. The power levels were slowly increased and then once a satisfactory power level achieved, left to run for a few hours to make sure that no weaknesses were found from usage. This process resulted in a slight redesign of the ^{1}H LGR. Originally the design called for two sets of two series capacitors in parallel. The final design uses two sets of three series capacitors in parallel. This reduces the voltage across each capacitor such that arcing is much less likely.

Experience has shown that we are operating at the maximum power on the ^{13}C channel before arcing at around 75 W. The ^{13}C solenoid is not arcing to the LGR since when arcing occurs, there is no corresponding spike showing on the ^{1}H reflected power. The next version of the probe this should be examined more carefully and verified so that the full 100 W available on the spectrometer can be used. However, even with this limitation on power, the efficiency of the ^{13}C channel is high enough for our purposes due to the 8 turns in the solenoid.

Shimming

The probe shims adequately without spending extensive time. The ^{13}C full width at half height for adamantane is 9 Hz at a sample length of 3.7 mm; the 0.55% linewidth is 83 Hz. For

the full rotor length, 11.7 mm, the half height linewidth is 11 Hz. A small foot is observed in the ^{13}C signal which is similar to inhomogeneous broadening we have observed in a commercial XC4 probe from Doty Scientific. Our lineshape would likely be improved by using zero susceptibility wire in the solenoid which is closest to the sample, but the opportunity to test this hypothesis has not arisen. Another source of inhomogeneity may be the capacitors for the ^{1}H coil. However, they are more physically distant from the sample, so we expect their contribution to the observed broadening to be less, relative to the ^{13}C coil wire.

RF-Induced Heating

To characterize RF performance with typical biological samples, test samples containing either D_2O or 0.15 M NaCl in D_2O were prepared. Experiments were performed using the full rotor volume as well as more restricted sample lengths. Rotors were sealed with PTFE tape gaskets and sample lengths were varied using Kel-F spacers.

To measure RF-induced heating, aqueous samples described above were doped with 20 mM thulium 1,4,7,10-tetraazacyclododecane-1,4,7,10-tetrakis(methylene phos-phonate) ($TmDOTP^{5-}$) (Macrocyclics) as the temperature dependencies of the exchangeable proton chemical shifts in $TmDOTP^{5-}$ are sensitive, linear, and well documented [23]. In particular, the H(6) proton provides a nicely resolved resonance for monitoring temperature changes. Because of its high ionic strength, the relatively small concentration of $TmDOTP^{5-}$ is expected to contribute noticeable RF loss. The sample rotation rate was regulated at 2 kHz, and bearing and drive air were supplied at room temperature. The sample temperature was regulated by means of an air stream cooled by a Bruker BCU-05 refrigeration unit and controlled by a BVT-3300. The RF heating experiment was run as follows: a presaturation pulse was applied to the probe for 40 ms at the test power level; this was followed by 5 ms of signal recovery before a standard pulse and acquire sequence. The duty cycle was maintained at a constant 3.8% while the

presaturation power was varied. Before signal averaging, 256 dummy scans were run (taking ~5 minutes) in order for the sample to reach a steady state temperature. Example spectra are shown in Figure 2-11.

The sample temperature rise due to RF irradiation can be seen in Figure 2-12. Even for the full rotor with 150 mM NaCl added to the $TmDOTP^{5-}$, the sample temperature increased less than 15 K. For the 20 mM $TmDOTP^{5-}$ samples, the heating was 7.3 K for a full-length sample (11.7 mm) at 168 kHz^2, 6.5 K for a 6.7 mm length sample at 257 kHz^2, and only 5.3 K for a 3.7 mm length sample at 280 kHz^2. From Figure 2-12 we can see that the 150 mM added NaCl roughly doubles the amount of RF heating in the sample. The longer samples, which extend closer to the ends of the LGR coil where the E field is known to be higher [24], reached a somewhat higher temperature than the 3.7 mm samples. Further samples incorporating lipids (50 wt %) were also tested (data not shown), with results similar to those seen for samples without salt. The relatively low RF heating observed even with higher salt conditions demonstrates that the LGR reduces the conservative E field within the sample to an acceptable level that will neither damage the sample nor significantly affect NMR measurements. This is particularly critical for high field spectroscopy since the conservative E field scales linearly with B_0 for a fixed coil inductance and B_1 field [18]. The voltage across a coil, which determines the conservative E field, is proportional to the impedance of the coil, $\omega_0 L$, where ω_0 is the Larmor frequency.

The use of low-inductance LGR for generation of ^1H RF fields had been shown to reduce RF heating in the sample by an order of magnitude during decoupling [18]. At the same time, our detection solenoid has multiple turns with inductance much higher than that of LGR. The strength of its conservative E field is mitigated by 4 times smaller Larmor frequency of ^{13}C.

Also, the amount of time the ^{13}C channel is transmitting is typically less than half the time the ^{1}H channel is transmitting, since the ^{13}C channel is only transmitting during the excitation part of the experiment while ^{1}H decoupling is used during both excitation and acquisition. To be safe, it is prudent to compare amounts of heat generated in the sample by each of the coils. A simple way to estimate heating contributions from each RF channel under normal operating conditions is to use changes in a probe's Q or 90° pulse length when switching between lossy and non-lossy samples [9,18]. For the Q measurement, the amount of RF power dissipating in the dielectrically lossy sample for each kHz^2 of RF field is shown in Equation 2-1 in units of W/kHz^2 where Q_{BIO} is the Q of a probe loaded with lossy (biological) sample and Q_{NL} and η_{NL} are the Q and power efficiency of a probe with a non-lossy reference.

$$q_{heat} = \left(\frac{Q_{NL}}{Q_{BIO}} - 1 \right) \cdot \frac{1}{\eta_{NL}} \tag{2-1}$$

For our measurements, we used 150 mM NaCl aqueous solution as a lossy sample. We chose water to serve as a non-lossy reference in order to maintain a similar dielectric constant and minimize retuning of the probe between the measurements of Q's. Both samples occupied full volume of the rotor (11.7 mm in length). The Q values and calculated heat dissipation rates, q_{heat}, are listed in Table 2-1 for both ^{1}H and ^{13}C channels.

To compare amounts of ^{13}C and ^{1}H heating induced in the sample under typical operating conditions, we estimate heating during an actual DQDRAWS experiment, an experiment used extensively in Dr. Long's lab. If the ^{1}H 93 kHz decoupling field is applied for 40 ms every second, the resulting heat dissipation is $1.02x(10^{-3})x93^{2}x0.04 = 0.35$ Joules into the sample per transient from the ^{1}H channel. The CP pulse and windowless excitation pulse train on the ^{13}C channel last half as long (20 ms) with RF fields held at 42.5 kHz, resulting in $4.67x42.5^{2}x0.02 = 0.17$ Joules of heat dissipated in to the sample. Thus, with the current probe design, the ^{13}C and

^{1}H channels contribute levels of heating on the same order of magnitude in the sample under normal operation.

Frictional Heating

The aqueous samples used for measurement of RF heating will not spin reliably at high sample rotation rates, so powdered lead nitrate ($Pb(NO_3)_2$) samples were used to measure frictional heating due to sample rotation. The temperature dependence of the ^{207}Pb chemical shift (156.4 MHz at 17.6 T) is also well documented [25]. To reduce the density of the MAS sample to levels relevant to biological samples, lead nitrate was mixed with NaCl. Fine, ground crystals of NaCl were mechanically mixed with fine, ground crystals of lead nitrate to achieve a mixture that was ~10% lead nitrate by weight and to achieve a biologically relevant density of ~2.4 g/cm^3. Spectra were collected using a standard pulse-acquire sequence at different sample rotation rates while holding the VT air temperature constant, Figure 2-13. A temperature equilibrium time of at least 10 minutes was used at each spinning speed. The full 57 mL sample volume of a thick walled rotor was used for these measurements. As shown in Figure 2-14, frictional sample heating was a maximum of 20 K when spinning at 13 kHz with a VT temperature of 298 K. It was noted that different bearing and drive pressures resulting in the same sample rotation rate cause varying levels of frictional heating.

Sensitivity Measurements

Measurements of sensitivity were performed using crystalline, natural abundance glycine (α-form) to allow comparison to other published work. The veracity of this standard for determining true S/N is limited given that the quality of the spectra are determined by a number of interrelated factors. Measurements were done for the full sample length (11.9 mm) in a thin-walled rotor (95.6 mL sample volume) with a sample mass of 115.4 mg spinning at 13 kHz. Sample lengths of 8.4 mm with 83.6 mg of sample, 6.9 mm with 70.1 mg of sample, and 3.9 mm

with 38.3 mg of sample were also examined. All spectra were collected at 13 kHz MAS. Signal was measured with ramped cross polarization sequence and 94 kHz SPINAL64 decoupling [26] during acquisition. A recycle delay of 30 seconds was used and signal was averaged over 1, 2, 4, and 8 scans with 2 dummy scans. Data (4096 points with a dwell of 10 ms) were Fourier transformed (without line broadening), phased, and maximum signal was measured between 40-50 ppm for comparison to noise between -70 and -90 ppm, Figure 2-15.

There are several formulae for calculating S/N. We have chosen a standard definition of S/N as being $P_H/(2*N_{RMS})$, where P_H is signal peak height and N_{RMS} is the root-mean-square noise amplitude. The sensitivity of the probe was also characterized for a variety of sample lengths in order to allow quantitative determination of optimal conditions for applications in which one may be examining mass-limited samples or concentration-limited samples with either simple pulse sequences or pulse sequences requiring high B_1 and high homogeneity on the ^{13}C and/or ^{1}H channels for the best possible performance. For these measurements, the S/N is normalized to the square root of the number of scans, allowing S/N comparisons independent of the number of scans, as well as to the weight of the individual samples.

As can be seen in Figure 2-16, the best S/N per mg is with the smallest sample length and remains high for sample lengths less than 6.9 mm (83% of the coil length). The best overall S/N is achieved for samples lengths on the order of 6.9 mm and longer. The signal quality does not increase greatly once the sample length is on the order of or greater than the coil length (8.3 mm) and resolution is compromised for the longer samples.

For most applications that are concentration-limited rather than mass-limited, a sample geometry in which the rotor wall is the minimum thickness possible for a given spinning speed is optimal. A sample length on the order of 7 mm provides optimal S/N with acceptable RF

homogeneity. Increasing the length of the sample beyond this point yields little in improving S/N and can even be detrimental to making quantitative measurements using standard ssNMR pulse sequences due to incomplete excitation of the full sample. For samples with limited quantities, the length of the sample can easily be adjusted using spacers allowing optimal S/N and RF performance.

Cross Polarization Measurements

For static cross-polarization (CP), the best magnetization transfer is at the Hartmann-Hahn matching condition, which means that both nuclei are nutating at similar rates during the spin lock [27]. When optimizing CP on a MAS sample, the best transfer conditions are when the nutation rates differ by +/-1 or 2 times the MAS rate [28,29]. A CP profile can be generated by holding the RF field constant for one resonant frequency and varying the RF field for the other resonant frequency through these matching conditions. The symmetry and widths of the peak matching conditions as well as the overall profile reveal how homogeneous the B_1 fields are at the two frequencies and how well they overlap with respect to the sample [5]. CP matching profiles using square spin lock pulses were collected by fixing the ^{1}H power at $(\omega_1/2\pi)$ ~35 kHz and varying the ^{13}C power, Figure 2-17. Measurements were performed at a sample rotation rate of 10 kHz on adamantane samples that were 3.7 and 11.7 mm in length. The symmetry of the peaks indicates that the homogeneity and the overlap of the B_1 fields are quite good. For the longer samples the matching conditions become broader and less symmetric but are still considerably better than multiply resonant coils, particularly those that are not properly balanced [5]. This emphasizes the importance of homogeneity for efficient cross polarization. The similarity of the measured profiles to an ideal profile is striking given that the two B_1 fields are generated by two orthogonal coils with different geometries.

Performance with a Biologically Relevant Sample

To illustrate the performance of the Low-E MAS probe we collected a standard CP-MAS spectrum on a microcrystalline protein. Lysozyme crystals were made by dissolving lysozyme at a concentration of ~50 mg/mL in deionized water. To this solution an equal volume of precipitation solution containing 3 M NaCl in 50 mM Tris buffer, pH 7.0 was added. The 1:1 mixture of protein solution and precipitation solution was left to evaporate at 40 °C for 2 days leading to crystal formation. The crystals were then collected by centrifugation and loaded into a rotor. A CPMAS spectrum, Figure 2-18, verifies the crystallinity of the sample.

Performance Using Multipulse Windowless RF Sequences

DQDRAWS (Double Quantum Dipolar Recoupling in A Windowless Sequence) experiments [30] were performed on a tripeptide, *G*AV, which is ^{13}C' enriched on the first two amino acids (the enriched peptide is diluted to 10% with natural abundance peptide and crystallized). These experiments are useful at intermediate rotation rates (5-8 kHz) for determining torsion angles in peptides [31], yet, like many recoupling sequences, their performance is dependent on the availability of strong, homogeneous B_1 fields [32]. For these experiments, the CP contact time was 2.2 ms, cw decoupling was applied during the mixing period and SPINAL64 decoupling [20] was applied during acquisition. The MAS rate was 5 kHz, which corresponds to a $\pi/2$ pulse of 5.88 ms for the rotor-synchronized DRAWS sequence. Even with a full rotor (sample length 11.7 mm) and 5 kHz rotation rate (~104 kHz ^{1}H field and 42.5 KHz ^{13}C field) a DQ excitation efficiency of >25% was achieved for the enriched spins, which have a dipolar coupling of ~250 Hz. A 2D DQ-CSA spectrum was collected with a t_1 increment of 20 μs and evolution of 5.12 ms with 48 scans per slice and demonstrates the improved resolution of these experiments with high magnetic field, Figure 2-19.

DQDRAWS data were also collected on a peptide in a lipid environment to demonstrate the utility of the low-E probe in examining protein structures under conditions in which the protein is inherently dilute. The sample consists of a 21-amino acid peptide (KL_4) isotopically enriched at two adjacent $^{13}C'$ positions and reconstituted in a lipid environment at a lipid:peptide molar ratio of >50:1. The sample was packed into a 6.7 mm length as this gave a reasonable filling factor for the amount of sample available. The DQ-filtered spectrum of this sample, Figure 2-20, demonstrates the advantages of using a selection filter to simplify complicated spectra. The 2D DQ-CSA data validate the ability of this technique at high fields to isolate and characterize different peptide conformations if suitable RF fields are available without compromising the integrity of the sample. Figure 2-20 also shows the improvements that can be had in fitting CSA-CSA correlation data by moving to a higher B_0 irrespective of the probes used due to the increased spans of the CSAs.

Table 2-1. Calculation of heat dissipation rates, q_{heat}, per kHz^2 or RF field.

Channel	Q_{NL}	Q_{BIO}	η_{NL}	q_{heat}
^{1}H	147	141	46.7	0.91
^{13}C	119	90	64	5.0

Q values listed are at -3 dB. Q values were also measured at -7 dB to verify consistency.

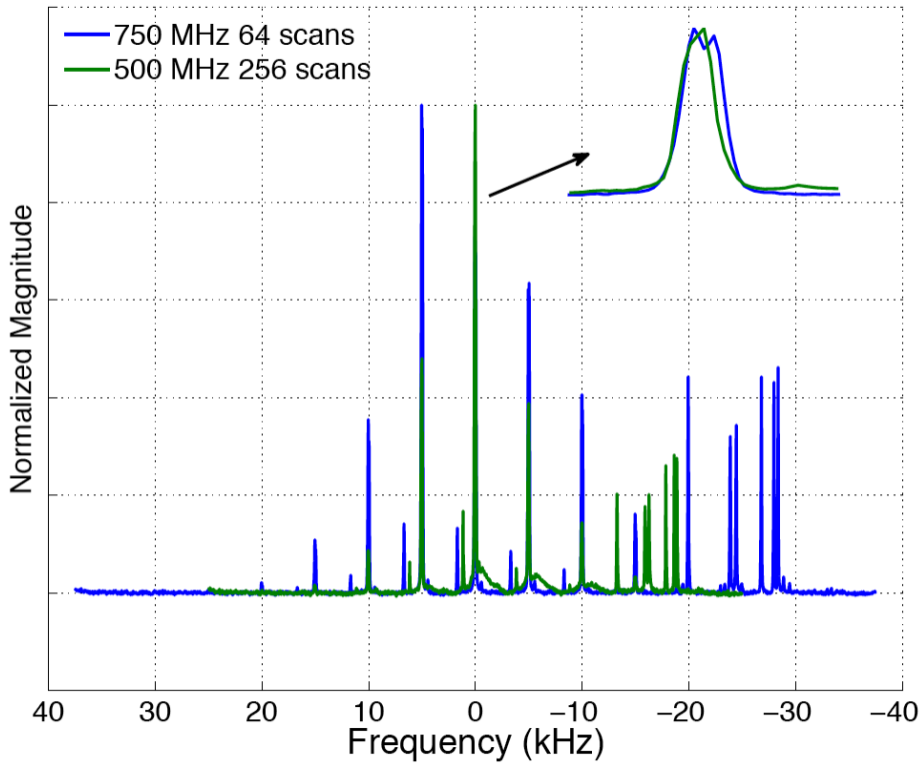

Figure 2-1. Spectra of ~25-30 mg of *G*GV peptide spinning at 5 kHz MAS at 500 MHz and 750 MHz. The green spectrum is taken at 500 MHz (11.7 T) and the blue spectrum at 750 MHz (17.6 T). The 750 MHz spectrum covers a wider set of frequencies (compare peak patterns) and has finer detail as shown in the subplot of +/- 500 Hz from the carbonyl resonance. The two carbonyl peaks cannot be distinguished at all at 500 MHz, but begin to separate at 750 MHz. They are even more distinguishable at 900 MHz. Also, the S/N of the two experiments is quite similar, as evidenced by the noise levels in the baselines, even though the spectrum at 750 MHz took 1/4 the time to acquire (64 scans vs 256 scans). The increase in CSA can be seen in the spinning sidebands of the 750 MHz data extending further out at higher magnitudes. The spinning sidebands are spaced at 5 kHz in both the 500 MHz and 750 MHz data.

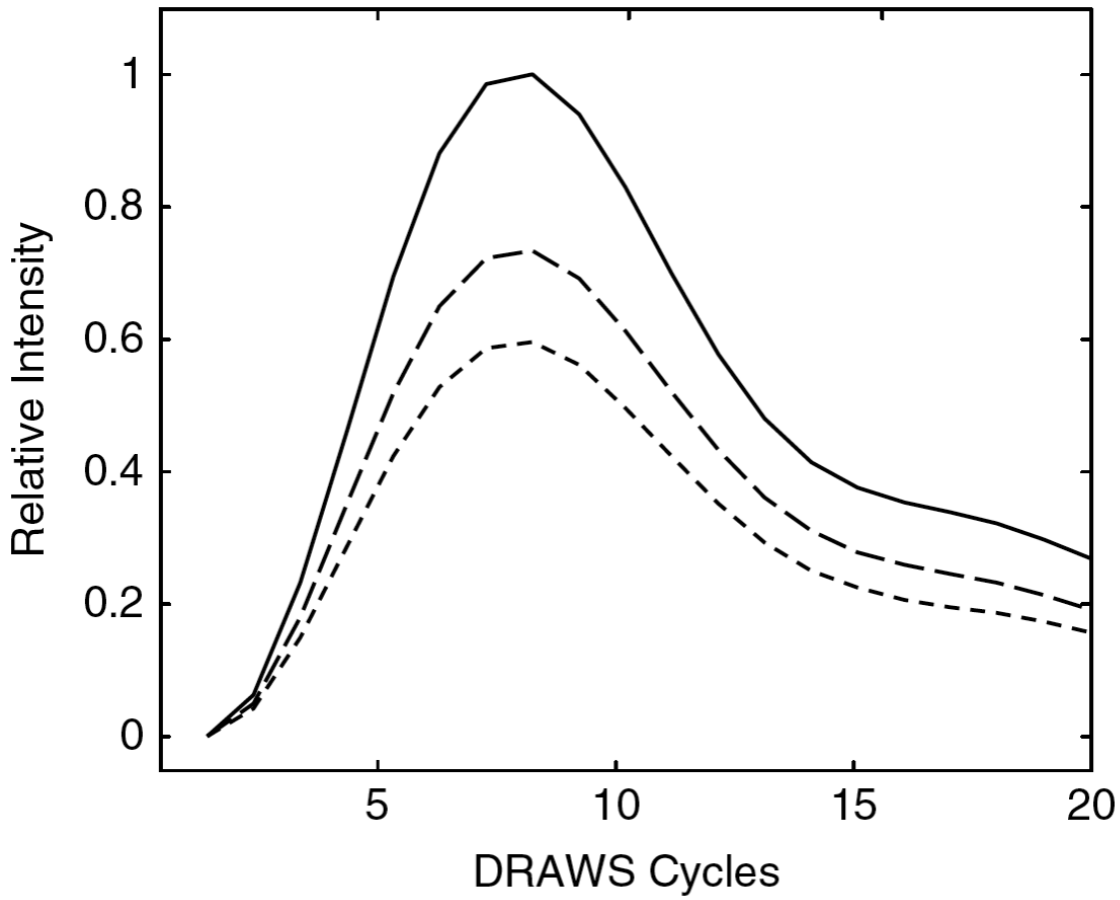

Figure 2-2. A double quantum recoupling experiment (DRAWS) simulated performance as a function of ^{13}C RF homogeneity at 750 MHz. Shown is the relative intensity of DQ-filtered signal as a function of mixing time. The curves are calculated for probes with relative homogeneities (810/90) of 95% (solid) 75% (long dashes) and 50% (short dashes). Improved homogeneity, particularly between 75% and 100% can significantly improve the amount of signal. This figure is used with permission from S. McNeill, P. Gor'kov, J. Struppe, W. Brey, and J. Long, "Optimizing ssNMR experiments for dilute proteins in heterogeneous mixtures at high magnetic fields," *Magnetic Resonance in Chemistry,* vol. 45, Dec. 2007, Figure 11 pp. S219.

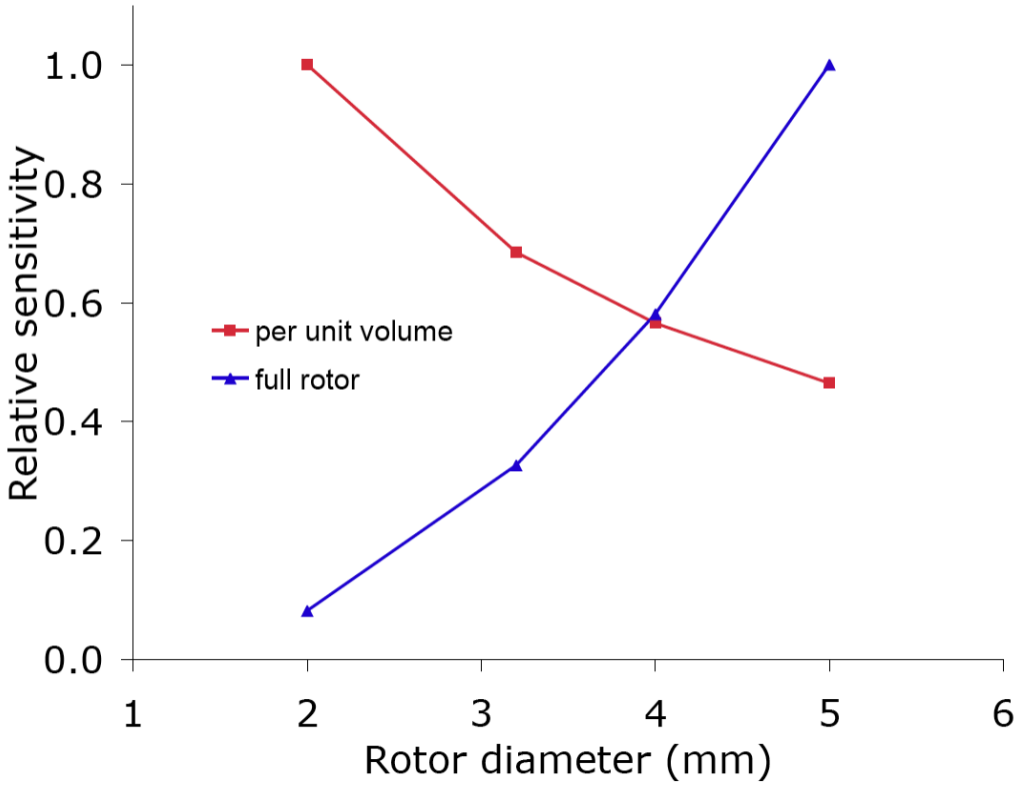

Figure 2-3. Relative sensitivity as a function of rotor diameter. The relative sensitivity of ssNMR as a function of rotor diameter increases as d^2 when looking at full rotor signal sensitivity, where d is the diameter of the rotor. It decreases as $1/d$ when sensitivity is measured per unit volume. The following assumptions were used in calculating this figure: 1) the solenoid coil diameter is 0.6 mm greater that the rotor diameter, 2) the length of each coil is twice its diameter, 3) a low-E type circuit is used on the ^{1}H channel to minimize wavelength effects, 4) the rotor has a wall 0.4 mm thick, 5) B_1 homogeneity remains constant for each rotor diameter since the coil length-to-diameter ratio remains constant, 6) the sample is constrained to 0.8 times the length of the coil. This figure shows that for mass limited samples, the smaller the rotor diameter the better, but for concentration limited samples, larger diameters are better. The sweet spot for both considerations is around a 4 mm diameter rotor.

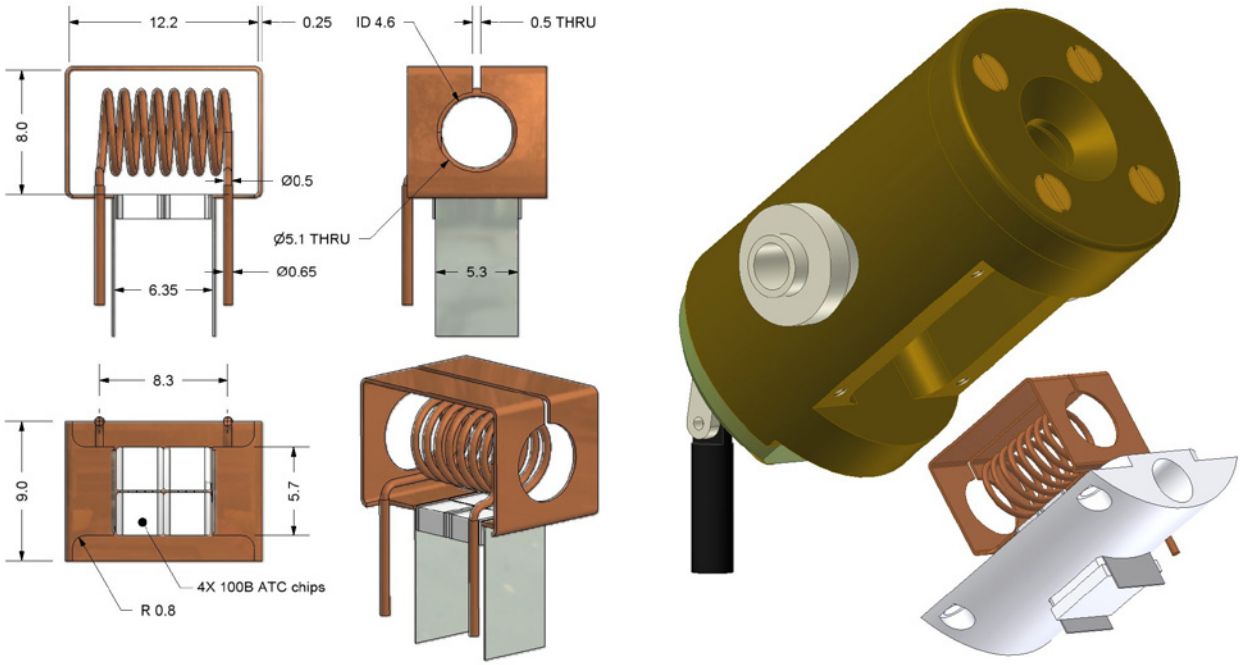

Figure 2-4. The dimensions of the coil assembly are in millimeters. The leads of the ^{13}C solenoid coil press fit into the Teflon platform, holding it suspended inside the ^{1}H LGR.

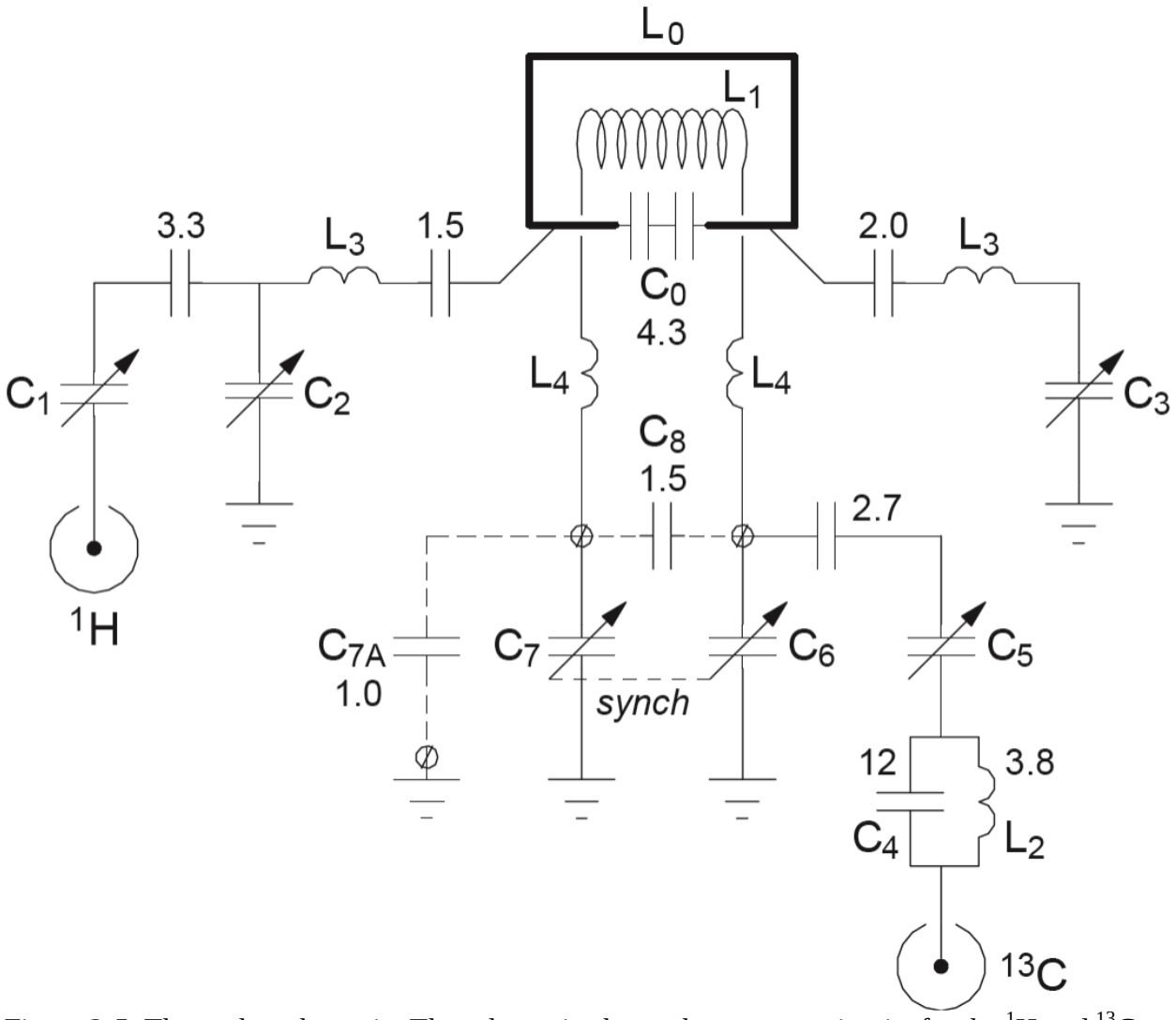

Figure 2-5. The probe schematic. The schematic shows the separate circuits for the ^{1}H and ^{13}C matching networks. Isolation is achieved through the orthogonal placement of the coils. L_0–C_0 forms the ^{1}H loop gap resonator with the detection solenoid inside (L_1). Inductors L_3 and L_4 (5–10 nH each) represent flexible leads connecting sample coils to the ^{1}H and ^{13}C circuits. C_1, C_2, and C_3 are variable capacitors for, respectively, matching, balancing, and tuning the ^{1}H LGR. In the low-frequency channel, C_5 is used for matching while C_6 and C_7 tuning capacitors are connected to a single tuning rod via a gear mechanism. Retuning to different observe nuclei (e.g. ^{15}N) is done by replacing a tuning chip, C_8, and a balancing chip, C_{7A}. A low-voltage ^{1}H rejection trap, L_2–C_4, is placed at the entry of the ^{13}C RF cable.

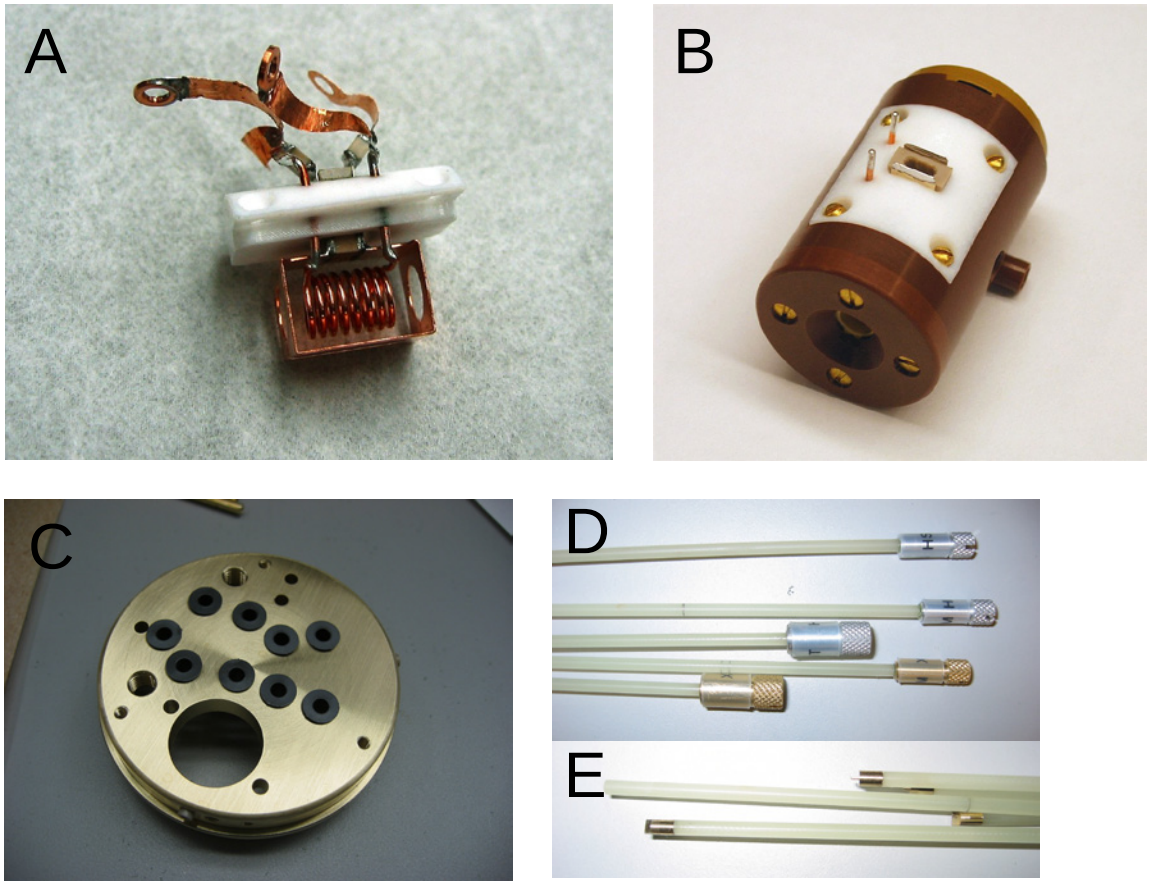

Figure 2-6. Pictures from assembling the probe. A) The coil assembly before inserting into the stator has the flexible leads attached and the external capacitors on the leads. B) After the coil assembly is inserted into the bottom of the stator the ^{13}C lead (wires) and the ^{1}H leads (plates on either side of the brown block in the middle of the Teflon) are visible. The brown (PEEK) block has a hole in it for cooling the LGR capacitors. C) The top plate for the body tube has guide and attachment holes. The holes that have tuning rods going through them have plastic lining to prevent wear. D) The handle ends of the tuning rods have different knobs depending on their use and are individually labeled. E) The tuning end of the tuning rods have metal screwdriver type ends epoxied on. The rod without a metal end will have a gear attached for turning the two ^{13}C tuning capacitors. F) The tuning capacitors are mounted on a copper plate. G) The RF lines are carefully bent around a hard object to line the connectors up with holes on the probe base. Note the small diameter air lines coming out as well.

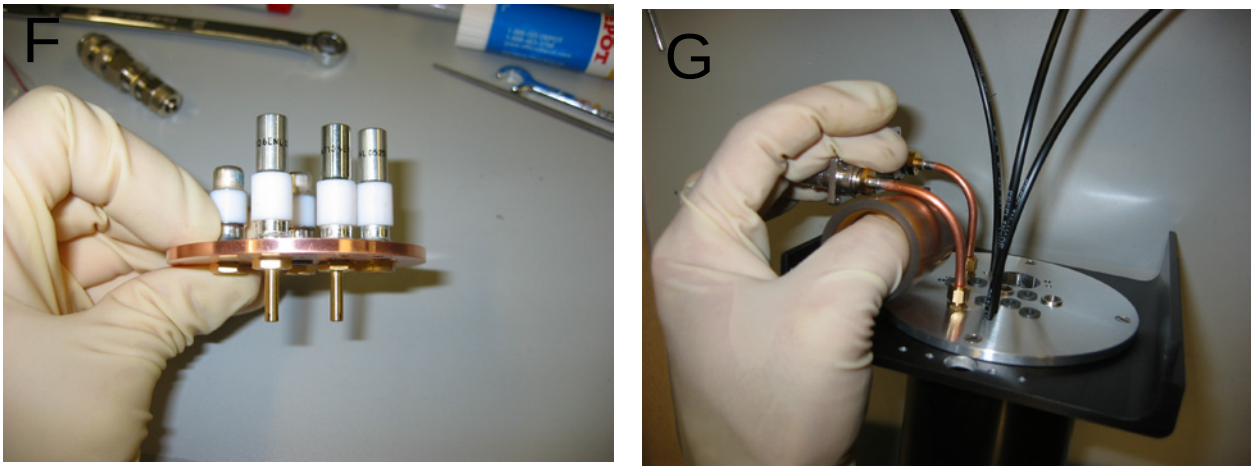

Figure 2-6. Continued

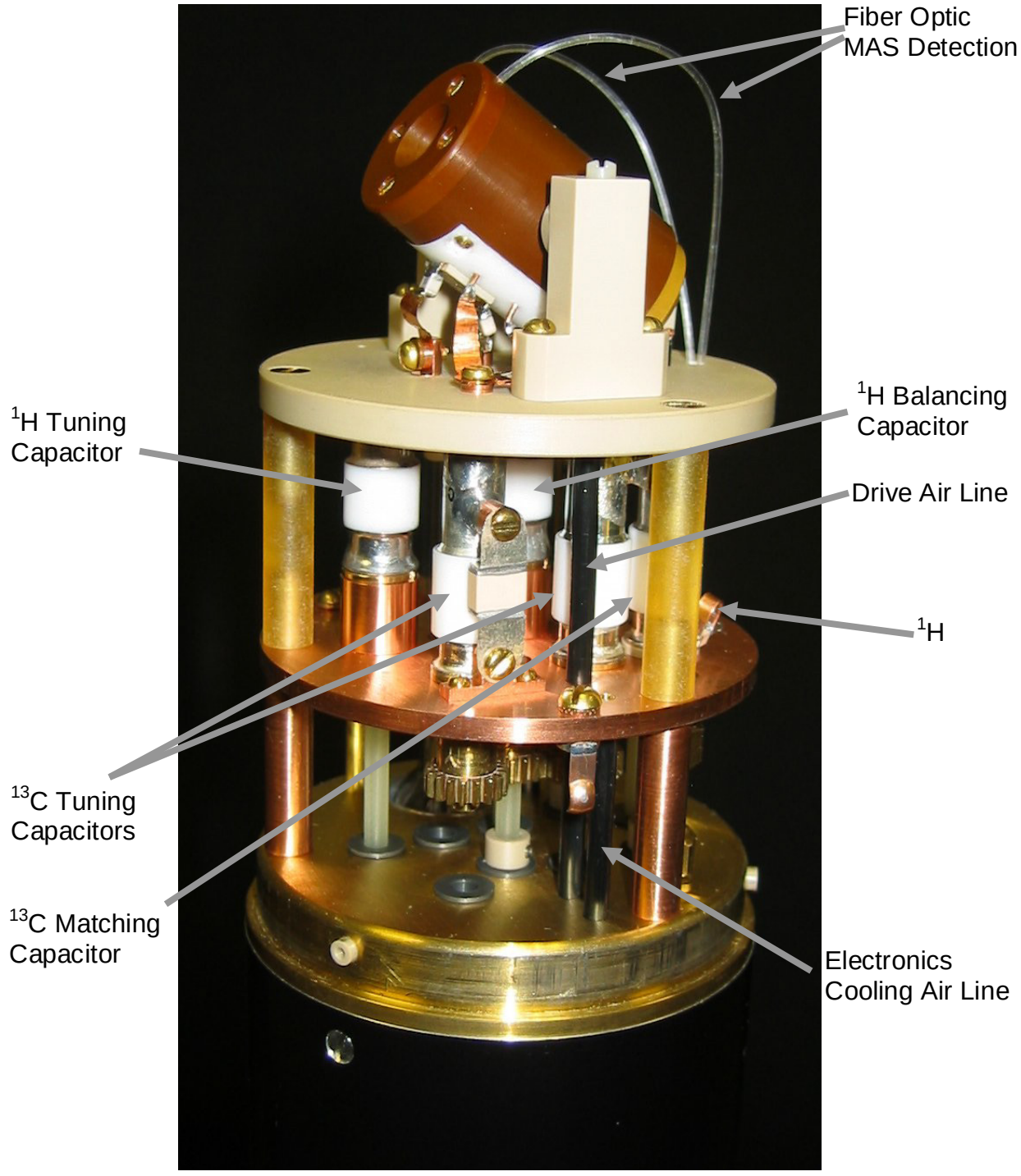

Figure 2-7. Labeled picture of the probe head.

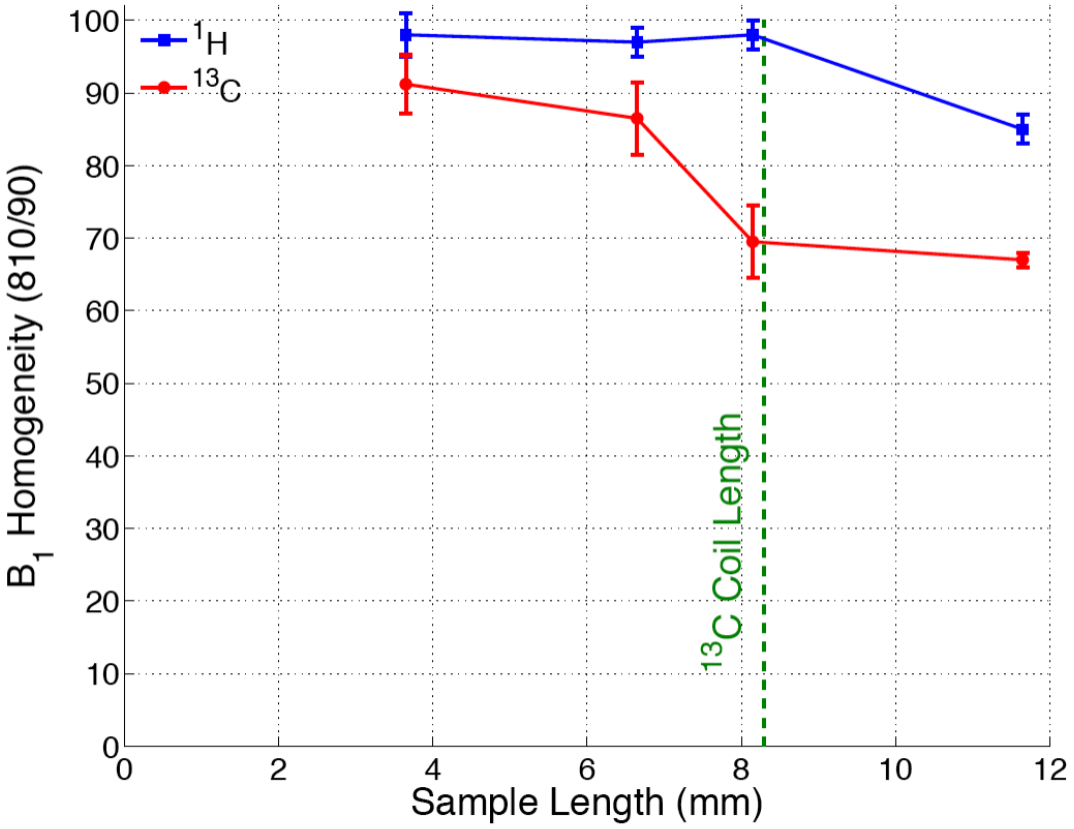

Figure 2-8. Homogeneity plot for both channels. The B_1 homogeneity characteristics for the ^{1}H (blue squares) and ^{13}C (red circles) channels as a function of sample length is high and improves when the sample is confined within the coil, as expected. The ^{13}C solenoidal coil length, 8.3 mm, is indicated by the green, dashed, vertical line.

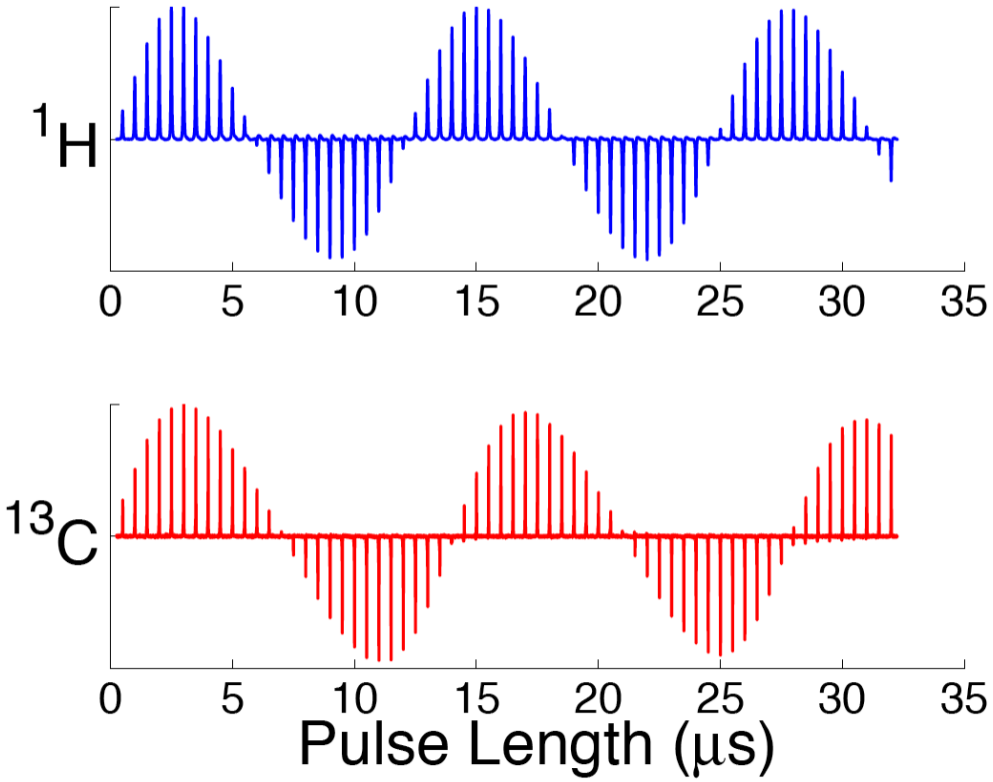

Figure 2-9. Nutation profiles for both channels. Example nutation profiles for the ^{1}H (blue) and ^{13}C (red) channels collected using a 6.7 mm long adamantane sample show that the homogeneity is very good on both channels even for this sample length. Each peak corresponds to the monitored resonance as a function of the pulse length in 0.5 µs increments. The ^{13}C profile used 3 s recycle delays, which is a little short for adamantane. This is why the peaks of the profile are biased rather than perfectly sinusoidal.

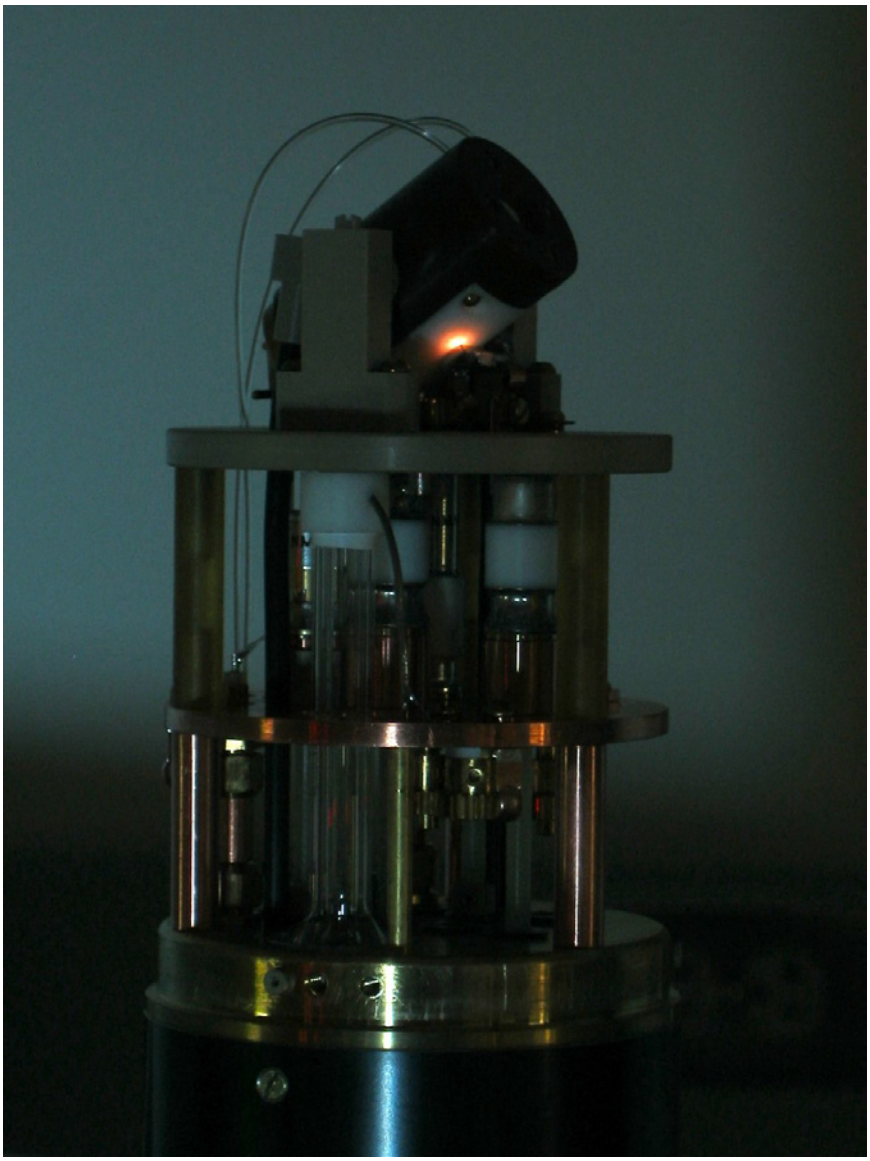

Figure 2-10. Probe arcing. During initial testing of the probe it arced seriously. This is a picture of what a probe is not supposed to do. The arc was across one of the capacitors in the ^{1}H LGR circuit. The arcing scorched the Teflon platform. The scorched parts of the platform had to be removed to prevent rearcing in those spots.

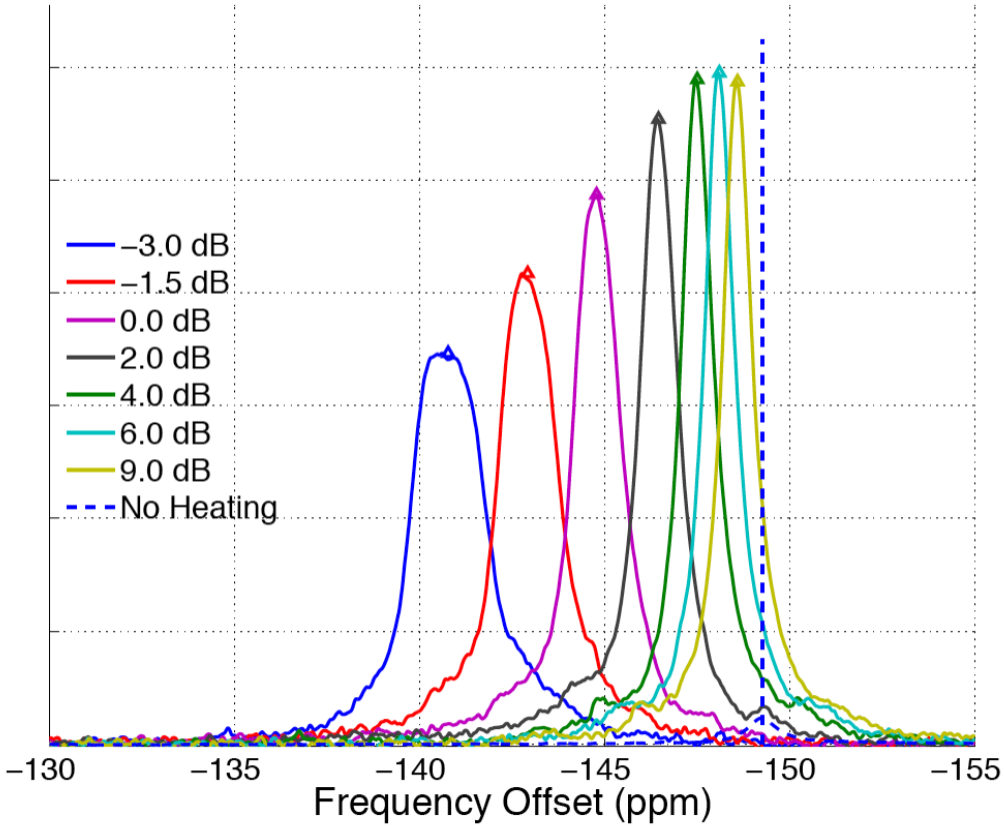

Figure 2-11. Example TmDOTP peaks with increasing RF input. An example TmDOTP H(6) heating run shows the peaks shift toward 0 ppm as the temperature increases. The temperature spread also increases as indicated by the increasing peak width. The temperature spread is a result of the cooling effects of the VT and MAS competing with the RF heating. The no heating peak is used as a reference point and the temperature of the sample is estimated using a slope from that point of 0.87 ppm/K from Zuo et al [23].

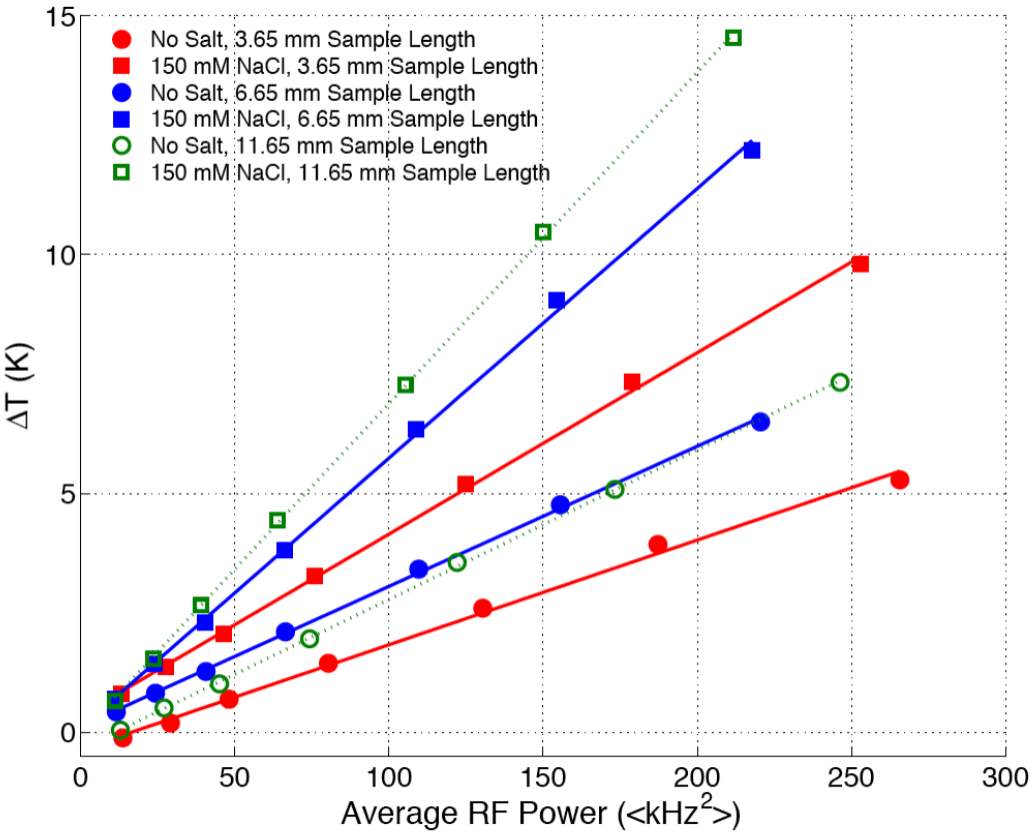

Figure 2-12. RF heating. The average RF sample heating at three different sample lengths and two salt concentrations are shown. Red filled circles are 20 mM $TmDOTP^{5-}$ with a 3.7 mm sample length. Red filled squares are 20 mM $TmDOTP^{5-}$ and 150 mM NaCl with a 3.7 mm sample length. Blue filled symbols correspond to the same solutions with a 6.7 mm sample length, and the empty, green, symbols correspond to the same solutions with an 11.7 mm sample length. Lines are linear fits to the data as a visual guide. Note that because of its high ionic strength, small concentrations of $TmDOTP^{5-}$ still contribute significantly to RF loss. The RF heating remains under 15 K for all samples and either decreasing the sample length or the salt concentration further reduces the heating. Average power was varied by keeping the duty cycle constant at 3.8% and varying the presaturation pulse power. A delay of 5 ms between the presaturation pulse and the acquire pulse was used and 256 dummy scans (~5 min) were run before acquiring to make sure the sample had reached equilibrium temperature.

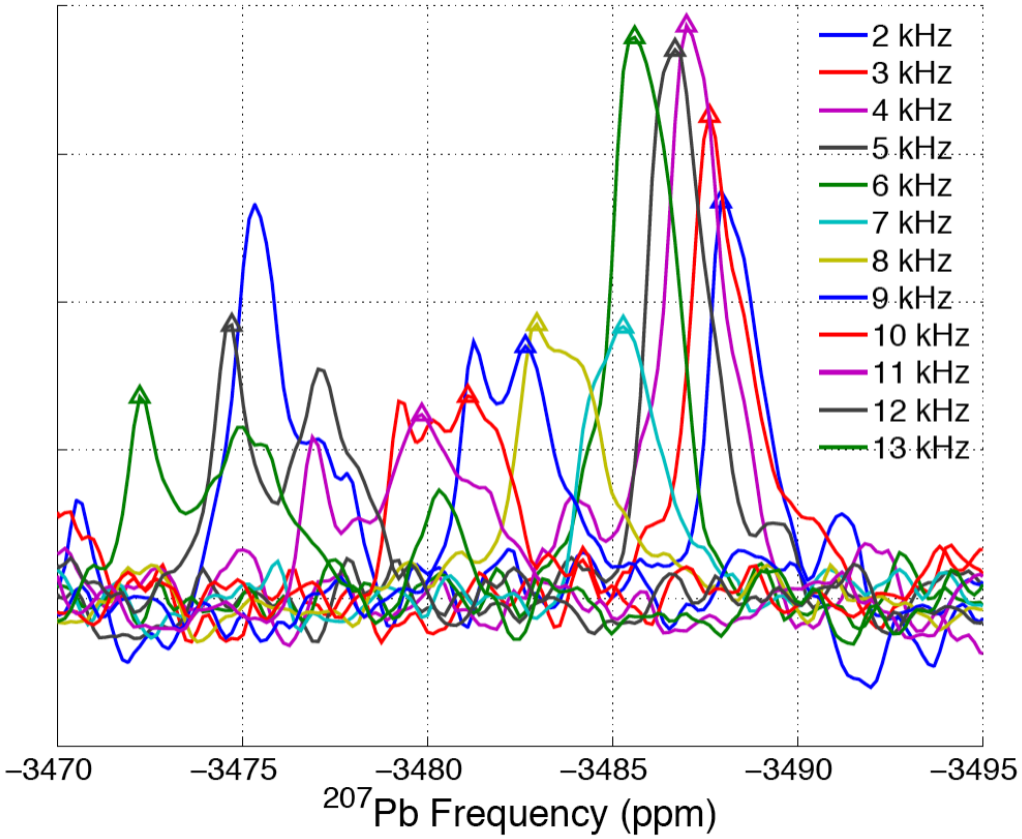

Figure 2-13. MAS heating spectra. These ^{207}Pb spectra are used to create the MAS heating curve shown in Figure 2-14. The peaks picked for the data are indicated by the triangles. The temperature gradient across the sample increases as MAS rate increases. Note that the left peak gets taller than the right peak above 11 kHz. The temperature is equal to 1.32*(peak location + 3714) [33].

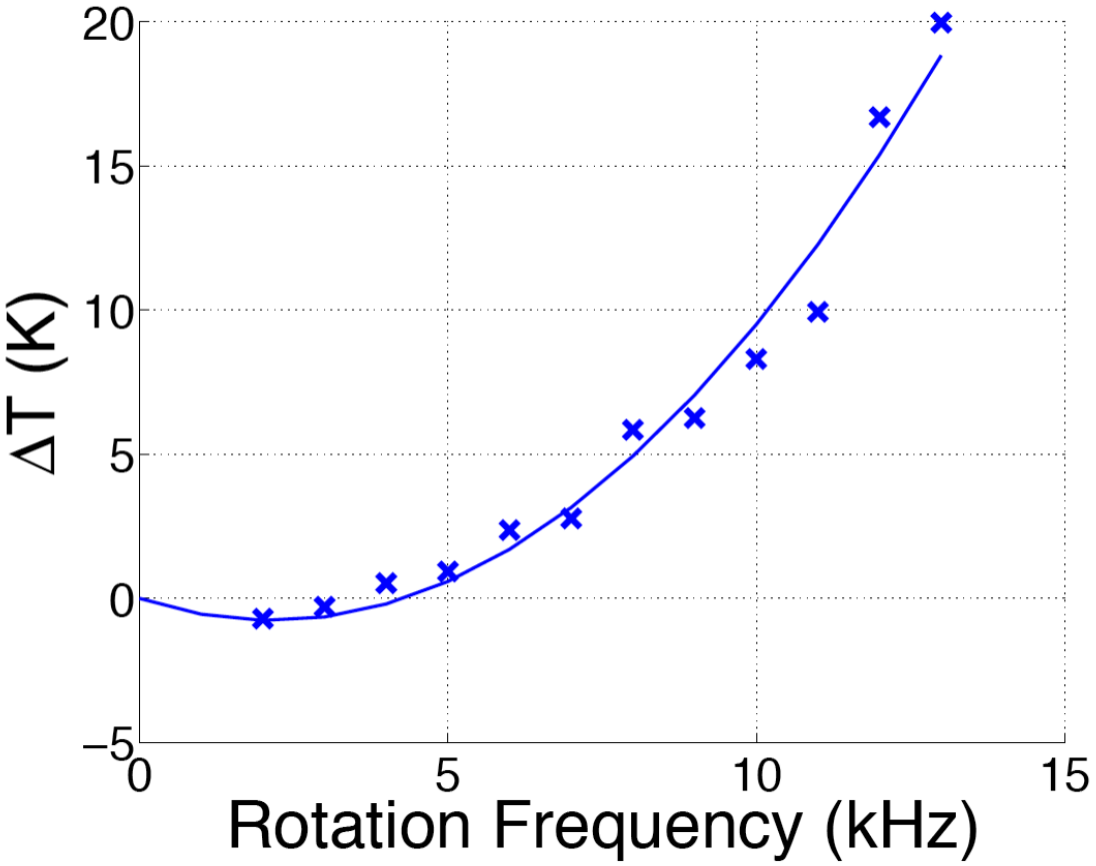

Figure 2-14. MAS heating. The frictional heating due to MAS reaches 20 K for speeds of 13 kHz, which is enough to be a problem for biological samples, but it can be mitigated by cooling the rotor with chilled VT gas. The sample temperature was monitored using a sample containing 10% lead nitrate ($Pb(NO_3)_2$) diluted with NaCl to reduce its density. The gas lines were kept at room temperature. The line is a quadratic fit through the origin. The initial temperature drop is due to Joule-Thomson cooling.

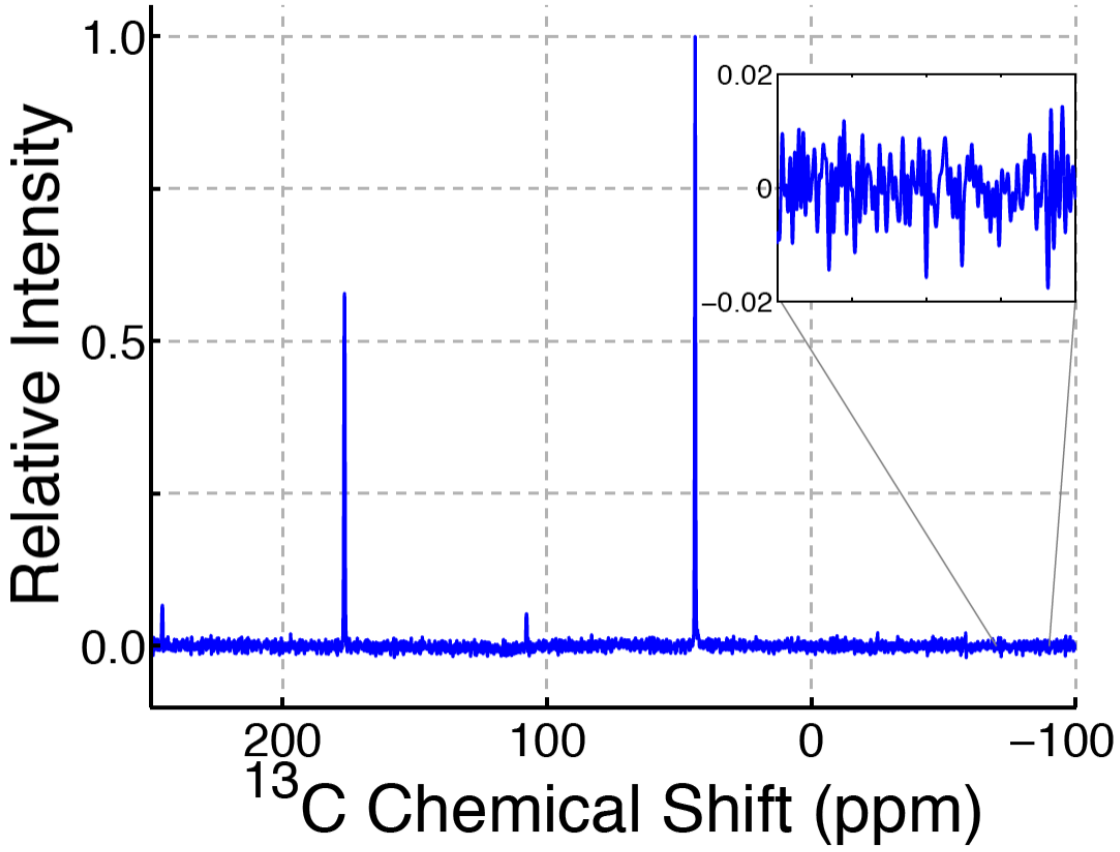

Figure 2-15. Glycine spectrum for measuring S/N. This is a ^{NA}GLY ^{13}C CPMAS spectrum after 8 scans with a sample length of 6.9 mm containing 67.3 mg of glycine spinning at 13 kHz. Inset is a magnification of the noise used in the S/N measurement. This particular spectrum has a S/N measurement of 301.

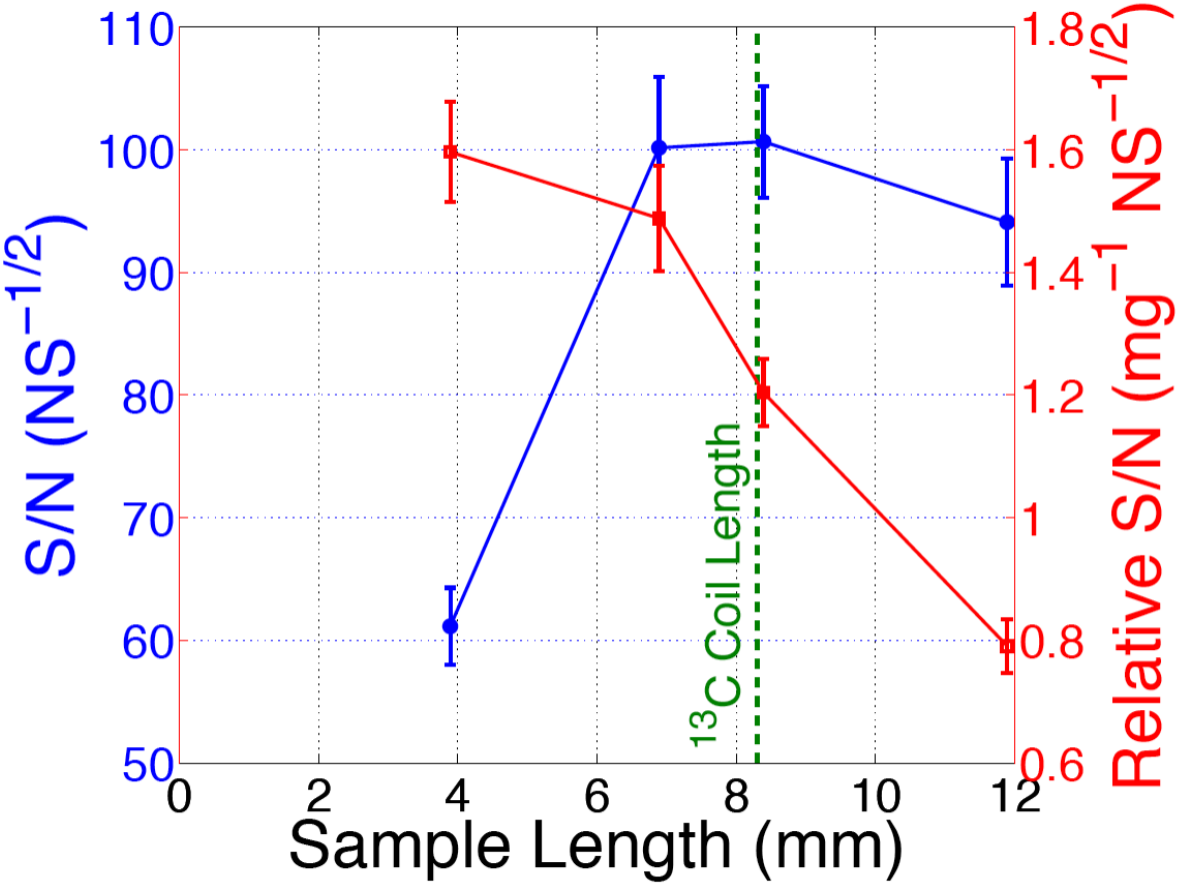

Figure 2-16. Signal-to-noise for various sample lengths. This shows S/N for ^{NA}GLY as a function of sample length. Shown are the S/N normalized for the number of scans (solid blue circles, left axis) as well as the S/N per unit mass (open red squares, right axis), also normalized for the number of scans.

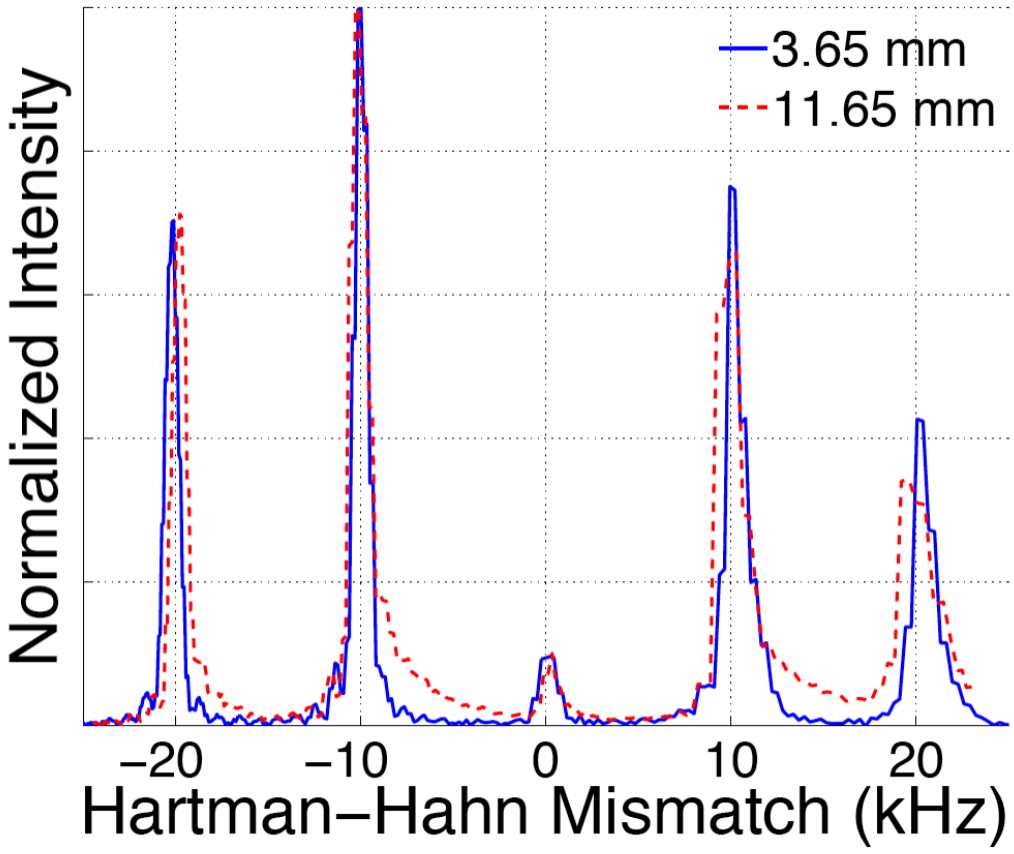

Figure 2-17. CPMAS matching condition profile for adamantane at 10 kHz MAS using a square spinlock pulse on both RF channels. The ^{1}H RF field was held constant at $(\omega_1/2\pi)$ = 35 kHz while the ^{13}C B_1 field was varied; the signal is graphed as a function of the RF mismatch. The solid blue line corresponds to a sample length of 3.7 mm; the dashed red line corresponds to a sample length of 11.7 mm. The X-axis is the average ^{13}C B_1 field minus the average ^{1}H B_1 field. The longer sample (11.7 mm, dotted line) has broader, weaker matching conditions. This is from the wider range of RF energy the sample is exposed to due to the lower homogeneity of the ^{13}C B_1 field. The shorter sample (3.7 mm, solid line) shows excellent agreement with theory with maximal signal at mismatch levels equal to integer values of the spinning speed.

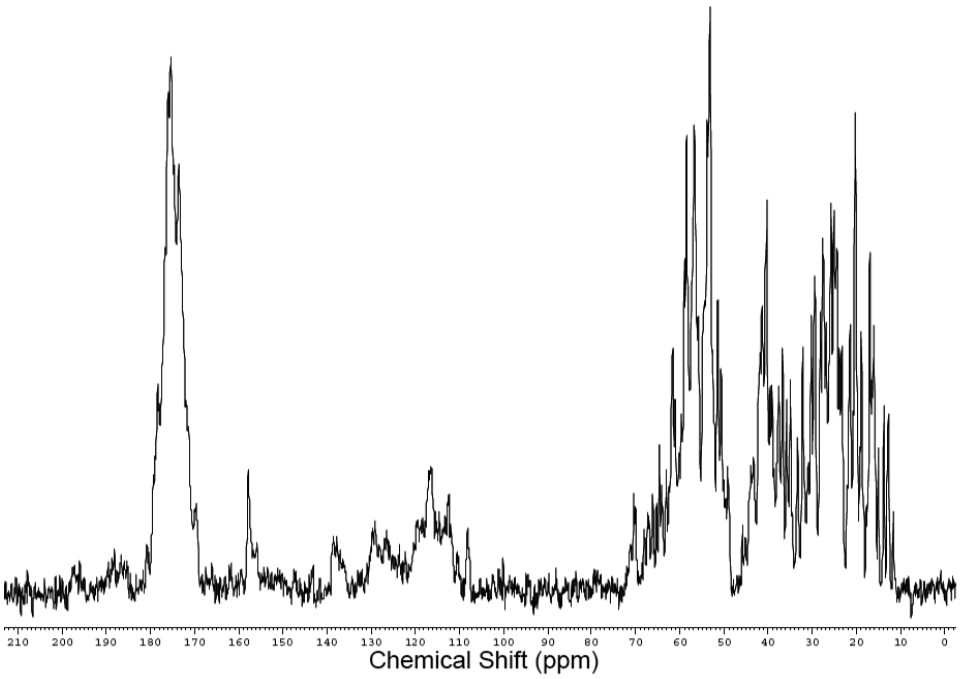

Figure 2-18. The ^{13}C CPMAS spectrum of nanocrystalline natural abundance lysozyme demonstrates the probe's capabilities on a crystalline sample.

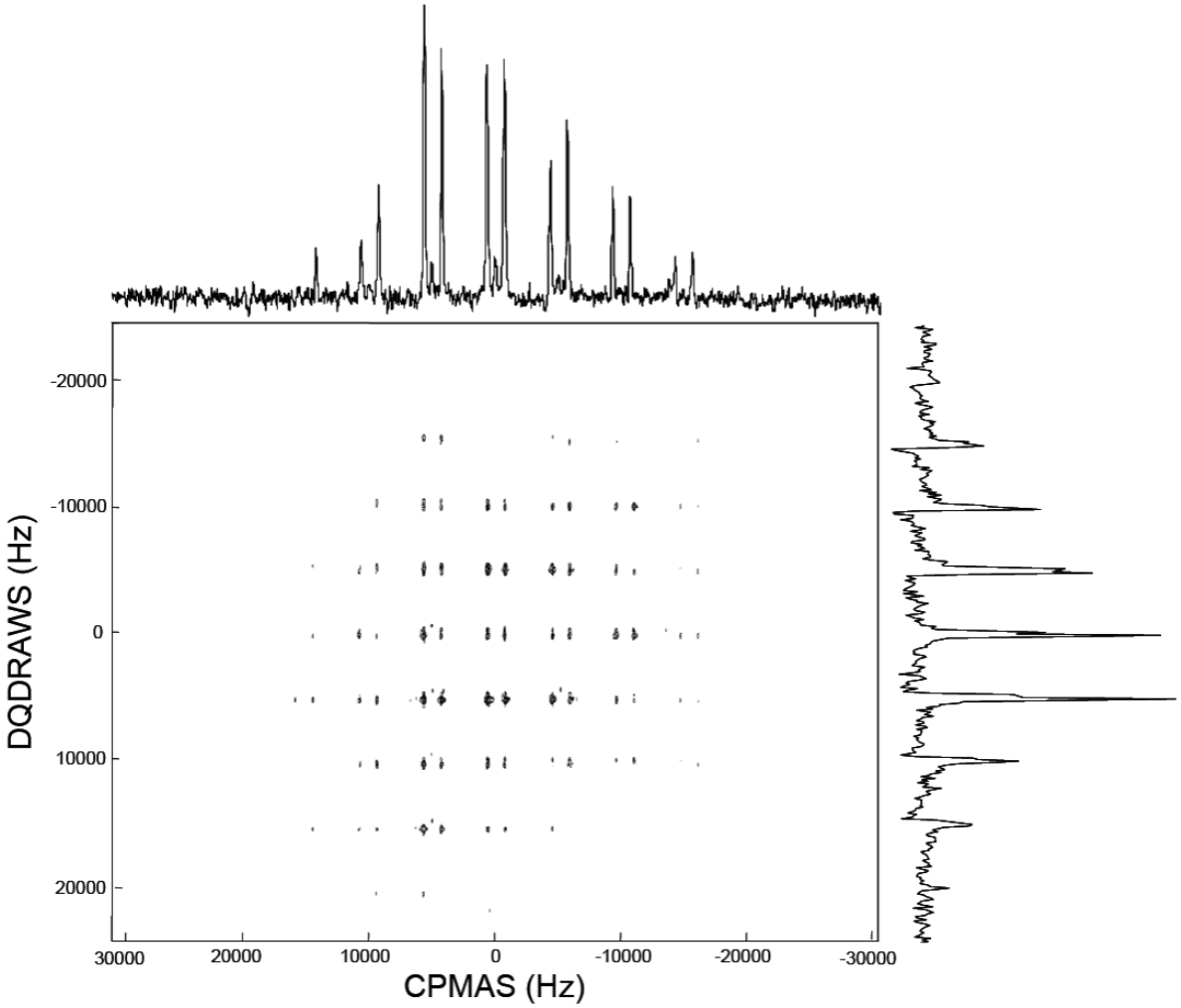

Figure 2-19. A 2D DQ-CSA spectrum of *G*AV demonstrates a dipolar recoupling experiment.

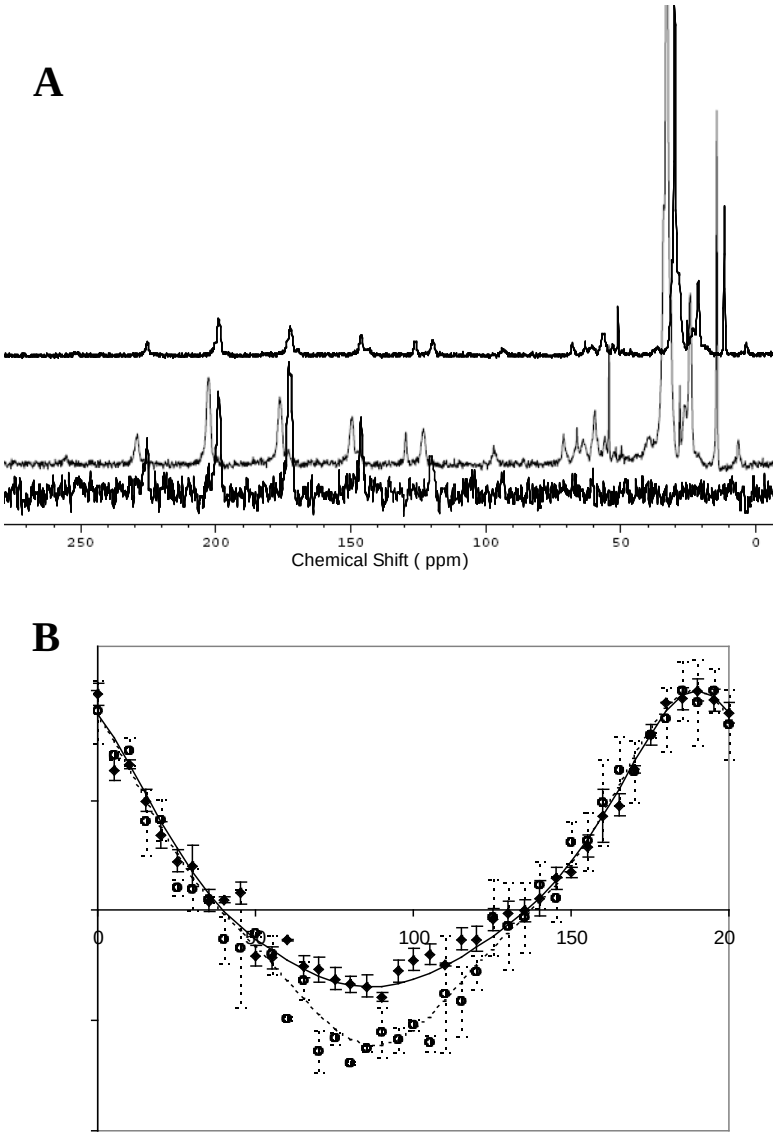

Figure 2-20. An example showing the improvements in signal quality in going to higher field. A) CPMAS and DQ-filtered spectra for the 21 amino acid peptide KL_4 13C'-enriched at positions L9 and L10 and incorporated into DPPC:POPG lipid vesicles at a peptide:lipid molar ratio > 1:50; the signals in the aliphatic region are primarily from the surrounding lipids. B) 2D DQ-CSA correlation data for the KL_4 sample along with best fit simulations. Closed and open symbols are data collected at 750 and 500 MHz, respectively; lines correspond to the signal trajectories for the best fit (ϕ,ψ) simulations at the two fields. C) χ^2 evaluation of simulations with varying ψ while holding ϕ at a value obtained from DQ buildup experiments; solid and dashed lines correspond to fitting of data at 750 and 500 MHz, respectively. Note the improved selectivity of the χ^2 evaluation at 750 MHz due to the increased CSAs at the higher magnetic field.

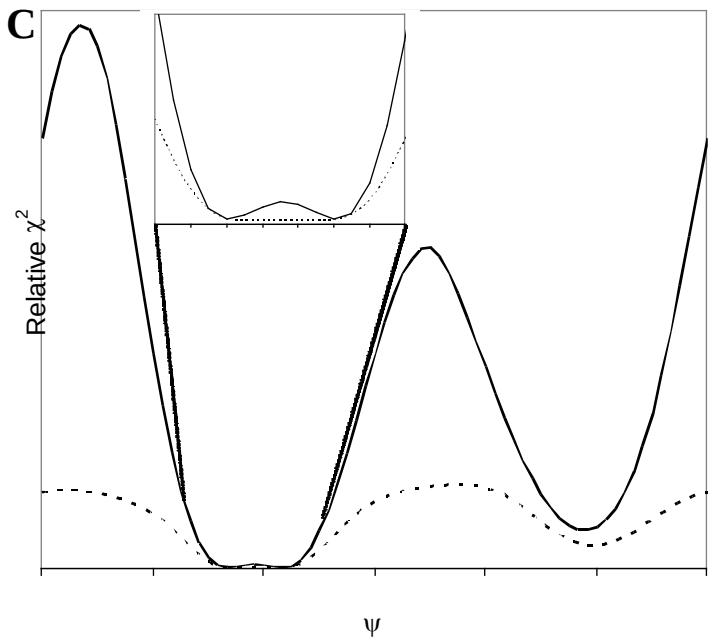

Figure 2-20. Continued

CHAPTER 3
RF EXCITATION PROFILES BACKGROUND AND PRIOR ART FOR OPTIMIZATION

Introduction

One method of optimizing NMR experiments is by improving the hardware used to run the experiments. NMR experiments can also be improved by optimizing the way NMR experiments run. An NMR experiment consists of transmitting RF energy into a sample and then detecting the resultant RF signal the sample emits. The RF signal transmitted into the sample can be as simple as a single pulse at a single amplitude, frequency, and phase, or it can be many hundreds of pulses each at a different amplitude, frequency, and phase.

Experimental setups are never perfect. As discussed in Chapter 2, the RF fields are not perfectly homogeneous. Also, the static magnetic fields are not perfectly homogeneous. The RF pulses have finite lengths and are not perfect in phase, frequency or amplitude. Even in a homogeneous magnetic field the various nuclei in an experiment may be resonant across frequencies spanning many kHz due to the chemical shielding described in Chapter 1. In fact, these frequencies are often the information desired from the experiment. However, exciting all of the nuclei uniformly across wide bandwidths can be difficult. Ideally, one could transmit exactly on resonance for all the nuclei of interest and transmit nothing at the resonant frequencies of all other nuclei. In practice this is not possible. Many experiments have been designed to measure the various properties of molecules; the goal of this research is to take some of these experiments and rebuild them to be more tolerant of the imperfections in the experimental process. In Chapter 2, hardware development addressed the problem inhomogeneity of the RF field around the sample. In this chapter I explore the effect of the frequency offsets due to chemical shifts.

A thorough understanding of the relationship between RF pulses and frequency offsets set the stage for Chapter 4 in which compensated pulses are designed and optimized by using computational optimization to design pulse sequences more tolerant to frequency offsets.

The goal of computational optimization is to find the sequence of RF energy that when transmitted into the sample, maximizes the signal of interest from the sample in the presence of various imperfections of the system. The RF energy has four main variables: length of time applied, magnitude, frequency, and phase.

Typically, the RF energy is not constant in an experiment, but is pulsed, which means that each of these parameters can and may be varied throughout an experiment. However, there are practical limitations. The sample and the hardware determine the overall length of time the RF can be applied, the maximum magnitude used, and the speed with which the phases and amplitudes can be changed. Exceeding the maximum length of time or magnitude can cause probe arcing and/or sample degradation through heating.

The RF frequencies used in an NMR experiment are determined by the nuclei of interest, their molecular environment, and the strength of magnet used. The base frequency for a nucleus of interest is determined by a (or magnetogyro) ratio characteristic for that nucleus times the strength of the magnetic field. For ^{1}H nuclei in a 17.6 T field, that frequency is 750 MHz; for ^{13}C nuclei in a 17.6 T field, that frequency is 188.6 MHz. The molecular environment around each nucleus also affects its resonant frequency; at 17.6 T variations of up to 40 kHz are seen for ^{13}C nuclei in biomolecules.

The phase of the RF, then, is the only completely free variable. Therefore, the goal of computational optimization is to find the sequence of RF pulses, defined by their length, magnitude, frequency, and phase, for a given set of nuclear parameters to obtain the desired

information about the nuclei using a NMR simulator. Once a suitable solution has been found, it is tested on a spectrometer to verify that it really works experimentally.

Mathematical Representations of NMR Experiments

To simulate an NMR experiment, one must have some understanding of the math behind NMR and how to accurately model an experiment using a simulator. Accurately modeling an experiment via simulation requires understanding NMR experiments, usually learned at an NMR facility, and how to run a simulator, usually learned from the manuals published about them. This section gives a brief overview of the mathematics used to model NMR experiments to aid in understanding both the simulations and possible approaches to optimization.

A quantum mechanical property of atomic nuclei is spin angular momentum, specified by their spin number. Nuclei with nonzero spin numbers are said to have nuclear spin and are NMR sensitive due to transitions between nuclear spin states. In the absence of a magnetic field, these states are equal in energy. When a magnetic field is applied, these spin states are no longer degenerate with an energy separation defined by the Larmor frequency, which is equal to minus the gyromagnetic ratio of the nucleus times the magnitude of the magnetic field. Since this is a resonant frequency, the applied RF is usually as close to the Larmor frequency as possible. Describing what happens when RF is applied is easier if we switch from a laboratory frame of reference, to a frame of reference rotating at the frequency of the applied RF. If the applied RF is at the Larmor frequency (on resonance), its effects can be described by simple rotations.

In the presence of a magnetic field the average magnetic field generated by a population of nuclei with spin will be aligned with the magnetic field and is referred to as bulk magnetization, which by convention lies along the z-axis at equilibrium. There are two main ways to track the bulk magnetization of spin-1/2 NMR active nuclei tumbling isotropically in a magnetic field using linear algebra. The first method uses the Pauli spin matrices as a basis set for a spin-1/2

system. This method follows the true quantum mechanical behavior of isolated spins most closely. The second method is to assume a population of nuclei with a net magnetization and to treat the bulk magnetization as a vector in 3D space and use vector rotation matrices to rotate the vector in 3D space in response to RF pulses and the resulting precession of the magnetization. In their simplest forms, both of these methods ignore relaxation and omit interactions added by the molecular environment of the nuclei.

Pauli Method

Spin-1/2 particles have 2 available states, usually represented by the vectors $[1\ 0]^T$ and $[0\ 1]^T$. Assuming a frame of reference in which the magnetic field is along Z, the Pauli matrices, Equations 3-1 through 3-3, are a convenient basis set for the superposition of states when operating on a spin-1/2 system.

$$I_x = \frac{1}{2}\begin{vmatrix} 0 & 1 \\ 1 & 0 \end{vmatrix} \tag{3-1}$$

$$I_y = \frac{1}{2}\begin{vmatrix} 0 & -i \\ i & 0 \end{vmatrix} \tag{3-2}$$

$$I_z = \frac{1}{2}\begin{vmatrix} 1 & 0 \\ 0 & -1 \end{vmatrix} \tag{3-3}$$

Each matrix can be thought of as representing the magnetization along one of the principle axes in 3D space. Bulk magnetization with arbitrary direction and magnitude can represented by linear combinations of the Pauli matrices as shown in Equation 3-4 where x_{arb}, y_{arb}, and z_{arb} are constants.

$$I_{arb} = x_{arb}I_x + y_{arb}I_y + z_{arb}I_z \tag{3-4}$$

Equation 3-4 can be thought of as the conversion from 3D Cartesian space to the 2x2 matrix quantum representation of the superposition of states. By convention, the static magnetic

field is along the Z-axis. The RF is applied in the X-Y plane and disturbs the system away from equilibrium. In the rotating frame, applying an RF field resonant to the energy splitting of a spin-1/2 particle rotates the bulk magnetization, initially at I_{init}, through an angle, α, about the axis of the RF field, I_{rot} as illustrated in Figure 3-1. The rate of this rotation is referred to as the nutation rate, ω. The nutation rate is the frequency (usually specified as $\omega/2\pi$ Hz) that the magnetization returns to I_{init} under continuous RF application. Nutation rate increases with applied RF power, so the stronger the field the higher the nutation rate. This leads to the confusing practice of referring to power levels in NMR by either RF power levels in the laboratory frame (in either watts or dB attenuation) or the nutation rate in the rotating frame. Nutation rate is the most standard method since it normalizes for any differences between NMR spectrometers and probes. The angle the bulk magnetization rotates through, α, is determined by the pulse length, measured in either units of time (usually μs) or angle (degrees or radians), and the RF nutation rate.

When the RF field is applied at exactly the resonant frequency of the nucleus of interest, i.e. "on resonance", the bulk magnetization rotates around I_{rot} through an angle of α degrees. However, if the RF field is off resonance, the bulk magnetization will experience a stronger effective field causing it to rotate through an angle greater than α around an axis rotated towards I_z from I_{rot}. This is because the off resonance part of the static magnetic field (B_0 in the Z direction) contributes to the magnetic field experienced by the spin in the rotating frame. This is explained by observing in the rotating frame. If a spin is rotating at a frequency different than the rotating frame is spinning at, it will appear to rotate along the Z axis at the difference between the rotating frame's frequency and the spin's frequency. The rotation around the Z axis can be modeled as an RF pulse around the Z axis. Thus, the two magnetic fields are at right angles and are combined through vector addition as illustrated in Figure 3-2. Thus, an RF signal applied $\Delta\omega$

from a spin's resonance with a power of ω_{nut}, rotates the spin with an effective magnetic field, ω_{eff}, which is the magnitude of the vector sum of $\Delta\omega$ and ω_{nut}, Equation 3-5.

$$\omega_{eff} = \sqrt{(\omega_{nut})^2 + (\Delta\omega)^2} \tag{3-5}$$

α_{eff} is the effective pulse length in degrees experienced by the spin, Equation 3-6.

$$\alpha_{eff} = \frac{\omega_{eff}}{\omega_{nut}}\alpha \tag{3-6}$$

The axis that the pulse drives the magnetization around is given by the vector sum of ω_{nut} and $\Delta\omega$. $\Delta\omega$ is always along the Z-axis since the rotating frame rotates around the Z-axis.

The bulk magnetization rotation that the RF pulse causes can be expressed mathematically as a complex exponential. For example, Equation 3-7, an RF pulse applied along the X-axis, or I_x, results in a rotation of α degrees around I_x.

$$\mathbf{R}_x(\alpha) = \exp(-i\alpha I_x) \tag{3-7}$$

Equation 3-7 can be represented using a matrix by doing a Taylor expansion of the exponentiation, Equation 3-8, where **E** is the identity matrix.

$$\mathbf{R}_x(\alpha) = \mathbf{E}\cos\frac{\alpha}{2} - i2I_x\sin\frac{\alpha}{2} = \begin{bmatrix} \cos\frac{\alpha}{2} & 0 \\ 0 & \cos\frac{\alpha}{2} \end{bmatrix} - \begin{bmatrix} 0 & i\sin\frac{\alpha}{2} \\ i\sin\frac{\alpha}{2} & 0 \end{bmatrix} = \begin{bmatrix} \cos\frac{\alpha}{2} & i\sin\frac{\alpha}{2} \\ i\sin\frac{\alpha}{2} & \cos\frac{\alpha}{2} \end{bmatrix} \tag{3-8}$$

Mathematically, the final position of the magnetization, I_{end}, after a pulse can be calculated by front multiplying the initial magnetization, I_z, by $\mathbf{R}_x$ and rear multiplying it by $\mathbf{R}_x^{-1}$ as shown in Equation 3-9.

$$I_{end} = \mathbf{R}_x(\alpha) I_z \mathbf{R}_x^{-1}(\alpha) \tag{3-9}$$

Due to the nature of $\mathbf{R}_x$, $\mathbf{R}_x^{-1}$ is just the complex conjugate of $\mathbf{R}_x$. Evaluating Equation 3-9 leads to Equation 3-10 for the final solution.

$$I_{end} = \frac{1}{2}\begin{vmatrix} \cos\alpha & i\sin\alpha \\ -i\sin\alpha & -\cos\alpha \end{vmatrix} \tag{3-10}$$

That solves the problem nicely for a rotation about I_x starting at I_z, but what is really needed is a more general solution for a rotation with an arbitrary starting point, rotation axis, and rotation angle. A general rotation about an axis, *A*, is shown in Equation 3-11.

$$\mathbf{R}_A(\alpha) = \mathbf{E}\cos\frac{\alpha}{2} - 2i\mathbf{n}\cdot\mathbf{I}\sin\frac{\alpha}{2} \tag{3-11}$$

n·I can be expressed in terms of ϕ, θ, and the Pauli matrices, Equation 3-12 and Figure 3-3.

$$\mathbf{n}\cdot\mathbf{I} = I_x\cos\phi\sin\theta + I_y\sin\phi\cos\theta + I_z\cos\theta \tag{3-12}$$

A rotation from I_{init} to I_{end} around axis A is implemented in Equation 3-13.

$$I_{end} = \mathbf{R}_A(\alpha)I_{init}\mathbf{R}_A^{-1}(\alpha) \tag{3-13}$$

It is important when using Equation 3-13 to remember to invert $\mathbf{R}_A$ rather than just take the complex conjugate like we could do for $\mathbf{R}_x$. Equations 3-11 and 3-13 allow for rotation about an arbitrary axis by an arbitrary angle and include off-resonance effects. The result is a 2x2 matrix and it is often handy to convert this result into Cartesian (*x*,*y*,*z*) components for visualization. Equation 3-14 shows how to do this.

$$(x,y,z) = 2 * real\left[trace(\mathbf{R}_A I_{init}\mathbf{R}_A^{-1}I_x), trace(\mathbf{R}_A I_{init}\mathbf{R}_A^{-1}I_y), trace(\mathbf{R}_A I_{init}\mathbf{R}_A^{-1}I_z)\right] \tag{3-14}$$

For further enlightenment, the best reading is Cavanagh et al [34], but some other good references are [1,35-38].

3D Vector Method

An alternate way of deriving the same information is to use the kinematic method of rotating vectors in 3-space. The rotation matrix for rotating an initial magnetization vector, I_{init}, about a unit vector $\mathbf{K} = [k_x\ k_y\ k_z]^T$ by α_{eff} degrees, is given in Equation 3-15 [39].

$$\mathbf{R}_K(\theta) = \begin{bmatrix} k_x k_x \upsilon\theta + c\theta & k_x k_y \upsilon\theta - k_z s\theta & k_x k_z \upsilon\theta + k_y s\theta \\ k_x k_y \upsilon\theta + k_z s\theta & k_y k_y \upsilon\theta + c\theta & k_y k_z \upsilon\theta - k_x s\theta \\ k_x k_z \upsilon\theta - k_y s\theta & k_y k_z \upsilon\theta + k_x s\theta & k_z k_z \upsilon\theta + c\theta \end{bmatrix} \quad (3\text{-}15)$$

Equation 3-15 uses the abbreviations $c\theta = cos\ \theta$, $s\theta = sin\ \theta$, and $\upsilon\theta = 1 - cos\ \theta$. Implementing $\mathbf{R}_K(\theta)$ is shown in Equation 3-16 and illustrated in Figure 3-4.

$$I_{end} = \mathbf{R}_K(\theta) I_{init} \quad (3\text{-}16)$$

This method gives the same results as the Pauli matrix method for isolated spin-1/2 nuclei tumbling isotropically and is physically more intuitive.

Off-Resonance and Non-Ideal Rotations

Off-resonance effects in NMR are important for several reasons. The ^{13}C spectrum of a peptide can span 300 ppm. Few of the carbon nuclei will be exactly on resonance for RF transmitted at a particular frequency, so it is important to use a pulse sequence that evenly excites all the nuclei of interest. Off-resonance causes non-ideal rotations. Imperfect rotations can also a result from the inhomogeneity of the applied RF field. The RF pulse is not exactly the same strength throughout the sample, some spins experience different RF rotations than others. In longer, multipulse experiments, these imperfections accumulate, sometimes causing destructive interference and significantly reducing the desired signal.

Solids

Switching from running experiments on liquid samples to solid samples significantly increases the complexity of simulating the experiment. Most of the work mentioned so far has been done assuming the NMR is performed on liquid samples. The molecules in a liquid tumble isotropically at a rate that is much faster than the timescale observable by NMR, including the RF pulses. This means that many of the interactions in a molecule are averaged out. The chemical shift anisotropy (CSA) and dipolar couplings in particular are averaged out. When the

molecules no longer tumble isotropically, such as in liquid crystals or lipid membranes, these interactions are not completely averaged out. This is also true for samples that have even more restricted movements such as in polymers, glasses, or crystals. Without the averaging effect of molecular motion, the CSA and dipolar effects can significantly affect pulse sequences' efficacy.

One method of removing or averaging out the CSA and other interactions is to mechanically spin a sample rapidly at what is termed the "magic angle." This can average out most effects, but gives the operator control of how much gets averaged out. Also, through careful RF pulsing in synchrony with the rotor, selected interactions of interest can be refocused.

Modifying the previous mathematical representations to model solid state NMR would require the addition of at least two main things: orientation dependence and time dependence. Since the molecules in a solid are no longer tumbling isotropically, the orientation of a sample inside the magnet matters. Doing single crystal NMR can show this well. Examples of what different crystal orientations might look like are shown in Figure 3-5. Crystallized solid samples are usually ground to fine powder which means that the NMR signal will be the sum of all crystal orientations, called a powder pattern, Figure 3-6. The time dependence is to model magic angle spinning. Magic angle spinning creates time dependence in the signal since each particle in the sample is at a different location throughout a revolution. Rather than reinventing the wheel and having to verify a new simulator for solids, I decided to use existing simulators that have been verified and accepted by the general NMR community.

Simulators

There are two ways to evaluate pulse sequences—experimentally and using numerical simulations. The use of optimizing pulse sequences using experimental evaluation, Figure 3-7 [40], has shown some success. This is the most reliable method of finding a pulse sequence that will actually achieve the signal you want. Unfortunately, it also has two major problems. First,

spectrometers are expensive and experiments can take a long time to run, particularly the more complex experiments. Most of the experimental time is from waiting for the sample to relax back to equilibrium after the last scan. For example, the inversion experiments discussed in Chapter 4 required a relaxation time of 7 seconds per scan. Each data point required 16 scans and at least 41 data points were required for each experiment giving a total experiment time of an hour and 15 minutes. It can take hundreds to thousands of evaluations to converge to a solution. This means that lots of spectrometer time has to be spent optimizing a particular pulse sequence before it can be used to find the molecular information we are seeking. The second problem is generalization. A pulse sequence optimized on one spectrometer will not necessarily work as well on another spectrometer or even the same spectrometer with a different probe. The solutions are often over fit to match the peculiarities of a particular spectrometer setup.

A faster, cheaper, and more general method of testing pulse sequences is to use a numerical simulation, Figure 3-8. An RF experiment that may take hours to run on a spectrometer can be run on a simulator in a fraction of a second with infinitely good S/N and no delays waiting for a sample to return to equilibrium. With a cluster of computers, very complex experiments can be run even faster or many experiments can be tried in parallel. Also, a simulation can be used to find a general solution that is not specific to any particular spectrometer or it can be used to optimize particular experiments for different spectrometers and probes with different RF characteristics. The one downside to using a simulator is that one has to be very careful that the setup between the simulator and the spectrometer eventually used to test the pulse sequence are as similar as possible and that the simulation accurately represents experimental limitations.

There are three published simulators currently used widely in the NMR community: Gamma [41], SIMPSON [42], and SPINEVOLUTION [43]. Independent labs also have developed a wide variety of simulation programs for their specific problems.

Gamma is a set of C++ libraries for doing NMR simulations that has also been ported to the Python language. It is used by writing a program that describes the experiment, and then compiling and running that program. Gamma was originally written at ETH Zürich, but is now maintained at the National High Magnetic Field Laboratory in Tallahassee, FL. The routines have not been optimized for speed, so they are slow. The need to write, compile, and then run the programs further slows and complicates methods when many, automatically generated simulations need to be run. Also, Gamma is no longer consistently maintained. Consequently, Gamma was not chosen as a simulator for this project.

In 2000, Dr. Niels Nielson's lab at the University of Aarhus published SIMPSON, a simulation program written primarily in C with input files written in TCL. Having the interface written in TCL makes it easier to work with than writing each simulation as its own executable. SIMPSON has bugs when using explicit phase cycling of RF pulses. Consequently, a filter command is often used. Phase cycling is a method of varying the overall transmitter and receiver phases to create destructive interference in the unwanted signals and constructive interference in the desired signal. The filter command just zeros out elements of the system density matrix rather than actually doing a phase cycle filter which oversimplifies the behavior of an NMR system.

Dr. Robert Griffin's lab published a simulator in 2006 called SPINEVOLUTION (referred to from here on as spinev). It is distributed as a compiled binary and uses a set of custom text files as input. Spinev has phase cycling explicitly built in, which is more representative of what is done on a real spectrometer.

Dr. Nathan Oyler designed a simulation program in collaboration with Dr. Manish Mehta and Dr. Joanna Long when he was in Dr. Gary Drobny's group at the University of Washington. It has been used in this project to verify results against the other simulation programs. It also allows explicit phase cycling. The interface is a bit complicated, is not automated easily, and the software is not optimized for speed. Additionally, it is not being actively maintained, so it was not used as the main simulator. However, it has been extensively verified by experimental data ensuring its robustness. Also, it allows the inclusion of relaxation parameters in the simulations.

Initially, SIMPSON was chosen as the simulator of choice for this research as it was being supported the most actively. It also had the most straightforward interface and is newer than Gamma. It is supported on both Windows and Linux and work was actively being done to get it compiled for Mac. Dr. Long, in collaboration with other members of the Drobny group at the University of Washington, had developed SIMPSON simulation routines [32]. Their code was very instructive in how SIMPSON works and how to accomplish a pulse sequence, like those in this research.

SIMPSON has the capability to divide the computations for a simulation across multiple computers, also referred to as parallel computing or clustering. SIMPSON sends the computations to specially configured compute nodes. Initially this was seen as a very useful feature, but since the configuration was not easily amenable to public clusters, and the initial simulations ended up quite short, SIMPSON's clustering feature ultimately proved to be unimportant.

SIMPSON also has problems with pulse sequences of certain lengths. These problems seem to arise from a numeric computation issue. It only worked consistently for even numbers of pulses between 2 and 22 (the maximum tried) and 5. For numbers that did not work, SIMPSON

gave an error that the sequence was not rotor synchronized. The amount of time it was off by was very small, so I suspect it is an accumulation of time quantization errors.

The simulations run for the last couple years have been run using spinev rather than SIMPSON. Support and upgrades for SIMPSON have stopped in the last couple years, but have continued for spinev. Also, spinev allows computations to be split into multiple processes on the same computer. This made some slow calculations faster and since it didn't require specially setup nodes, was easily implemented on the HPC cluster.

Minimization

Minimization is the process of finding the best sequence of RF pulses for a given a set of experimental constraints, such as hardware limitations or molecular interactions, that leads to an experimental outcome most close to ideal. Generally, experimental NMR is governed by many parameters. However, a parameter which can be controlled, such as RF, can be varied and its effects monitored.

An analytical method for minimization is to take a differentiable function, find where its first derivative equals zero, check the second derivative to make sure it is a minimum, and test each point to see which has the smallest value. This works well for a function that is fully differentiable. Not all experiments can easily be described analytically or are differentiable.

A numerical method of minimization is to check the gradient at a specific location (the initial position), and change parameters in the opposite direction of the gradient to find a point with a lower function value. Repeating this process can eventually lead to a minimum. This is referred to as gradient decent. However, there is no guarantee that the minimum found is the global minimum. If the process is repeated "enough" times with different initial conditions, then the lowest minimum found is likely to be the global minimum, or at least close to it. Figure 3-9

illustrates the difference between global and local minima. If the surface being searched is not smooth, the search process may never find the global minimum.

Gradient decent is convenient because the gradient at each point can be found numerically and does not require closed analytical expressions. One simple method of finding the gradient is numerical differentiation where the function is sampled on each side of the current point to determine the slope. There are many methods of gradient decent, some of which work better and some of which are easier to implement. If the function being minimized is not analytically differentiable, the function evaluation is time consuming, or the function is of high dimension, gradient decent may not be the best method of minimization.

Another method of minimizing a function is by using a simplex algorithm. A simplex is a group of points in the search space. At each step of the algorithm, one or more points are moved in a direction likely to lead towards a minimum. Usually, when minimizing N variables, there are N+1 vertices to the simplex. The Matlab fminsearch function uses the Nelder-Mead Simplex method [44].

The Nelder-Mead simplex (NM) method is a direct search algorithm that for a problem of N variables maintains a convex hull of N+1 vertices that surround a nonzero volume. Given a set of N+1 vertices, $\{x_1, x_2, \ldots, x_N, x_{N+1}\}$, ordered from best function value to worst function value, $\{f_1, f_2, \ldots, f_N, f_{N+1}\}$, each iteration of the NM algorithm returns either a new point that has a function value less than f_{N+1}, or a set of N new points, which, together with x_1, form the new simplex. Each iteration of the NM algorithm has up to 5 steps: order, reflect, expand, contract, and shrink.

ORDER. Order the N+1 vertices such that $f(x_1) \leq f(x_2) \leq \ldots \leq f(x_N) \leq f(x_{N+1})$.

REFLECT. Compute the point x_r, called the reflection point, from Equation 3-17.

$$x_r = \bar{x} + \rho(\bar{x} - x_{N+1}) \tag{3-17}$$

$\bar{x}$ is the centroid of the N best points, Equation 3-18. If $f_1 \le f_r = f(x_r) < f_N$, x_r joins the simplex. Terminate the iteration.

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{3-18}$$

EXPAND. Calculate the expansion point, x_e, if $f_r < f_1$, Equation 3-19.

$$x_e = \bar{x} + \chi(x_r - \bar{x}) = (1 + \rho\chi)\bar{x} - \rho\chi x_{N+1} \tag{3-19}$$

x_e joins the simplex if $f_e = f(x_e) < f_r$, otherwise, x_r joins the simplex. Terminate the iteration.

CONTRACT. If x_r is strictly less than x_{N+1}, $f_N \le f_r < f_{N+1}$, calculate an outside contraction, Equation 3-20.

$$x_c = \bar{x} + \gamma(x_r - \bar{x}) = (1 + \rho\gamma)\bar{x} - \rho\gamma x_{N+1} \tag{3-20}$$

If $f_c = f(x_c) \le f_r$, x_c joins the simplex and terminate the iteration. Otherwise, perform a shrink calculation. Calculate an inside contraction if $f_r \ge f_{N+1}$, Equation 3-21.

$$x_{cc} = \bar{x} - \gamma(\bar{x} - x_{N+1}) = (1 - \gamma)\bar{x} + \gamma x_{N+1} \tag{3-21}$$

If $f_{cc} = f(x_{cc}) < f_{N+1}$, x_{cc} joins the simplex and terminate the iteration. Otherwise, perform a shrink calculation.

SHRINK. The new, unordered list of vertex points is $\{x_1, v_2, \ldots, v_{N+1}\}$ where v_i is calculated from Equation 3-22.

$$v_i = x_1 + \sigma(x_i - x_1) \tag{3-22}$$

Examples for N = 2 are shown in Figure 3-10. There are four parameters that have to be chosen for the NM method. Their standard values are $\rho = 1$, $\chi = 2$, $\gamma = 0.5$, and $\sigma = 0.5$ [45].

Clustered Computing

Clustered computing allows many optimizations or calculations to be done at the same time on parallel processors. This significantly decreases throughput time for solving problems. The University of Florida's High-Performance Computing Center (hpc.ufl.edu) has more than 1800 processor cores available for research projects. Computer clusters were found to be most useful in this research for doing many optimizations simultaneously rather than for speeding up individual calculations. When doing pulse optimization, each simulation depends on a previous one, so it is not possible to split the problem up to have each node doing a different set of simulations related to the same problem. Each simulation typically takes less than 30 seconds

and each optimization can take 4-24 hrs. This is why it is most useful to use each node to run a different set of optimizations.

When submitting a job to the HPC cluster it is important to understand the job queuing algorithm. A job will be queued fastest if its time and memory requirements can be accurately estimated. One way to check memory usage is to run a job and check it while it is running. Also, the email that a job can send when finished gives a listing of how much memory the job actually used. Estimating the amount of time a job will use isn't as straightforward as it would seem. The problem is that jobs run at slightly (and sometimes not so slightly) different speeds depending on what else a node is doing while the job is running. Generally adding 20% more time than calculated is a good rule of thumb. For the research presented here, the HPC cluster was used to do many optimizations simultaneously. This freed the author's primary workstation for other tasks, and allowed many more optimizations to be run simultaneously.

We are very fortunate to have this cluster of computers available for our free use here at the University of Florida. The project is supported by industry partners, university sponsorship, and particularly, individual professors writing grants that have funds set aside for the purchase and maintenance of a computer cluster, which the HPC provides for them by way of queuing priority on the number of nodes funded.

Liquids Examples of Compensation by Composite Pulses

Pulsed Fourier NMR has the problem of reality to deal with. Pulse lengths are never perfectly accurate. Frequencies are not all on resonance. The homogeneity of the magnetic fields involved is never perfect. Many people have worked on designing NMR experiments to make them less sensitive to these issues. What follows is an overview of a select subset of these papers. A search on the authors will reveal many other papers that cover more topics than are presented here.

Freeman et al. did some of the initial work showing how the introduction of a random length delay can suppress anomalies caused by acquisition being synchronous to an event [46]. This forms the basis for most methods of suppressing imperfections. If an imperfection can be made to cycle in some manner, it can be averaged out. Phase cycling [47] is one such method of doing this. Pulses are tried from all the symmetric phases in a cycle so that imperfections are averaged out. There are some more complex versions of phase cycling that help to deal with imperfections in going into and out of the double quantum state.

The idea of replacing a single pulse with multiple pulses was invented by Malcolm Levitt in 1978 in Ray Freeman's group [48]. This idea was given the name "composite pulses" to denote a group of pulses used to accomplish the same functionality as a single pulse. In the original paper, the authors replace an inversion 180° pulse with a phase of X (0°) with 3 pulses, a 90°(X) 180°(Y) 90°(X). This simple composite pulse quite effectively compensates for both errors in pulse lengths and off resonance frequencies. The payment is a longer pulse sequence requiring more RF power to be absorbed by the sample. Figure 3-11 shows the trajectories of a single 180 degree pulse at various frequency offsets. Note that the terminal points are in the back quadrant of the sphere. Figure 3-12 shows the terminal points as a function of offset frequency normalized to the nutation rate, $\Delta\omega/\omega$, and their projection onto I_z. This gives some idea of the bandwidth of a 180 pulse and shows how it is dependent on ω. Figure 3-13 shows the trajectories of a 90°(-X) 180°(-Y) 90°(-X) pulse sequence at negative frequency offsets. Figure 3-14 shows the terminal points and projection onto I_z for this pulse sequence. The useful bandwidth has been increased from that of the single 180 pulse.

The Freeman group went on to explore composite pulses more fully [49] with their focus on 90° degree pulses and 180° inversion pulses and concluded "It has not yet been possible to devise a composite sequence suitable for this first application, the 180° refocusing pulse." [49]

In 1981 the group came closer to dealing with the issues of refocusing pulses. They found that replacing the $R_Y(\pi)$ pulse of a Carr-Purcell spin-echo experiment with a composite pulse $R_X(\pi/2)R_Y(\pi)R_X(\pi/2)$ removes the phase shifting on even numbered echoes [50]. It doesn't suppress the phase shifting on the odd numbered echoes. The group also published a more complicated composite pulse sequence to compensate for inhomogeneity and frequency offset [51].

Others were also in the field shortly after the Freeman group, including a description of another 3-pulse sequence that compensates for frequency offset [52]. This paper derives a method for finding composite pulses that compensate over a wider range of frequencies. Tycho later shows how using the Magnus expansion of the deviation propagator allows pulse sequences to be derived that not only compensate in magnitude of the signal, but phase as well [53]. These pulse sequences are useful for both inversion and refocusing situations since the phase deviation of the output due to inhomogeneity or frequency offset is minimal or at least significantly lower than that of Levitt's or the standard single pulses.

Solids Examples

In 2000, Lyndon Emsley's group decided explore numerically optimized pulse sequences [54]. They set out to solve the problem of homonuclear decoupling by varying the phase of a constant RF amplitude signal. Their approach is different than what other groups have generally done, so it is worth noting. They started with the idea of optimizing the BLEW-12 decoupling sequence [55]. More pulses than 12 were needed to improve the pulse sequence, but the more

parameters to optimize, the harder optimization becomes since each parameter is an increase in dimensionality. Their solution was to treat RF phase as a waveform and to optimize Fourier components of the desired waveform rather than actual points in a waveform.

To simplify computation they imposed a few constraints on the Fourier components to maintain known properties of good homonuclear decoupling sequences. In order to keep the RF propagator unity over a cycle, they chose to only change the first half of a cycle. The second half is just the first half with π added to each of the Fourier components. This has the side effect of causing the pulse sequence to also have time reversal symmetry. Time reversal symmetry causes all the odd order terms of the Magnus expansion to vanish. From Tycko's work, this makes the pulse sequences more likely to be robust to frequency offset issues. Sakellariou et al. optimized over a range of dipolar couplings and power mismatches to achieve a pulse sequence more robust to both issues.

Sakellariou ran their optimization by first generating $2x10^6$ random sets of coefficients. The 1000 best of these sets were then chosen to be optimized using a least-squares steepest decent algorithm. The result of Sakellariou's work is the DUMBO-1 sequence. It is a 64 step sequence based on the BLEW-12 sequence for homonuclear decoupling. They tested it using a variety of samples by observing the ^{13}C CPMAS signal with the decoupling sequence applied to the ^1H channel during acquisition. The DUMBO-1 sequence is a bit more robust to dipolar couplings and RF inhomogeneity than the BLEW-12 and FSLG sequences that they compared it with.

In 2003, Dr. Emsley's group published a paper on optimizing heteronuclear decoupling using the same techniques [56]. Here they optimize over the residual bilinear proton-carbon terms of the effective Hamiltonian rather than just looking at the final output of a simulated

experiment. One of the challenges was to properly model the proton bath interacting with the ^{13}C spin. They finally settled on a phenomenological model which had to be set by trial and error between simulation and experiment. Their basic method of optimization was to try a set of 10^6 randomly generated Fourier coefficient sets. The best of these randomly generated coefficient sets were then optimized using a steepest decent procedure. They optimized over frequency offsets and proton couplings [56]. The results were the DROOPY and SDROOPY pulse sequences. These sequences performed about on par with existing pulse sequences, SPINAL-64 in particular, with respect to frequency offsets. However, the DROOPY-1 and SDROOPY-1 sequences were quite a bit more robust with respect to inhomogeneity, as modeled by power misset, than SPINAL-64.

A second publication on optimizing heteronuclear decoupling pulse sequences used a spectrometer for evaluation rather than a simulator [40]. This is an important change since it moves from optimizing using a theoretical model to optimizing using a specific, real system. They found that modeling the proton bath around a ^{13}C spin was problematic, so they implemented simplex optimization using a Bruker AU program on their 500 MHz AVANCE spectrometer.

They show good results for a comparatively simple decoupling scheme. They reduced the number of parameters to 2, down from 6 complex coefficients in their previous work. The results are good across different samples for a specific magnet, probe, RF power, and MAS rate. Due to the use of an isotopically labeled test sample ([2^{13}C]-glycine) and the reduction to two parameters, the simplex algorithm converges in about 50 iterations, which take about 30 minutes for their setup. They also say that the result seems to be stable over time for the same hardware setup.

Unfortunately, the results from this method are not likely to be very general across different magnets, probes, power levels, or MAS rates. If one wanted to optimize over any of these things, the length of the optimization procedure would increase substantially. Even optimizing over 10 different power levels to attempt to simulate probe inhomogeneity increases the length of the optimization at least 10-fold—from 30 minutes to 5 hours. That is assuming that the simplex algorithm still converges at the same rate. From experience in my research, the simplex algorithm converges much more slowly with an increase in number of parameters. Thus, optimizing a more complicated problem, which has more parameters, would also significantly increase the amount of time required to optimize a solution.

Optimizing across different magnet field strengths is a prohibitively hard problem using this method for nearly all users. It requires running experiments simultaneously on multiple magnets, the collecting and analyzing the results in a central way. With most magnets networked, this is not an impossible task, but would be very complex and very expensive.

Magnet time is very valuable. When working at a user facility such as we have here at UF, magnet time varies from $6/hr for the AMRIS 500 MHz magnet in off hours to $26/hr for the AMRIS 750 MHz magnet in peak hours.

Another method of optimizing pulse sequences is to simulate using optimal control theory. The idea of applying optimal control to design pulse sequences that are optimized for a particular situation was first published in 1986 with respect to MRI [57,58]. The Conolly publication and a publication 10 years later [59] are probably the best fundamental papers on using optimal control in magnetic resonance. The Rosenfeld paper also shows how to use mathematical programming to solve pulse sequence optimization problems.

This idea was picked up by Skinner et al. [60] in 2003 and applied to broadband excitation in liquids NMR. Khaneja et al. looked into the coupled spin problem using optimal control [61].

In 2004, Dr. Nielsen's lab described using optimal control to improve solid state dipolar recoupling experiments [62]. This very brief JACS Communication tells of using optimal control theory to find a better pulse sequence for coherence transfer from ^{13}C to ^{15}N for double-cross-polarization experiments. They achieved an improvement of 53% for a $^{13}C^{\alpha}$,^{15}N-glycine spinning at 10 kHz with ^{1}H decoupling in excess of 100 kHz. Unfortunately, due to the short nature of the article, there is not much information on how they did this experiment but it seemed to work for the compound they modeled in their optimizations.

In 2004, Mikhail Veshtort, from Robert Griffin's group, wrote an article about optimizing frequency selective pulses for both liquid and solid state NMR using his simulation program SPINEVOLUTION (released to the public in 2006) [63]. He uses 20-30 parameters which are usually optimized using a nonlinear least-squares method after carefully selecting "good" solutions to start the optimization process. This makes it less likely for the optimization routine to get stuck in a non-optimal solution. Again, he provides the final answers, but details are limited and do not provide guidance for how the optimization procedure may be applied to a different pulse sequence or situation.

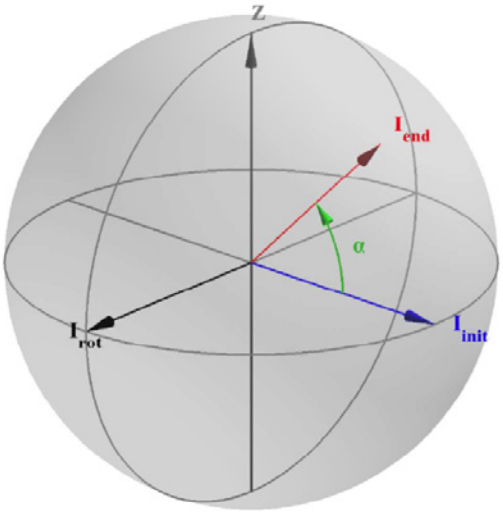

Figure 3-1. On resonance rotation. An on resonance RF field along I_{rot} rotates magnetization from an arbitrary initial point, I_{init}, an angle of α around I_{rot} to end at I_{end}.

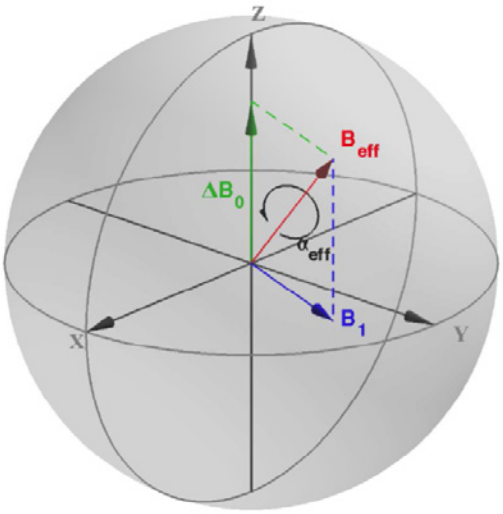

Figure 3-2. Off resonance rotation. The effective field, B_{eff}, of an off resonance RF pulse, B_1, is the vector sum of B_1 and the offset frequency, ΔB_0. The off resonance pulse will rotate the magnetization through an angle of α_{eff} around B_{eff}.

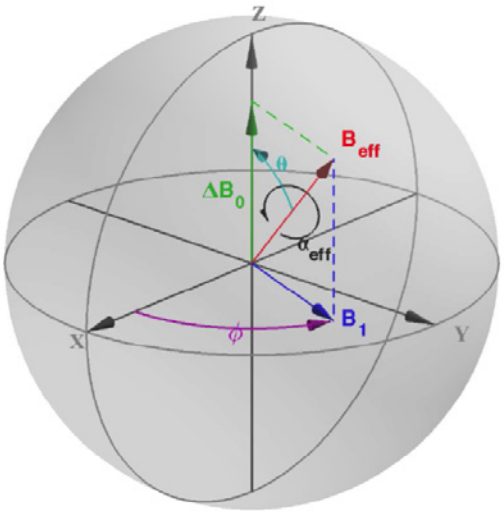

Figure 3-3. Pauli angle definitions. The definitions of ϕ, and θ for rotation about an arbitrary axis using the Pauli equation method are shown. B_1 is the applied magnetic field at a frequency offset from resonance by ΔB_0. B_{eff} is the effective field felt by the spins which are then rotated by an angle of α_{eff} around B_{eff}.

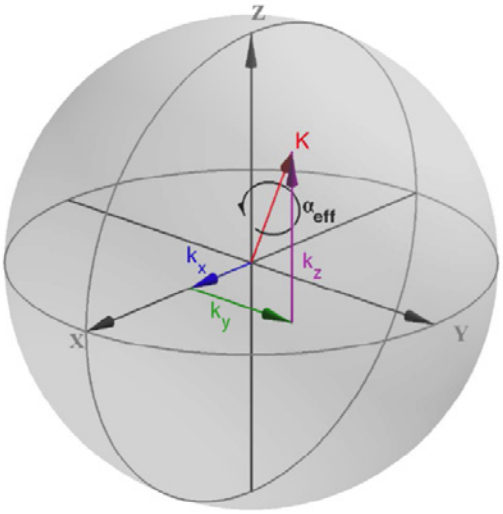

Figure 3-4. Vector rotation definitions. The definitions for rotating about an arbitrary axis using the 3D vector rotation method from kinematics are shown here. K, a unit vector made up of $[k_x, k_y, k_z]^T$, is the effective magnetic field, B_{eff}. The spins are then rotated by α_{eff} around K.

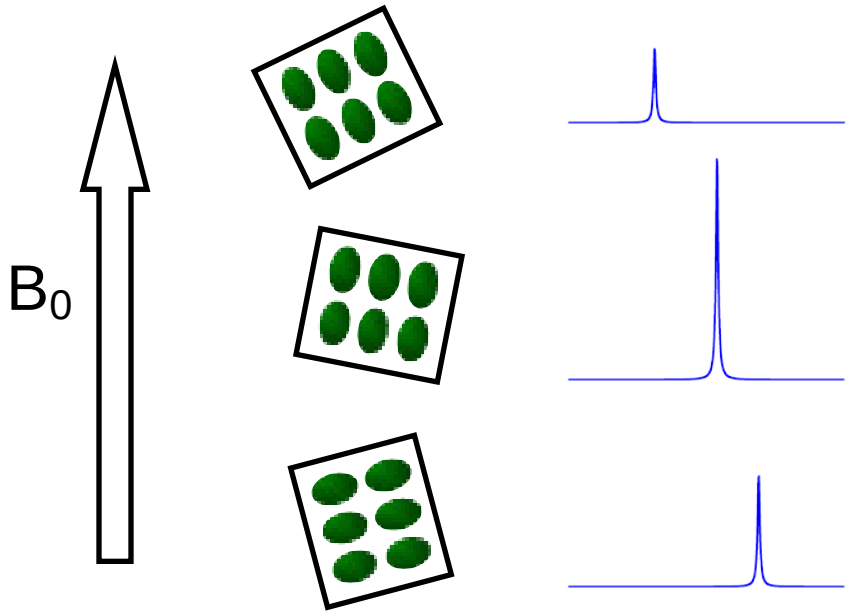

Figure 3-5. Orientation dependence of solid state NMR. A single crystal gives a different spectra depending the molecular orientation. This is called chemical shift anisotropy (CSA) and is because the currents induced in the electrons around a nucleus is dependant on the orientation with relation to B_0.

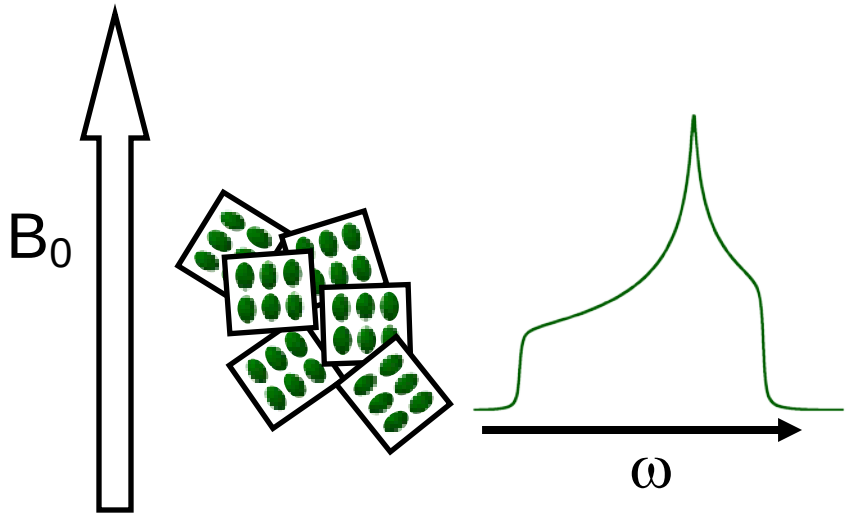

Figure 3-6. Solid state powder pattern. A powder pattern is the sum of all the possible molecular orientations weighted by their probability.

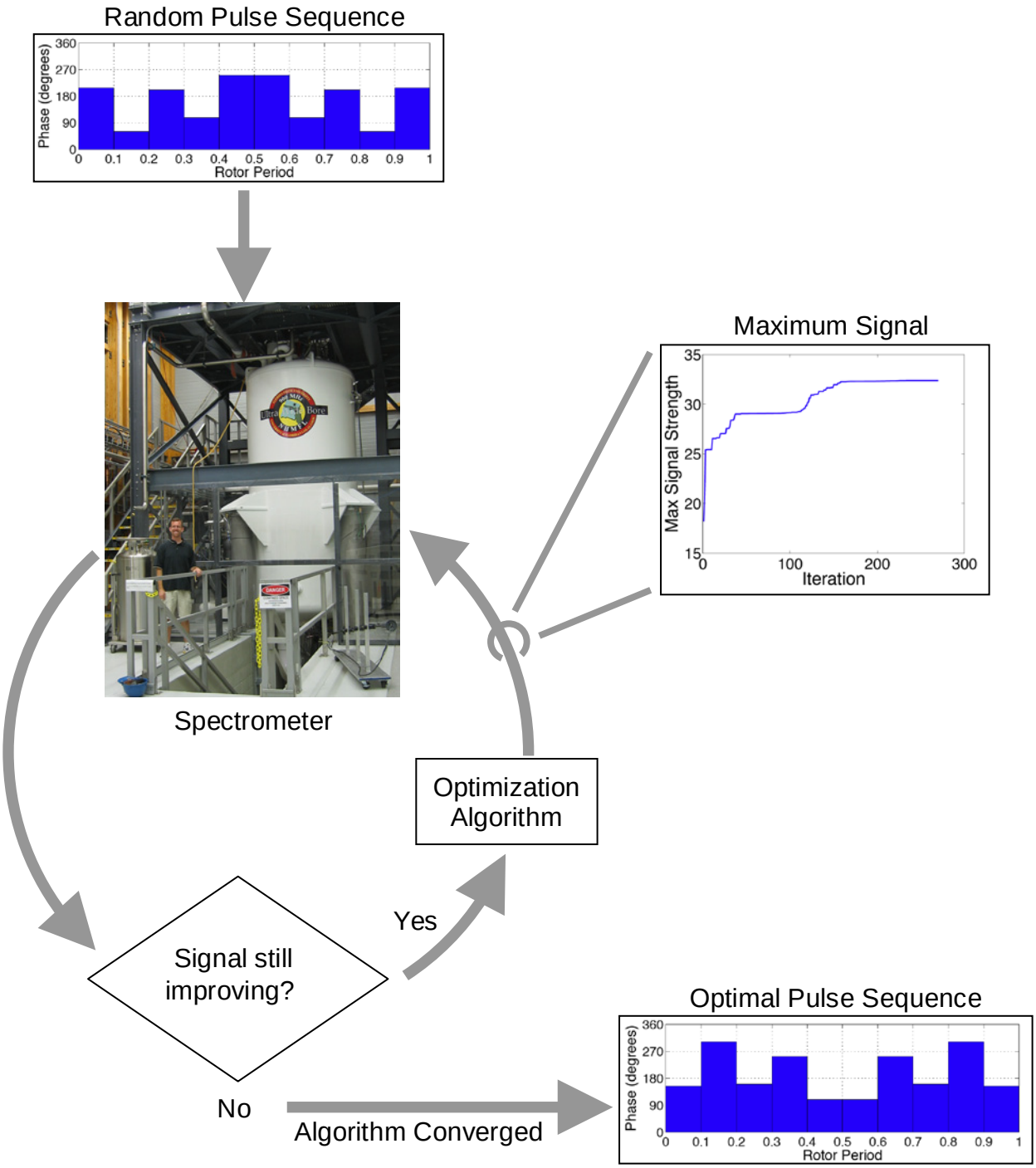

Figure 3-7. Spectrometer in the loop optimization. For spectrometer in the loop optimization, optimization starts with a randomly generated pulse sequence. This is evaluated on the spectrometer. The optimization algorithm determines how to change the pulse sequence to improve its quality. This new pulse sequence is then tested on the spectrometer. This is continued until the quality no longer improves. The system exits with an optimized pulse sequence.

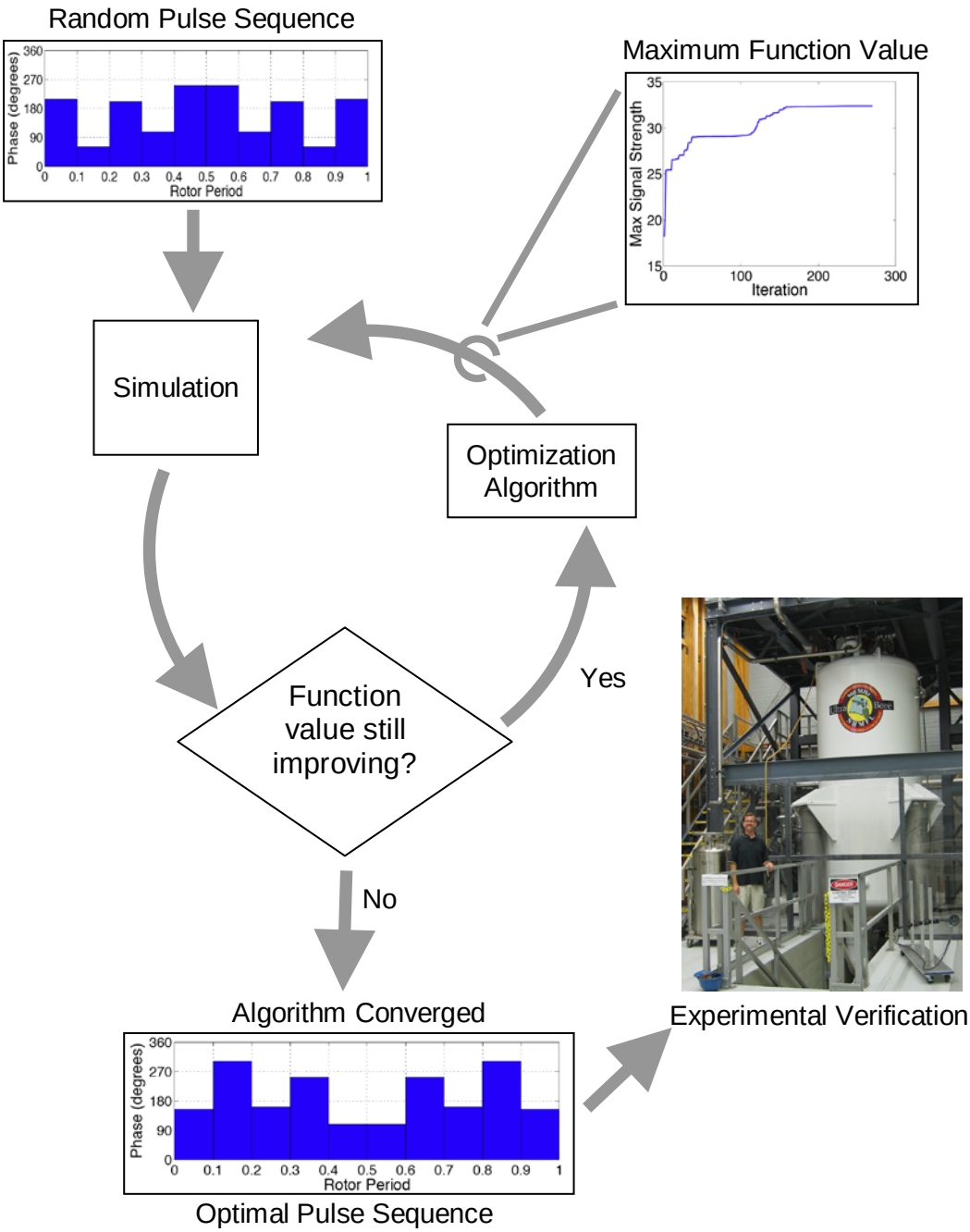

Figure 3-8. Simulation based optimization. For simulation based optimization, optimization starts with a randomly generated pulse sequence. This is evaluated using the simulator of your choice. The optimization algorithm determines how to change the pulse sequence to improve its quality. It goes back to the simulator where it is evaluated. This is continued until the quality no longer improves. The pulse sequence is then tested on a spectrometer (or two) to verify its capability to solve real life problems.

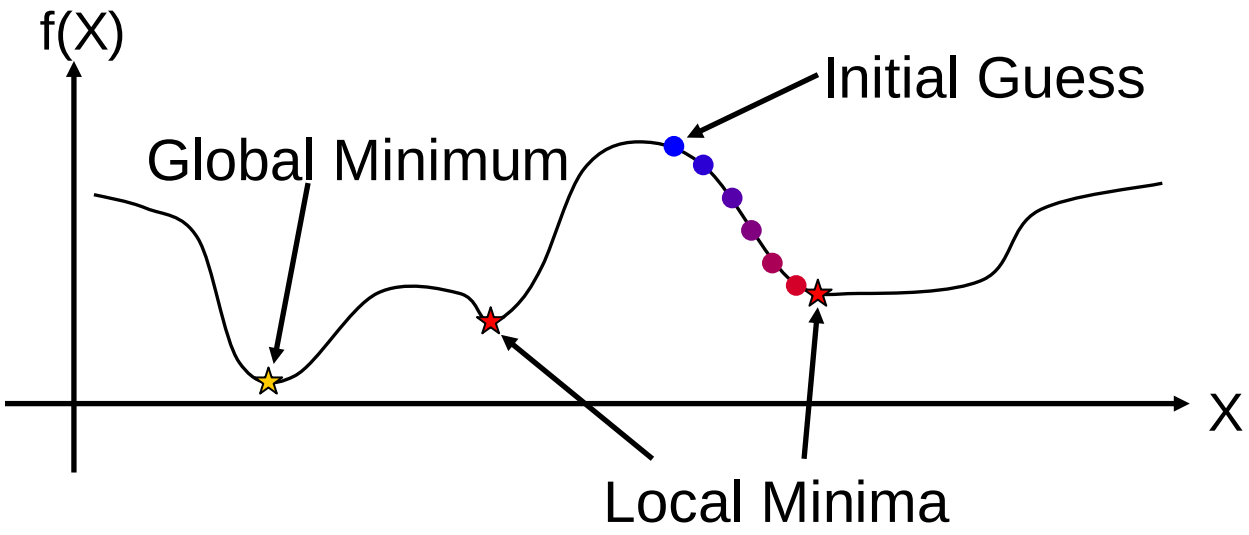

Figure 3-9. Minimization illustration. This shows an example of gradient decent minimization. The example function has 3 minima with one global minimum. Starting at the initial guess, gradient decent follows the gradient downwards until a minimum is found. In the example shown, the minimum found is not the global minimum. Typically gradient decent is run many times starting from random points with the expectation that eventually the global minimum will be found.

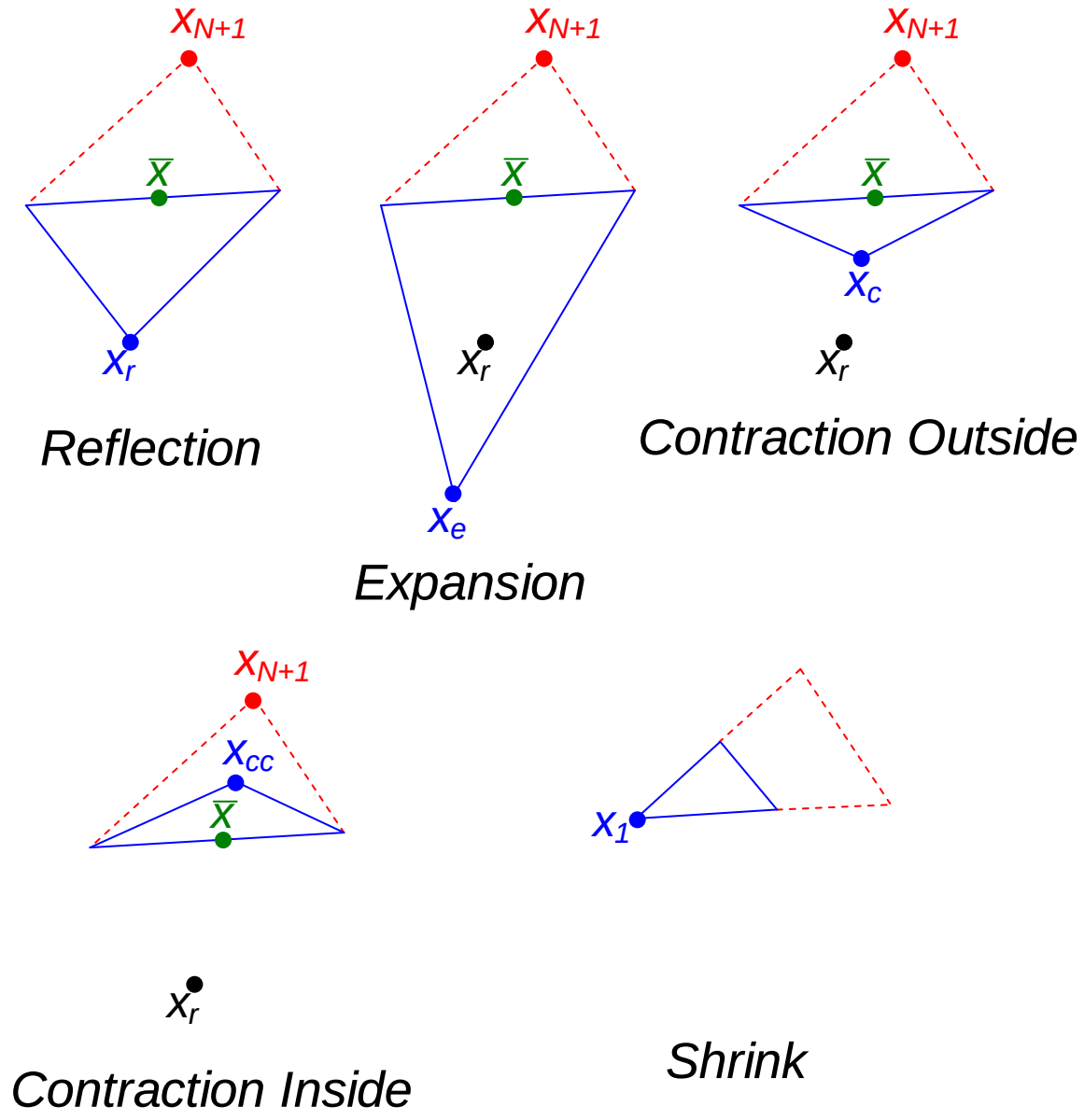

Figure 3-10. The 5 steps that the Nelder-Mead simplex method can take are shown for N = 2.

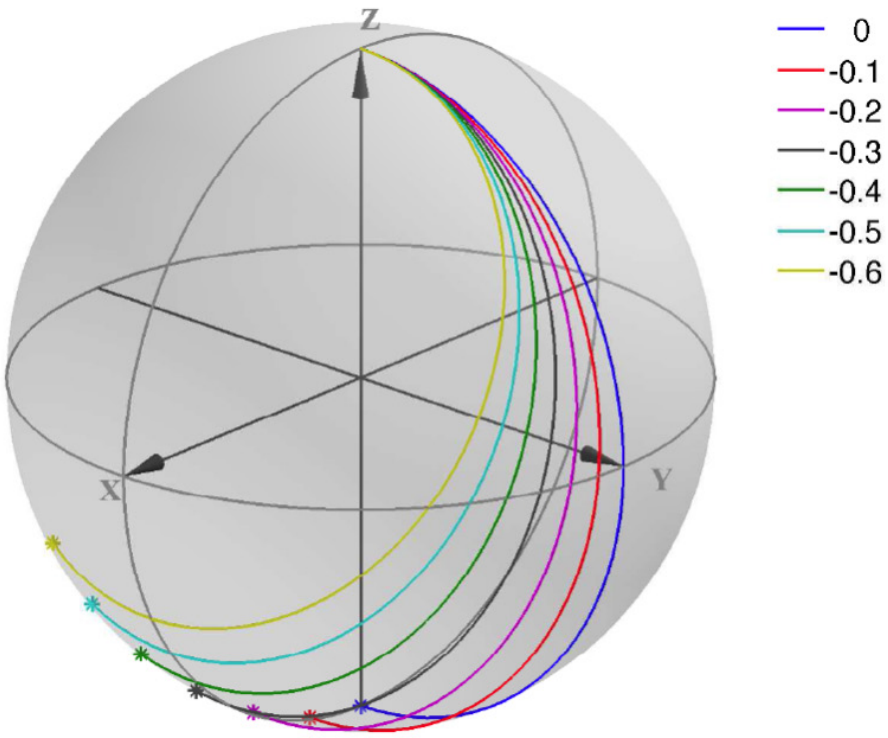

Figure 3-11. Off resonance trajectories for a 180 degree pulse. Trajectories for a 180_{-x} pulse at frequency offsets ($\Delta\omega/\omega$) shown starting at I_z. The endpoints are marked with asterisks.

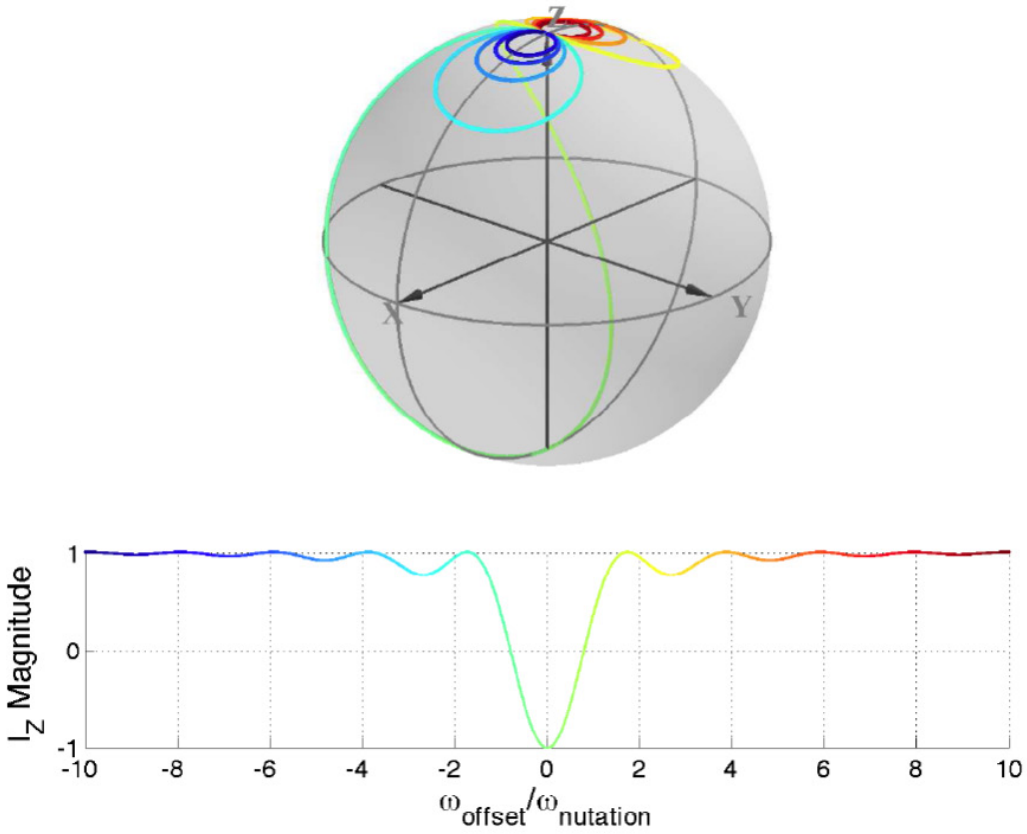

Figure 3-12. Endpoints of off resonance 180 degree pulses. The endpoints for a 180_{-x} pulse at frequency offsets ($\Delta\omega/\omega$) shown starting at I_z. The sphere shows the endpoints in 3D while the graph shows the projection onto I_z. The projection onto I_z shows the effectiveness of the pulse, disregarding phase. By $\Delta\omega/\omega = 2$, no inversion is observed.

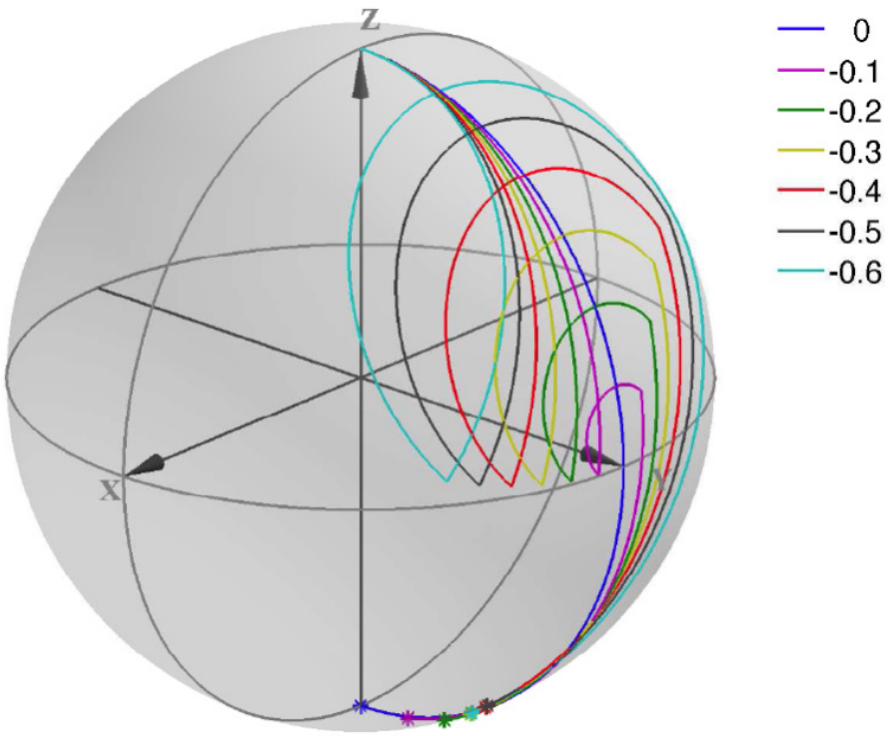

Figure 3-13. Trajectories for 90-180-90 composite pulse. Trajectories for a $90_{-x}180_{-y}90_{-x}$ pulse at frequency offsets ($\Delta\omega/\omega$) shown starting at I_z. The endpoints are marked with asterisks. This more complicated pulse sequence brings the final state of the magnetization closer to $-I_z$ than just the 180.

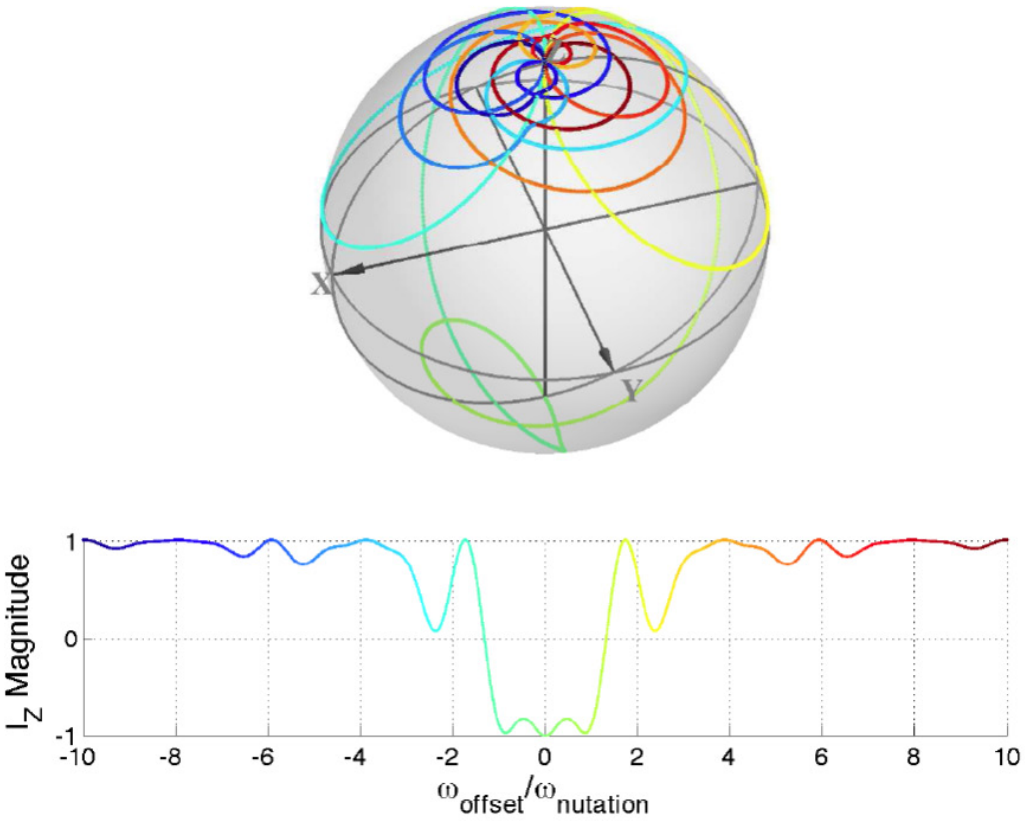

Figure 3-14. Endpoints for 90-180-90 composite pulse. Endpoints for a $90_{-x}180_{y}90_{-x}$ composite pulse at frequency offsets ($\Delta\omega/\omega$) shown starting at I_z. The usefulness of the composite pulse does not extend much past $\Delta\omega/\omega = 1$, but is much flatter than the single 180 pulse, giving it a wider useful bandwidth.

CHAPTER 4
OPTIMIZATION OF RF EXCITATION PROFILES

The general methodology of optimizing pulse sequences followed in this dissertation was shown in Figure 3-6. Optimization starts with a randomly generated pulse sequence. For all the pulse sequences optimized, the initial phases are set randomly. For some pulse sequences (inversion and refocusing pulse sequences) the lengths of the pulses are also initially set randomly. For other pulse sequences (such as DRAWS, which has to be rotor synchronized) the pulse lengths are fixed. The randomly generated pulse sequence is tested via an NMR simulator to give a measure of how well the pulse sequence works.

A minimization routine (usually fminsearch from Matlab) then continues by algorithmically changing the free parameters (phases and possibly the pulse lengths) in such a way that the pulse sequence evaluates better. This continues until the minimization routine cannot improve the pulse sequence above some threshold. The routine then stops optimizing that pulse sequence and generates a new random starting point, continuing around the optimization circle again. This is repeated as many times as resources allow. The best pulse sequences obtained are then tested on a spectrometer to verify that they work in real life.

Three types of pulse sequences were optimized for this work: inversion, refocusing, and DRAWS.

Optimizing Inversion Pulse Sequences

Introduction

An ideal inversion pulse sequence takes magnetization that is along the +Z-axis and rotates it to the –Z-axis. These are commonly used in inversion recovery studies to determine T_1, the constant for spin-lattice relaxation. For typical molecular species and NMR spectrometers, inversion pulse sequences are imperfect. One aspect of their imperfection is that they don't invert

all frequencies equally. For example, take a single 180° pulse which would be 10 μs long for a 50 kHz RF field. This is a very simple pulse sequence that only has one pulse besides the readout 90 pulse, shown in Figure 4-1. A frequency Δω from the center frequency will experience a nutation rate, ω_{eff}, as shown in Equation 4-1 and explained in more detail in chapter 3.

$$\omega_{eff} = \sqrt{(\omega_{nut})^2 + (\Delta\omega)^2} \tag{4-1}$$

Another way to state it is: offset frequencies will experience a faster nutation rate than the center frequency. They will also rotate around an axis that is the vector sum of the nutation axis and the offset frequency along the Z axis as discussed in Chapter 3. The 10 μs pulse that gives perfect inversion at the center frequency will not give perfect inversion at an offset frequency. It will overshoot by some amount. This can be seen in Figure 4-2 where the trajectories for ±0.5 Δω/ω are shown. Figure 4-3 shows where the bulk magnetization ends up after a 180° pulse around –X.

This problem can be solved with better pulse sequences for inversion [51,52,64]. The results using SPINEVOLUTION [43] (hereafter referred to as spinev) at a spin rate of 10 kHz and power of 30 kHz are shown in Figure 4-4 for some of these pulse sequences. The pulse sequence by Keniry et al (magenta in Figure 4-4 starting with a pulse length of 38°)[64] gives the widest bandwidth so it is used as a comparison in this work. The composite pulse by Bai et al. (yellow in Figure 4-4) [65] is also a good comparison pulse sequence because it uses pulse lengths that are not integer multiples of 90°.

Experimental verification of these pulse sequences were run to make sure that the spectrometer setup was correct and the simulations agreed well with experiment before optimizations were run. Starting with the single 180° pulse, there was a large discrepancy between the experiment and simulation. The lower offset frequency side of the experimental data

drooped below the simulated data. After trying a variety of things, it was determined that the problem was that the recycle delay was too short. Examples of data for different recycle delays are shown in Figure 4-5 and shows that a recycle delay of 10 seconds is sufficient. Further testing showed that 7 seconds of recycle delay were adequate for adamantane. Experimental data compared to simulated data for a single 180° inversion pulse are shown in Figure 4-6 for a few power levels.

A representation of the lengths and phases of the pulse sequence by Keniry et al. is shown in Figure 4-7 [64]. Note that this pulse sequence uses 558 degrees, which is just over three times as long as a single 180 pulse. A comparison of experimental to simulated data is shown in Figure 4-8 for three power levels for an adamantane sample 3.7 mm long.

These two examples showed that experimental and simulated data match quite well for the low-E probe described in Chapter 2. This is likely due to the high homogeneity (96% on ^{13}C and ~100% on ^{1}H) of the RF and the modern spectrometer used (Bruker Avance II). Using spinev, I have optimized pulse sequences to give a better inversion profile than those in the above references when used in ssNMR.

Methods

Inversion pulse sequence optimization started with testing a 180° pulse sequence over a range of ±100 kHz frequency offsets. This was done both in simulation using spinev and experimentally on a 750 MHz spectrometer. The idea of finding an optimal pulse sequence is to use multiple pulses, with differing lengths and phases to invert a larger range of frequencies more evenly. Optimizations were run with one pulse through 8 pulses with the phase and lengths unrestricted. Minimization was done using Matlab's fminsearch function. The basic procedure for optimizing is as follows. The random number generator in Matlab is reset to something random, usually 100*clock. This is important since the random number generator is initialized to

the same point each time Matlab is started. If it isn't reinitialized, the set of random numbers it generates will be the same every time it is run. A set of random pulse lengths and phases is generated. Both are represented in degrees between 0 and 360. This is because fminsearch works better if all of the parameters it is varying are on the same order of magnitude. fminsearch is run starting at this random initialization. The process is repeated as many times as possible.

Evaluation of an inversion pulse sequence is done by the function spinev_eval_inv_dly.m. This function separates out the pulse length parameters from the pulse phase parameters. If desired, it can add an intermediate pulse between the specified pulses that has a phase halfway between its adjacent pulses. The idea was to model phase slew, which has not turned out to be a significant parameter in fitting the data. Next, this function writes a pp file that contains the pulses, phases, powers, and frequency offsets for the trial inversion pulse sequence. Doing a system call to spinev follows this. The function takes advantage of the new macro feature introduced in spinev 3.3 that allows variables within a pulse program to be specified on the command line when that program is called. This is how the pp file, frequency offsets, and power offsets are specified. Also, the eval function makes use of the new spinev feature of having the results returned to the command line. Matlab reads the command line return and thus saves a file write and read cycle, which tend to be slow. The end of the eval function returns the sum of the polarization divided by the number of simulations done. This normalizes the results to be -1 for perfect inversion. Normalizing the result this way makes for easy comparison to an ideal inversion pulse sequence. The eval function also checks to make sure that the spinev expiring warning does not cause problems and can add an extra multiplier (usually -1 if used) in case the function is used for evaluating something other than an inversion pulse sequence (this same function is also used in refocusing pulse sequence optimization).

All of this is done many times (hundreds to thousands) for different numbers of pulses. In order to get other work done while this is happening, the optimizations are usually run on the HPC cluster. The best solutions are then written as a Bruker pulse program and tested on a spectrometer using adamantane (Acros Organics) as the sample. Comparisons were then made between the simulated results and the experimental results to determine if the simulation was actually making sense.

One somewhat amusing observation was over fitting. To optimize for broad spectral width, points were initially optimized from -100 kHz to +100 kHz every 50 kHz. When the intermediate points were tested via simulation, it showed that the optimization process had only minimized the signal at the points where the optimization was tested, and the other frequencies were quite high as shown in Figure 4-9. This proves that it is important to sample over the region of interest and to double check how well solutions generalize. Most of the optimizations were done with a 10 kHz step in frequency offset, but some were done with a 1 kHz step to see if the result would be a flatter bandwidth.

Initially, pulse sequences were optimized over both power offset, to model power missets and inhomogeneity, and frequency offset. It is important to keep in mind that there are different ways to look at power offsets. One way is to change the transmit power (nutation rate) but not the pulse length. This simulates an incorrectly calibrated pulse length. Another way to simulate the same thing is to change the pulse length, but not the power level. This way is not how we typically think about modeling inhomogeneity. The third way to look at power offset is to have the power and pulse length scale together. This is important to think about in case the pulse sequence is optimized for a specific power setting or MAS rate. This became particularly important when fitting the experimental data. Initially, the experimental data was run at a

correctly calibrated nutation rate of 62.5 kHz while the simulations ran at 50 kHz. In trying to match simulation and experiment, there was a 0.8 frequency offset scaling factor due to the fact that 50 divided by 62.5 is 0.8. Once that issue was solved, the experimental data fit the simulation data quite well.

Optimizing over different power levels turned out not to be necessary for using pulse sequences on the 750 MHz magnet with the Low-E MAS probe. The homogeneities of the B_1 fields on this probe are high enough that inhomogeneities do not need to be modeled.

As the optimizations continued, two modifications to the method were added. First, instead of just trying to minimize the average output (projection on to I_z), the system was changed to minimize the sum-squared-error between a desired frequency offset profile and the simulated frequency offset profile. This allowed arbitrary profiles to be generated.

The second modification to the method was to use constrained optimization as implemented in Matlab's fmincon function. This was done as an attempt to prevent composite pulses with pulse lengths shorter than the spectrometer can actually generate. This changed the minimization method from Nelder-Mead simplex to a sequential quadratic programming method. This is a conjugate gradient decent type method which uses an perturbation method of calculating the gradient for functions lacking a gradient function. Setting the amount of perturbation turned out to be critical in getting this method to work. It needed to be increased to be large enough to see a gradient but the results were no better than the unconstrained optimizations.

Results

The pulse lengths and phases of the best pulse sequence from unconstrained optimization using the Nelder-Mead method are shown in Figure 4-10. This composite pulse is just over 6 times the length of a single 180° pulse. Note that all of the pulse lengths and phases are arbitrary,

i.e. not equal to an integer multiple of 90°. Figure 4-11 shows a comparison of a single 180° pulse, the pulse sequence by Keniry et al., and my optimized pulse sequence at 32 kHz RF power. The correlation between simulation and experiment is quite good and the bandwidth over which the magnetization is inverted is also quite good. My optimized composite pulse has a wider bandwidth, slightly higher ripple, and a longer RF excitation than the pulse sequence by Keniry.

Figure 4-12 shows a comparison of the inversion data for a sample 3.7 mm long and a sample that is 11.7 mm long. The homogeneity was ~96% for the 3.7 mm sample and ~76% for the 11.7 mm sample. While both sample lengths match the simulation relatively well, with the longer sample does not fit quite as well. This is to be expected since the longer sample has more spins resonant at frequencies farther from the transmission frequency.

Optimizing Refocusing Pulse Sequences

Introduction

Inversion pulse sequences take bulk magnetization that lies on the +Z-axis and move it to the -Z-axis. There are times when instead it is desired to take magnetization that has precessed in the XY-plane and move it to the opposite side of the XY-plane. This is typically called a refocusing pulse.

When an on-resonance signal is observed in the rotating frame, it is stationary in the XY-plane (ignoring any relaxation). An off-resonance signal will precess around the XY-plane at a rate equal to the how far off resonance it is. This is the cause of first order phase errors when looking at a spectrum that encompasses a broad range of frequencies. One method to remove these phase errors is to use a spin-echo pulse sequence. “The importance of the spin-echo in modern NMR techniques can hardly be over emphasized” [35]. A typical spin-echo sequence consists of a delay, τ, a 180° refocusing pulse, and a second delay of τ, Figure 4-13. In ssNMR τ

typically equals a rotor period minus half the length of the 180° pulse. The idea for this pulse sequence is that the signals that are above the resonance frequency will be moving away from resonance in one direction and those below resonance in the opposite. If a 180° pulse inverts the whole XY-plane, both of these signals will now be moving back towards their starting point. At the end of the second τ, all the signals will be back where they started with no first order phase errors, Figure 4-14.

The difficulty is that signals that are not on resonance do not experience the same effective pulse length as those on resonance. The goal of optimizing refocusing pulses is to overcome this problem by finding a composite pulse sequence which flips a much broader range of frequencies 180 degrees. This problem has been explored, particularly in liquid state NMR. Examples include: [48-51,53,66,64,65].

Similar to the inversion pulse sequence problem, pulse sequences that are optimized for liquid-state NMR do not always transfer over to ssNMR directly. The simplest refocusing pulse, the 180° pulse, can be used as an example, Figure 4-15. In a liquids simulation, the signal strength goes through a series of beat patterns of fast oscillations that increase in strength as the resonance offset increases as shown by the blue trace. The green trace shows the same simulation using the same RF power, but now in a solids sample with a MAS rate of 10 kHz. The beat pattern has changed to a slower oscillation of increasing strength. It turns out that this oscillation frequency for the solids signal is dependent on both MAS rate and RF power level. Figure 4-16 shows a series of simulations across 4 power levels. The oscillations increase in frequency with increased RF power. The oscillations decrease in frequency with increased MAS rate, Figure 4-17.

Using spinev, I have optimized pulse sequences to give a better refocusing profile than those in the above references when used in ssNMR.

Methods

The first big problem is coming up with a simulation program that accurately describes the refocusing problem. Spinev provides the capabilities to make runtime chosen pulse lengths, but the method was not straightforward at first. An example refocusing pulse sequence input file is shown in Figure 4-18. The spin system shown approximates adamantane and the pulse sequence has 3 pulses. The first and last are the delays, τ. The middle pulse is the composite pulse, specified by the $P replaced by the name of the file containing the composite inversion pulse sequence using the –macro command line option. The foffs scan_par is used to simulate across the different offset frequencies. The $F is also replaced using the –macro command line option. It has the form of low:step:high frequency offsets. The $P is the range of powers to simulate over, is also replaced using the –macro command line option and has the same form as the frequency offsets. Due to the high homogeneity of the probe used in this research, the optimizations were only done at one power level which models perfect homogeneity.

Figure 4-19a shows a typical composite pulse input file (the name of which replaces the $P in Figure 4-18). The first column is the pulse lengths, the second is pulse powers, the third is pulse phases, and the forth is pulse frequency offsets. Figure 4-19b shows the commands used to run a refocusing experiment using spinev.

Initially, differences arose between the simulation data and the experimental data. The problem turned out to be the method of automatically calculating the τ lengths. The initial attempts at calculating them inadvertently turned on the RF during them in the simulations. When attempting to model proton coupling during the τ's the program was rewritten with all the

values hard coded. This greatly improved the match between simulation and experiment. Attempting to reautomate the calculations revealed the location of the error being that the power was on during the τ's.

Most of the methods for optimizing refocusing pulse sequences are very similar to those used in inversion pulse sequences. A generic pulse sequence program is written for spinev. A Matlab master program is written (very similar to the one for inversion) to run fminsearch, which varies the pulse lengths and phase for N pulses. N was varied from one to ten and the optimizations were run on the HPC cluster. Since the signal across various frequencies seemed to fluctuate quite a bit, the frequencies were run in 2 kHz steps initially for optimization. This seemed like a good compromise between calculation time and sampling width for optimizations.

Results

The results for three refocusing pulse sequences are shown in Figures 4-20 to 4-22. Figure 4-20 shows the experimental and simulated results for using a single 180 degree pulse for refocusing. The simulation and experiment match quite well. This data was taken with an RF power of 51.2 kHz and a MAS rate of 10 kHz across an offset frequency range of +/- 50 kHz. Figure 4-21 shows data with the same settings for a composite refocusing pulse developed by Bai et al. [65,66]. It was optimized using simulated annealing to be phase-distortionless composite pulses. They were demonstrated as inversion composite pulses, but being phase distortionless, they should work well as refocusing pulses too [53] and the simulation supports this. The experimental data does not fit as well as the 180 data does. This may be the result of a slight misset of the spectrometer either in specifying the true resonance correctly or in calibrating the length of the pulses correctly. Even with it not fitting well, the response is far superior to that of a 180 pulse. Figure 4-22 shows a composite pulse obtained through the research presented in

this paper. It is marginally longer than the pulse sequence by Bai et al., 1224 degrees vs 1166 degrees. The simulation and experimental data fit better than the data in Figure 4-21, but still not as well as that for the 180. The more complex composite pulses in a more complicated situation provide for more chance of divergence between simulation and experiment. A comparison of all three pulse sequences is shown in Figure 4-23.

To numerically compare these composite pulses, the average value across a frequency-offset range of +/-2 $\Delta\omega/\omega$ is compared. The 180 has an average value of 0.4422 for simulation, the composite pulse by Bai et al. has an average value of 0.5598 for experimental data and 0.5893 for simulation, and the composite pulse optimized for this research has an average value of 0.8594 for experimental data and 0.8910 for simulated data. Both of the composite pulses are better across the frequency offsets than a 180 pulse, with the composite from this research being the best. If comparing across only +/-1 $\Delta\omega/\omega$ the tables turn a bit. Bai et al.'s pulse turns out to be better than the pulse sequence from this research over the narrow range. Both composite pulses are still better than the 180.

The composite pulses do come at a cost. Bai et al.'s pulse sequence is 6.5 times as long as the 180 and the composite from this research is 6.8 times as long as the 180. This is a tradeoff that has to be understood well when using composite pulses. They do not always function well as straight drop-in replacements for a single 180 pulse.

Figure 4-24 shows a comparison of the pulse sequence optimized for this research at two different sample lengths, 3.7 mm and 11.7 mm. The difference is much more drastic here than with the inversion pulse sequences. The longer sample matches the simulation noticeably worse than in the inversion pulse sequences. The simulation assumes perfect homogeneity and neither of the composite pulse sequences were optimized to reduce the effects of inhomogeneity.

Optimizing the DRAWS Pulse Sequence

Introduction

The pulse sequences being optimized so far are simple pulse sequences that are building blocks for much more complex experiments. One of the main pulse sequences used in the Long lab is the DRAWS pulse sequence [30], Figure 4-25, therefore this was the first pulse sequence optimization was attempted on. This pulse sequence is designed to determine internuclear distances in doubly labeled samples via windowless dipolar recoupling. In simpler terms, this means that the distance between two ^{13}C labeled amino acids in a peptide or protein can be measured. Windowless means that it is a pulse sequence where the transmitter stays on during the whole excitation period. Some pulse sequences have periods where the transmitter is turned off and the spins are allowed to freely precess for a period of time. The buildup form of the DRAWS pulse sequence does not have any free precession times. However, once DRAWS has been used to build up double quantum (DQ) coherence, it is possible to let the DQ coherences evolve before running DRAWS in reverse to bring the signal back to single quantum states where it can be directly observed. This has been shown to be quite useful in finding backbone torsion angles of peptides [31]. It has also been used by our lab to help illuminate the structure of a peptide mimic of lung surfactant protein B, one of the proteins essential to lung function [67].

The theoretical maximum efficiency for DRAWS varies depending on the particular sample being run, but they average in the mid to low forty percent range [32]. However, the experimental efficiencies are lower, in the high twenty to low thirty percent range [32]. It would be very nice if the pulse sequence could be optimized to make it easier to achieve the theoretical maximum efficiency and possibly increase this theoretical maximum. This could involve optimizing a pulse sequence for each specific sample, but ideally, a more general result could be

obtained. It might also be handy to optimize other double quantum recoupling experiments to make them easier to setup to achieve their maximum efficiency.

Double quantum recoupling efficiency (DRAWS in particular) diminishes as the static field, B_0, strength increases due to interactions of the increased size of the CSA [32]. This counteracts the signal increase from the improved sensitivity these higher fields bring. It would be nice if a pulse sequence could be found that is not so sensitive to the increase in static field strength.

These recoupling experiments (again, DRAWS in particular) are also quite sensitive to B_1 inhomogenieties. Reduced homogeneity noticeably reduces the DQ efficiency. It would be nice if a pulse sequence could be found that is also more tolerant to inhomogeneity.

The goal of this dissertation section was to optimize the DRAWS pulse sequence to improve its efficiency and robustness with respect to B_0 field strength and inhomogeneity of the B_1 field.

Methods

Several methods of optimizing the DRAWS pulse sequence were tried. The bulk of them, particularly at first, used the SIMPSON simulator [42]. The original DRAWS pulse sequence is shown in Figure 4-25. Each basic R group consists of 10 pulses. Eight of them are 360 degrees in length (measured in nutation distance) and two of them are 90-degree pulses, Figure 4-26.

The first method was to use Fourier components to define a pulse sequence in the manner of Sakellariou et al. [54]. The method assumes a windowless experiment at constant amplitude RF. The only variable factors are the phases throughout the excitation. Rather than treating the phase of every pulse as a variable, which could be hard to optimize if there were hundreds of pulses, they reduced the number of variables by treating the phase throughout the experiment as the waveform described by a Fourier series. The parameters to optimize are the complex Fourier

coefficients. They chose to use 6 complex coefficients, which leads to 12 parameters. I tried this Fourier component optimization method without successfully finding pulse sequences with better properties than the original DRAWS pulse sequence when I first started on the project of optimizing pulse sequences. It might be worth revisiting in the future with the simpler experiments currently under development.

The next optimization method replaced the R group with 10 equal length pulses. The length of each R group was still τ_R, one rotor period, and the same supercycling of the 4 R groups, R-Rbar-Rbar-R, was used. The power level was also fixed at the same level that would have been used for a regular DRAWS sequence. The new R group also maintained reverse symmetry, so that only the first half of the R group was optimized. The second half of the R group was simply a reflection of the first half. Some optimizations were done without this constraint, but they never produced pulse sequences even as good as the original DRAWS pulse sequence. The optimization procedure searched for the phases for the first half of the pulse sequence (the second half used a reflection of them) that produced the maximum signal. The buildup curve was done at each optimization step out to 10 supercycles and the maximum value in those 10 supercycles was maximized. This method worked well and was used for the bulk of the optimizations done on DRAWS.

The last method of optimizing DRAWS used the original pulse lengths from DRAWS, rather than 10 equal length pulses. Just the phases for each pulse were varied to see if a better set of phases could be found.

Results

Both the experiments optimized in SIMPSON and in spinev were run on the 750 Mhz spectrometer using the low-E MAS probe. Neither sets of experimental data matched the

predicted simulation data in shape or intensity. A representative example of a pulse sequence optimized using spinev and pulse lengths the same length as the original DRAWS is shown in Figure 4-27. The simulation and experimental results are shown in Figure 4-28. The simulation showed that there should be an increase in signal by about 30%, but the experimental results showed a decrease in signal to about 60% of the original DRAWS signal. This set of data also gives some clues about how optimizing this kind of pulse sequence could be improved in the future. Note that the original DRAWS data does not match up well between simulation and experiment. Tentative results suggest that matching the theoretical spin system to the experimental spin system better could reduce this discrepancy. It was originally hoped that the optimized pulse sequences would be robust across variations in the spin system. Experimental data suggests otherwise.

In working on the inversion and refocusing problems, a method of optimization emerged. Start by running a simulation of the experiment. Next run the experiment. Try to fit the experimental data with the simulation. It maybe helpful to run optimizations of the experiment, run the experiments and try to fit both the original and optimized data with simulation. Once the simulations and experiments match well, move on to more optimizations and better optimization methods.

In the future, attempts should be made to find simulations that accurately describe the experimental results from implementing the optimized pulse sequences.

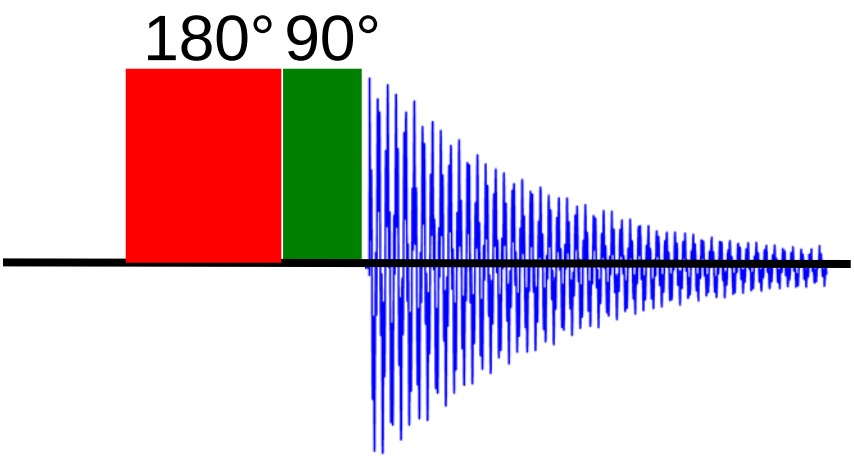

Figure 4-1. Inversion pulse sequence. An inversion pulse sequence rotates the magnetization from along Z 180 degrees to be along -Z. In order to observe this signal, a 90 degree pulse is used to rotate the magnetization back to the X-Y plane. To test the frequency robustness of an inversion pulse, the inversion pulse is transmitted off resonance, but the 90 degree pulse is kept on resonance.

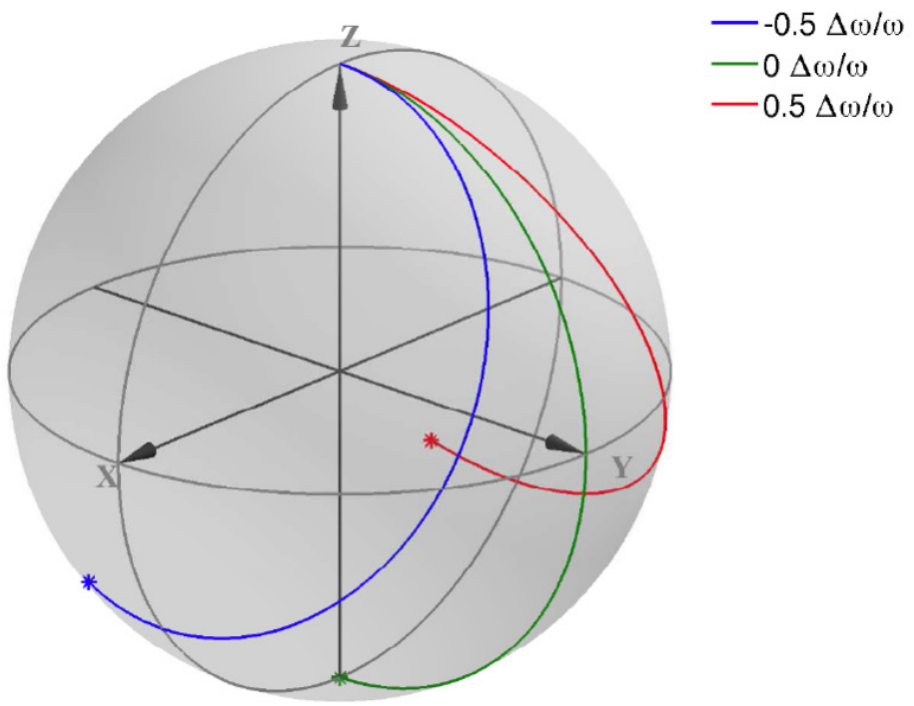

Figure 4-2. Inversion trajectories for 180 degree pulse. Single 180° pulse around -X trajectory paths for a spin on resonance (green), at +0.5 Δω/ω (red) and -0.5 Δω/ω (blue). A Δω/ω of 0.5 is equivalent to a frequency offset of 30 kHz for an RF power level of 60 kHz. This simulation does not model crystallites or magic angle spinning. In other words it assumes a liquid.

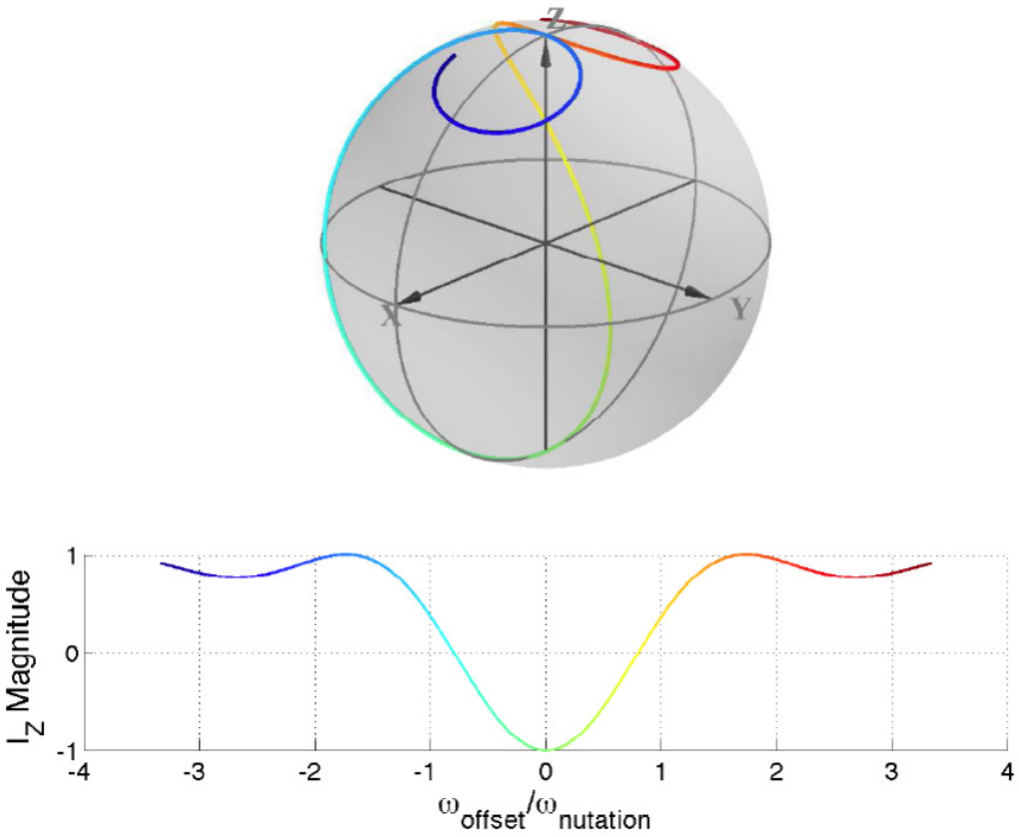

Figure 4-3. Endpoints for 180 degree pulse with varying frequecy offset. Endpoints of a single 180° pulse around -X while varying the $\Delta\omega/\omega$. A 0.5 $\Delta\omega/\omega$ as shown in Figure 4-2 would be in either the cyan or yellowish green regions. This simulation does not model crystallites or magic angle spinning. In other words it assumes a liquid.

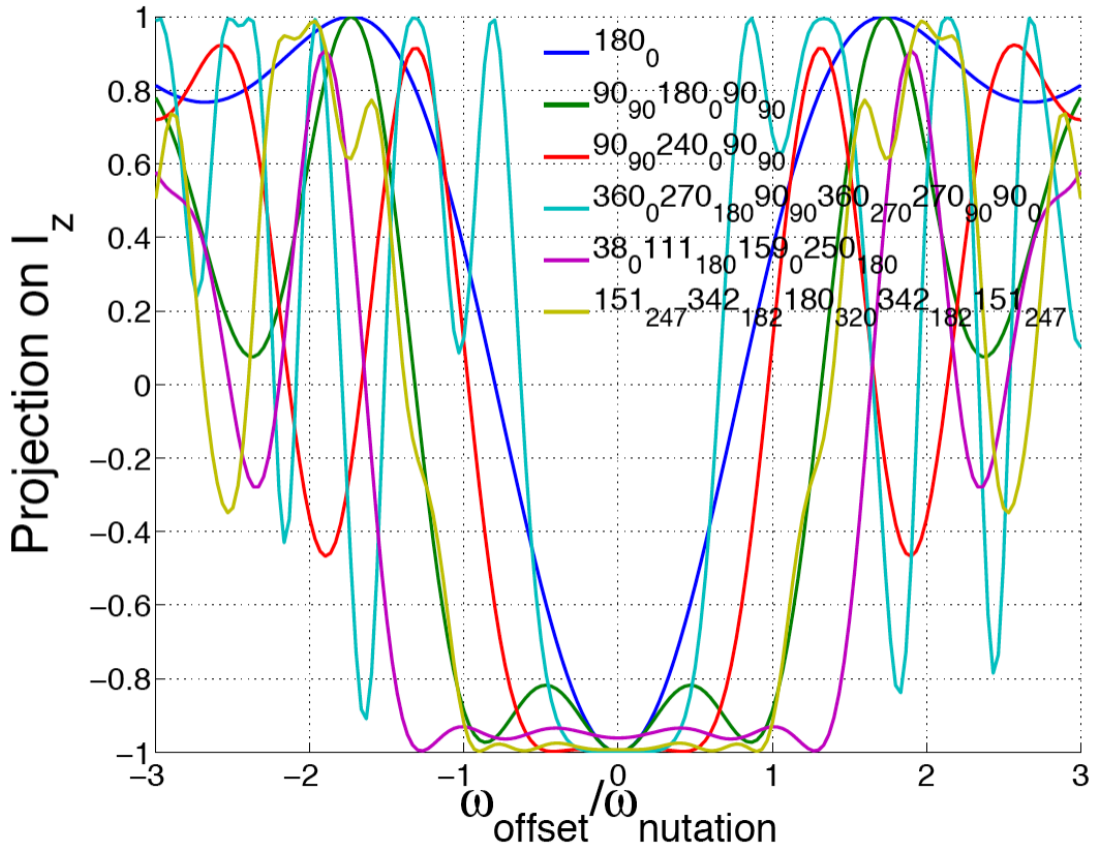

Figure 4-4. Simulations using SPINEVOLUTION of various liquids composite inversion pulses simulating an adamantane-like solid spinning at 10 kHz. A regular 180 pulse around X is in blue. References for the rest of the sequences is as follows: green [48], red [49], cyan [51], magenta [64], yellow [65,66].

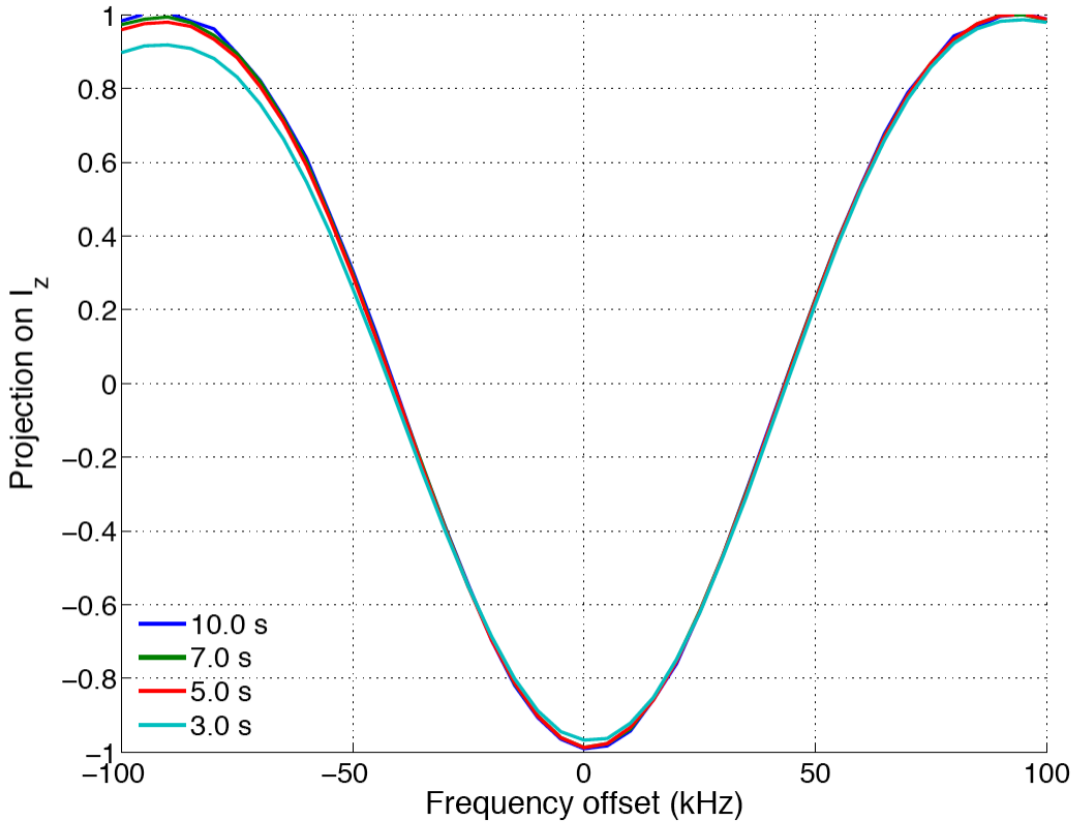

Figure 4-5. Finding a useful recycle delay. Initial Δf experiments did not match simulation well. This was because the recycle time for the sample used (adamantane) was too short. Here are some examples of what the results are for various recycle times. It turns out that a recycle delay of 7 seconds is usually enough for adamantane.

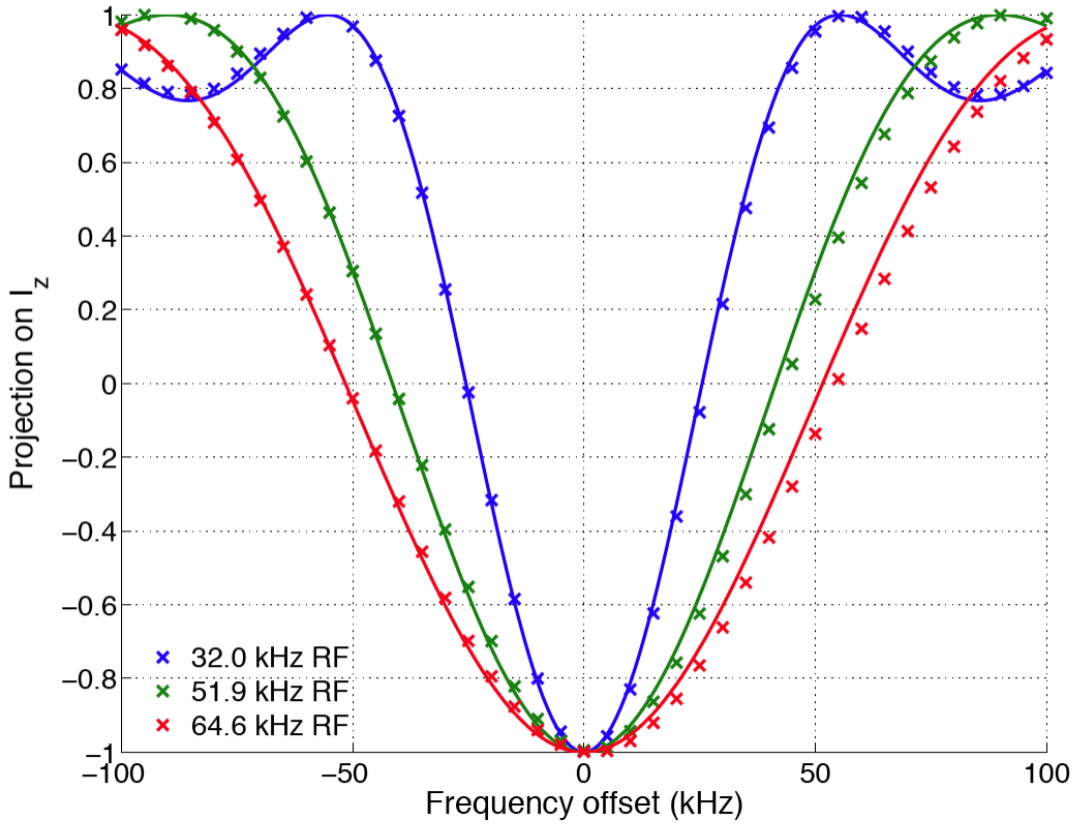

Figure 4-6. Comparison of experimental and simulation data (SPINEVOLUTION) for a single 180° inversion pulse. The pulse sequence was a 180° followed by a 90° to bring the bulk magnetization back into the X-Y plane for observation. The transmit frequency offset was only changed on the 180° pulse. The experimental data is from integrating the adamantane resonance at 38.48 ppm for a sample length of 3.7 mm.

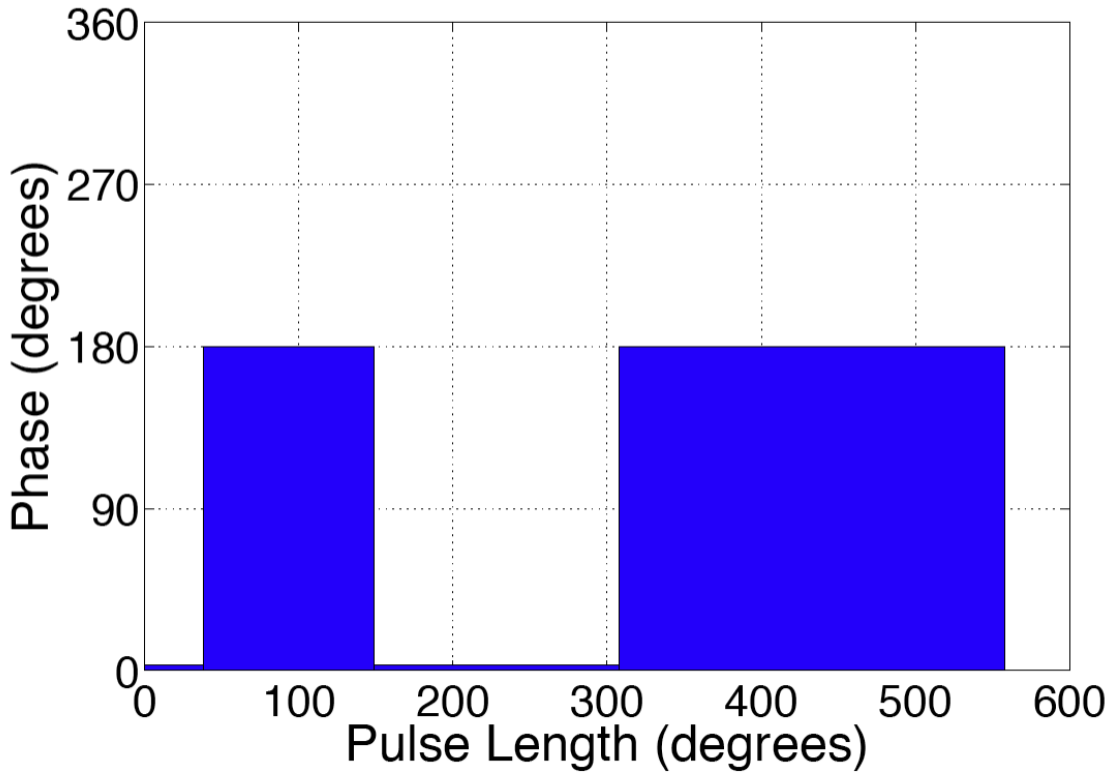

Figure 4-7. Composite inversion 180° pulse showing the lengths and phases of the pulses. This pulse sequence was developed by Sanctuary et al. [68,64] via the series expansion of the offset angle method.

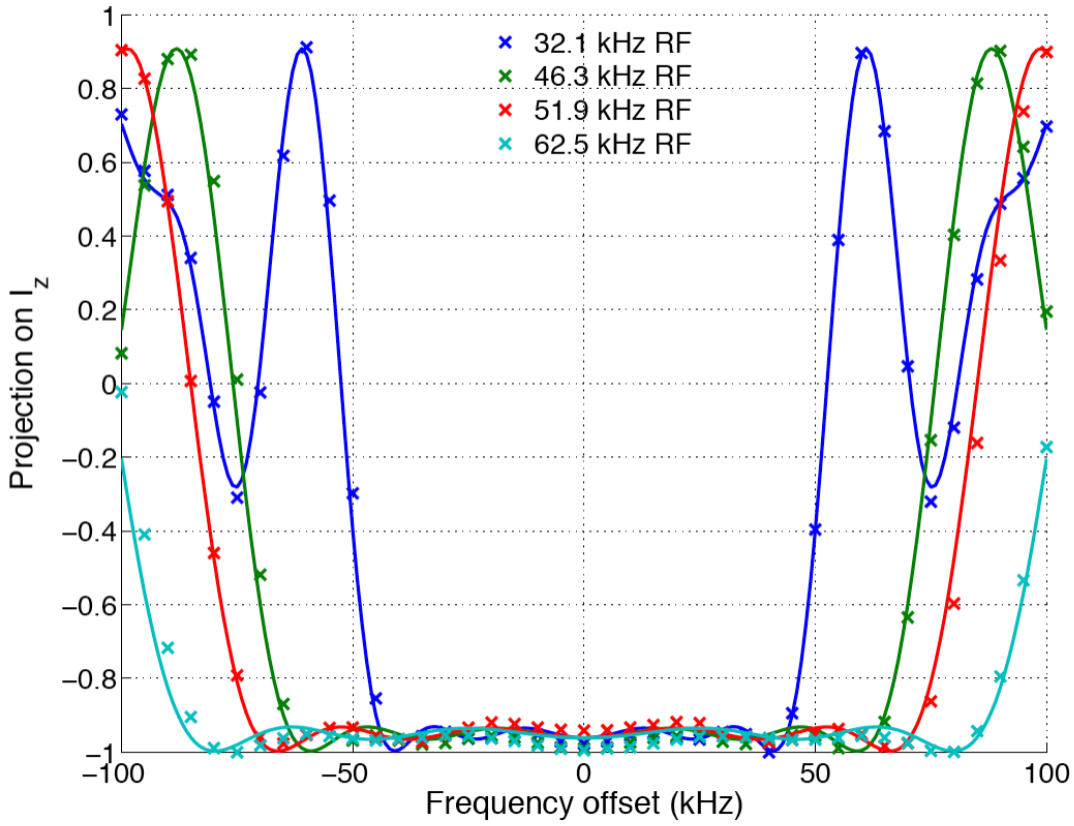

Figure 4-8. Comparison of experimental and simulation data for a composite inversion pulse. This pulse sequence was developed by Keniry and Sanctuary [64] via the series expansion of the offset angle method. Note how flat the response is in the middle. Experimental data is integration of the adamantane peak at 38.48 ppm for a sample length of 3.7 mm.

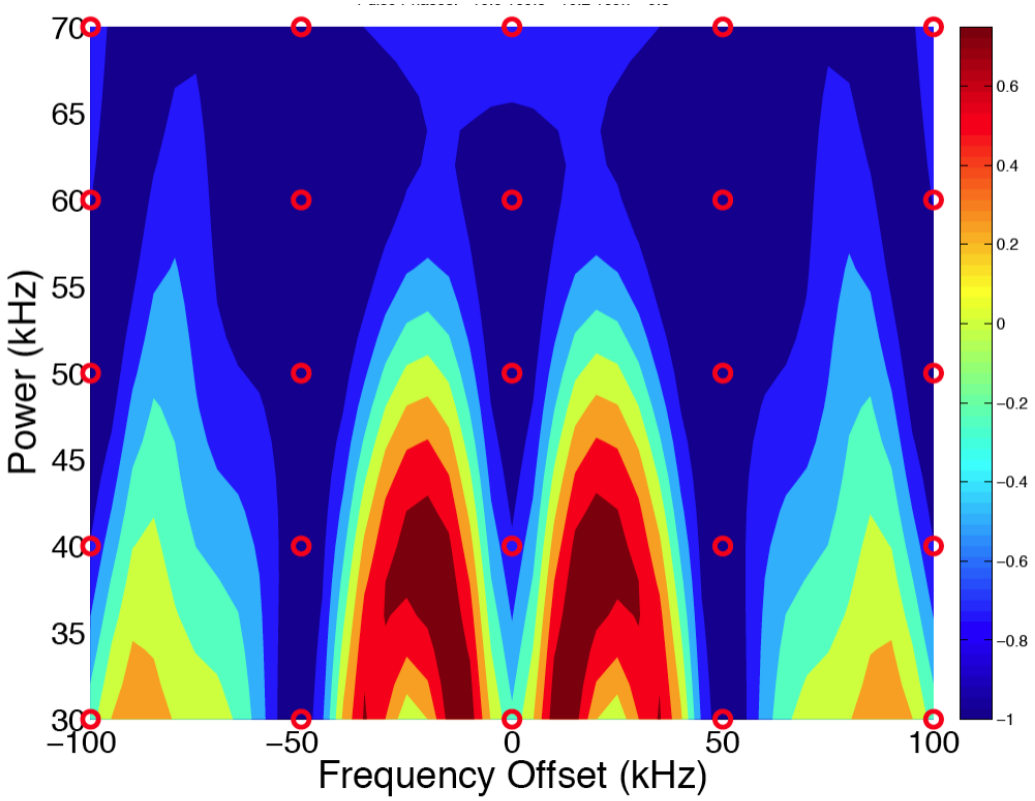

Figure 4-9. Optimization undersampling. The pulse sequence was optimized by minimizing the signal at the points indicated by the red circles. The signal is nicely minimized at these points, but not in-between which is an example of over fitting or maybe under fitting. A tighter grid of points is needed. Since the homogeneity of the 750 MHz probe is so high, optimizing over different power levels is not needed. Therefore, all the time can be focused on optimizing over more points at the desired power level (usually 50 kHz).

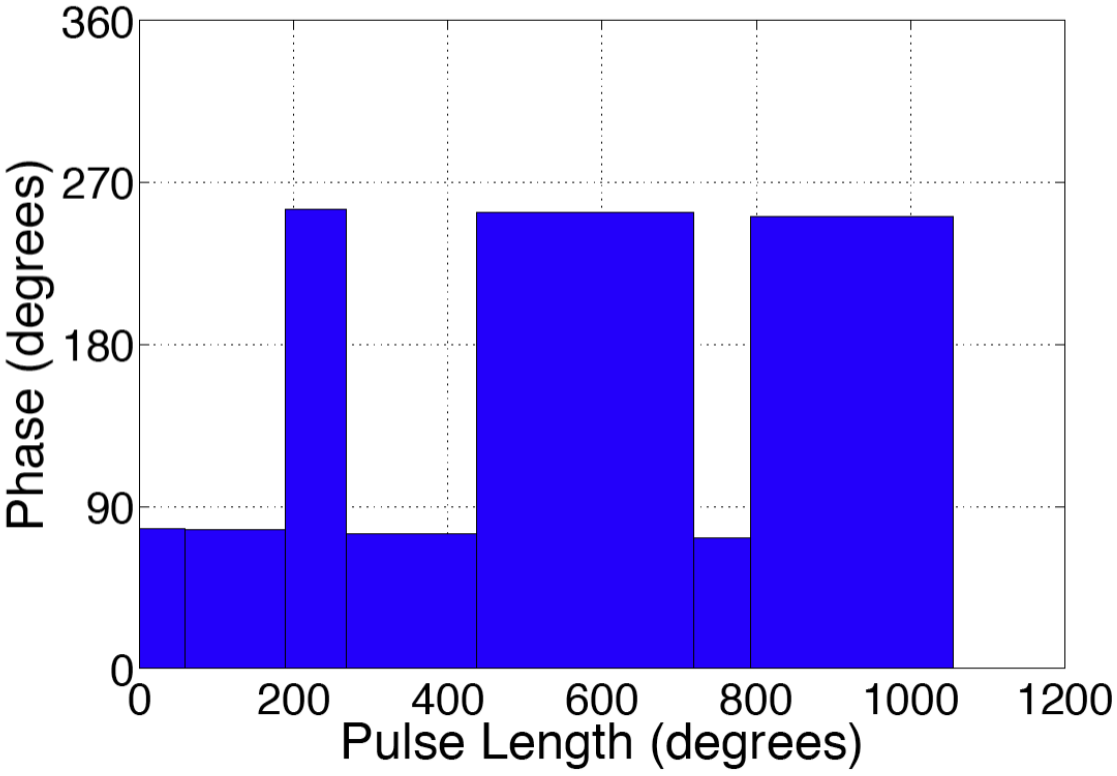

Figure 4-10. An optimized composite inversion 180° pulse showing the lengths and phases of the pulses. This composite pulse was optimized by the author during this research using the Nelder-Mead simplex method and spinev. This pulse sequence is just under 6 times the length of a single 180° pulse.

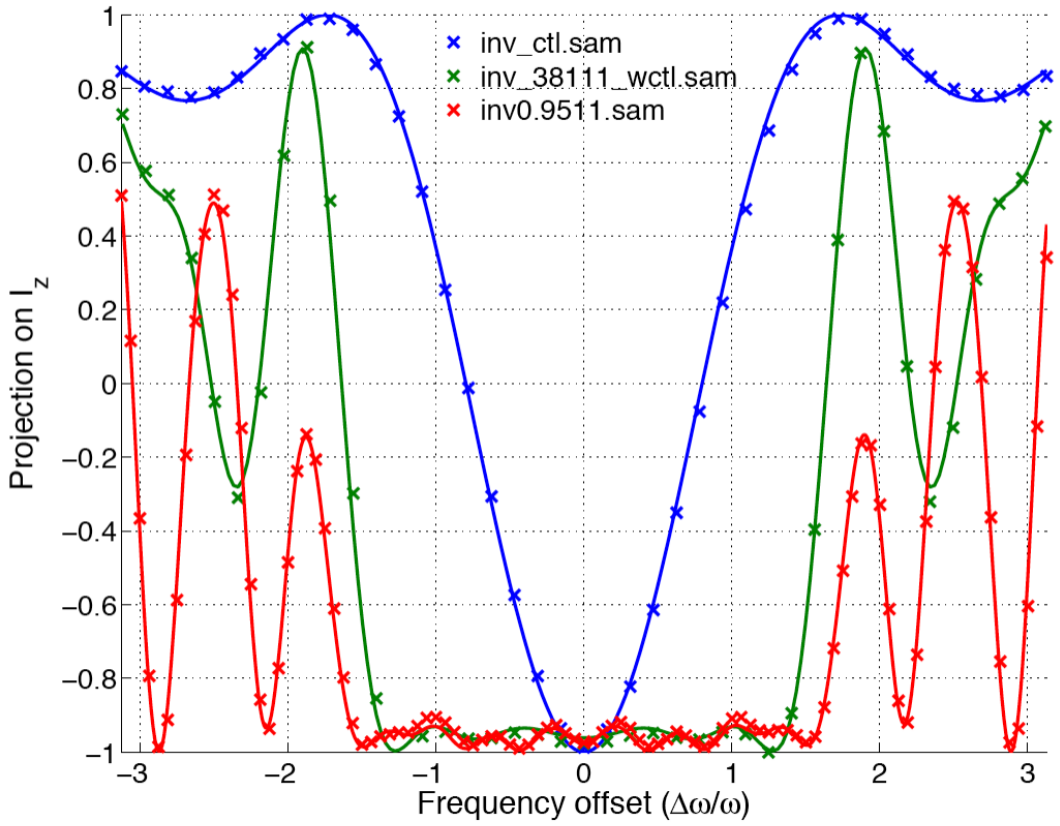

Figure 4-11. Comparison of various inversion composit pulses. This figure compares all three pulse sequences discussed so far at the lowest power level. The lowest power level has the narrowest bandwidth. A standard single 180° pulse is in blue. The composite by Keniry and Sanctuary [64] is shown in green and the pulse sequence optimized for this research is shown in red. Note that the optimized composite pulse has a wider bandwidth at a minimal expense of greater ripple.

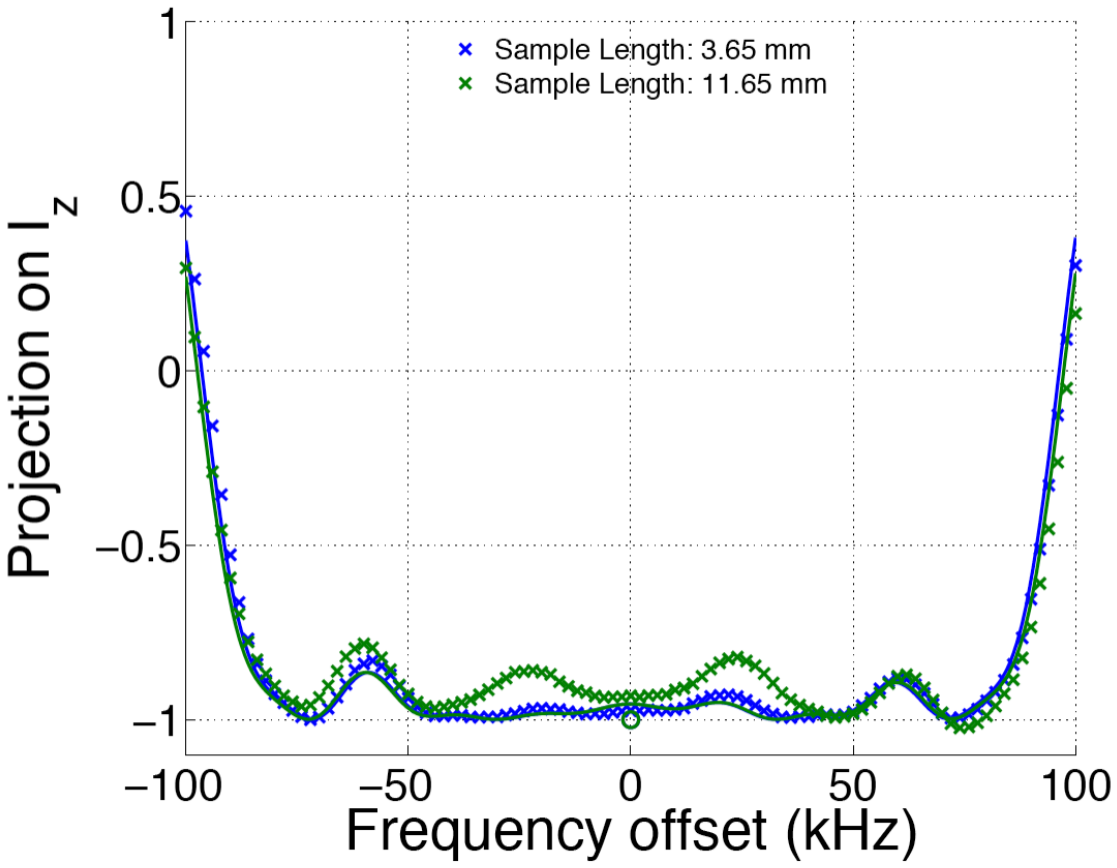

Figure 4-12. Inversion homogeneity match to simulation. The match between simulation (green line) and experiment (X's) is not as good for longer samples as it is for shorter samples. This example is for an optimized pulse sequence at 52 kHz RF. The homogeneity of the ^{13}C channel for the 3.7 mm sample is ~96% and for the 11.7 mm sample is ~76%.

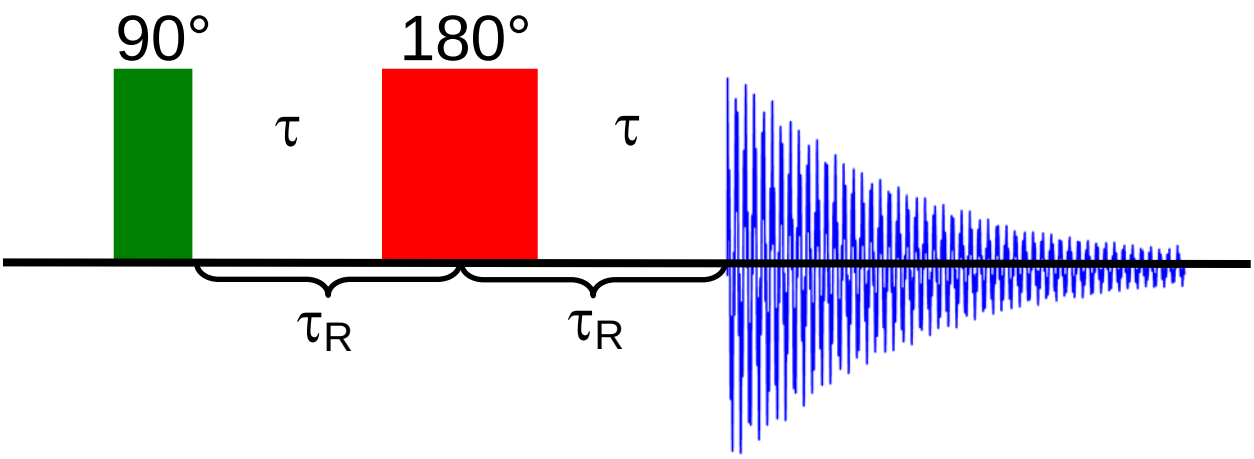

Figure 4-13. Refocusing pulse sequence. A refocusing pulse sequence begins by rotating the magnetization into the X-Y plane. Following that, there is a delay of τ, where τ is equal to a rotor period, τ_R, minus half of the length of the 180 refocusing pulse. Another τ follows the 180 pulse before acquisition commences.

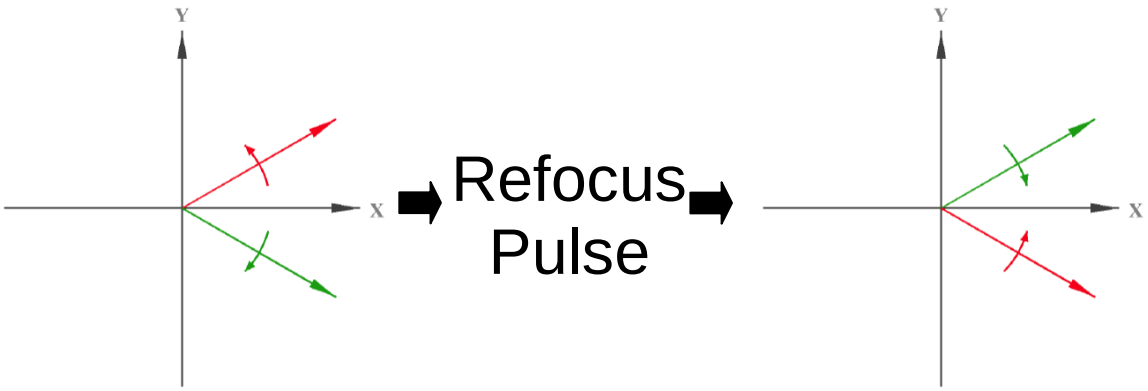

Figure 4-14. Refocusing explaination. In the rotating frame of reference, spins at a higher frequency than the reference frequency will rotate away from their starting position (X here) in one direction (red), while those at a lower frequency will rotate in the opposite direction (green). A refocusing pulse inverts the X-Y plane so the resonances exchange positions while retaining their direction of rotation. At some time, τ, after the refocusing pulse, the spins will all be aligned along their starting position again.

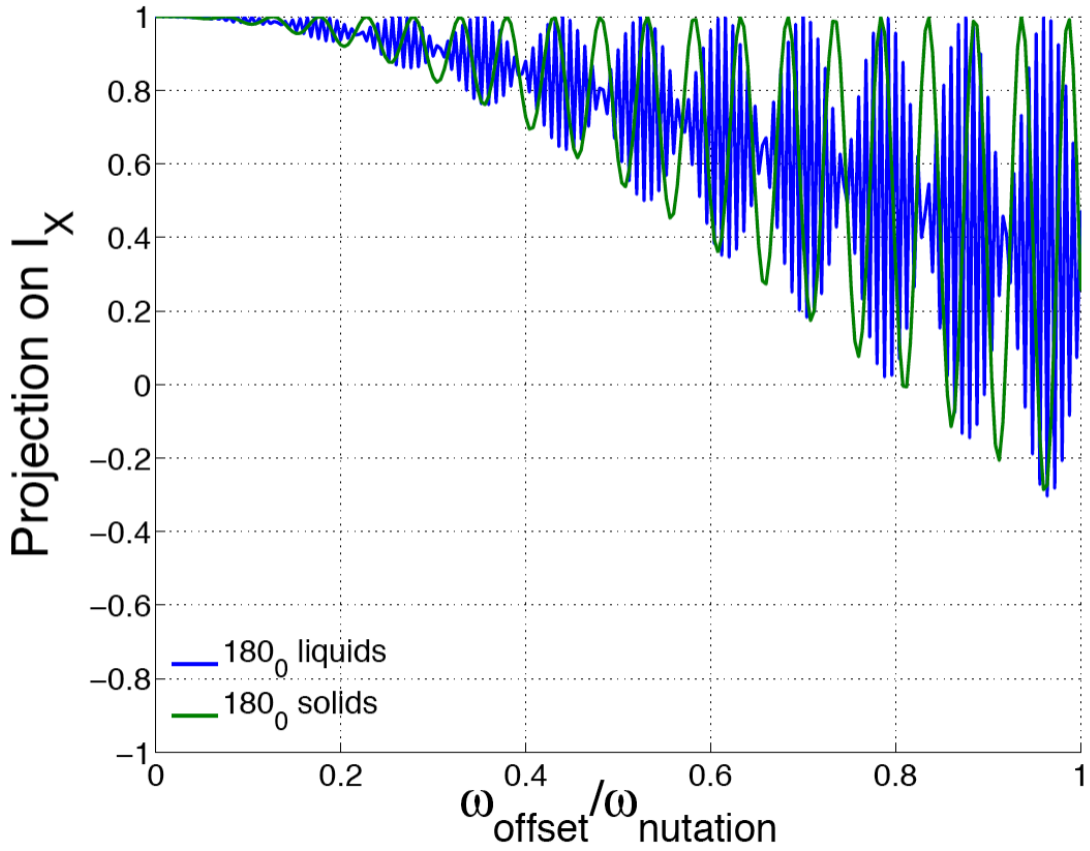

Figure 4-15. Refocusing comparison between liquids and solids. Simulations of a liquids refocusing experiment and a solids refocusing experiment at 10 kHz MAS show how different the results can be between the two cases. Both use the same values for τ and a spin system that approximates adamantane.

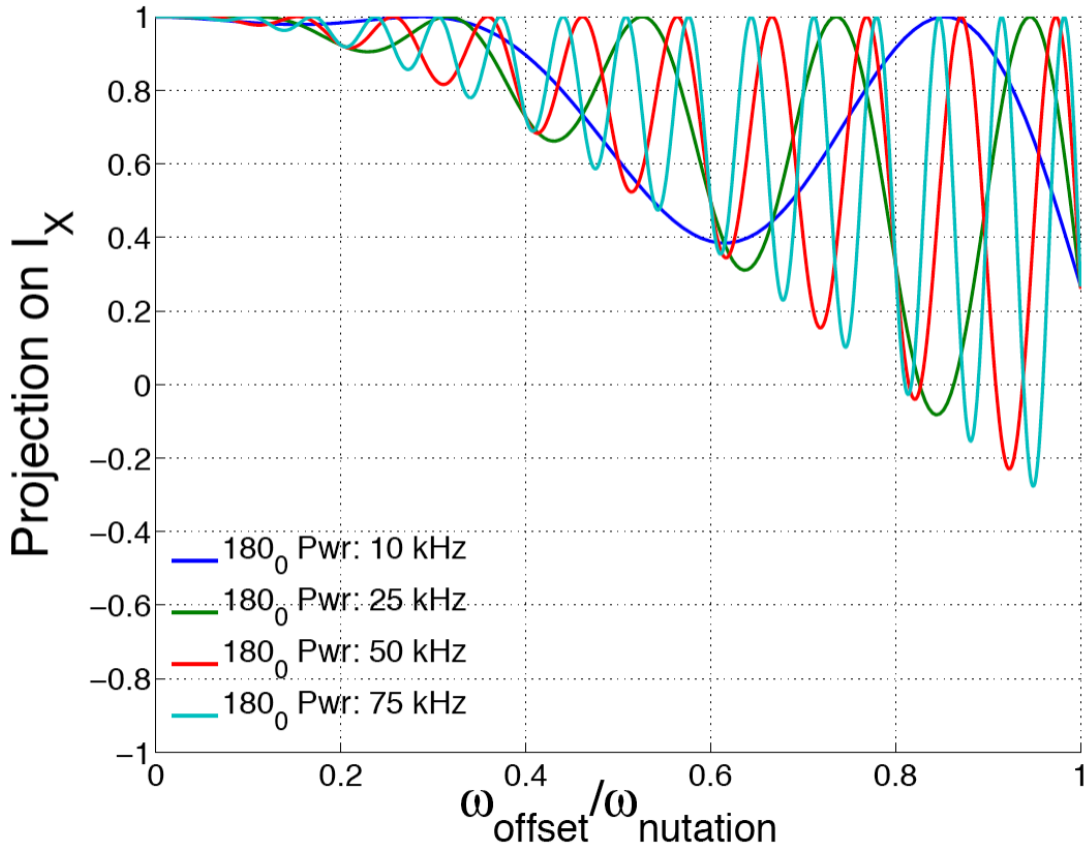

Figure 4-16. Solids refocusing power response. The response of a refocusing pulse is sensitive to both the MAS rate and the power of the applied pulse. This figure shows simulations of how the response varies according to power level for a solid sample with spin parameters that approximate adamantane with a MAS rate of 10 kHz.

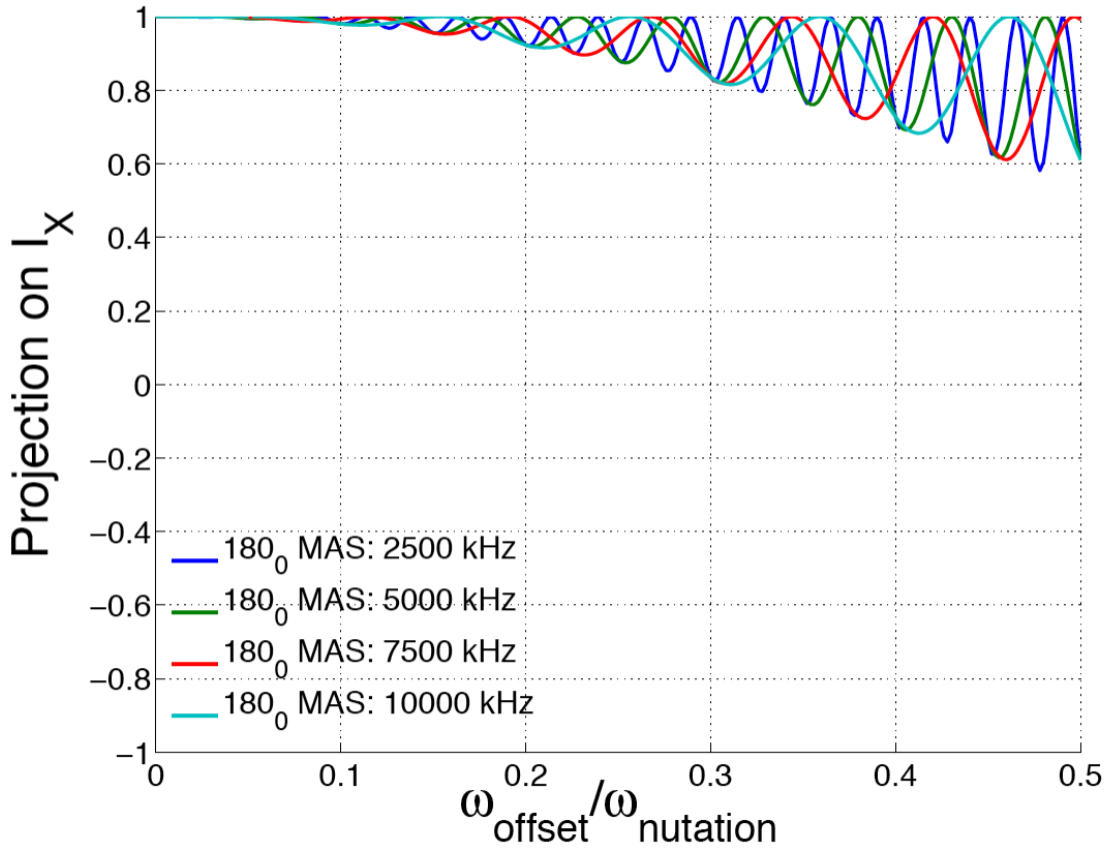

Figure 4-17. Solids refocusing MAS rate response. Simulations of a 180 refocusing pulse around +X at various MAS rates using a spin model that approximates adamantane show how the response varies with MAS rate.

```
****** The System *******
spectrometer(MHz)  750
spinning_freq(kHz) 10.000000
channels           C13
nuclei             C13
atomic_coords      *
cs_isotropic       0 ppm
csa_parameters     1 -4 0.1 0 0 0 ppm
j_coupling         *
quadrupole         *
dip_switchboard    *
csa_switchboard    *
exchange_nuclei    *
bond_len_nuclei    *
bond_ang_nuclei    *
tors_ang_nuclei    *
groups_nuclei      *
******* Pulse Sequence *****************************
CHN 1
timing(usec)         1 $P 1
power(kHz)           0  * 0
phase(deg)           0  * 0
freq_offs(kHz)       0  * 0
phase_cycling        13 11(RCV)
******* Variables *********************************
* scan across frequency offsets
scan_par     foffs/$F/
             freq_1_1=foffs
* scan across power missets
scan_par     noffs/$N/
             power_1_1=noffs
* Set the first and last pulses back to zero power
             power_1_1_1=0*noffs
             power_1_1(nall)=0*noffs
Tr=1000/spinning_freq
n=size(pulse_1_1,1)-1
nall=n+1
pulse_1_1_1=Tr-sum(pulse_1_1(["2:n"]))/2
pulse_1_1(nall)=pulse_1_1_1
******* Options ************************************
rho0               I1x
observables        I1x
EulerAngles        rep700.dat
n_gamma            50
line_broaden(Hz)   *
zerofill           *
FFT_dimensions     *
options            -scheck3
*** refocus.spv ***
*** Automatically created by /Users/mcnese/Seth/bin/write_spinev_refocus ***
*** Created at 08-Jan-2009 09:22:23 ***
```

Figure 4-18. A typical spinev input file for simulating a refocusing experiment. This file requires 3 command line inputs: $P, the name of the file containing the refocusing pulses, $F, the frequency range and step in the form of low:step:high, and $N, the powers to scan over also in the form of low:step:high.

a)

```
27.848861 32.258065 253.819629 0.000000
23.164910 32.258065 154.993246 0.000000
10.180519 32.258065 328.578250 0.000000
23.384791 32.258065 153.043209 0.000000
9.007919 32.258065 236.164760 0.000000
4.779880 32.258065 309.853269 0.000000
13.310008 32.258065 305.223512 0.000000
0.016393 32.258065 196.789120 0.000000
*** Duration Pwr Phase Offset ***
*** refocus_opt.pp ***
*** Automatically created by /Users/mcnese/Seth/bin/write_spinev_pp ***
*** Created at 08-Jan-2009 09:22:24 ***
```

b)

```
source ~/.profile; spinev refocus.spv -t -macro\$P="refocus_opt.pp" \
   -macro\$F=-100.00:10.00:100.00 -macro\$N=50.00:1.00:50.00 -re -split2 -renice10
```

Figure 4-19. spinev input pp file and commandline call. A) A typical pulse input file for a spinev main program such as that shown in Figure 4-19. The first column is the length of the pulses in microseconds. The second column is the power for each of the pulses in kHz. The third column is the phase of the pulses in degrees. The last column is the frequency offset for the pulses in Hz. B) The command line argument used to run the scripts in Figure 4-19 and 4-20a. It sources .profile because the terminal MATLAB loads does not automatically do this. .profile updates the paths to include spinev. The -t argument returns the output to the command line where matlab reads it in. The -macro commands do text replacement in the .spv file. -re makes the script only return the real part of the data. -split2 splits the process into two threads for faster execution on a computer with at least 2 processor cores. The -renice10 command nices the threads so that they don't take over the computer.

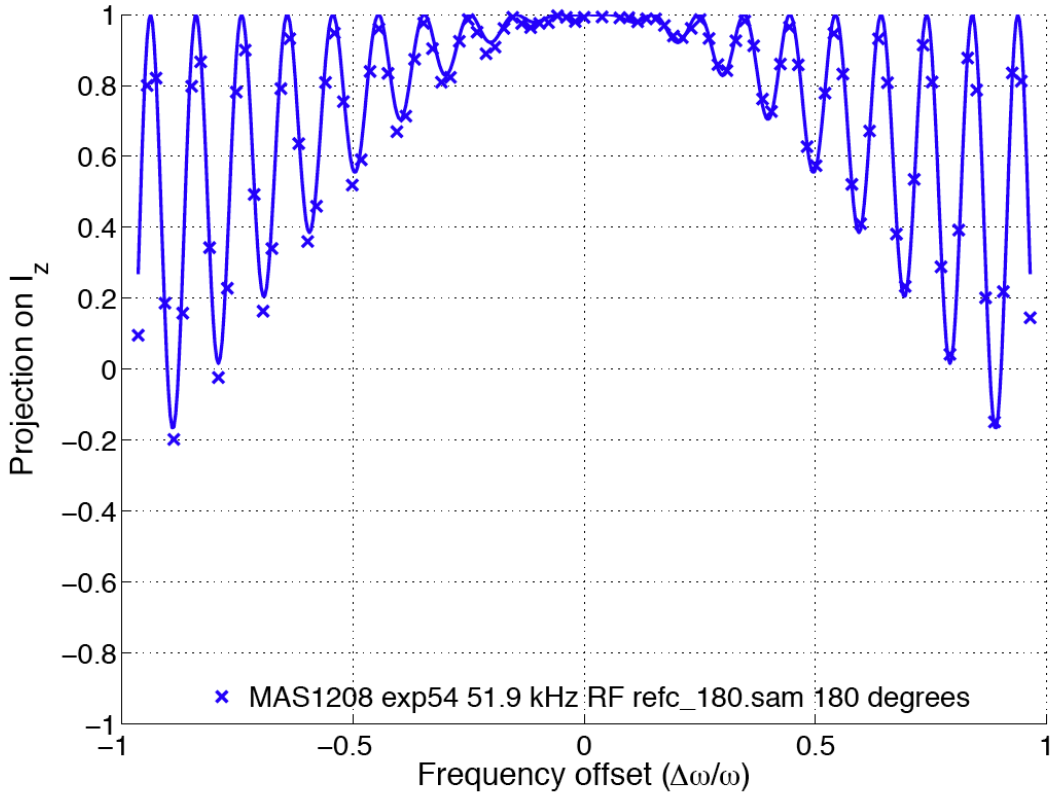

Figure 4-20. Simulation and experiment for a single 180 refocusing pulse. The simulation and experimental data for a 180 refocusing pulse match very well. This data is at 51.9 kHz RF power and 10 kHz MAS rate. The average value across Δω/ω of +/-1 for the experimental data is 0.7246 and 0.7580 for the simulation data. This gives some idea of the refocusing ability of this pulse sequence. The simulated average value across a Δω/ω of +/-2 is 0.4422.

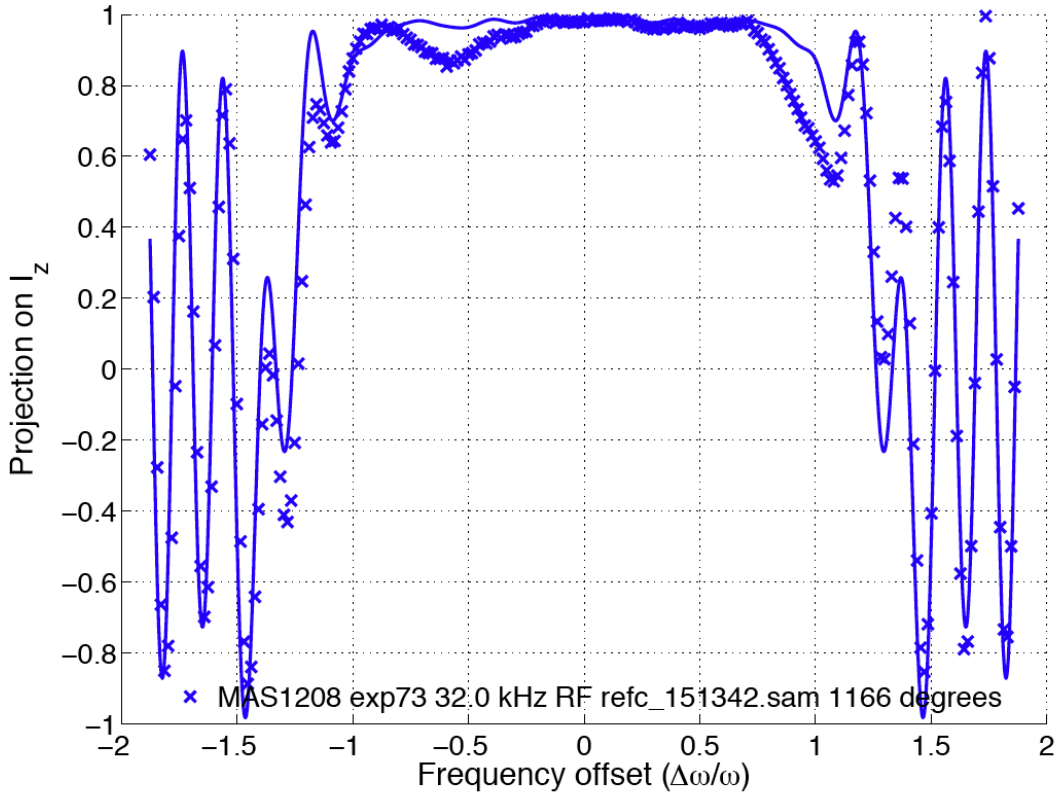

Figure 4-21. The refocusing data and simulation for the composite pulse by Bai et al. [65]. The simulation and experimental data don't fit as well as the 180 shown in Figure 4-20, but it shows a remarkable improvement in bandwidth from the 180. The average value across Δω/ω of +/-2 for the experimental data is 0.5598 and 0.5893 for the simulation data. From Figure 4-21, the 180 has an average value of 0.4422 over +/-2 Δω/ω so this pulse sequence is definitely better than just a 180. Over +/-1 Δω/ω this composite pulse has an average value of 0.9330/0.9696 experimental/simulated which is significantly better than just a 180 pulse.

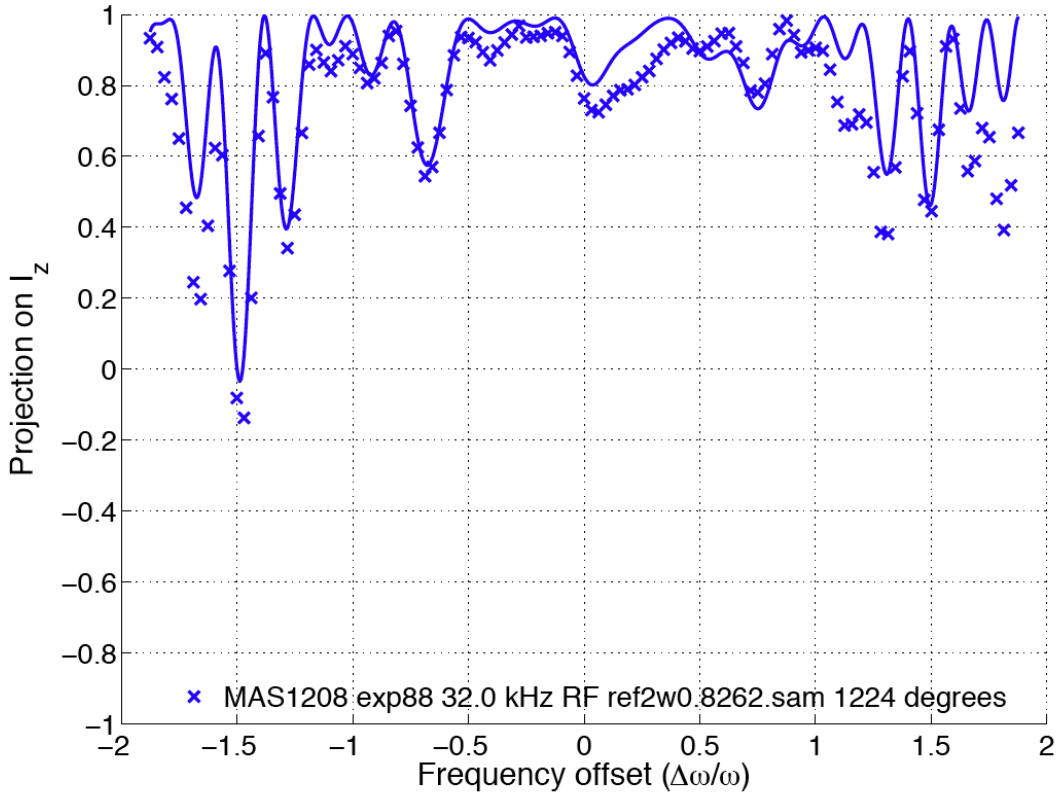

Figure 4-22. The simulation and experimental data optimized in this research. The average value across Δω/ω of +/-2 for the experimental data is 0.7490 and 0.8392 for the simulation data. This gives some idea of the refocusing ability of this pulse sequence. Over +/-1 Δω/ω this composite pulse has an average value of 0.8594/0.8910 experimental/simulated so this pulse sequence is significantly better than just a 180 pulse for both frequency offset ranges. The composite pulse in Figure 4-21 does better over the narrower range, but this pulse sequence does better over the wider frequency offset range.

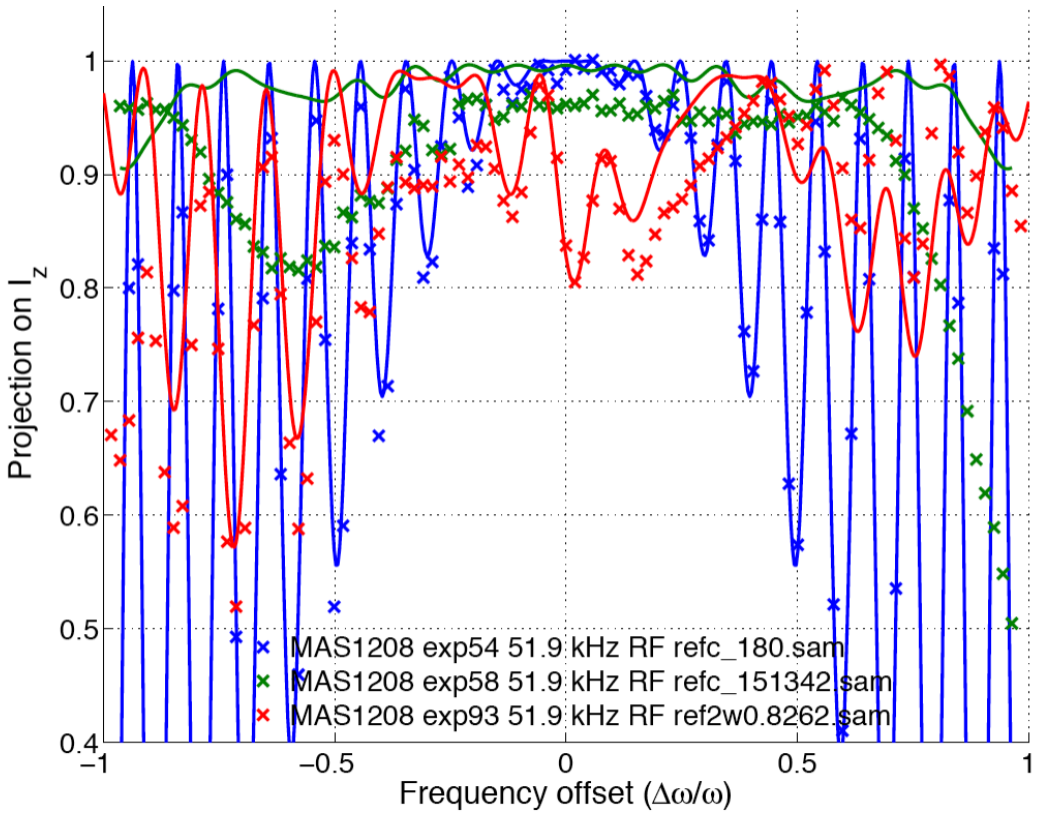

Figure 4-23. A comparison of all three refocusing pulses. The bandwidth of both composite pulses is better than that of a single 180.

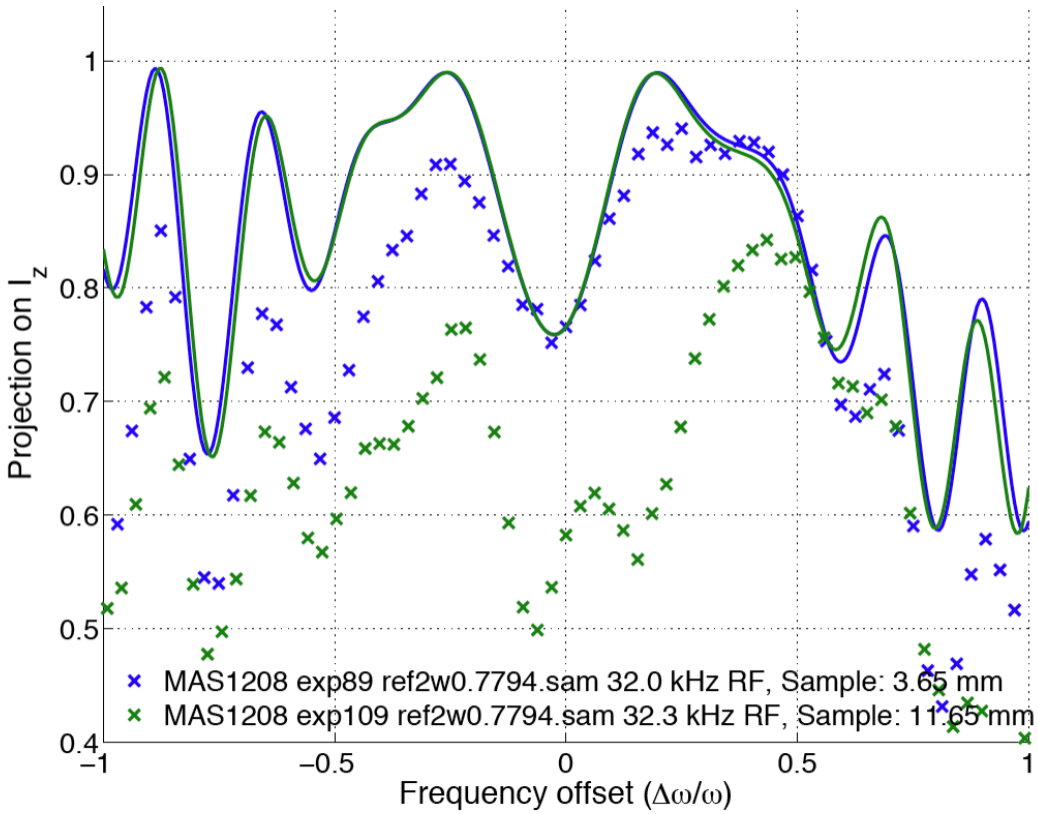

Figure 4-24. Homogeneity and refocusing pulse simulations. Comparing experiment (X's) to simulation (lines) for long samples (11.7 mm green) as well as short samples (3.7 mm blue) for a composite pulse sequence optimized in this research shows that just like for the inversion pulse sequences, the match gets worse. Modeling inhomogeneity may help reconcile the differences.

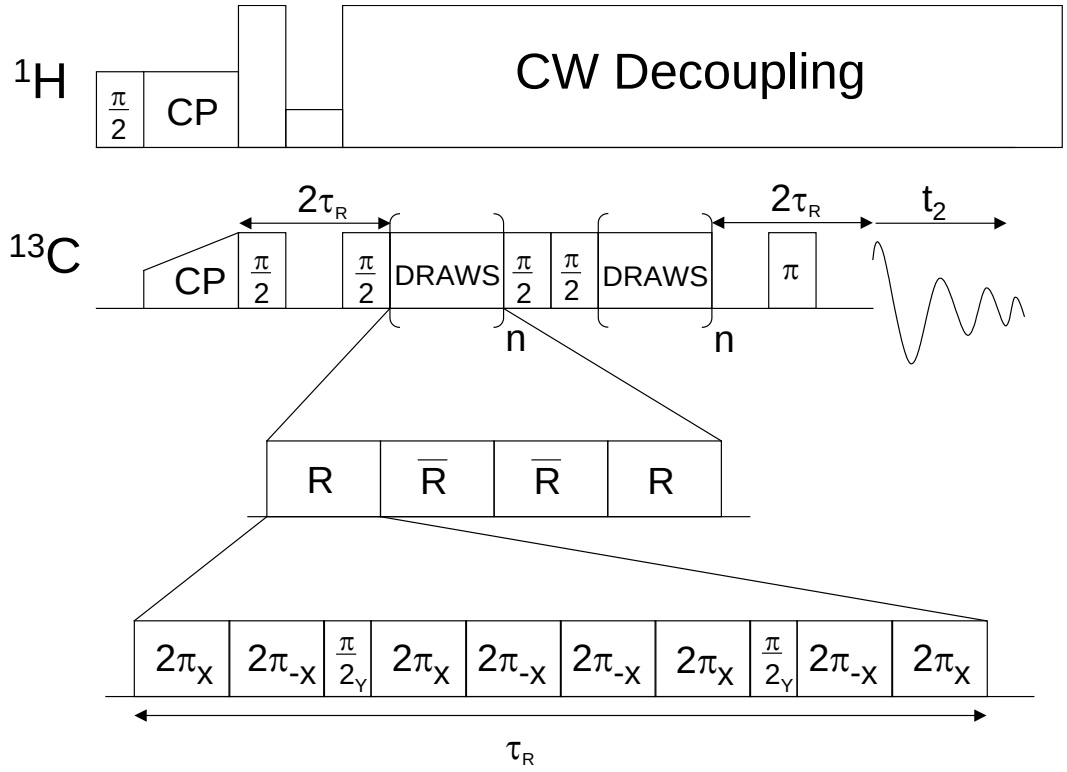

Figure 4-25. The DRAWS pulse sequence. The DRAWS block is repeated n times where n usually is between 1 and 10. R-bar is an R group with the phases shifted by 180°. The Z filter at the beginning ($\pi/2$-delay-$\pi/2$ over $2\tau_R$) and the echo at the end (the π pulse in the middle of $2\tau_R$) are both not critical and were usually not implemented for this research. The first DRAWS group is the excitation part of the sequence which creates the double quantum (DQ) coherence. The second DRAWS group converts the DQ coherence back into single quantum (SQ) for observation.

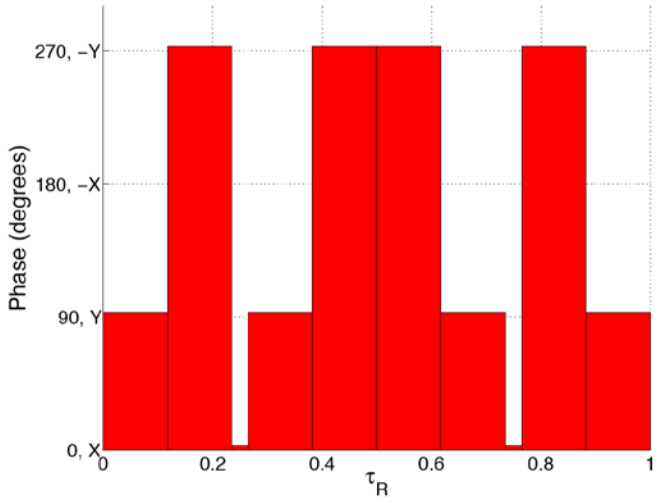

Figure 4-26. The original DRAWS R group pulse lengths and phases for comparison with the optimized DRAWS in Figure 4-27.

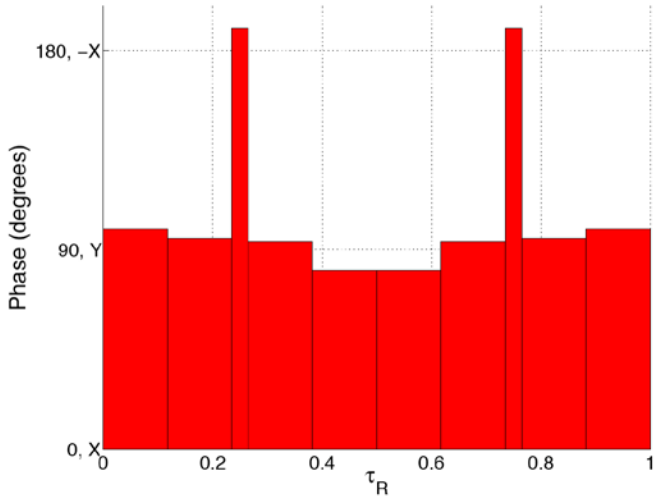

Figure 4-27. The optimized DRAWS R group pulse lengths (the same as original DRAWS) and phases.

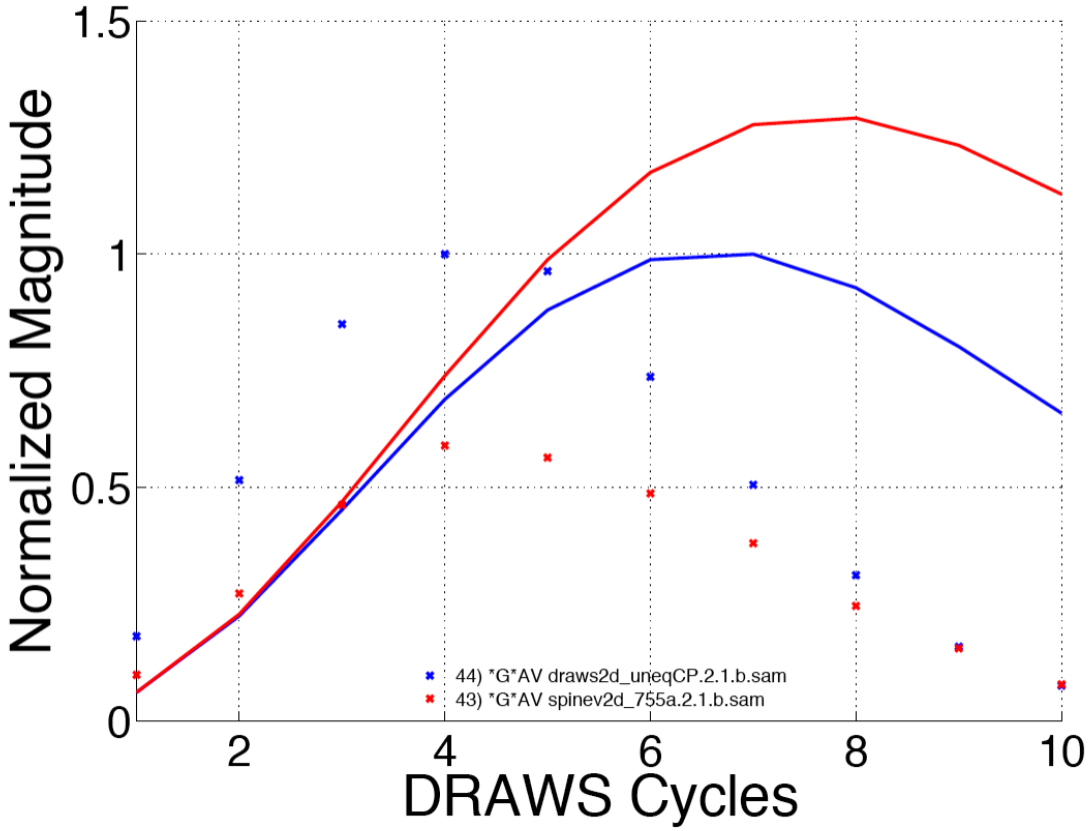

Figure 4-28. DRAWS experimental and simulation data. An example of simulation (lines) and experimental (x's) data for regular DRAWS (blue) and an optimized form (red) is shown where the data is normalized to the magnitude of the regular DRAWS for both simulation and experiment. The regular DRAWS experiment versus simulation do not match. This is most likely due to differences between the simulated spin system and the experimental spin system.

CHAPTER 5
CONCLUSIONS AND FUTURE DIRECTIONS

Conclusions

This research presents hardware and software solutions to many of the problems facing biological ssNMR at high fields. The low-E 750 MHz MAS probe was thoroughly characterized in order to determine the RF efficiency of each channel, maximum achievable B_1 fields, homogeneity of the B_1 fields, isolation between channels, stability at high power, RF-induced sample heating, frictional heating under MAS, spectral linewidths, and signal/noise on standard compounds. Under normal operating conditions, a ^{1}H B_1 RF field of $(\omega_1/2\pi) = 93$ kHz and homogeneity (810°/90°) of 93% can be obtained with a sample length of 8.4 mm corresponding to a volume of 80 μL. With a higher power amplifier, we should be able to exceed 110 kHz decoupling fields based on bench measurements. ^{13}C B_1 RF fields greater than $(\omega_1/2\pi) = 70$ kHz with a homogeneity (810°/90°) of 70% are routinely observed for this sample length; the ^{13}C B_1 homogeneity can be increased to 89% with a 6.7 mm sample length. Under full ^{1}H decoupling for long periods of time, sample heating due to the high RF field is minimal even for samples containing physiological levels of salt. We have not noticed any sample degradation in heat sensitive samples after extensive experimentation. The power handling characteristics, B_1 fields, and homogeneities make this an ideal probe for applying the full range of MAS solid state NMR experiments, including sequences which use extended periods of continuous RF pulsing on both channels, to biological samples which are inherently dilute. This probe is providing high quality data to further scientific research of biological problems. In particular, the mechanism of lung surfactant protein B is currently being studied. Other studies have already been run using this probe, including attempts to analyze different Brazilian soils.

A system for optimizing pulse sequences for ssNMR was also developed, demonstrated, and is running. This system was demonstrated on the two standard pulse sequences used to test pulse optimization systems: the inversion experiment and the refocusing experiment. In both cases, pulse sequences were derived which had a wider bandwidth than existing pulse sequences and had extremely good agreement between experiment and simulation. These pulse sequences should be useful in maintaining high signal strength and phase coherence in future research. The methods of optimization and verification allow them to be easily extended to more complex situations in future research.

The combination of the new probe and the method for optimizing pulse sequences for use at higher fields opens many opportunities for new research on biological solids.

Future Directions

Probe Development

A static low-E probe for a 360 MHz magnet in chemistry is currently being built. It will provide a sturdy workhorse probe for the 360 magnet since all the other probes are older and less efficient than this probe will be. It serves as a platform for us to redesign our circuits for implementation in narrow bore magnets. Two improvements to the probes current design should be considered. First, use zero susceptibility wire to reduce the foot in spectra. Second, make the ^{13}C coil longer to improve its homogeneity in order to make use of the whole sample volume. This should be feasible since the cavity in the stator is long enough to fit a longer solenoid and LGR.

The next logical steps in the development of new low-E MAS probes is the addition of a third RF channel and their redesign for narrow bore magnets. A triple resonance probe would allow simultaneous excitation at ^{15}N, ^{1}H, and ^{13}C frequencies. This is very useful in the study of biological solids. Preliminary design work has been done on this front for a 750 MHz triple

resonance MAS probe. A narrow bore low-E MAS probe is the ultimate goal since most of the NMR magnets are narrow bore due to issues of cost and siting. Narrow bore means the diameter of the bore is 51-54 mm versus the 89 mm bore that the 750 MHz magnet has.

Pulse Sequence Optimization

Both inversion and refocusing composite pulses need to be implemented into more complicated experiments to demonstrate their usefulness. The usual process for pulse sequence development is to develop a solution, test it on a standard sample (adamantane with the pulses at a frequency offset is generally used), and then demonstrate its usefulness on a more complex sample. This work has been started for the pulse sequences developed in this research, but still in progress.

Alternate methods of optimization should also be implemented. The basic framework that was developed for this research is amenable to other optimization methods. One potential avenue is to use genetic algorithms to find optimal pulse sequences. Additionally, Niels Nielson's group in Denmark recently released a new version of SIMPON which has their optimal control method built in [69]. It would be instructive to get a system running here at UF using this new version to see if it compares favorably with the optimization method developed in this research.

Getting optimized DRAWS running would be a big boost to the research here at UF. Dr. Long uses it heavily in the research on lung surfactants. Lung surfactant samples are dilute and not always stable for long periods at high temperature, so being able to run DRAWS experiments more quickly would help to ensure sample integrity and also allow more efficient use of the NMR spectrometers.

In working through the pulse optimization problems, the necessity of comparing experiment with theory on a regular basis became paramount. The best approach is to start by writing a simulation for a simple pulse sequence that needs optimizing. Next, run some

experiments on the spectrometer and observe the difference between the initial simulations and the experiment. Work on the experimental side until any discrepancies are minimal. Switch back to the simulations and try to make the simulations match the experiments. I found that this was usually where the breakthroughs came. Once you have a model that accurately reproduces experimental data, the experiment can be modified to achieve what is actually desired, i.e. homogeneity improved, moving/changing where proton decoupling is on, etc, all the time making sure that an accurate simulation is maintained. Another area where constrained optimization may be better is to limit how long the final pulse sequence can be to some smaller value. The refocusing pulse sequences found here and elsewhere tend to be on the order of 7 times longer than a hard 180 pulse. It would be interesting to try to minimize that time while keeping the same broadband capabilities.

The path for the next person working on optimizing pulse sequences should probably start by optimizing inversion pulse sequences. This provides a straightforward method for testing optimizations systems since the correlation between simulation and experiment is very good in this case. Once the optimization system/method is running well on inversion pulse sequences, move on to optimizing refocusing pulse sequences. The correlation between simulation and experiment diverges a bit more in this case, but is still quite good. Implementing the refocusing pulses in a more complicated pulse program reveals how well the whole system works. When both inversion and refocusing pulse sequence optimizations are running well, then the more complicated problem of interest should be tackled. This method will build up understanding of the whole process quickly so that problems with the more complicated pulse sequence optimization are more easily solved and understood. It might be even better to start by optimizing for liquids since the mathematics behind liquids is much simpler than solids. This

would provide a basic understanding of what is going on mathematically much faster than starting with solids. In either case, it was very instructive for the author to write his own simulator for liquids simulations. Modifying them to solids might also have been instructive, but is hard to justify since several good simulators are already in existence.

In the future it would be very interesting to explore spectrometer in the loop optimizations to see if there are better and faster ways to do this than have been presented in the literature so far. A combination of *in silico* and spectrometer in the loop optimization would probably generate the absolute best pulse sequences. It is possible to have the simulator try to match the current spectra exactly between each acquisition so that the optimization scheme can make faster steps towards optimality rather than using only the data from the spectrometer.

APPENDIX A
DATA PROCESSING TECHNIQUES

The functions I wrote for processing and analyzing data can mostly be found in ~/Seth/bin on etude.mbi.ufl.edu or CITS identifier BBJQQ. They are also duplicated on the Long_lab network drive under Seth/bin.

Visualization Aids

The scripts for visualization using vectors and a sphere are included here. The vectors were all made using Arrow3 version 5 available from the Matlab Central File Exchange (http://www.mathworks.com/matlabcentral/fileexchange/14056).

Nutation Rate Visualization Script

```
clear
cd /Users/mcnese/Seth/Presentations/Dissertation

plot_name = 'Nut_alpha_about_Ix.pdf';
p = 0; % 1 to plot, 0 to not
r = 1;
N = 256;
a = 30*pi/180; % Angle to rotate through
lite_grey = [0.5 0.5 0.5];
grey = [0.5 0.5 0.5];
sphere_color = [1 1 1];

t=(0:N)*2*pi/N;
zs = zeros(size(t));
fig_handle = figure;
orient landscape
set(fig_handle, 'color', 'w')
plot3(r*cos(t), r*sin(t), zs, '-', 'linewidth', 2, 'color', lite_grey);
hold all
axis equal
plot3(r*cos(t), zs, r*sin(t), '-', 'linewidth', 2, 'color', lite_grey);
arrow3([0, 0, -r], [0, 0, r], 'l2') % Z axis
arrow3([0, -r, 0], [0, r, 0], 'l2') % Y axis
arrow3([-r 0 0], [r, 0, 0], 'l2') % X axis
text(-1.1*r, 0, 0, 'B_1', 'fontsize', 20, 'fontweight', 'bold', ...
    'Fontname', 'Times New Roman', 'color', 'k')
text(0.3, 0, 0.8*r, 'I_i_n_i_t', 'fontsize', 20, 'fontweight', 'bold', ...
    'Fontname', 'Times New Roman', 'color', 'b')
text(0, 0, 1.1*r, 'B_0', 'fontsize', 20, 'fontweight', 'bold', 'Fontname', ...
    'Times New Roman', 'color', grey)
text(0, 1.1*r, 0, 'Y', 'fontsize', 20, 'fontweight', 'bold', 'Fontname', ...
    'Times New Roman', 'color', grey)
text(1.1*r, 0, 0, 'X', 'fontsize', 20, 'fontweight', 'bold', 'Fontname', ...
    'Times New Roman', 'color', grey)

% create the sphere
[X, Y, Z] = sphere(N);
c(:,:,1) = ones(size(X));
c(:,:,1) = sphere_color(1);
c(:,:,2) = sphere_color(2);
c(:,:,3) = sphere_color(3);
hs = surf(r*X, r*Y, r*Z, c);
shading flat
```

```
grid off
set(gca, 'XTickLabel', [], 'YTickLabel', [], 'ZTickLabel', [])
set(gca, 'XColor','w', 'YColor','w', 'ZColor','w')
set(gca, 'XTick', [], 'YTick', [], 'ZTick', [])
set(hs, 'EdgeAlpha', 0)
set(hs, 'FaceAlpha', 0.25)

arrow3([0, 0, 0], [0, 0, 1], 'b2') % Iinit
arrow3([0, 0, 0], [-1, 0, 0], 'k2') % Irot
arrow3([0, 0, 0], [0, sin(a), cos(a)], '_g2') % Iend
arrow3([0 0 0], [0 sin(a) 0], 'r2')
arrow3([0 sin(a) 0], [0 sin(a), cos(a)], '--k2', 0, 0)
text(0, 1.1*sin(a), 1.1*cos(a), 'I_e_n_d', 'fontsize', 20, 'fontweight', ...
    'bold', 'Fontname', 'Times New Roman', 'color', [0 0.7 0])
plot3(zs(1:floor(a/(2*pi)*N)), r/2*sin(t(1:floor(a/(2*pi)*N))), ...
    r/2*cos(t(1:floor(a/(2*pi)*N))), '-', 'linewidth', 2, 'color', 'k');
arrow3([0 r/2*sin(t(floor(a/(2*pi)*N))) r/2*cos(t(floor(a/(2*pi)*N)))], ...
    [0 r/2*sin(t(floor(a/(2*pi)*N)+1)) r/2*cos(t(floor(a/(2*pi)*N)+1))], 'k2', 1, 2)
text(0, 0.6*sin(a/2), 0.6*cos(a/2), '\alpha', 'fontsize', 20, 'fontweight', ...
    'bold', 'Fontname', 'Times New Roman', 'color', 'k')

axis vis3d
axis off
view(132, 22)
camzoom(1.5)
h_light = camlight;
lighting gouraud
material dull
set(h_light, 'color', [1 1 1])
set(h_light, 'Position', [-5 5 10])

% set(gcf,'renderer','zbuffer') % turns sphere opaque

if(p)
    print('-dpdf', plot_name)
    disp(['Printed: ' plot_name])
end
```

Pauli Definitions Figure Script

The script for visualizing the definitions for Pauli rotation matrices demonstrates how to do more complex visualization with arrow3 and sphere.

```
clear
cd /Users/mcnese/Seth/SethsPapers/Dissertation/figs

plot_name = 'Fig4-3_Rotation_arbitrary_axis.pdf';
p = 0; % 1 to plot, 0 to not
phi = 70*pi/180; % angle from X
theta = 45*pi/180; % angle from Z
angle_prefs = [0.5 0.12]; % length and offset from A for angle circle
r = 1;
N = 256;
lite_grey = [0.5 0.5 0.5];
grey = [0.5 0.5 0.5];
sphere_color = [1 1 1];

t=(0:N)*2*pi/N;
zs = zeros(size(t));
os = ones(size(t));
% create the figure
fig_handle = figure;
orient landscape
set(fig_handle, 'color', 'w')

% create the axes
plot3(r*cos(t), r*sin(t), zs, '-', 'linewidth', 2, 'color', lite_grey);
```

```
hold all
axis equal
plot3(r*cos(t), zs, r*sin(t), '-', 'linewidth', 2, 'color', lite_grey);
arrow3([-r, 0, 0], [r, 0, 0], 'd2')
arrow3([0, -r, 0], [0, r, 0], 'd2')
arrow3([0, 0, -r], [0, 0, r], 'd2')
text(1.1*r, 0, 0, 'X', 'fontsize', 20, 'fontweight', 'bold', ...
    'Fontname', 'Times New Roman', 'color', grey)
text(0, 1.1*r, 0, 'Y', 'fontsize', 20, 'fontweight', 'bold', ...
    'Fontname', 'Times New Roman', 'color', grey)
text(0, 0, 1.1*r, 'Z', 'fontsize', 20, 'fontweight', 'bold', 'Fontname', ...
    'Times New Roman', 'color', grey)

% create the sphere
[X, Y, Z] = sphere(N);
c(:,:,1) = ones(size(X));
c(:,:,1) = sphere_color(1);
c(:,:,2) = sphere_color(2);
c(:,:,3) = sphere_color(3);
hs = surf(r*X, r*Y, r*Z, c);
shading flat
grid off
set(gca, 'XTickLabel', [], 'YTickLabel', [], 'ZTickLabel', [])
set(gca, 'XColor','w', 'YColor','w', 'ZColor','w')
set(gca, 'XTick', [], 'YTick', [], 'ZTick', [])
set(hs, 'EdgeAlpha', 0)
set(hs, 'FaceAlpha', 0.25)

% plot the real information
% Vectors
arrow3([0, 0, 0], [sin(theta)*cos(phi), sin(theta)*sin(phi), cos(theta)], ...
    'r2') % Rotation axis
arrow3([0, 0, 0], [sin(theta)*cos(phi), sin(theta)*sin(phi), 0], 'b2') % B1
arrow3([0, 0, 0], [0, 0, cos(theta)], '_g2') % dB0
% lines
plot3([sin(theta)*cos(phi) sin(theta)*cos(phi)], ...
    [sin(theta)*sin(phi) sin(theta)*sin(phi)], [0 cos(theta)], '--', ...
    'linewidth', 2, 'color', [0 0 1])
plot3([sin(theta)*cos(phi) 0], [sin(theta)*sin(phi) 0], ...
    [cos(theta) cos(theta)], '--m', 'linewidth', 2, 'color', [0 0.7 0])
% arcs with arrowheads
tphi = t(1:floor(phi/(2*pi)*N));
Nphi = length(tphi);
plot3(sin(theta)*cos(tphi), sin(theta)*sin(tphi), zs(1:floor(phi/(2*pi)*N)), ...
    '-', 'linewidth', 2, 'color', [0.7 0 0.7]);
arrow3([sin(theta)*cos(t(floor(phi/(2*pi)*N))) ...
    sin(theta)*sin(t(floor(phi/(2*pi)*N))) 0], ...
    [sin(theta)*cos(t(floor(phi/(2*pi)*N)+1)) ...
    sin(theta)*sin(t(floor(phi/(2*pi)*N)+1)) 0], '_m2', 1, 2)
text(1.1*sin(theta)*cos(phi/2), 1.1*sin(theta)*sin(phi/2), 0, '$\bf\phi$',...
    'interpreter', 'latex', 'fontsize', 24, 'fontweight', 'bold', ...
    'color', [0.7 0 0.7])

ttheta = linspace(0,theta,Nphi);
plot3(r/2*sin(ttheta)*cos(phi), r/2*sin(ttheta)*sin(phi), ...
    r/2*cos(ttheta), '-', 'linewidth', 2, 'color', [0 0.7 0.7]);
arrow3([r/2*sin(ttheta(3))*cos(phi) r/2*sin(ttheta(3))*sin(phi) ...
    r/2*cos(ttheta(3))], [0 0 r/2], '_c2', 1, 2)
text(1.1*r/2*sin(theta/2)*cos(phi), 1.1*r/2*sin(theta/2)*sin(phi), ...
    1.1*r/2*cos(theta/2), '\theta', 'fontsize', 20, 'fontweight', ...
    'bold', 'color', [0 0.7 0.7])

text(1.1*sin(theta)*cos(phi), 1.1*sin(theta)*sin(phi), 1.1*cos(theta), ...
    'B_e_f_f', 'fontsize', 20, 'fontweight', 'bold', 'color', 'r')
text(1.1*sin(theta)*cos(phi), 1.1*sin(theta)*sin(phi), 0, 'B_1', ...
    'fontsize', 20, 'fontweight', 'bold', 'color', 'b')
text(0.14, -0.14, 0.4*cos(theta), '\DeltaB_0', 'fontsize', 20, 'fontweight', ...
    'bold', 'color', [0 0.7 0])

% angle circle
Otheta = theta+atan(angle_prefs(2)/angle_prefs(1)); % offset angle from Z-axis
```

```
O = [angle_prefs(1)*sin(Otheta)*cos(phi) angle_prefs(1)*sin(Otheta)*sin(phi) ...
    angle_prefs(1)*cos(Otheta)];
arc_pts = rot3d(O, [sin(theta)*cos(phi), sin(theta)*sin(phi), cos(theta)], ...
    -45:1:280);
plot3(arc_pts(:,1), arc_pts(:,2), arc_pts(:,3), '-k', 'linewidth', 2)
arrow3([arc_pts((end-5), 1), arc_pts((end-5), 2), arc_pts((end-5), 3)], ...
    [arc_pts(end, 1) arc_pts(end, 2) arc_pts(end, 3)], 'k', 0.5, 1.1)
text(0, 0.25, 0.12, '\alpha_e_f_f', 'fontsize', 20, 'color', 'k', ...
    'fontweight', 'bold')

% setup the view and lighting
axis vis3d
axis off
view(132, 22)
camzoom(1.5)
h_light = camlight;
lighting gouraud
material dull
set(h_light, 'color', [1 1 1])
set(h_light, 'Position', [-5 5 10])

% set(gcf,'renderer','zbuffer') % turns sphere opaque

if(p)
    print('-dpdf', plot_name)
    disp(['Printed: ' plot_name])
end
```

Pulse Path Calculation and Visualization

I wrote some scripts for pulse path visualizations. The scripts assume a liquid environment.

Pulse path calculation

```
function [xyz, line_h] = pulse_path2(cur, plength, b1h, ax_raw, foffs, npts, p, usec)
% function [xyz, line_h] = pulse_path2(cur, plength, b1h, ax_raw, foffs,
% npts, p, usec)
%
% cur     = the current vector position
% plength = the ideal pulse length in degrees
% b1h     = the b1 nutation rate in Hz
% ax      = the ideal axis of rotation give in [x y z] components
% foffs   = the frequency offset in Hz
% npts    = the number of points to calculate for
% if p exists and is non-zero, the trajectory will be plotted in 3D
% usec    = plength is in microseconds rather than degrees
% xyz     = a list of npts [x y z] components of the pulse trajectory
% line_h  = the plotted line's handle

% 2008 September 8 SAM

% define some constants
i = sqrt(-1);
Ix = 0.5*[0 1; 1 0];
Iy = 0.5*[0 -i; i 0];
Iz = 0.5*[1 0; 0 -1];
if(exist('usec', 'var'))
    plength = plength/(1/b1h*1e6)*360; % convert from useconds to degrees
end
ax = ax_raw/sqrt(sum(ax_raw.^2)); % normalize the rotation axis

Iinit = cur(1)*Ix + cur(2)*Iy + cur(3)*Iz;

% calculate effective pulse length
b1effM = sqrt(b1h^2 + foffs^2);
peff = b1effM/b1h*plength; % effective pulse length in degrees
% disp(['peff = ' num2str(peff, '%.1f') ' usec'])
```

```
% Rp is a rotation of a about an arbitrary axis p degrees from the X-axis
% in the XY plane and th degrees from the Z-axis
% p_hyp is the hypotonuse of triangle for X and Y of rotation axis (phi)
p_hyp = sqrt(ax(1)^2 + ax(2)^2);
th_hyp = sqrt(p_hyp^2 + (ax(3)+(foffs/b1h))^2); % same for theta from Cavanagh

a = linspace(0, peff, npts)*(pi/180);
xyz = zeros(npts, 3);
for ii = 1:npts
    % Rp from Cavanagh's Protein NMR Spectroscopy pp. 48
    % Rp = eye(2)*cos(a/2) - 2*i*sin(a/2)*(Ix*cos(p)*sin(th) + Iy*sin(p)*sin(th)...
    %     + Iz*cos(th));
    Rp = eye(2)*cos(a(ii)/2) - 2*i*sin(a(ii)/2)*(Ix*(ax(1)/p_hyp)*(p_hyp/th_hyp)...
        + Iy*(ax(2)/p_hyp)*(p_hyp/th_hyp) + Iz*((ax(3)+(foffs/b1h))/th_hyp));
    xyz(ii,:) = real(2*[trace(Rp*Iinit*inv(Rp)*Ix) trace(Rp*Iinit*inv(Rp)*Iy)...
        trace(Rp*Iinit*inv(Rp)*Iz)]);
end

if(p)
    figure(p)
    line_h = plot3(xyz(:,1), xyz(:,2), xyz(:,3), '-', 'linewidth', 2);
else
    line_h = [];
end
```

Pulse path visualization

The script that uses pulse_path2.m to visualize the paths was used in several of the figures in this document.

```
clear
cd /Users/mcnese/Seth/SethsPapers/Dissertation/figs

p = 0; % 1 to print figures
pwr = 50; % kHz
base360 = 20; %usec
plengths = base360; % usec
percent_offset = 30;
% f = [0 percent_offset/100 -percent_offset/100]*pwr*1000; % Hz
f = [0 -0.1 -0.2 -0.3 -0.4 -0.5 -0.6]*pwr*1000; % Hz
plots_fname = 'Fig4-6_9018090FreqOffset.pdf';

cur_dir = pwd;
if(p)
    delete(fullfile(cur_dir, plots_fname));
end

h = figure('name','90-180-90 frequency offsets');
orient landscape
plot_3D_axes2(h);

L = length(f);
for ii = plengths
    e = zeros(L, 3);
    for jj = 1:length(f)
        [path_data] = pulse_path2([0 0 1], ii/4, pwr*1000, ...
        [-1 0 0], f(jj), 50, 0, 1);
        hline(jj,1) = plot3(path_data(:,1), path_data(:,2), path_data(:,3),...
            '-', 'linewidth', 2);
        [path_data] = pulse_path2(path_data(end,:), ii/2, pwr*1000, ...
        [0 -1 0], f(jj), 50, 0, 1);
        plot3(path_data(:,1), path_data(:,2), path_data(:,3), '-', ...
            'linewidth', 2, 'color', get(hline(end), 'color'))
        [path_data] = pulse_path2(path_data(end,:), ii/4, pwr*1000, ...
        [-1 0 0], f(jj), 50, 0, 1);
        plot3(path_data(:,1), path_data(:,2), path_data(:,3), '-', ...
```

```
            'linewidth', 2, 'color', get(hline(end), 'color'))
        plot3(path_data(end,1), path_data(end,2), ...
            path_data(end,3), '*', 'color', get(hline(end), 'color'),...
            'linewidth', 2, 'markersize', 10)
    end
end

% annotation('textbox', [0 0.73 1 0.25], 'string', ...
%     {['Inversion 90(-x)-180(-y)-90(-x)', ' [' sprintf('%.1f, ', ...
%     f/pwr/1000) '] \DeltaF/F Offsets']}, 'edgecolor', 'none', 'fontsize', ...
%     20, 'horizontalalignment', 'center');

legend(hline, num2str(f'/pwr/1000))
set(gca, 'fontsize', 20)
legend boxoff

if(p)
    print('-dpdf', plots_fname)
    disp(['Printed: ' plot_name])
end
```

Trajectory Endpoint Visualization

The endpoint visualization uses pulse_path2.m, but in a slightly different way to show where the endpoints of a pulse sequence are over a range of frequency offsets.

```
clear
cd /Users/mcnese/Seth/SethsPapers/Dissertation/figs


p = 0; % 1 to print figures
% pwrs = [10 25 50 75 100 500];
pwr = 10;
base360 = 1/pwr*1e3; %usec
plengths = base360; % usec
f = -100000:50:100000;
plots_fname = 'Inversion9018090FreqOffsetEndpts.ps';

cur_dir = pwd;
if(p)
    delete(fullfile(cur_dir, plots_fname));
end

L = length(f);
for ii = plengths
    e = zeros(L, 3);
    for jj = 1:length(f)
        [path_data] = pulse_path2([0 0 1], ii/4, pwr*1000, ...
        [-1 0 0], f(jj), 1, 0, 1);
        [path_data] = pulse_path2(path_data(end,:), ii/2, pwr*1000, ...
        [0 -1 0], f(jj), 1, 0, 1);
        e(jj,:) = pulse_path2(path_data(end,:), ii/4, pwr*1000, ...
        [-1 0 0], f(jj), 1, 0, 1);
    end

    h = figure('name','Inversion pulses ab initio');
    orient landscape
    subplot(3,1,[1 2])
    plot_3D_axes2(h);
    h3D = scatter3(e(:,1), e(:,2), e(:,3), 10, linspace(0, 1, length(f)), 'filled');
    % Create title for 3D axes
    annotation('textbox', [0 0.73 1 0.25], 'string', ...
        {['Inversion 90(-x)-180(-y)-90(-x) Terminal Points']}, ...
        'edgecolor', 'none', 'fontsize', 20, 'horizontalalignment', 'center');

    subplot(3,1,3)
    xproj = e*[0; 0; 1]; % dot product with x-axis
```

```
    hold all; grid on
    scatter(f/1000/pwr, xproj, 6, f/1000, 'filled')
    a = axis;
    axis([a(1) a(2) -1 1])
    set(gca, 'fontsize', 16)
    xlabel('\omega_o_f_f_s_e_t/\omega_n_u_t_a_t_i_o_n', 'fontsize', 20)
    ylabel('I_Z Magnitude', 'fontsize', 20)

    if(p)
        print('-dpsc', '-append', plots_fname)
    end
end
```

Data Preparation

Most of the processing functions and scripts used for this document expect that the data be in .mat files. The processed data should be in a file named expNN.mat where NN is the experiment number. For instance, the processed data for experiment 19 in the dataset MAS1208 is saved as exp19.mat in the folder MAS1208 under ~/Seth/750Data. The integrated data is saved as a separate file named expNNint.mat. The processing script named ndnmr_sam will generate these files after processing all data by pushing the “Save Processed FIDs” and “Save Integration Data” buttons. The data processing program that Neil Wargo has been working on saves the data in a different format as processed_data.mat and integration_data.mat in the actual data folder. For example the data for experiment 99 in the dataset MAS1208 is saved as ~/Seth/750Data/MAS1208/99/processed_data.mat. I wrote a conversion function called long2ndnmr.

Long2ndnmr.m

```
function long2ndnmr(exps)
% function long2ndnmr(exps)
%
% This function goes through the directories specified by the numbers in
% exps loading processed_data.mat and integration_data.mat if they exist
% and saving them as expXX.mat and expXXint.mat in the directory where the
% function is called from.
% 2009 January 01

% Seth McNeill

data_fname = 'processed_data.mat';
int_fname = 'integration_data.mat';
start_dir = pwd;
```

```
for ii = exps
    ii_txt = num2str(ii);
    cd(ii_txt);
    if(exist(data_fname, 'file'))
        load(data_fname)
        expname = ['exp' ii_txt];
        eval([expname ' = all_params.data;']);
        eval(['save ' start_dir filesep expname '.mat ' expname]);
        disp(['Saved: ' start_dir filesep expname '.mat'])
        if(exist(int_fname, 'file'))
            load(int_fname)
            eval([expname 'int = nmr_integration_data;']);
            eval(['save ' start_dir filesep expname 'int.mat ' expname 'int'])
            disp(['Saved: ' start_dir filesep expname 'int.mat'])
        else
            disp([ii_txt ': no ' int_fname])
        end
    else
        disp(['Directory: ' num2str(ii) ' skipped (no ' data_fname ')'])
    end
    cd(start_dir)
end
```

Bruker_data.m

There is also a function for collecting information from the acqu and acqus files that a Bruker spectrometer creates when an experiment is run.

```
function data = bruker_data(exp_num, directory)
%% function data = bruker_data(exp_num, directory)
%
% This function extracts data for zg_2d and zgig_2d experiments on Bruker
% spectrometers. If directory is not supplied, it assumes that it is called
% from the base directory of the experiments (ie the directory that has all
% the numbered directories in it).
%
% It looks at the acqus file and the parameters P, TD, PL, ...
%
% Seth McNeill
% 2007 July 24

%% Basic initializations
acqu1_fname = 'acqus';
acqu2_fname = 'acqu2s';
acqu2a_fname = 'acqu2';
title_fname = fullfile('pdata', '1', 'title');
twod = 0; % is a 2D experiment

start_dir = pwd;
exp_num_txt = num2str(exp_num, '%.0f');
if(~exist('directory', 'var'))
    directory = start_dir;
end
cd(fullfile(directory, exp_num_txt))



%% read in the acqu1 file
fid = fopen(acqu1_fname, 'r');
if(fid == -1)
    error(['Cannot open file: ' acqu1_fname ' for reading.'])
end
ii = 1;
acqu1_txt{ii} = fgetl(fid);
while(~isfloat(acqu1_txt{ii}))
    ii = ii + 1;
```

```
    acqu1_txt{ii} = fgetl(fid);
end
fclose(fid);

% Remove the non-string cell at then end of acqu1_txt
acqu1_txt(end) = [];


%% read in the acqu2 file
if(exist(acqu2_fname, 'file'))
    twod = 1;
    fid = fopen(acqu2_fname, 'r');
    if(fid == -1)
        error(['Cannot open file: ' acqu2_fname ' for reading.'])
    end
    jj = 1;
    acqu2_txt{jj} = fgetl(fid);
    while(~isfloat(acqu2_txt{jj}))
        jj = jj + 1;
        acqu2_txt{jj} = fgetl(fid);
    end
    fclose(fid);

    % Remove the non-string cell at then end of acqu1_txt
    acqu2_txt(end) = [];
end

%% read in the acqu2 file
if(exist(acqu2a_fname, 'file'))
    fid = fopen(acqu2a_fname, 'r');
    if(fid == -1)
        error(['Cannot open file: ' acqu2a_fname ' for reading.'])
    end
    jj = 1;
    acqu2a_txt{jj} = fgetl(fid);
    while(~isfloat(acqu2a_txt{jj}))
        jj = jj + 1;
        acqu2a_txt{jj} = fgetl(fid);
    end
    fclose(fid);

    % Remove the non-string cell at then end of acqu1_txt
    acqu2a_txt(end) = [];
end

%% Start collecting data
% TD
tokens = regexp(acqu1_txt, '\#\#\$TD=\s*(\d*)', 'tokens');
empty_check = cellfun('isempty', tokens);
[kk, kk] = find(empty_check == 0);
data.TD = str2num(tokens{kk}{1}{1});

% BF1
tokens = regexp(acqu1_txt, '\#\#\$BF1=\s*([\d\.]*)', 'tokens');
empty_check = cellfun('isempty', tokens);
[kk, kk] = find(empty_check == 0);
data.BF1 = str2num(tokens{kk}{1}{1});

% NS
tokens = regexp(acqu1_txt, '\#\#\$NS=\s*([\d\.]*)', 'tokens');
empty_check = cellfun('isempty', tokens);
[kk, kk] = find(empty_check == 0);
data.NS = str2num(tokens{kk}{1}{1});

% SFO1
tokens = regexp(acqu1_txt, '\#\#\$SFO1=\s*([\d\.]*)', 'tokens');
empty_check = cellfun('isempty', tokens);
[kk, kk] = find(empty_check == 0);
data.SFO1 = str2num(tokens{kk}{1}{1});

% SW_h
```

```
tokens = regexp(acqu1_txt, '\#\#\$SW_h=\s*([\d\.]*)', 'tokens');
empty_check = cellfun('isempty', tokens);
[kk, kk] = find(empty_check == 0);
data.SW_h = str2num(tokens{kk}{1}{1});

% TE - temperature
tokens = regexp(acqu1_txt, '\#\#\$TE=\s*([\d\.]*)', 'tokens');
empty_check = cellfun('isempty', tokens);
[kk, kk] = find(empty_check == 0);
data.TE = str2num(tokens{kk}{1}{1});

% P
tokens = regexp(acqu1_txt, '\#\#\$(P)=', 'tokens');
empty_check = cellfun('isempty', tokens);
[kk, kk] = find(empty_check == 0);
data.P = [str2num(acqu1_txt{kk+1}) str2num(acqu1_txt{kk + 2})];

% PL
tokens = regexp(acqu1_txt, '\#\#\$(PL)=', 'tokens');
empty_check = cellfun('isempty', tokens);
[kk, kk] = find(empty_check == 0);
data.PL = str2num(acqu1_txt{kk+1});

% SP
tokens = regexp(acqu1_txt, '\#\#\$(SP)=', 'tokens');
empty_check = cellfun('isempty', tokens);
[kk, kk] = find(empty_check == 0);
data.SP = str2num(acqu1_txt{kk+1});

% NUC - Nucleus 1
tokens = regexp(acqu1_txt, '\#\#\$NUC1=\s*<(\w*)', 'tokens');
empty_check = cellfun('isempty', tokens);
[kk, kk] = find(empty_check == 0);
nuc_base = tokens{kk}{1}{1};
data.nuc = regexprep(nuc_base, '(\d)', '^$1');

% PULPROG - Nucleus 1
tokens = regexp(acqu1_txt, '\#\#\$PULPROG=\s*<([^>]*)', 'tokens');
empty_check = cellfun('isempty', tokens);
[kk, kk] = find(empty_check == 0);
pulprog_base = tokens{kk}{1}{1};
data.pulprog = regexprep(pulprog_base, '(\_)', '\\$1');

% D
tokens = regexp(acqu1_txt, '\#\#\$(D)=', 'tokens');
empty_check = cellfun('isempty', tokens);
[kk, kk] = find(empty_check == 0);
data.D = [str2num(acqu1_txt{kk+1}) str2num(acqu1_txt{kk + 2})];



if(twod)
    % TD2 from acqu2s - the actual number of spectra taken
    tokens = regexp(acqu2_txt, '\#\#\$TD=\s*(\d*)', 'tokens');
    empty_check = cellfun('isempty', tokens);
    [kk, kk] = find(empty_check == 0);
    data.TD2 = str2num(tokens{kk}{1}{1});

    % The number of points requested in the second dimension
    tokens = regexp(acqu2a_txt, '\#\#\$TD=\s*(\d*)', 'tokens');
    empty_check = cellfun('isempty', tokens);
    [kk, kk] = find(empty_check == 0);
    data.TD2a = str2num(tokens{kk}{1}{1});

    data.expt = data.P(2):data.P(12):(data.P(12)*data.TD2a + data.P(2) - data.P(12));
else
    data.expt = 1/data.SW_h:1/data.SW_h:(data.TD/data.SW_h/2);
end


% read the title and remove empty lines
```

```
fid = fopen(title_fname, 'r');
if(fid == -1)
    warning(['Cannot open file: ' title_fname ' for reading.'])
    data.title = [];
else
    full_title = textscan(fid, '%s', 'Delimiter', '\n');
    empty_check = cellfun('isempty', full_title{1});
    [kk] = find(empty_check == 1);
    full_title{1}(kk) = [];
    data.title = full_title{1};
end


cd(start_dir);
```

Simulating Experiments with SPINEVOLUTION

Simulating an experiment using spinev requires several files. The first two describe the simulation parameters for Matlab which get passed on to the simulator. A .par file contains the general parameters for the simulation and a .ss file contains a description of the spin system to be simulated. The data from both of these files is compiled into a settings structure in Matlab.

inv.par

inv.par contains the settings for doing an inversion simulation with spinev.

```
# inv.par
#
# Parameter file for inversion optimizations
#
# Seth McNeill
# 2008 April 22
# 2008 June 18 uncommented so write_spinev_inv.m would work

spin_rate 10000
crystal_file rep700.dat
gamma_angles 50
start_operator   I1z
detect_operator  I1z
proton_frequency 750e6

ss_fname adam.ss # spin system file
in_fname inv_adam.spv # the name of the input file to call, should be same as par(name)
pulse_fname inv_opt.pp
write_spv_fcn write_spinev_inv
npulses [1 2 3 4 5 6 7] # number of pulses in group optimized
freq_rng [-100 10 100]
power    50 # kHz, power for 1-pulse at perfect homogeneity
pwr_rng [50 1 50]

min_convergence 0.0001 # fminsearch minimum convergence
simulator_fname spinev
sim_opts -split2 -renice10
fcn spinev_eval_inv

nopts 15 # number of optimizations
max_cpu_time 600  # hours

data_dir data
```

adam.ss

adam.ss contains a description of a spin system based on adamantane.

```
# adam.ss
#
# This is a spin system file for admantane for nutation experiments
#
# The format is the same as in a spinsys section
# of a SIMPSON input file.
#
# Seth McNeill
# 2007 February 25

channels 13C
nuclei 13C
shift 1 0p -4p 0.1 0 0 0
```

load_par.m

The function that loads information from a .par file is load_par.m

```
function data = load_par(par_filename, data)
% function data = load_par(par_filename, data)
%
% This function load in the parameters that simpson will use in it's par
% structure from a file. The file uses the # symbol for comments. Otherwise
% items in the first column are added as fields in data with the value of
% what is in the second column. If data is passed in, the fields are added
% to it before it is returned.
%
% Seth McNeill
% 2006 March 29

filename = par_filename;

if(~exist('data','var')) % must create data if wasn't passed in
    data = [];
end

try
    file_lines = textread(filename, '%s', 'delimiter', '\n', 'whitespace', '');
catch
    err = lasterr;
    if(~isempty(strfind(err, 'File not found')))
        error(['File not found: ' filename])
    else
        rethrow(lasterr)
    end
end
commentless_lines = regexprep(file_lines, '\s*(#.*)', ''); % clear out comments
clean_lines = regexprep(commentless_lines, '(^\s*)|(\s*$)', ''); % clear white space
jj = 1;
for ii = 1:length(clean_lines)
    if(~isempty(clean_lines{ii}))
        full_lines{jj} = clean_lines{ii}; % collect only nonempty lines left after cleaning
        jj = jj + 1;
    end
end
nlines = length(full_lines);

pars_cell = regexp(full_lines, '(\w+)\s+(.+)', 'tokens'); % separate out fields from values
if(nlines) % check to make sure that there are some pars to read in
    for ii = 1:nlines
        value = str2num(pars_cell{ii}{1}{2}); % This is empty for nonnumeric values
        if(isempty(value))
            eval(['data.' pars_cell{ii}{1}{1} ' = ''' pars_cell{ii}{1}{2} ''';']); % strings
```

```
        else
            eval(['data.' pars_cell{ii}{1}{1} ' = ' pars_cell{ii}{1}{2} ';']); % numbers
        end
    end
else
    error('No data in pars file!')
end
```

load_ss.m

The function that loads a .ss file is load_ss.m

```
function data = load_ss(spinsys_filename, data)
% function data = load_ss(spinsys_filename, data)
%
% This function load in the spins system that simpson will use in it's spinsys
% structure from a file. The file uses the # symbol for comments. Otherwise
% items in the first column are added as fields in data with the value of
% what is in the rest of the line. If data is passed in, the fields are added
% to it before it is returned.
%
% It was written with the expectation of two nuclei including dipole
% information. If this isn't the case some modifications to the code will
% be required.
%
% Seth McNeill
% 2006 March 30

filename = [spinsys_filename];

if(~exist('data','var')) % must create data if wasn't passed in
    data = [];
end

file_lines = textread(filename, '%s', 'delimiter', '\n', 'whitespace', '');
commentless_lines = regexprep(file_lines, '\s*(#.*)', ''); % clear out comments
clean_lines = regexprep(commentless_lines, '(^\s*)|(\s*$)', ''); % clear white space
jj = 1;
for ii = 1:length(clean_lines)
    if(~isempty(clean_lines{ii}))
        full_lines{jj} = clean_lines{ii}; % collect only nonempty lines left after cleaning
        jj = jj + 1;
    end
end
nlines = length(full_lines);

spinsys_cell = regexp(full_lines, '(\w+)\s+(.+)', 'tokens'); % separate out fields from values
nshifts = 0;
if(nlines) % check to make sure that there are some fields to read in
    for ii = 1:nlines
        if(strcmp('shift', spinsys_cell{ii}{1}{1}))
            nshifts = nshifts + 1;
            eval(['data.' spinsys_cell{ii}{1}{1} '{' num2str(nshifts) ...
                '} = ''' spinsys_cell{ii}{1}{2} ''';']); % shifts
        else
%         value = str2num(spinsys_cell{ii}{1}{2}); % This is empty for nonnumeric values
%         if(isempty(value))
            % strings
            eval(['data.' spinsys_cell{ii}{1}{1} ' = ''' spinsys_cell{ii}{1}{2} ''';']);
%         else
%             % numbers
%             eval(['data.' spinsys_cell{ii}{1}{1} ' = [' spinsys_cell{ii}{1}{2} '];']);
%         end
        end
    end
else
    error('No data in spinsys file!')
end
```

Matlab settings structure

A settings structure in Matlab contains the information loaded from the .par and .ss file.

```
settings =
           spin_rate: 10000
         crystal_file: 'rep700.dat'
         gamma_angles: 50
       start_operator: 'I1z'
      detect_operator: 'I1z'
     proton_frequency: 750000000
            ss_fname: '../adam.ss'
            in_fname: 'inv_adam.spv'
         pulse_fname: 'inv_opt.pp'
        write_spv_fcn: 'write_spinev_inv'
              npulses: [4 5 6]
             freq_rng: [-50 1 50]
                power: 30
              pwr_rng: [30 1 30]
      min_convergence: 1.0000e-04
      simulator_fname: 'spinev'
             sim_opts: '-renice10'
                  fcn: 'spinev_eval_inv_dly'
               nopts: 200
         max_cpu_time: 166
             data_dir: 'data'
             channels: '13C'
               nuclei: '13C'
                shift: {'1 0p -4p 0.1 0 0 0'}
           start_time: [2008 12 7 17 46 1.2814]
       start_datestr: 'spinev_opt_inv2 started: 07-Dec-2008 17:46:01'
            len_phase: [1x14 double]
             plengths: [1x7 double]
               pphase: [1x7 double]
             hom_fval: -0.9511
           total_time: 9.3884e+04
                fname: [1x77 char]
```

write_spinev_inv.m

The information in the settings structure is written as an input file to spinev using a write_spinev_XX function. For inversions, use write_spinev_inv.m

```
function write_spinev_inv(settings)
% function write_spinev_inv(settings)
%
% This function creates a spinev file from data in the settings structure
% and saves it as settings.in_fname to do an inversion experiment.
%
% Seth McNeill
% 2007 June 18



fid = fopen(settings.in_fname, 'w');
if(fid == -1)
    error(['Cannot open ' settings.in_fname ' for writing']);
end

fprintf(fid, '****** The System *******\n');
fprintf(fid, 'spectrometer(MHz)  %.0f\n', settings.proton_frequency/1e6);
fprintf(fid, 'spinning_freq(kHz) %f\n', settings.spin_rate/1000);

fprintf(fid, 'channels           %s\n', regexprep(settings.channels, ...
```

```
    '([0-9]*)([^ ]*)', '$2$1'));
fprintf(fid, 'nuclei             %s\n', regexprep(settings.nuclei, ...
    '([0-9]*)([^ ]*)', '$2$1'));

% dipole_cell = regexpi(settings.dipole, '^[0-9]*\s[0-9]*\s([-0-9]*).*', 'tokens');
% dipole_str = cell2mat(dipole_cell{1});
% fprintf(fid, 'atomic_coords      %f\n', dipole_dist(settings.nuclei, ...
%     abs(str2num(dipole_str))));
fprintf(fid, 'atomic_coords      *\n');

if(isfield(settings, 'shift'))
    %% Assume everything is in ppms
    csa_cells = regexpi(settings.shift, '^[0-9]*\s([\w\.\-]*)\s(.*)', 'tokens');
    csa_iso = [];
    for ii = 1:length(settings.shift)
        csa_cells_clean{ii} = regexprep(csa_cells{ii}{1}, '[a-zA-Z]', '');
        csa_iso = [csa_iso ' ' csa_cells_clean{ii}{1}];
    end
    fprintf(fid, 'cs_isotropic       %s ppm\n', csa_iso(2:end));
    for jj = 1:length(settings.shift)
        fprintf(fid, 'csa_parameters     %.0f %s ppm\n', jj, csa_cells_clean{jj}{2});
    end
else
    fprintf(fid, 'cs_isotropic         *\n');
    fprintf(fid, 'csa_parameters       *\n');
end
fprintf(fid, 'j_coupling         *\n');
fprintf(fid, 'quadrupole         *\n');
fprintf(fid, 'dip_switchboard    *\n');
fprintf(fid, 'csa_switchboard    *\n');
fprintf(fid, 'exchange_nuclei    *\n');
fprintf(fid, 'bond_len_nuclei    *\n');
fprintf(fid, 'bond_ang_nuclei    *\n');
fprintf(fid, 'tors_ang_nuclei    *\n');
fprintf(fid, 'groups_nuclei      *\n');
fprintf(fid, '******* Pulse Sequence ****************************\n');
fprintf(fid, 'CHN 1\n');
fprintf(fid, 'timing(usec)         $P\n');
fprintf(fid, 'power(kHz)           *\n');
fprintf(fid, 'phase(deg)           *\n');
fprintf(fid, 'freq_offs(kHz)       *\n');
fprintf(fid, '******* Variables *********************************\n');
fprintf(fid, '* scan across frequency offsets\n');
fprintf(fid, 'scan_par     foffs/$F/\n');
fprintf(fid, '                   freq_1_1=foffs\n');
fprintf(fid, '* scan across power missets\n');
fprintf(fid, 'scan_par     noffs/$N/\n');
fprintf(fid, '          power_1_1=noffs\n');
fprintf(fid, '******* Options **********************************\n');
fprintf(fid, 'rho0               %s\n', settings.start_operator);
fprintf(fid, 'observables        %s\n', settings.detect_operator);
fprintf(fid, 'EulerAngles        %s\n', settings.crystal_file);
if(isnumeric(settings.gamma_angles))
    fprintf(fid, 'n_gamma            %.0f\n', settings.gamma_angles);
else
    fprintf(fid, 'n_gamma            %s\n', settings.gamma_angles);
end
fprintf(fid, 'line_broaden(Hz)   *\n');
fprintf(fid, 'zerofill           *\n');
fprintf(fid, 'FFT_dimensions     *\n');

fprintf(fid, '*** %s ***\n', settings.in_fname);
fprintf(fid, '*** Automatically created by %s ***\n', mfilename('fullpath'));
fprintf(fid, '*** Created at %s ***', datestr(clock));
fclose(fid);
```

spinev_eval_inv_dly.m

To run a spinev simulation use a spinev_eval_XX function. For inversions, use spinev_eval_inv_dly.m.

```
function [sm, d] = spinev_eval_inv_dly(len_phase, settings)
% function [sm, d] = spinev_eval_inv_dly(len_phase, settings)
%
% Evaluates an inversion composite pulse. len_phase is a vector where the
% odd elements are pulse lengths and the even elements are pulse phases.
% The output is the sum of the inversion over the specified frequency
% offsets and power offsets from settings.foffs and settings.pwroffs. These
% should be 3 element vectors [min step max].
%
% This version adds prepulse delays and postpulse delays.

% Seth McNeill
% 2008 April 21

Llp = length(len_phase);
if(mod(Llp, 2))
    error('len_phase has to be an even length vector')
end

if(size(len_phase, 2) == 1)
    len_phase = len_phase';
end
odd_ind = 1:2:Llp;
even_ind = 2:2:Llp;
plengthsd = len_phase(odd_ind)'; % pulse lengths in degrees
plengths_init = plengthsd/360*(1/settings.power)*1000; % pulse lengths in microseconds
pphase_init = len_phase(even_ind)'; % pulse phases
ppwr_init = ones(Llp/2, 1)*settings.power; % pulse powers

% Add pre and post pulse delays
if(isfield(settings, 'pre_pul_dly'))
    plengths_pre = reshape([settings.pre_pul_dly*ones(1, Llp/2); ...
        plengths_init'], Llp, 1);
    pphase_pre = reshape([zeros(1, Llp/2); pphase_init'], Llp, 1);
    ppwr_pre = reshape([zeros(1, Llp/2); ppwr_init'], Llp, 1);
    if(isfield(settings, 'post_pul_dly'))
        plengths = reshape([reshape(plengths_pre, 2, Llp/2); ...
            settings.post_pul_dly*ones(1, Llp/2)], 3*Llp/2, 1);
        pphase = reshape([reshape(pphase_pre, 2, Llp/2); zeros(1, Llp/2)], 3*Llp/2, 1);
        ppwr = reshape([reshape(ppwr_pre, 2, Llp/2); zeros(1, Llp/2)], 3*Llp/2, 1);
    else
        plengths = plengths_pre;
        pphase = pphase_pre;
        ppwr = ppwr_pre;
    end
elseif(isfield(settings, 'post_pul_dly'))
    plengths = reshape([plengths_init'; settings.post_pul_dly*ones(1, Llp/2)], Llp, 1);
    pphase = reshape([pphase_init'; zeros(1, Llp/2)], Llp, 1);
    ppwr = reshape([ppwr_init'; zeros(1, Llp/2)], Llp, 1);
else
    plengths = plengths_init;
    pphase = pphase_init;
    ppwr = ppwr_init;
end
pfoffs = zeros(length(plengths), 1); % pulse frequency offsets

pp_fname = settings.pulse_fname;
write_spinev_pp(plengths, ppwr, pphase, pfoffs, pp_fname);


freq_rng = settings.freq_rng; % [low step high]
pwr_rng = settings.pwr_rng; % [low step high]
```

```
freq_str = [num2str(freq_rng(1), '%.2f') ':' num2str(freq_rng(2), '%.2f') ...
    ':' num2str(freq_rng(3), '%.2f')];
pwr_str = [num2str(pwr_rng(1), '%.2f') ':' num2str(pwr_rng(2), '%.2f') ...
    ':' num2str(pwr_rng(3), '%.2f')];

% run the simulation
if(~isfield(settings, 'sim_opts'))
    settings.sim_opts = [];
end
% disp(['source ~/.profile; spinev ' settings.in_fname ' -t -macro\$P="' ...
%     pp_fname '" -macro\$F=' freq_str ' -macro\$N=' pwr_str ' -re ' settings.sim_opts])
[s, r] = system(['source ~/.profile; spinev ' settings.in_fname ' -t -macro\$P="' ...
    pp_fname '" -macro\$F=' freq_str ' -macro\$N=' pwr_str ' -re ' settings.sim_opts]);
if(s)
%     error(r)
    warning(r)
    disp(['source ~/.profile; spinev ' settings.in_fname ' -t -macro\$P="' ...
        pp_fname '" -macro\$F=' freq_str ' -macro\$N=' pwr_str ' -re ' settings.sim_opts])
    sm = inf;
    return
end

% d is the data without the string header
if(isempty(strfind(r, 'Warning')))
    d = str2num(r(strfind(r, 'loaded')+7:end));
else
    d = str2num(r(strfind(r, 'loaded')+7:(strfind(r, 'Warning')-1)));
end
if(isfield(settings, 'fit_values'))
    if(size(settings.fit_values) ~= size(d(:,2:end)))
        error('settings.fit_values is the incorrect size! %dx%d instead of %dx%d', ...
            size(settings.fit_values, 1), size(settings.fit_values, 2), ...
            size(d,1), size(d,2)-1)
    end
    sm = sum(sum((d(:,2:end) - settings.fit_values).^2))/numel(d(:, 2:end));
else
    % first column is the frequencies, normalized to make -1 perfect
    sm = sum(sum(d(:, 2:end)))/numel(d(:, 2:end));
end
if(isfield(settings, 'opt_mult')) % to set value negative for maximization problems.
    sm = sm*settings.opt_mult;
end
```

write_spinev_pp.m

The last function needed for simulation is the function that creates the pp file for spinev.

This is write_spinev_pp.m

```
function write_spinev_pp(plengths, ppwr, pphase, pfoffs, fname)
% function write_spinev_pp(plengths, ppwr, pphase, pfoffs)
%
% Writes a spinev pp file to be called from a spinev spv file. It creates a
% file with 4 columns. The first contains plengths--the pulse lengths in
% microseconds. The second column is pulse power in kHz (typically 50). The
% third is pulse phase (pphase) in degrees (0-360). The last column is
% frequency offset in kHz.

% Seth McNeill
% 2008 April 21

% columnize input data
if(size(plengths, 1) == 1)
    plengths = plengths';
end
if(size(ppwr, 1) == 1)
    ppwr = ppwr';
end
```

```
if(size(pphase, 1) == 1)
    pphase = pphase';
end
if(size(pfoffs, 1) == 1)
    pfoffs = pfoffs';
end

if(length(plengths) ~= length(ppwr) || length(ppwr) ~= length(pphase) || length(pphase) ~=
length(pfoffs))
    error('All inputs must be the same length except the filename')
end

datamat = [plengths ppwr pphase pfoffs];

fid = fopen(fname, 'w');
if(fid == -1)
    error(['Cannot open ' fname]);
end

fprintf(fid, '%f %f %f %f\n', datamat');

fprintf(fid, '*** Duration Pwr Phase Offset ***\n');
fprintf(fid, '*** %s ***\n', fname);
fprintf(fid, '*** Automatically created by %s ***\n', mfilename('fullpath'));
fprintf(fid, '*** Created at %s ***', datestr(clock));

fclose(fid);
```

Simulating Experiments and Comparing to Experimental Data

Use the following script to compare experimental and simulation data. The .mat file contains a settings structure for a particular pulse sequence. The structure for a 38-111 sequence is shown after the script.

```
% requires inv.par and adam.ss to be in the same directory
% copied and modified from MAS0808proc.m

clear
cd /Users/mcnese/Seth/SethsPapers/Dissertation/figs

p = 0; % 1 for print, 0 otherwise
resim = 0; % 1 to resimulate the data
do_phase_plot = 0; % 1 to do phases subplot

plots_fname = 'ch4invComparisonExpvsSimx3.pdf';


expind = 0;

expind = expind + 1;
exps_all(expind).dataset = 'MAS1208';
exps_all(expind).exps = 19;
exps_all(expind).sample_lengths = {'3.65'}; % mm
exps_all(expind).sim_fname = '~/Seth/PulseOptimization/spinevOpt/inversion/180.mat';
exps_all(expind).leg_txt = '';

expind = expind + 1;
exps_all(expind).dataset = 'MAS0808';
exps_all(expind).exps = 29;
exps_all(expind).sample_lengths = {'3.65'}; % mm
exps_all(expind).sim_fname = '~/Seth/PulseOptimization/spinevOpt/inversion/38111.mat';

expind = expind + 1;
exps_all(expind).dataset = 'MAS1208';
exps_all(expind).exps = 77;
```

```
exps_all(expind).sample_lengths = {'3.65'}; % mm
exps_all(expind).sim_fname = ...
    '/Users/mcnese/Seth/PulseOptimization/spinevOpt/inversion/HPC/20081204_7p_2/data/inv_-
0.9511_7_20081207.mat';
exps_all(expind).leg_txt = '';


cur_dir = pwd;
if(p)
    delete(fullfile(cur_dir, plots_fname));
end

h_mfig = figure('name', '38111 Inv exp vs sim at different nutation rates');
orient landscape; hold on; grid on
leg_txt = {};
h_exp = [];
for mm = 1:length(exps_all)
    cd(['~/Seth/750Data/' exps_all(mm).dataset])
    exps = exps_all(mm).exps;
    first_exp = min(exps);
    last_exp = max(exps);
    nkk = length(exps);
    kk = 1;
    for ii = exps
        % Collect experimental data
        eval(['expname = ''exp' num2str(ii) ''';']);
        disp(['%%%%%% Now evaluating: ' exps_all(mm).dataset ' ' expname])
        load([expname 'int'])
%         load(fullfile(num2str(ii), 'foffs'))
        foffs = load(fullfile(num2str(ii), 'fq1list'));
        foffs = flipud(foffs); % proved by asymmetry of refocusing pulses
        eval(['exprint = real(' expname 'int);']);
        expinfo = bruker_data(ii);
        if(length(foffs) == length(exprint))
            h_exp(end+1) = plot(foffs/1000/(1/(expinfo.P(2)*4e-3)), ...
                exprint/max(abs(exprint)), 'linestyle', 'none', ...
                'linewidth', 2, 'markersize', 10);
        else  % to account for control inversion at beginning
            h_exp(end+1) = plot(foffs/1000/(1/(expinfo.P(2)*4e-3)), ...
                exprint(2:end)/abs(exprint(1)), 'linestyle', 'none', ...
                'linewidth', 2, 'markersize', 10);
        end
%         leg_txt{end+1} = [exps_all(mm).dataset ' ' expname ' ' ...
%             num2str(1/(expinfo.P(2)*4e-3), '%.1f') ' kHz RF'];
%         leg_txt{end+1} = expinfo.pulprog;
        if(~isempty(exps_all(mm).sim_fname))
            load(exps_all(mm).sim_fname);

            % simulate the data
            if(exist([expname 'sim.mat'], 'file') && ~resim)
                load([expname 'sim.mat']);
            else
                load(exps_all(mm).sim_fname);

                % simulate the data
                if(~isfield(settings, 'shift'))
                    settings = load_par([cur_dir filesep 'inv.par'], settings);
                    settings = load_ss([cur_dir filesep settings.ss_fname], settings);
                    if(~isfield(settings, 'write_spv_fcn'))
                        settings.write_spv_fcn = 'write_spinev_inv';
                    end
                end
                if(isfield(settings, 'write_spv_fcn'))
                    % write the spv file (main spinev file)
                    feval(settings.write_spv_fcn, settings);
                end
                orig_freq_rng = settings.freq_rng;
                orig_pwr_rng = settings.pwr_rng;
                orig_power = settings.power;
                settings.freq_rng = [foffs(1)/1000 ...
                    (foffs(end) -  foffs(1))/(length(foffs)-1)/1000/5 foffs(end)/1000];
```

```
                settings.power = 1/(expinfo.P(2)*4e-3);
                settings.pwr_rng = [1/(expinfo.P(2)*4e-3) 1 1/(expinfo.P(2)*4e-3)];
                if(isfield(settings, 'fit_values') && ~isempty(settings.fit_values))
                    settings.fit_values = ...
                        [settings.freq_rng(1):settings.freq_rng(2):settings.freq_rng(3)]';
                end
                [sum_min, opt_sim_data] = feval(settings.fcn, ...
                    settings.len_phase, settings);
                save([expname 'sim.mat'], 'opt_sim_data');
                settings.freq_rng = orig_freq_rng;
                settings.pwr_rng = orig_pwr_rng;
                settings.power = orig_power;
            end
            plot(opt_sim_data(:,1)/(1/(expinfo.P(2)*4e-3)), opt_sim_data(:,2), '-', ...
                'linewidth', 2, 'color', get(h_exp(end), 'color'))
        end
        leg_txt{end+1} = [expinfo.pulprog ' ' ...
            num2str(sum(settings.len_phase(1:2:end)), '%.0f') ' degrees'];

        kk = kk + 1;
        co=get(gca,'ColorOrder');
        set(gca,'ColorOrder',co([2:end 1],:));
    end
end


cd(cur_dir)

% finish up the figure and print it if desired
figure(h_mfig)
h1 = gca;
set(h1, 'fontsize', 18)
% xlabel(h1, 'Frequency offset (kHz)', 'fontsize', 20)
xlabel(h1, 'Frequency offset (\Delta\omega/\omega)', 'fontsize', 20)
ylabel(h1, 'Projection on I_z', 'fontsize', 20)
title(h1, {['Inversion Comparison at ' num2str(1/(expinfo.P(2)*4e-3), '%.1f') ...
    ' kHz RF'], ['\fontsize{10}' datestr(now, 'yyyy mmm dd HH:MM:SS')]}, ...
    'fontsize', 20)
legend(h_exp, leg_txt, 'location', 'n')
legend boxoff
axis tight
a = axis;
axis(h1, [a(1) a(2) -1 1])
if(p)
    print('-dpdf', fullfile(cur_dir, plots_fname))
    disp(['Printed: ' fullfile(cur_dir, plots_fname)])
end
```

Settings for 38-111 simulation

```
settings =
              power: 50
          in_fname: 'inv2.spv'
        pulse_fname: 'inv_opt.pp'
            npulses: 4
           freq_rng: [-100 10 100]
            pwr_rng: [50 1 50]
    min_convergence: 1.0000e-04
    simulator_fname: 'spinev'
           sim_opts: '-split2 -renice10'
                fcn: 'spinev_eval_inv'
              nopts: 0
       max_cpu_time: 600
           data_dir: 'data'
         start_time: [2008 6 13 10 44 13.4249]
      start_datestr: 'settings created: 13-Jun-2008 10:45:28'
          len_phase: [38 0 111 180 159 0 250 180]
           plengths: [2.1111 6.1667 8.8333 13.8889]
```

```
           pphase: [0 180 0 180]
         hom_fval: 0
       total_time: 0
            fname: [1x54 char]
```

Fit over frequency offset and power offset

The following function was used to create Figure 4-9 showing the points that are used in optimization and the fit across both frequency offset and power offset. The filename should be for a .mat file created from an optimization.

```
function settings = plot_inv_opt(data_fname)
% function settings = plot_inv_opt(data_fname)
%
% Plots a dataset saved from optimizing an inversion pulse.

% Seth McNeill
% 2008 April 24

freq_rng = [-100 5 100];
pwr_rng = [30 2 70];
pwr = 50;

load(data_fname);
npulses = length(settings.plengths);

ppwr = ones(npulses, 1)*settings.power; % pulse powers
pfoffs = zeros(npulses, 1); % pulse frequency offsets

if(~isfield(settings, 'shift'))
    settings = load_par([cur_dir filesep 'refocus.par'], settings);
    settings = load_ss([cur_dir filesep settings.ss_fname], settings);
    if(~isfield(settings, 'write_spv_fcn'))
        settings.write_spv_fcn = 'write_spinev_refocus';
    end
end

orig_freq_rng = settings.freq_rng;
orig_pwr_rng = settings.pwr_rng;
orig_power = settings.power;
settings.freq_rng = freq_rng;
settings.power = pwr;
settings.pwr_rng = pwr_rng;

disp(['Writing ' settings.in_fname ' with ' settings.write_spv_fcn])
feval(settings.write_spv_fcn, settings) % write the SPV file
disp(['Started: ' datestr(now)])
start_time = clock;
[sum_min, opt_sim_data] = feval(settings.fcn, settings.len_phase, settings);
stop_time = clock;
sim_time_sec = etime(stop_time, start_time);
disp(['Sim took: ' sec2str(sim_time_sec) ' hh:mm:ss'])

dp = opt_sim_data';

% reset settings back to the originals
settings.freq_rng = orig_freq_rng;
settings.power = orig_power;
settings.pwr_rng = orig_pwr_rng;


fs = freq_rng(1):freq_rng(2):freq_rng(3);
ps = pwr_rng(1):pwr_rng(2):pwr_rng(3);
```

```
orig_fs = settings.freq_rng(1):settings.freq_rng(2):settings.freq_rng(3);
orig_ps = settings.pwr_rng(1):settings.pwr_rng(2):settings.pwr_rng(3);
sx = repmat(orig_fs', 1, length(orig_ps));
sy = repmat(orig_ps, length(orig_fs), 1);

figure('name',['Contour ' data_fname])
orient landscape
hold all
grid on
[C, hcon] = contourf(fs, ps, dp(2:end,:), -1:0.25:1);
h_s = plot(sx, sy, 'ko', 'linewidth', 3, 'markersize', 10);
set(hcon,'linecolor', 'none')
legend(h_s(1), 'Optimization Pts');
legend boxoff
ax_con = gca;
set(gca, 'fontsize', 20)
ylabel('Power (kHz)', 'fontsize', 24)
xlabel('Frequency Offset (kHz)', 'fontsize', 24)
title({'\fontsize{20}Optimized Inversion Composite pulse', ...
    ['\fontsize{14}Number of Pulses: ' num2str(npulses) ', New Fval = ' num2str(sum_min)], ...
    ['\fontsize{10}Pulse Lengths: ' num2str(settings.plengths, '%.1f ')], ...
    ['Pulse Phases: ' num2str(settings.pphase, '%.1f ')], ['File: ' titlize(data_fname)]})
colorbar
% colormap('gray')

keyboard
```

Liquids and Solids Simulation Comparison

It was instructive to compare liquids and solids for the refocusing experiment to show that solids are quite a bit different. The code for doing this comparison uses functions introduced elsewhere in this appendix and follows. This code does take a few minutes to complete since the simulations take some time.

```
%% 180 Refocusing solid vs liquid simulation
% requires inv.par and adam.ss to be in the same directory

clear
cd /Users/mcnese/Seth/SethsPapers/Dissertation/figs

p = 0; % 1 for print, 0 otherwise
sim_plot = 1; % To do a simulation at the "real" RF level

base_settings_fname = '~/Seth/PulseOptimization/spinevOpt/refocusing/180.mat';
power_base = 50; % kHz
freq_rng = [0 0.2 100];
spin_rate = 5000;
plots_fname = ['ch5180SolidvsLiquid_' num2str(power_base, '%.0f') 'kHz' ...
    num2str(spin_rate/1000, '%.1f'), 'MAS.ps'];


expind = 0;
expind = expind + 1;
exps_all(expind).len_phase = [180 0];
% expind = expind + 1;
% exps_all(expind).len_phase = [90 90 180 0 90 90];
% expind = expind + 1;
% exps_all(expind).len_phase = [90 90 240 0 90 90];
% expind = expind + 1;
% exps_all(expind).len_phase = [360 0 270 180 90 90 360 270 270 90 90 0];
% expind = expind + 1;
% exps_all(expind).len_phase = [38 0 111 180 159 0 250 180];
```

```
% expind = expind + 1;
% exps_all(expind).len_phase = [336 0 246 180 10 90 74 270 10 90 246 180 336 0];
% expind = expind + 1;
% exps_all(expind).len_phase = [151 247 342 182 180 320 342 182 151 247];

cur_dir = pwd;
if(p)
    delete(fullfile(cur_dir, plots_fname));
end

h_mfig = figure('name', 'Claridge Refocusing Solids Sims');
orient landscape; hold all; grid on
leg_txt = {};
h_exp = [];
for mm = 1:length(exps_all)
    %%%% Liquids Simulation %%%%
    load(base_settings_fname);
    disp(['Running: [' num2str(exps_all(mm).len_phase) '] liquids'])

    if(~isfield(settings, 'shift'))
        settings = load_par([cur_dir filesep 'refocus.par'], settings);
        settings = load_ss([cur_dir filesep settings.ss_fname], settings);
        if(~isfield(settings, 'write_spv_fcn'))
            settings.write_spv_fcn = 'write_spinev_refocus';
        end
    end
    % Modify settings for each pulse sequence
    settings.freq_rng = freq_rng;
    settings.power = power_base;
    settings.pwr_rng = [power_base 1 power_base];
    settings.len_phase = exps_all(mm).len_phase;
    settings.pphase = settings.len_phase(2:2:end);
    settings.plengths = settings.len_phase(1:2:end)/360*1/settings.power*1000;
    settings.spin_rate = '*';
    settings = rmfield(settings, 'shift'); % sets it to liquids
    settings.crystal_file = '*';
    settings.gamma_angles = '*';
    feval(settings.write_spv_fcn, settings); % write the spv file (main spinev file)

    [sum_min, opt_sim_data] = feval(settings.fcn, settings.len_phase, settings);
    plot(opt_sim_data(:,1)/power_base, opt_sim_data(:,2), '-', ...
        'linewidth', 2)
    leg_txt{end+1} = [sprintf('%.0f_{%.0f}', [settings.len_phase(1:2:end); ...
        settings.len_phase(2:2:end)]) ' liquids'];


    %%%% Solids simulation %%%%
    load(base_settings_fname);
    disp(['Running: [' num2str(exps_all(mm).len_phase) '] solids'])

    if(~isfield(settings, 'shift'))
        settings = load_par([cur_dir filesep 'refocus.par'], settings);
        settings = load_ss([cur_dir filesep settings.ss_fname], settings);
        if(~isfield(settings, 'write_spv_fcn'))
            settings.write_spv_fcn = 'write_spinev_refocus';
        end
    end
    % Modify settings for each pulse sequence
    settings.freq_rng = freq_rng;
    settings.power = power_base;
    settings.pwr_rng = [power_base 1 power_base];
    settings.len_phase = exps_all(mm).len_phase;
    settings.pphase = settings.len_phase(2:2:end);
    settings.plengths = settings.len_phase(1:2:end)/360*1/settings.power*1000;
    settings.spin_rate = spin_rate;
    feval(settings.write_spv_fcn, settings); % write the spv file (main spinev file)

    [sum_min, opt_sim_data] = feval(settings.fcn, settings.len_phase, settings);
    plot(opt_sim_data(:,1)/power_base, opt_sim_data(:,2), '-', ...
        'linewidth', 2)
    leg_txt{end+1} = [sprintf('%.0f_{%.0f}', [settings.len_phase(1:2:end); ...
```

```
        settings.len_phase(2:2:end)]) ' solids'];


    %%%% finish up the figure and print it if desired %%%%
    cd(cur_dir)
    figure(h_mfig)
    set(gca, 'fontsize', 20)
    xlabel('\omega_o_f_f_s_e_t/\omega_n_u_t_a_t_i_o_n', 'fontsize', 32)
    ylabel('Projection on I_X', 'fontsize', 32)
    title({['Refocusing from Claridge pp. 344 at ' num2str(power_base) ' kHz (solids)'], ...
        ['\fontsize{10}' datestr(now, 'yyyy mmm dd HH:MM:SS')]}, 'fontsize', 20)
    legend(leg_txt, 'location', 'sw')
    legend boxoff
    axis tight
    ax = axis;
    axis([0 ax(2) -1 1])
    if(p)
        print('-dpsc', fullfile(cur_dir, plots_fname))
        disp(['Printed: ' fullfile(cur_dir, plots_fname)])
    end

end
```

Refocusing Power Level Comparison Code

```
%% 180 Refocusing simulation at various Power levels
% requires inv.par and adam.ss to be in the same directory

clear
cd /Users/mcnese/Seth/SethsPapers/Dissertation/figs

p = 0; % 1 for print, 0 otherwise

plots_fname = 'ch5RefSim_PwrLevels.pdf';
base_settings_fname = '~/Seth/PulseOptimization/spinevOpt/refocusing/180.mat';
power_base = [10 25 50 75]; % kHz
freq_rng = [0 0.1 75];
spin_rate = 10000;

expind = 0;
expind = expind + 1;
exps_all(expind).len_phase = [180 0];
% expind = expind + 1;
% exps_all(expind).len_phase = [90 90 180 0 90 90];
% expind = expind + 1;
% exps_all(expind).len_phase = [90 90 240 0 90 90];
% expind = expind + 1;
% exps_all(expind).len_phase = [360 0 270 180 90 90 360 270 270 90 90 0];
% expind = expind + 1;
% exps_all(expind).len_phase = [38 0 111 180 159 0 250 180];
% expind = expind + 1;
% exps_all(expind).len_phase = [336 0 246 180 10 90 74 270 10 90 246 180 336 0];
% expind = expind + 1;
% exps_all(expind).len_phase = [151 247 342 182 180 320 342 182 151 247];

cur_dir = pwd;
if(p)
    delete(fullfile(cur_dir, plots_fname));
end

h_mfig = figure('name', 'Claridge Refocusing Solids Sims at various MAS rates');
orient landscape; hold all; grid on
leg_txt = {};
h_exp = [];
for mm = 1:length(exps_all)
    load(base_settings_fname);
    for nn = 1:length(power_base)
        disp(['Running: [' num2str(exps_all(mm).len_phase) '] at ' num2str(power_base(nn)) ' kHz
Pwr'])
        if(~isfield(settings, 'shift'))
```

```
            settings = load_par([cur_dir filesep 'refocus.par'], settings);
            settings = load_ss([cur_dir filesep settings.ss_fname], settings);
            if(~isfield(settings, 'write_spv_fcn'))
                settings.write_spv_fcn = 'write_spinev_refocus';
            end
        end
        % Modify settings for each pulse sequence
        settings.freq_rng = freq_rng;
        settings.power = power_base(nn);
        settings.pwr_rng = [power_base(nn) 1 power_base(nn)];
        settings.len_phase = exps_all(mm).len_phase;
        settings.pphase = settings.len_phase(2:2:end);
        settings.plengths = settings.len_phase(1:2:end)/360*1/settings.power*1000;
        settings.spin_rate = spin_rate;
        feval(settings.write_spv_fcn, settings); % write the spv file (main spinev file)

        [sum_min, opt_sim_data] = feval(settings.fcn, settings.len_phase, settings);
        plot(opt_sim_data(:,1)/power_base(nn), opt_sim_data(:,2), '-', ...
            'linewidth', 2)
        leg_txt{end+1} = [sprintf('%.0f_{%.0f}', ...
            [settings.len_phase(1:2:end); settings.len_phase(2:2:end)]) ...
            ' Pwr: ' num2str(power_base(nn)) ' kHz'];
    end
end

cd(cur_dir)

% finish up the figure and print it if desired
figure(h_mfig)
set(gca, 'fontsize', 20)
xlabel('\omega_o_f_f_s_e_t/\omega_n_u_t_a_t_i_o_n', 'fontsize', 32)
ylabel('Projection on I_X', 'fontsize', 32)
title({['Refocusing from Claridge pp. 344 at ' num2str(spin_rate/1000) ' kHz MAS'], ...
    ['\fontsize{10}' datestr(now, 'yyyy mmm dd HH:MM:SS')]}, 'fontsize', 20)
legend(leg_txt, 'location', 'sw')
legend boxoff
axis tight
ax = axis;
axis([0 1 -1 1])
if(p)
    print('-dpdf', fullfile(cur_dir, plots_fname))
    disp(['Printed: ' fullfile(cur_dir, plots_fname)])
end
```

Refocusing MAS Rate Comparison Code

```
%% 180 Refocusing simulation at various MAS rates
% requires inv.par and adam.ss to be in the same directory

clear
cd /Users/mcnese/Seth/SethsPapers/Dissertation/figs

p = 0; % 1 for print, 0 otherwise
sim_plot = 1; % To do a simulation at the "real" RF level

plots_fname = 'ch5RefSimMASrates_50kHz1w.pdf';
base_settings_fname = '~/Seth/PulseOptimization/spinevOpt/refocusing/180.mat';
power_base = 50; % kHz
freq_rng = [0 0.1 25];
spin_rate = [2500 5000 7500 10000];


expind = 0;
expind = expind + 1;
exps_all(expind).len_phase = [180 0];
% expind = expind + 1;
% exps_all(expind).len_phase = [90 90 180 0 90 90];
% expind = expind + 1;
% exps_all(expind).len_phase = [90 90 240 0 90 90];
% expind = expind + 1;
```

```
% exps_all(expind).len_phase = [360 0 270 180 90 90 360 270 270 90 90 0];
% expind = expind + 1;
% exps_all(expind).len_phase = [38 0 111 180 159 0 250 180];
% expind = expind + 1;
% exps_all(expind).len_phase = [336 0 246 180 10 90 74 270 10 90 246 180 336 0];
% expind = expind + 1;
% exps_all(expind).len_phase = [151 247 342 182 180 320 342 182 151 247];

cur_dir = pwd;
if(p)
    delete(fullfile(cur_dir, plots_fname));
end

h_mfig = figure('name', 'Claridge Refocusing Solids Sims at various MAS rates');
orient landscape; hold all; grid on
leg_txt = {};
h_exp = [];
for mm = 1:length(exps_all)
    load(base_settings_fname);
    for nn = 1:length(spin_rate)
        disp(['Running: [' num2str(exps_all(mm).len_phase) '] at ' ...
            num2str(spin_rate(nn)) ' kHz MAS'])
        if(~isfield(settings, 'shift'))
            settings = load_par([cur_dir filesep 'refocus.par'], settings);
            settings = load_ss([cur_dir filesep settings.ss_fname], settings);
            if(~isfield(settings, 'write_spv_fcn'))
                settings.write_spv_fcn = 'write_spinev_refocus';
            end
        end
        if(sim_plot)
            % Modify settings for each pulse sequence
            settings.freq_rng = freq_rng;
            settings.power = power_base;
            settings.pwr_rng = [power_base 1 power_base];
            settings.len_phase = exps_all(mm).len_phase;
            settings.pphase = settings.len_phase(2:2:end);
            settings.plengths = settings.len_phase(1:2:end)/360*1/settings.power*1000;
            settings.spin_rate = spin_rate(nn);
            feval(settings.write_spv_fcn, settings); % write the spv file (main spinev file)

            [sum_min, opt_sim_data] = feval(settings.fcn, settings.len_phase, settings);
            plot(opt_sim_data(:,1)/power_base, opt_sim_data(:,2), '-', ...
                'linewidth', 2)
            leg_txt{end+1} = [sprintf('%.0f_{%.0f}', ...
                [settings.len_phase(1:2:end); settings.len_phase(2:2:end)]) ...
                ' MAS: ' num2str(spin_rate(nn)/1000) ' kHz'];
        end
    end
end

cd(cur_dir)

% finish up the figure and print it if desired
figure(h_mfig)
set(gca, 'fontsize', 20)
xlabel('\omega_o_f_f_s_e_t/\omega_n_u_t_a_t_i_o_n', 'fontsize', 32)
ylabel('Projection on I_X', 'fontsize', 32)
title({['Refocusing from Claridge pp. 344 at ' num2str(power_base) ' kHz (solids)'], ...
    ['\fontsize{10}' datestr(now, 'yyyy mmm dd HH:MM:SS')]}, 'fontsize', 20)
legend(leg_txt, 'location', 'sw')
legend boxoff
axis tight
ax = axis;
axis([0 ax(2) -1 1])
if(p)
    print('-dpdf', fullfile(cur_dir, plots_fname))
    disp(['Printed: ' fullfile(cur_dir, plots_fname)])
end
```

Refocusing Experimental Comparison to Simulation

The code to load the experimental data and run the simulations that go with the experimental data is shown here.

```
%% Refocusing Experimental Individual
% requires inv.par and adam.ss to be in the same directory
% copied and modified from MAS0808proc.m

clear
cd /Users/mcnese/Seth/Presentations/Dissertation

p = 0; % 1 for print, 0 otherwise
sim_plot = 1; % To do a simulation at the "real" RF level
resim = 0; % 1 to resimulate the data
do_phase_plot = 0; % 1 to do phases subplot

dww = 2; % the plot limit in abs(deltaomega/omega)
plots_fname = 'refocus_individual.ps';


expind = 0;


%%%% Claridge based sequences %%%%
expind = expind + 1;
exps_all(expind).dataset = 'MAS1208';
% exps_all(expind).exps = [44 54];
exps_all(expind).exps = 54;
exps_all(expind).sample_lengths = {'3.65'}; % mm
exps_all(expind).sim_fname = '~/Seth/PulseOptimization/spinevOpt/refocusing/180.mat';
exps_all(expind).leg_txt = '180';
expind = expind + 1;
exps_all(expind).dataset = 'MAS1208';
exps_all(expind).exps = 73;
exps_all(expind).sample_lengths = {'3.65'}; % mm
exps_all(expind).sim_fname = '~/Seth/PulseOptimization/spinevOpt/refocusing/180.mat';
exps_all(expind).len_phase = [151 247 342 182 180 320 342 182 151 247];
exps_all(expind).leg_txt = '151-342';

%%%% My sequences %%%%
expind = expind + 1;
exps_all(expind).dataset = 'MAS1208';
exps_all(expind).exps = 88;
exps_all(expind).sample_lengths = {'3.65'}; % mm
exps_all(expind).sim_fname = ...
    ['/Users/mcnese/Seth/PulseOptimization/spinevOpt/refocusing/HPC/' ...
    'ref2w20090102_8p_15/data/ref2w_-0.8262_8_20090102.mat'];
exps_all(expind).leg_txt = 'My optimum';


cur_dir = pwd;
if(p)
    delete(fullfile(cur_dir, plots_fname));
end

for mm = 1:length(exps_all)
    h_mfig = figure('name', 'Refocusing experimental comparison');
    orient landscape; hold on; grid on
    leg_txt = {};
    h_exp = [];
    cd(['~/Seth/750Data/' exps_all(mm).dataset])
    exps = exps_all(mm).exps;
    first_exp = min(exps);
    last_exp = max(exps);
    nkk = length(exps);
    kk = 1;
```

```
    for ii = exps
        % Collect experimental data
        eval(['expname = ''exp' num2str(ii) ''';']);
        disp(['%%%%%% Now evaluating: ' exps_all(mm).dataset ' ' expname])
        load([expname 'int'])
        foffs = load(fullfile(num2str(ii), 'fq1list'));
        foffs = flipud(foffs); % proved by asymmetry of refocusing pulses
        eval(['exprint = real(' expname 'int);']);
        expinfo = bruker_data(ii);
        exp_pwr = (1/(expinfo.P(2)*4e-3)); % experimental power level
        expind = abs(foffs/1000/exp_pwr) <= dww; %indices of the points of interest
        if(length(foffs) == length(exprint))
            h_exp(end+1) = plot(foffs(expind)/1000/exp_pwr, ...
                exprint(expind)/max(abs(exprint)), 'linestyle', 'x', ...
                'linewidth', 2, 'markersize', 10);
            % average value of interest area
            exp_fitvalue = sum(exprint(expind))/sum(expind)/max(abs(exprint));
        else  % to account for control inversion at beginning
            h_exp(end+1) = plot(foffs(expind)/1000/exp_pwr, ...
                exprint([false; expind])/abs(exprint(1)), 'linestyle', 'x',...
                'linewidth', 2, 'markersize', 10);
             % average value of interest area
            exp_fitvalue = sum(exprint([false; expind]))/sum(expind)/abs(exprint(1));
        end
%         leg_txt{end+1} = [exps_all(mm).dataset ' ' expname ' ' ...
%             num2str(1/(expinfo.P(2)*4e-3), '%.1f') ' kHz RF ' expinfo.pulprog];
%         leg_txt{end+1} = expinfo.pulprog;
        leg_txt{end+1} = [exps_all(mm).leg_txt ' exp'];

        if(isfield(exps_all(mm), 'sim_fname') && ~isempty(exps_all(mm).sim_fname))
            load(exps_all(mm).sim_fname);
            % simulate the data
            if(~isfield(settings, 'shift'))
                settings = load_par([cur_dir filesep 'refocus.par'], settings);
                settings = load_ss([cur_dir filesep settings.ss_fname], settings);
                if(~isfield(settings, 'write_spv_fcn'))
                    settings.write_spv_fcn = 'write_spinev_refocus';
                end
            end
            if(sim_plot)
                % Just do a simulation plot at the correct RF frequency
                if(isfield(exps_all(mm), 'len_phase') && ~isempty(exps_all(mm).len_phase))
                    settings.len_phase = exps_all(mm).len_phase;
                end
                if(exist([expname 'sim.mat'], 'file') && ~resim)
                    load([expname 'sim.mat']);
                else
                    orig_freq_rng = settings.freq_rng;
                    orig_pwr_rng = settings.pwr_rng;
                    orig_power = settings.power;
                    settings.freq_rng = [foffs(1)/1000 ...
                        (foffs(end) -  foffs(1))/(length(foffs)-1)/1000/5 ...
                        foffs(end)/1000];
                    settings.power = 1/(expinfo.P(2)*4e-3);
                    settings.pwr_rng = [1/(expinfo.P(2)*4e-3) 1 1/(expinfo.P(2)*4e-3)];
                    disp(['Writing ' settings.in_fname ' with ' settings.write_spv_fcn])
                    feval(settings.write_spv_fcn, settings) % write the SPV file
                    disp(['Started: ' datestr(now)])
                    start_time = clock;
                    [sum_min, opt_sim_data] = feval(settings.fcn, ...
                        settings.len_phase, settings);
                    stop_time = clock;
                    sim_time_sec = etime(stop_time, start_time);
                    save([expname 'sim.mat'], 'opt_sim_data');
                end
                simind = abs(opt_sim_data(:,1)/exp_pwr) <= dww;
                sim_fitvalue = sum(opt_sim_data(simind, 2))/sum(simind);
                h_exp(end+1) = plot(opt_sim_data(simind,1)/exp_pwr, ...
                    opt_sim_data(simind,2), '-', 'linewidth', 2, 'color', ...
                    get(h_exp(end), 'color'));
        leg_txt{end+1} = [exps_all(mm).leg_txt ' sim'];
```

```
%                 leg_txt{end+1} = [exps_all(mm).dataset ' ' expname ' ' ...
%                     num2str(1/(expinfo.P(2)*4e-3), '%.1f') ' kHz RF ' ...
%                     expinfo.pulprog ' ' num2str(sum(settings.len_phase(1:2:end)), ...
%                     '%.0f') ' degrees'];
            end
        end

        kk = kk + 1;
        co=get(gca,'ColorOrder');
        set(gca,'ColorOrder',co([2:end 1],:));
    end

    cd(cur_dir)

    % finish up the figure and print it if desired
    figure(h_mfig)
    h1 = gca;
    set(h1, 'fontsize', 18)
    % xlabel(h1, 'Frequency offset (kHz)', 'fontsize', 20)
    xlabel(h1, 'Frequency offset (\Delta\omega/\omega)', 'fontsize', 20)
    ylabel(h1, 'Projection on I_z', 'fontsize', 20)
    title(h1, {['Refocus Comparison'], ...
        ['\fontsize{10}' datestr(now, 'yyyy mmm dd HH:MM:SS')]}, 'fontsize', 20)
    legend(h_exp, leg_txt, 'location', 's')
    legend boxoff
    % a = axis;
    % axis(h1, [a(1) a(2) -1 1])
    axis([-dww dww -1 1])
    % axis([-1 1 0.4 1.05])
    if(p)
        %     print('-dpdf', fullfile(cur_dir, plots_fname))
        print('-dpsc', '-append', fullfile(cur_dir, plots_fname))
        disp(['Printed: ' fullfile(cur_dir, plots_fname)])
    end

end
```

DRAWS Experimental to Simulation Comparison

Matlab Script

```
%% Spinev optimized DRAWS spinev2d_755a DRAWS Normalized

%  MAS0907 exps = [8:10 16:18 42:44 91:93]; % Buildup Experiment numbers

clear
cd /Users/mcnese/Seth/Presentations/Dissertation

p = 0; % 1 for print, 0 otherwise
sim_plot = 1; % To do a simulation at the "real" RF level
% resim = 0; % 1 to resimulate the data
int_region = [1208:1298,1307:1371,1569:1624,1660:1732,1922:1976,2012:2058,...
    2274:2311,2356:2401,2609:2645,2708:2745]; % region to integrate over

plots_fname = 'SpinevDRAWS_norm.pdf';

expind = 0;

expind = expind + 1;
exps_all(expind).dataset = '~/Seth/750Data/MAS0907';
exps_all(expind).name{1} = 'Original DRAWS';
exps_all(expind).name{2} = 'Optimized DRAWS';
exps_all(expind).sample = '*G*AV';
% exps_all(expind).exps = [8:9 42 44 91 93];
% exps_all(expind).exps = [8:9 44];
exps_all(expind).exps = [44 43];
exps_all(expind).int_region = int_region; % region to integrate over
exps_all(expind).par_fname = 'uneq10.par';
exps_all(expind).phases(1,:) = [16384 49152 0 16384 49152]/65536*360;
```

```
exps_all(expind).phases(2,:) = [18088 17309 34612 17062 14680]/65536*360;

cur_dir = pwd;
if(p)
    delete(fullfile(cur_dir, plots_fname));
end

h_mfig = figure('name', 'DRAWS Buildups');
orient landscape; hold all; grid on
leg_txt = {};
h_exp = [];
for mm = 1:length(exps_all)
    exps = exps_all(mm).exps;
    first_exp = min(exps);
    last_exp = max(exps);
    nkk = length(exps);
    kk = 1;
    for ii = exps
        % Collect experimental data
        eval(['expname = ''exp' num2str(ii) ''';']);
        disp(['%%%%%% Now evaluating: ' exps_all(mm).dataset ' ' expname])
        cd(exps_all(mm).dataset)
        expinfo = bruker_data(ii);
        %%%% integrate over a fixed region %%%%
        load(expname);
        eval(['expdata = ' expname ';'])
        clear(expname)
        exprint(:,kk) = real(sum(expdata(exps_all(mm).int_region, :)));
        %%%%%%
        h_exp(end+1) = plot(1:(length(exprint(:,kk))-1), ...
            exprint(2:end,kk)/max(exprint(2:end,1)), 'linewidth', 2, ...
            'linestyle', 'none');
%         leg_txt{end+1} = [num2str(ii) ') ' exps_all(mm).sample ' ' expinfo.pulprog];
        leg_txt{end+1} = [exps_all(mm).name{kk} ' exp'];

        if(sim_plot)
            cd(cur_dir)
            settings = load_par(exps_all(mm).par_fname);
            settings = load_ss(settings.ss_fname, settings);
            [mx, draws, all_data] = feval(settings.fcn, ...
                exps_all(mm).phases(kk,:), settings);
            sim_data(:,kk) = all_data(2:end,2);
            h_exp(end+1) = plot(1:length(sim_data(:,kk)), ...
                sim_data(:,kk)/max(sim_data(:,1)), '-', 'linewidth', 2, ...
                'color', get(h_exp(end), 'color'));
           leg_txt{end+1} = [exps_all(mm).name{kk} ' sim'];
        end
        kk = kk + 1;
    end

end

figure(h_mfig)
set(gca, 'fontsize', 28)
xlabel('DRAWS Cycles', 'fontsize', 36)
ylabel('Normalized Magnitude', 'fontsize', 36)
title({['DRAWS Buildups'], ['\fontsize{12}Integration region: [' ...
    num2str(exps_all(mm).int_region(1)) ':' ...
    num2str(exps_all(mm).int_region(end)) ']'], ...
    ['\fontsize{10}' datestr(now, 'yyyy mmm dd HH:MM:SS')]},...
    'fontsize', 20)
legend(h_exp, leg_txt, 'location', 's', 'fontsize', 12)
legend boxoff
ax = axis;
axis([1 10 ax(3:4)])

cd(cur_dir)
if(p)
    print('-dpdf', fullfile(cur_dir, plots_fname))
    disp(['Printed: ' fullfile(cur_dir, plots_fname)])
end
```

Parameter File

```
# uneq10.par
#
# Parameter file for matmin_mult optimizations
#
# Seth McNeill
# 2007 January 12

spin_rate 5000
sw        50000 # Spectral Width, I think this is changed in the simpson file
np        11 # number of DRAWS cycles to try
ni        20 # number of points along inhomogeneity sample length
#crystal_file zcw3_50.dat
crystal_file rep100.dat
gamma_angles 7
start_operator   I1x+I2x
detect_operator  I1p+I2p
proton_frequency 750e6
# use_cluster      0
Csigma  0.7
maxL    0.5
R_fname   draws.pp # name of the file containing the R group
phase_in 1111
phase_out 1234
phase_rcv 1432
t360s 8.5

ss_fname test.ss # Spin system file
min_convergence 0.0001 # fminsearch minimum convergence
#simpson_fname /Users/mcnese/apps/simpson # this is for etude
#simpson_fname /usr/local/bin/simpson # this is for ascaris/briggsae
#simpson_fname ~/bin/simpson # this is for hpc
simulator_fname spinev
in_fname draws.spv # the name of the input file to call, should be same as par(name)
sim_opts -vclk
stdout_fname spinev_out.txt
max_cpu_time 500
fcn spinev_eval_neg_uneq10

stop_fname stopfile.txt
err_fname draws_errors.txt
```

CP Simulation Code

The MRC paper has several simulations of CP matching conditions for both static and MAS samples. The code here was used to make those figures. Note that the input files are for SIMPSON instead of spinev.

Static Square CP

Spin system

```
# CP0.ss
#
# This is a spin system file for 1H-13C cross polarization simulations.
#
# The format is the same as in a spinsys section
# of a SIMPSON input file.
```

```
#
# Seth McNeill
# 2007 June 19

channels 1H 13C
nuclei 1H 13C
dipole 1 2 -2500 0 0 0
shift 1 0 0 0 0 0 0
shift 2 0p 0p 1 0 0 0
```

SIMPSON input file

```
# CPrampStat.in
#
# This is the base SIMPSON input file for calculating static ramped CP
# curves. It reads in a .par file and a .ss file for the pars and spinsys.
#
# Seth McNeill
# 2007 June 15


source "~/Seth/bin/sutil.in"
#source "~/bin/sutil.in"

global par_fname ss_fname
set par_fname "./CPs/CP_C75_H95_0kHz.par"
set ss_fname "CP0.ss"


proc pulseq {} {
    global par

    incr par(npulseq)

    for {set ii 0} {$ii <= $par(np)} {incr ii} {
        reset
        pulse $par(cpmix) [expr $ii*$par(dHrf)] 0 $par(Xrf) 0
        acq
    }
}

proc main {} {
   global par spinsys start_time par_fname ss_fname

   init_log

   read_par_file $par_fname
   read_ss_file $ss_fname

   log_spinsys
   set dl [expr $par(maxL)/$par(ni)]
   puts "dl = $dl"

    set par(npulseq) 1

   for {set j 0} {$j <= $par(ni)} {incr j} {
      set calc_start_time [runtime]
      set L [expr $j*$dl]
      set par(dHrf) [expr $par(dHrfbase)*exp(-pow($L, 2)/(2*pow($par(Hsigma), 2)))]
      set par(Xrf) [expr $par(Xrfbase)*exp(-pow($L, 2)/(2*pow($par(Xsigma), 2)))]
      puts "Starting j = $j, L = $L, Crf = $par(Xrf)"
      log_pars
      set fd [fsimpson]
# Save data
#      fsave $fd $par(name)_$L.fid
#      set f [open $par(name)_$L.xy w]
#      for {set i 1} {$i <= $par(np)} {incr i} {
#          puts $f "[expr ($i-1)*$par(dHrfbase)/1000.0] [findex $fd $i -re]"
#      }
#      close $f
```

```
# Sum the data, doubling all points that aren't the center point
      if {$j == 0} {
         set s [fdup $fd]
         set f [open $par(name)_0_ideal.xy w]
         for {set i 1} {$i <= $par(np)} {incr i} {
             puts $f "[expr ($i-1)*$par(dHrfbase)/1000.0] [findex $s $i -re]"
         }
         close $f
      } else {
         fexpr $fd {$re*2} {$im*2}
        fadd $s $fd
      }
      funload $fd
      set calc_time [expr [runtime] - $calc_start_time]
      set avg_calc_time [expr [runtime]/($j+1)]
      set est_time_left [expr ($par(ni) - ($j+1))*$avg_calc_time]
      set est_tt [hhmmss $est_time_left]
      set tt [hhmmss [runtime]]
      write_log "Calculation took $calc_time seconds, total time is $tt"
      write_log "Avg: $avg_calc_time, Est Left: $est_tt"
   }
# Save summed data
   set f [open $par(name)_0_sum.xy w]
   for {set i 1} {$i <= $par(np)} {incr i} {
       puts $f "[expr ($i-1)*$par(dHrfbase)/1000.0] [findex $s $i -re]"
   }
   close $f
   funload $s
   close_log
}
```

Parameter file

```
# CP_C75_H50_0kHz.par
#
# This is a test set of parameters for a 13C nutation curve with 50%
# homogeneity.
#
# Seth McNeill
# 2007 June 12

spin_rate 0
np        90 # number of 1H powers to try
ni        20 # number of points along inhomogeneity sample length
crystal_file rep168
gamma_angles 1
start_operator   I1x
detect_operator  I2p
proton_frequency 750e6
use_cluster      0

Hsigma   1.29
Xsigma   0.84
maxL     0.5
dHrfbase 1000
Xrfbase  50000
cpmix    3000
maxRamp  100  # maximum power percentage for ramp
minRamp  50  # minimum power percentage for ramp
nRamp    20

simulator_fname /Users/mcnese/apps/simpson # this is for etude
#simulator_fname /usr/local/bin/simpson # this is for ascaris/briggsae
#simulator_fname ~/bin/simpson # this is for hpc
stdout_fname simpson_out6.txt
```

Static Ramped CP

SIMPSON input file

```
# CPrampStat.in
#
# This is the base SIMPSON input file for calculating static ramped CP
# curves. It reads in a .par file and a .ss file for the pars and spinsys.
#
# Seth McNeill
# 2007 June 15


source "~/Seth/bin/sutil.in"
#source "~/bin/sutil.in"

global par_fname ss_fname
set par_fname "./CPs/CP_C75_H95_0kHz.par"
set ss_fname "CP0.ss"


proc pulseq {} {
    global par

    set dPulse [expr $par(cpmix)/$par(nRamp)]
    set dRamp [expr ($par(maxRamp)/100.0 - $par(minRamp)/100.0)/$par(nRamp)]
#    write_log "n = $n, tr = $tr, dPulse = $dPulse, dRamp = $dRamp, npulseq = $par(npulseq)"
    incr par(npulseq)

    for {set ii 0} {$ii <= $par(np)} {incr ii} {
        reset
        for {set jj 0} {$jj < $par(nRamp)} {incr jj} {
            pulse $dPulse [expr $ii*$par(dHrf)*($par(maxRamp)/100.0 - $jj*$dRamp)] 0 $par(Xrf) 0
        }
        acq
    }
}

proc main {} {
   global par spinsys start_time par_fname ss_fname

   init_log

   read_par_file $par_fname
   read_ss_file $ss_fname

   log_spinsys
   set dl [expr $par(maxL)/$par(ni)]
   puts "dl = $dl"

    set par(npulseq) 1

   for {set j 0} {$j <= $par(ni)} {incr j} {
      set calc_start_time [runtime]
      set L [expr $j*$dl]
      set par(dHrf) [expr $par(dHrfbase)*exp(-pow($L, 2)/(2*pow($par(Hsigma), 2)))]
      set par(Xrf) [expr $par(Xrfbase)*exp(-pow($L, 2)/(2*pow($par(Xsigma), 2)))]
      puts "Starting j = $j, L = $L, Crf = $par(Xrf)"
      log_pars
      set fd [fsimpson]
# Save data
#      fsave $fd $par(name)_$L.fid
#      set f [open $par(name)_$L.xy w]
#      for {set i 1} {$i <= $par(np)} {incr i} {
#          puts $f "[expr ($i-1)*$par(dHrfbase)/1000.0] [findex $fd $i -re]"
#      }
#      close $f
# Sum the data, doubling all points that aren't the center point
      if {$j == 0} {
```

```
         set s [fdup $fd]
         set f [open $par(name)_0_ideal.xy w]
         for {set i 1} {$i <= $par(np)} {incr i} {
             puts $f "[expr ($i-1)*$par(dHrfbase)/1000.0] [findex $s $i -re]"
         }
         close $f
      } else {
         fexpr $fd {$re*2} {$im*2}
         fadd $s $fd
      }
      funload $fd
      set calc_time [expr [runtime] - $calc_start_time]
      set avg_calc_time [expr [runtime]/($j+1)]
      set est_time_left [expr ($par(ni) - ($j+1))*$avg_calc_time]
      set est_tt [hhmmss $est_time_left]
      set tt [hhmmss [runtime]]
      write_log "Calculation took $calc_time seconds, total time is $tt"
      write_log "Avg: $avg_calc_time, Est Left: $est_tt"
   }
# Save summed data
   set f [open $par(name)_0_sum.xy w]
   for {set i 1} {$i <= $par(np)} {incr i} {
       puts $f "[expr ($i-1)*$par(dHrfbase)/1000.0] [findex $s $i -re]"
   }
   close $f
   funload $s
   close_log
}
```

Parameter file

```
# CP_C75_H50_0kHz.par
#
# This is a test set of parameters for a 13C nutation curve with 50%
# homogeneity.
#
# Seth McNeill
# 2007 June 12

spin_rate 0
np        90 # number of 1H powers to try
ni        20 # number of points along inhomogeneity sample length
crystal_file rep168
gamma_angles 1
start_operator   I1x
detect_operator  I2p
proton_frequency 750e6
use_cluster      0

Hsigma   1.29
Xsigma   0.84
maxL     0.5
dHrfbase 1000
Xrfbase  50000
cpmix    3000
maxRamp  100  # maximum power percentage for ramp
minRamp  50  # minimum power percentage for ramp
nRamp    20

simulator_fname /Users/mcnese/apps/simpson # this is for etude
#simulator_fname /usr/local/bin/simpson # this is for ascaris/briggsae
#simulator_fname ~/bin/simpson # this is for hpc
stdout_fname simpson_out6.txt
```

MAS Square CP

Spin system file

```
# CP.ss
#
# This is a spin system file for 1H-13C cross polarization simulations.
#
# The format is the same as in a spinsys section
# of a SIMPSON input file.
#
# Seth McNeill
# 2007 June 12

channels 1H 13C
nuclei 1H 13C
dipole 1 2 -2500 0 0 0
shift 1 0 0 0 0 0 0
shift 2 80p -80p 1 0 0 0
```

SIMPSON input file

```
# CP.in
#
# This is the base SIMPSON input file for calculating CP curves. It reads
# in a .par file and a .ss file for the pars and spinsys.
#
# Seth McNeill
# 2007 June 12


source "~/Seth/bin/sutil.in"
#source "~/bin/sutil.in"

global par_fname ss_fname
set par_fname "./CPs/CP_C75_H50_10kHz.par"
set ss_fname "CP.ss"


proc pulseq {} {
    global par

#    maxdt 1
    maxdt 0.5

    set tr [expr 1.0e6/$par(spin_rate)]
    set n [expr int($par(cpmix)/$tr)-1]

    for {set i 0} {$i <= $par(np)} {incr i} {
        reset
        pulse $tr [expr $i*$par(dHrf)] 0 $par(Xrf) 0
        store 1
        prop 1 $n
        acq
    }
}

proc main {} {
   global par spinsys start_time par_fname ss_fname

   init_log

   read_par_file $par_fname
   read_ss_file $ss_fname

   log_spinsys
   set dl [expr $par(maxL)/$par(ni)]
   puts "dl = $dl"
```

```
   for {set j 0} {$j <= $par(ni)} {incr j} {
      set calc_start_time [runtime]
      set L [expr $j*$dl]
      set par(dHrf) [expr $par(dHrfbase)*exp(-pow($L, 2)/(2*pow($par(Hsigma), 2)))]
      set par(Xrf) [expr $par(Xrfbase)*exp(-pow($L, 2)/(2*pow($par(Xsigma), 2)))]
      puts "Starting j = $j, L = $L, Crf = $par(Xrf)"
      log_pars
      set fd [fsimpson]
# Save data
#      fsave $fd $par(name)_$L.fid
#      set f [open $par(name)_$L.xy w]
#      for {set i 1} {$i <= $par(np)} {incr i} {
#          puts $f "[expr ($i-1)*$par(dHrfbase)/1000.0] [findex $fd $i -re]"
#      }
#      close $f
# Sum the data, doubling all points that aren't the center point
      if {$j == 0} {
         set s [fdup $fd]
         set f [open $par(name)_$par(spin_rate)ideal.xy w]
         for {set i 1} {$i <= $par(np)} {incr i} {
             puts $f "[expr ($i-1)*$par(dHrfbase)/1000.0] [findex $s $i -re]"
         }
         close $f
      } else {
         fexpr $fd {$re*2} {$im*2}
        fadd $s $fd
      }
      funload $fd
      set calc_time [expr [runtime] - $calc_start_time]
      set tt [hhmmss [runtime]]
      write_log "Calculation took $calc_time seconds, total time is $tt"
   }
# Save summed data
   set f [open $par(name)_$par(spin_rate)sum.xy w]
   for {set i 1} {$i <= $par(np)} {incr i} {
       puts $f "[expr ($i-1)*$par(dHrfbase)/1000.0] [findex $s $i -re]"
   }
   close $f
   funload $s
   close_log
}
```

Parameter file

```
# CP_C75_H50_10kHz.par
#
# This is a test set of parameters for a 13C nutation curve with 50%
# homogeneity.
#
# Seth McNeill
# 2007 June 12

spin_rate 10000
np        90 # number of DRAWS cycles to try
ni        20 # number of points along inhomogeneity sample length
crystal_file rep30
gamma_angles 7
start_operator   I1x
detect_operator  I2p
proton_frequency 750e6
use_cluster      0

Hsigma   0.68
Xsigma   0.84
maxL     0.5
dHrfbase 1000
Xrfbase  50000
cpmix    3000
maxRamp  100  # maximum power percentage for ramp
```

```
minRamp  50  # minimum power percentage for ramp
nRamp    20

simulator_fname /Users/mcnese/apps/simpson # this is for etude
#simulator_fname /usr/local/bin/simpson # this is for ascaris/briggsae
#simulator_fname ~/bin/simpson # this is for hpc
stdout_fname simpson_out1.txt
```

MAS Ramped CP

SIMPSON input file

```
# CPramp.in
#
# This is the base SIMPSON input file for calculating ramped CP curves. It
# reads in a .par file and a .ss file for the pars and spinsys.
#
# Seth McNeill
# 2007 June 12


source "~/Seth/bin/sutil.in"
#source "~/bin/sutil.in"

global par_fname ss_fname
set par_fname "./CPs/CPramp_C75_H95_10kHz.par"
set ss_fname "CP.ss"


proc pulseq {} {
    global par

#    maxdt 1
    maxdt 0.5

    set tr [expr 1.0e6/$par(spin_rate)]
    set n [expr int($par(cpmix)/$tr)]
    set Ramp [expr $n*$tr]
    set dPulse [expr $Ramp/$par(nRamp)]
    set dRamp [expr ($par(maxRamp)/100.0 - $par(minRamp)/100.0)/$par(nRamp)]
#    write_log "n = $n, tr = $tr, Ramp = $Ramp, dPulse = $dPulse, dRamp = $dRamp, npulseq =
$par(npulseq)"
    incr par(npulseq)

    for {set ii 0} {$ii <= $par(np)} {incr ii} {
        reset
        for {set jj 0} {$jj < $par(nRamp)} {incr jj} {
            pulse $dPulse [expr $ii*$par(dHrf)*($par(maxRamp)/100.0 - $jj*$dRamp)] 0 $par(Xrf) 0
        }
        acq
    }
}

proc main {} {
   global par spinsys start_time par_fname ss_fname

   init_log

   read_par_file $par_fname
   read_ss_file $ss_fname

   log_spinsys
   set dl [expr $par(maxL)/$par(ni)]
   puts "dl = $dl"

    set par(npulseq) 1

   for {set j 0} {$j <= $par(ni)} {incr j} {
      set calc_start_time [runtime]
```

```
      set L [expr $j*$dl]
      set par(dHrf) [expr $par(dHrfbase)*exp(-pow($L, 2)/(2*pow($par(Hsigma), 2)))]
      set par(Xrf) [expr $par(Xrfbase)*exp(-pow($L, 2)/(2*pow($par(Xsigma), 2)))]
      puts "Starting j = $j, L = $L, Crf = $par(Xrf)"
      log_pars
      set fd [fsimpson]
# Save data
#      fsave $fd $par(name)_$L.fid
#      set f [open $par(name)_$L.xy w]
#      for {set i 1} {$i <= $par(np)} {incr i} {
#          puts $f "[expr ($i-1)*$par(dHrfbase)/1000.0] [findex $fd $i -re]"
#      }
#      close $f
# Sum the data, doubling all points that aren't the center point
      if {$j == 0} {
         set s [fdup $fd]
         set f [open $par(name)_$par(spin_rate)ideal.xy w]
         for {set i 1} {$i <= $par(np)} {incr i} {
             puts $f "[expr ($i-1)*$par(dHrfbase)/1000.0] [findex $s $i -re]"
         }
         close $f
      } else {
         fexpr $fd {$re*2} {$im*2}
         fadd $s $fd
      }
      funload $fd
      set calc_time [expr [runtime] - $calc_start_time]
      set avg_calc_time [expr [runtime]/($j+1)]
      set est_time_left [expr ($par(ni) - ($j+1))*$avg_calc_time]
      set est_tt [hhmmss $est_time_left]
      set tt [hhmmss [runtime]]
      write_log "Calculation took $calc_time seconds, total time is $tt"
      write_log "Avg: $avg_calc_time, Est Left: $est_tt"
   }
# Save summed data
   set f [open $par(name)_$par(spin_rate)sum.xy w]
   for {set i 1} {$i <= $par(np)} {incr i} {
       puts $f "[expr ($i-1)*$par(dHrfbase)/1000.0] [findex $s $i -re]"
   }
   close $f
   funload $s
   close_log
}
```

Parameter file

```
# CPramp_C75_H50_10kHz.par
#
# This is a test set of parameters for a 13C nutation curve with 50%
# homogeneity.
#
# Seth McNeill
# 2007 June 12

spin_rate 10000
np        90 # number of 1H powers to try
ni        20 # number of points along inhomogeneity sample length
crystal_file rep30
gamma_angles 7
start_operator   I1x
detect_operator  I2p
proton_frequency 750e6
use_cluster      0

Hsigma   1.29
Xsigma   0.84
maxL     0.5
dHrfbase 1000
Xrfbase  50000
cpmix    3000
```

```
maxRamp  100  # maximum power percentage for ramp
minRamp  50  # minimum power percentage for ramp
nRamp    20

simulator_fname /Users/mcnese/apps/simpson # this is for etude
#simulator_fname /usr/local/bin/simpson # this is for ascaris/briggsae
#simulator_fname ~/bin/simpson # this is for hpc
stdout_fname simpson_out.txt
```

APPENDIX B
OPTIMIZATION SCRIPTS

Inversion Optimization

create_run.sh

This script creates the needed directories and runs an inversion optimization. The usage can be seen by running the script with no arguments. The run_fname is the name of the run file created by compiling the functions. This script does expect that the MCR files for running compiled Matlab code are located at: ~/bin/mcr/v78/v78. Running mcrinstaller in Matlab will show you where the binary file is located that will install the mcr files. The mcr files must be installed for compiled Matlab to run.

Useage: create_run.sh run_fname par_fname npulses

```
#!/bin/bash
# create_run.sh
#
# This file should write a submitable script to run an inversion optimization
# and the submits it to the cluster.
#
# INPUTS:
#     run_fname - the name of the compiled matlab script
#     par_fname - name of the par file to use
#     npulses - number of pulses to use in optimization
#
# Seth McNeill
# 2008 June 20

if [ $# -lt 3 ] # check to make sure there are at least 2 commandline inputs
then
   echo "Useage: create_run.sh run_fname par_fname npulses"
   exit 1
fi

run_fname=$1
par_fname=$2
npulses=$3
mat_prgm_fname=`echo $run_fname | sed 's/run_//' | sed 's/.sh//'`
# grab the name base from the input file
name_base=`grep "^name_base" $par_fname | awk '{ print $2 }'`
if [ -z "$name_base" ]
then
   name_base="inv"
fi
# grab the maximum cpu time to use from the input file
max_cpu_time=`grep "^max_cpu_time" $par_fname | awk '{ print $2 }'`
# check to see if more than one cpu is going to be used
multicpu=`grep "^sim_opts" $par_fname | grep split`
if [ -n "$multicpu" ]
then
   echo $multicpu
   echo "How many cores does this script use?"
   read ncpus
else
```

```
   ncpus=1
fi
echo "this is going to use $ncpus cores"

# Generate the directory for everything to run in
dir_base="$UFHPC_SCRATCH/$USER/inversion"
date_str=`date "+%Y%m%d"` # Creates a string YYYYMMDD

if [ -d $dir_base ]
then
   echo "$dir_base already exists"
else
   echo "Creating $dir_base"
   mkdir $dir_base
fi

dir_main="$dir_base/$name_base${date_str}_${npulses}p" # name of main directory to work out of
if [ -d $dir_main ] # if the main directory already exists, create append a number until it
doesn't exist
then
   ii=2
   while [ -d $dir_main ]
   do
      dir_main="$dir_base/$name_base${date_str}_${npulses}p_${ii}"
      ii=`expr $ii + 1`
   done
fi
echo "Creating $dir_main"
mkdir $dir_main

cp $run_fname $dir_main
cp $par_fname $dir_main
cp $mat_prgm_fname $dir_main

exec 3<> $dir_main/q_run  # open a file named q_run for reading and writing

now_str=`date "+%Y %b %d %H:%M:%S"` # date string for file creation

# Write the script
echo "#!/bin/bash" >&3
echo "### $dir_main/q_run" >&3
echo "### Automatically generated $now_str" >&3
echo >&3
echo "### Set the job name to something better than the script name" >&3
echo "#PBS -N $name_base$npulses" >&3
echo >&3
echo "### Set the error out directory" >&3
echo "#PBS -e $dir_main" >&3
echo  >&3
echo "### Set the standard output directory" >&3
echo "#PBS -o $dir_main" >&3
echo  >&3
echo "### Set the cpu time, number of cpus, and memory requirements" >&3
echo "#PBS -l walltime=$max_cpu_time:00:00" >&3
echo "#PBS -l nodes=1:ppn=$ncpus:gige" >&3
echo "#PBS -l mem=100mb" >&3
echo "" >&3
echo "### Set the report email address" >&3
echo "#PBS -M mcnese@mbi.ufl.edu" >&3
echo "" >&3
echo "### Set the report email options" >&3
echo "#PBS -m ae" >&3
echo "### Make PBS look in the directory where the command was submitted from:" >&3
echo "cd $dir_main" >&3
echo "" >&3
echo "### execute the program:" >&3
echo 'echo $PBS_NODEFILE' >&3
echo "$dir_main/$run_fname ~/bin/mcr/v78/v78 $par_fname $npulses > stdout.txt" >&3
echo "" >&3
echo "### End the script cleanly:" >&3
echo "exit 0" >&3
```

```
exec 3>&- # close file director 3

qsub $dir_main/q_run

exit 0
```

Typical Inversion Optimization Parameter File

```
# inv_fit.par
#
# Parameter file for inversion optimizations to fit the response to fit_values
#
# Seth McNeill
# 2008 April 22
# 2008 June 18 uncommented so write_spinev_inv.m would work

spin_rate 10000
crystal_file rep700.dat
gamma_angles 50
start_operator   I1z
detect_operator  I1z
proton_frequency 750e6

#pre_pul_dly 0
#post_pul_dly 2

ss_fname ../adam.ss # spin system file
in_fname inv_adam.spv # the name of the input file to call, should be same as par(name)
pulse_fname inv_opt.pp
write_spv_fcn write_spinev_inv
npulses [4 5 6] # number of pulses in group optimized
freq_rng [-50 1 50]
power    60 # kHz, power for 1-pulse at perfect homogeneity
pwr_rng [60 1 60]
fit_values [ones(50,1); -1; ones(50,1)]
name_base fit1

min_convergence 0.0001 # fminsearch minimum convergence
simulator_fname spinev
sim_opts -split4
fcn spinev_eval_inv_dly

nopts 200 # number of optimizations
max_cpu_time 166  # hours

#html_dir html/
#html_fname conv.html
data_dir data
#stop_fname_base stopopt.ctl
#stdout_fname spinev_out.txt
```

Compiling Matlab

A typical script for compiling Matlab code is as follows. All the common files I use in Matlab are stored in /home/mcnese/bin/matlab on the HPC cluster where this compiling script was run.

```
function compile_inv()
% function compile_inv()
%
% This function compiles the code for the hpc cluster for the inversion optimizations
%
% Seth McNeill
```

```
% 2008 June 19

start_time = clock;
mcc -R -nodisplay -m -d bin -I . spinev_opt_inv2.m -a /home/mcnese/bin/matlab/*.m

!rm bin/*.c

total_time = etime(clock, start_time);
disp(sprintf('Total compile time was: %.1f s', total_time));
```

Inversion Optimization Matlab Script

```
function spinev_opt_inv2(par_fname, npulses_in)
% function spinev_opt_inv2(par_fname, npulses_in)
%
% Optimizes inversion pulse using spinevolution.
% Pass in the name of the par file to use. If npulses_in is specified it
% ignores the settings.npulses and does the number of pulses passed in.
% 2008 April 22
% 2008 June 18 added spin system and writing function specification options

% Seth McNeill

% need to add stop after setttings.max_cpu_time

cont_pwr_rng = [30 2 70]; % range for power for contour plot
cont_freq_rng = [-100 5 100]; % range for frequency for contour plot

settings = load_par(par_fname);
settings = load_ss(settings.ss_fname, settings);

% check to see if npulses_in has been specified
if(nargin > 1)
    if(ischar(npulses_in))
        npulses = str2num(npulses_in);
    else
        npulses = npulses_in;
    end
else
    npulses = settings.npulses;
end

% reset the random number generator state
cc = clock;
rand('twister', sum(1000*cc(6))); % Reset random number generator
% rand('state', sum(100*clock)); % Reset for old version of Matlab

for jj = 1:length(npulses)
    clear len_phase fval
    for ii = 1:settings.nopts
        start_time = clock;
        start_datestr = [mfilename ' started: ' datestr(now)];
        settings.start_time = start_time;
        settings.start_datestr = start_datestr;
        if(~exist(settings.data_dir, 'dir')) % create data_dir if it doesn't exist
            mkdir(settings.data_dir)
        end

        feval(settings.write_spv_fcn, settings); % write the spv file (main spinev file)
        init_coeffs = 360*rand(1,2*npulses(jj));

        %%%% Do the optimization %%%%
        options = optimset('Display','iter',... % outfcn defined below
            'TolFun', settings.min_convergence, 'TolX', settings.min_convergence, ...
            'TolCon', settings.min_convergence, 'Diagnostics', 'on');
        [len_phase(ii,:), fval(ii,1), exitflag, output] = fminsearch(@(x) ...
            feval(settings.fcn, x, settings), init_coeffs, options)
        output.message
        settings.len_phase = len_phase(ii,:);
        plengthsd = len_phase(ii,1:2:size(len_phase, 2)); % pulse lengths in degrees
```

```
    settings.plengths = plengthsd/360*(1/settings.power)*1000; % pulse lengths (microseconds)
    settings.pphase = len_phase(ii,2:2:size(len_phase, 2)); % pulse phases
    settings.hom_fval = fval(ii, 1);
    settings.output = output;
    end_time = clock;
    settings.total_time = etime(end_time, start_time);

    %%%% Save the data %%%%
    if(isfield(settings, 'name_base'))
        fname_base = sprintf('%s_%.4f_%.0f_%.0f%02.0f%02.0f', ...
            settings.name_base, fval(ii), npulses(jj), start_time(1), ...
            start_time(2), start_time(3));
    else
        fname_base = sprintf('inv_%.4f_%.0f_%.0f%02.0f%02.0f', ...
            fval(ii), npulses(jj), start_time(1), start_time(2), start_time(3));
    end
    % check for duplicate file names (should be rare)
    if(exist(fullfile(settings.data_dir, [fname_base '.mat']), 'file'))
        file_num = 2;
        filename_base_test = [fname_base '_' num2str(file_num)];
        while(exist(fullfile(settings.data_dir, [filename_base_test '.mat']), 'file'))
            file_num = file_num + 1;
            filename_base_test = [fname_base '_' num2str(file_num)];
        end
        fname_base = filename_base_test;
    end
    settings.fname = fullfile(pwd, settings.data_dir, [fname_base '.mat']);
    save(settings.fname, 'settings')
end

[mn, mnind] = min(fval);
% Reduce the step size and redo the simulation for the best optimization
freq_rng = settings.freq_rng;
pwr_rng = settings.pwr_rng;
if(isfield(settings, 'fit_values'))
    fit_values_saved = settings.fit_values;
    settings = rmfield(settings, 'fit_values');
end
settings.freq_rng = cont_freq_rng;
settings.pwr_rng = cont_pwr_rng;
[sum_min, d] = feval(settings.fcn, len_phase(mnind,:), settings);

fig_h = figure('name',['Contour ' num2str(npulses(jj)) ' pulses']);
orient landscape
hold all
grid on
[C, hcon] = contourf(settings.pwr_rng(1):settings.pwr_rng(2):settings.pwr_rng(3), ...
    settings.freq_rng(1):settings.freq_rng(2):settings.freq_rng(3), ...
    d(:,2:end), -1:0.25:1);
set(hcon,'linecolor', 'none')
colorbar
colormap('gray')
set(gca, 'fontsize', 20)
xlabel('Power (kHz)', 'fontsize', 24)
ylabel('Frequency Offset (kHz)', 'fontsize', 24)
zlabel('Iz magnitude')
title({'\fontsize{20}Optimized Inversion Composite pulse', ...
    ['\fontsize{14}Number of Pulses: ' num2str(npulses(jj)) ', Fval = ' num2str(mn)], ...
    ['\fontsize{10}Pulse Lengths: ' ...
    num2str(len_phase(mnind,1:2:size(len_phase, 2)), '%.1f ')], ...
    ['Pulse Phases: ' num2str(len_phase(mnind,2:2:size(len_phase, 2)), '%.1f ')]})
pause(0.1) % to make the figure show

% save the figure
if(isfield(settings, 'name_base'))
    fig_fname_base = sprintf('%s_%.4f_%.0f_%.0f%02.0f%02.0f', settings.name_base,...
        mn, npulses(jj), start_time(1), start_time(2), start_time(3));
else
    fig_fname_base = sprintf('inv_%.4f_%.0f_%.0f%02.0f%02.0f', mn, ...
        npulses(jj), start_time(1), start_time(2), start_time(3));
end
```

```
    % check for duplicate file names (should be rare)
    if(exist(fullfile(settings.data_dir, [fig_fname_base '.ps']), 'file'))
        file_num = 2;
        filename_base_test = [fig_fname_base '_' num2str(file_num)];
        while(exist(fullfile(settings.data_dir, [filename_base_test '.ps']), 'file'))
            file_num = file_num + 1;
            filename_base_test = [fig_fname_base '_' num2str(file_num)];
        end
        fig_fname_base = filename_base_test;
    end
    print('-dpsc', '-append', fullfile(pwd, settings.data_dir, [fig_fname_base '.ps']));

    % reset the step size back to original
    settings.freq_rng = freq_rng;
    settings.pwr_rng = pwr_rng;
    if(exist('fit_values_saved', 'var') && ~isempty(fit_values_saved))
        settings.fit_values = fit_values_saved;
        clear fit_values_saved
    end
    if(isdeployed)
        close(fig_h)
    end
end
% keyboard
```

Maximum Filename Check

This script looks for the maximum result under the data subdirectory of the current directory.

```
#!/bin/bash
# check.sh
# simple check of current optimizations
# 2008 August 19 SAM

pd=`pwd`
nview_base=5 # number of files in each dir to view
data_dirs=`ls -d $1*`
echo ============================================================

topt=0
for d in $data_dirs
do
   cd $d
   nview=$nview_base
   dochk=`ls -1 data | wc -l`
   if [ $dochk -gt 0 ]
   then
      nopts=`ls -lA data/*.mat | wc -l`
      echo "$d: $nopts optimizations"
      if [ $nopts -gt 0 ]
      then
         if [ $nopts -lt $nview ]
         then
            nview=$nopts
         fi
         ls -lA data/ | tail -n $nview
      fi
   else
      nopts=0
      echo "$d: $nopts optimizations"
   fi
   topts=$(($topts+$nopts))
   cd $pd
done

echo
```

```
date
echo "Total Optimizations: $topts"
echo

exit 1
```

Minimum Filename Check

This script checks the data filenames in the data subdirectories of the directories that match the wildcard passed in to display the names of the files with the lowest results.

```
#!/bin/bash
# mincheck.sh
# simple check of current optimizations
# 2008 August 19 SAM

pd=`pwd`
nview_base=5 # number of files in each dir to view
data_dirs=`ls -d $1*`
echo ============================================================

topt=0
for d in $data_dirs
do
   cd $d
   nview=$nview_base
   dochk=`ls -1 data | wc -l`
   if [ $dochk -gt 0 ]
   then
      nopts=`ls -lA data/*.mat | wc -l`
      echo "$d: $nopts optimizations"
      if [ $nopts -gt 0 ]
      then
         if [ $nopts -lt $nview ]
         then
            nview=$nopts
         fi
         ls -lA data/ | head -n $((${nview} + 1)) | tail -n $nview
      fi
   else
      nopts=0
      echo "$d: $nopts optimizations"
   fi
   topts=$(($topts+$nopts))
   cd $pd
done

echo
date
echo "Total Optimizations: $topts"
echo

exit 1
```

Refocusing Optimization

Refocusing Matlab Script

```
function spinev_opt_refocus(par_fname, npulses_in)
% function spinev_opt_refocus(par_fname, npulses_in)
%
% Optimizes refocusing pulse using spinevolution.
% Pass in the name of the par file to use. If npulses_in is specified it
```

```
% ignores the settings.npulses and does the number of pulses passed in.
% 2008 August 12

% Seth McNeill

% need to add stop after setttings.max_cpu_time

cont_pwr_rng = [30 2 70]; % range for power for contour plot
cont_freq_rng = [-100 5 100]; % range for frequency for contour plot

settings = load_par(par_fname);
settings = load_ss(settings.ss_fname, settings);

% check to see if npulses_in has been specified
if(nargin > 1)
    if(ischar(npulses_in))
        npulses = str2num(npulses_in);
    else
        npulses = npulses_in;
    end
else
    npulses = settings.npulses;
end

% reset the random number generator state
cc = clock;
rand('twister', sum(1000*cc(6))); % Reset random number generator
% rand('state', sum(100*clock)); % Reset for old version of Matlab

for jj = 1:length(npulses)
    clear len_phase fval
    for ii = 1:settings.nopts
        start_time = clock;
        start_datestr = [mfilename ' started: ' datestr(now)];
        settings.start_time = start_time;
        settings.start_datestr = start_datestr;
        if(~exist(settings.data_dir, 'dir')) % create data_dir if it doesn't exist
            mkdir(settings.data_dir)
        end

        feval(settings.write_spv_fcn, settings); % write the spv file (main spinev file)
        init_coeffs = 360*rand(1,2*npulses(jj));

        %%%% Do the optimization %%%%
        options = optimset('Display','iter',... % outfcn defined below
            'TolFun', settings.min_convergence, 'TolX', settings.min_convergence, ...
            'TolCon', settings.min_convergence, 'Diagnostics', 'on');
        [len_phase(ii,:), fval(ii,1), exitflag, output] = fminsearch(@(x) ...
            feval(settings.fcn, x, settings), init_coeffs, options)
        output.message
        settings.len_phase = len_phase(ii,:);
        plengthsd = len_phase(ii,1:2:size(len_phase, 2)); % pulse lengths in degrees
        settings.plengths = plengthsd/360*(1/settings.power)*1000; % pulse lengths (microseconds)
        settings.pphase = len_phase(ii,2:2:size(len_phase, 2)); % pulse phases
        settings.hom_fval = fval(ii, 1);
        setings.output = output;
        end_time = clock;
        settings.total_time = etime(end_time, start_time);

        %%%% Save the data %%%%
        if(isfield(settings, 'name_base'))
            fname_base = sprintf('%s_%.4f_%.0f_%.0f%02.0f%02.0f', ...
                settings.name_base, fval(ii), npulses(jj), start_time(1), ...
                start_time(2), start_time(3));
        else
            fname_base = sprintf('inv_%.4f_%.0f_%.0f%02.0f%02.0f', ...
                fval(ii), npulses(jj), start_time(1), start_time(2), start_time(3));
        end
        % check for duplicate file names (should be rare)
        if(exist(fullfile(settings.data_dir, [fname_base '.mat']), 'file'))
            file_num = 2;
```

```
            filename_base_test = [fname_base '_' num2str(file_num)];
            while(exist(fullfile(settings.data_dir, [filename_base_test '.mat']), 'file'))
                file_num = file_num + 1;
                filename_base_test = [fname_base '_' num2str(file_num)];
            end
            fname_base = filename_base_test;
        end
        settings.fname = fullfile(pwd, settings.data_dir, [fname_base '.mat']);
        save(settings.fname, 'settings')
    end

    [mn, mnind] = min(fval);
    % Reduce the step size and redo the simulation for the best optimization
    freq_rng = settings.freq_rng;
    pwr_rng = settings.pwr_rng;
    settings.freq_rng = cont_freq_rng;
    settings.pwr_rng = cont_pwr_rng;
    [sum_min, d] = feval(settings.fcn, len_phase(mnind,:), settings);

    fig_h = figure('name',['Contour ' num2str(npulses(jj)) ' pulses']);
    orient landscape
    hold all
    grid on
    [C, hcon] = contourf(settings.pwr_rng(1):settings.pwr_rng(2):settings.pwr_rng(3), ...
        settings.freq_rng(1):settings.freq_rng(2):settings.freq_rng(3), ...
        d(:,2:end), -1:0.25:1);
    set(hcon,'linecolor', 'none')
    colorbar
    colormap('gray')
    set(gca, 'fontsize', 20)
    xlabel('Power (kHz)', 'fontsize', 24)
    ylabel('Frequency Offset (kHz)', 'fontsize', 24)
    zlabel('Iz magnitude')
    title({'\fontsize{20}Optimized Inversion Composite pulse', ...
        ['\fontsize{14}Number of Pulses: ' num2str(npulses(jj)) ', Fval = ' num2str(mn)], ...
        ['\fontsize{10}Pulse Lengths: ' ...
        num2str(len_phase(mnind,1:2:size(len_phase, 2)), '%.1f ')], ...
        ['Pulse Phases: ' num2str(len_phase(mnind,2:2:size(len_phase, 2)), '%.1f ')]})
    pause(0.1) % to make the figure show

    % save the figure
    fig_fname_base = sprintf('inv_%.4f_%.0f_%.0f%02.0f%02.0f', mn, npulses(jj), start_time(1),
...
        start_time(2), start_time(3));
    % check for duplicate file names (should be rare)
    if(exist(fullfile(settings.data_dir, [fig_fname_base '.ps']), 'file'))
        file_num = 2;
        filename_base_test = [fig_fname_base '_' num2str(file_num)];
        while(exist(fullfile(settings.data_dir, [filename_base_test '.ps']), 'file'))
            file_num = file_num + 1;
            filename_base_test = [fig_fname_base '_' num2str(file_num)];
        end
        fig_fname_base = filename_base_test;
    end
    print('-dpsc', '-append', fullfile(pwd, settings.data_dir, [fig_fname_base '.ps']));

    % reset the step size back to original
    settings.freq_rng = freq_rng;
    settings.pwr_rng = pwr_rng;
    if(isdeployed)
        close(fig_h)
    end
end
% keyboard
```

Refocusing Parameter File

```
# refocus.par
#
# Parameter file for inversion optimizations
```

```
#
# Seth McNeill
# 2008 August 12

spin_rate 10000
crystal_file rep700.dat
gamma_angles 50
start_operator   I1x
detect_operator  I1x
proton_frequency 750e6

#pre_pul_dly 0.5
#post_pul_dly 1.5

ss_fname ../adam.ss # spin system file
in_fname refocus.spv # the name of the input file to call, should be same as par(name)
pulse_fname refocus_opt.pp
write_spv_fcn write_spinev_refocus
npulses [4 5 6] # number of pulses in group optimized
freq_rng [-100 2 100]
power    50 # kHz, power for 1-pulse at perfect homogeneity
pwr_rng [50 1 50]
opt_mult -1 # multiplier for optimization value, set to -1 for maximization
name_base ref2w

min_convergence 0.0001 # fminsearch minimum convergence
#min_convergence 100 # fminsearch minimum convergence
simulator_fname spinev
sim_opts -split4
fcn spinev_eval_inv_dly

nopts 200 # number of optimizations
max_cpu_time 166  # hours

#html_dir html/
#html_fname conv.html
data_dir data
#stop_fname_base stopopt.ctl
#stdout_fname spinev_out.txt
```

APPENDIX C
PULSE SEQUENCES

Nutation Figure 2-9

zg_2d.sam

```
# 1 "/opt/pv4.0/exp/stan/nmr/lists/pp/zg_2d.sam"
;zg_2d.sam
;avance-version (02/05/31)
;1D sequence


# 1 "/opt/pv4.0/exp/stan/nmr/lists/pp/Avance.incl" 1
;Avance2.incl
;   for 1
;
;avance-version (03/02/17)

;$Id: Avance2.incl,v 1.10 2003/02/25 14:46:08 ber Exp $
# 6 "/opt/pv4.0/exp/stan/nmr/lists/pp/zg_2d.sam" 2


1 ze
  0.1u rpu1
2 d1           ; recycle delay
  p1 ph1       ; observe pulse
  go=2 ph31
  30m wr #0 if #0  zd
  "p1 = p1 + p11"
  lo to 2 times td1  ; number of spectra to take
  ;30m mc #0 to 2 F0(zd)
  ;30m wr #0 if #0 zd
  ;p1 = p1 + p11
  ;lo to 2 times td1 ; # spectra to take
exit


ph1=0 2 2 0 1 3 3 1
ph31=0 2 2 0 1 3 3 1


;pl1 : f1 channel - power level for pulse (default)
;p1 : f1 channel -  high power pulse
;p11 : f1 channel - increment added to p1
;d1 : relaxation delay; 1-5 * T1
;NS : 1 * n, total number of scans: NS * TD0
;td1 : # spectra to take


;$Id: zg,v 1.7 2002/06/12 09:05:19 ber Exp $
```

zgig_2d.sam

```
# 1 "/opt/pv4.0/exp/stan/nmr/lists/pp/zgig_2d.sam"
;zg_2d.sam
;avance-version (02/05/31)
;1D sequence


# 1 "/opt/pv4.0/exp/stan/nmr/lists/pp/Avance.incl" 1
;Avance2.incl
;   for 1
;
;avance-version (03/02/17)
```

```
;$Id: Avance2.incl,v 1.10 2003/02/25 14:46:08 ber Exp $
# 6 "/opt/pv4.0/exp/stan/nmr/lists/pp/zgig_2d.sam" 2



1 ze
  0.1u rpu1
  d11 pl12:f2
2 30m do:f2
  d1           ; recycle delay
  p1 ph1       ; observe pulse
  go=2 ph31 cw:f2
  30m  do:f2 wr #0 if #0  zd
  "p1 = p1 + p11"
  lo to 2 times td1  ; number of spectra to take
  ;30m mc #0 to 2 F0(zd)
  ;30m wr #0 if #0 zd
  ;p1 = p1 + p11
  ;lo to 2 times td1 ; # spectra to take
exit


ph1=0 2 2 0 1 3 3 1
ph31=0 2 2 0 1 3 3 1


;pl1 : f1 channel - power level for pulse (default)
;p1 : f1 channel -  high power pulse
;p11 : f1 channel - increment added to p1
;d1 : relaxation delay; 1-5 * T1
;NS : 1 * n, total number of scans: NS * TD0
;td1 : # spectra to take


;$Id: zg,v 1.7 2002/06/12 09:05:19 ber Exp $
```

TmDOTP Heating Pulse Sequence Figure 2-11 zgps_p18.sam

```
# 1 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/zgps_p18.sam"
;zgps_p18.sam
;avance-version (02/05/31)
;1D sequence with presaturation
;using regular pulse for off-resonance presaturation


# 1 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/Avance.incl" 1
;Avance2.incl
;   for 1
;
;avance-version (06/02/20)
;
;$CLASS=HighRes Incl
;$COMMENT=

;$Id: Avance.incl,v 1.2 2006/09/13 12:12:04 chjo Exp $
# 7 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/zgps_p18.sam" 2


define list<frequency> foffs = <$FQ1LIST>

;"d12=20u"


# 1 "mc_line 14 file /opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/zgps_p18.sam expanding
definition part of mc command before ze"
define delay MCWRK
define delay MCREST
```

```
"MCWRK = 0.333333*30m"
"MCREST = d1 - 30m"
    dccorr
# 14 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/zgps_p18.sam"
1 ze
# 1 "mc_line 14 file /opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/zgps_p18.sam expanding
definition of mc command after ze"
# 15 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/zgps_p18.sam"
# 1 "mc_line 15 file /opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/zgps_p18.sam expanding start
label for mc command"
2 MCWRK  * 2
LBLF0, MCWRK pl2:f1 foffs:f1
  MCREST
# 16 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/zgps_p18.sam"
  (p18 ph29):f1
  d12 pl1:f1 fq=0:f1
  (p1 ph1):f1
  go=2 ph31
# 1 "mc_line 20 file /opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/zgps_p18.sam expanding mc
command in line"
  MCWRK wr #0
  MCWRK zd
  lo to LBLF0 times td0

  MCWRK
# 21 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/zgps_p18.sam"
exit


ph1=0 2 2 0 1 3 3 1
ph29=0
ph31=0 2 2 0 1 3 3 1


;pl1 : f1 channel - Pulse power level (default)
;pl2 : f1 channel - warming pulse power level
;sp6: f1 channel - shaped pulse for presaturation
;p1 : f1 channel -  90 degree high power pulse
;p18: f1 channel - presaturation pulse length
;d12: delay for power switching                     [20 usec]
;l6: p18 * l6 = total duration of presaturation
;NS: 1 * n, total number of scans: NS * TD0

;use 100msec pulse of square shape defined by 1000 points



;$Id: zgps,v 1.8 2002/06/12 09:05:24 ber Exp $
```

Inversion Pulse Sequences Figure 4-11

inv_ctl.sam

```
# 1 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/inv_ctl.sam"
;inv_ctl.sam
;inversion 180 varied across frequency offsets with a control pulse at the
;beginning of the sequence.
;2D sequence
;2008 July 04

;$COMMENT=Standard 180 inversion pulse
;$CLASS=Solids
;$DIM=2D
;$TYPE=inversion
;$OWNER=SethMcNeill

# 1 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/Avance.incl" 1
```

```
;Avance2.incl
;   for 1
;
;avance-version (06/02/20)
;
;$CLASS=HighRes Incl
;$COMMENT=


;$Id: Avance.incl,v 1.2 2006/09/13 12:12:04 chjo Exp $
# 13 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/inv_ctl.sam" 2

;define list<frequency> foffs = <inv_offs41>
define list<frequency> foffs = <$FQ1LIST>
"p2=p29*2"
"p30=p1*2"
define loopcounter nfoffs
"nfoffs=td1-1"

# 1 "mc_line 21 file /opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/inv_ctl.sam dc-measurement
inserted automatically"
    dccorr
# 21 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/inv_ctl.sam"
1 ze
  d11 pl12:f2
2 30m do:f2
  d1
  p30 ph30
  p1 ph1
  go=2 ph31 cw:f2
  30m  do:f2 wr #0 if #0  zd
  1m

  d11 pl12:f2
3 30m do:f2
  d1 foffs:f1     ; recycle delay and reset frequency
  p2 ph2          ; inversion pulse
  p1 ph1 fq=0:f1  ; 90 pulse at 0 Hz offset
  go=3 ph31 cw:f2
  30m  do:f2 wr #0 if #0  zd
  foffs.inc
  lo to 3 times nfoffs  ; number of spectra to take
exit


ph1=0 0 0 0 1 1 1 1 2 2 2 2 3 3 3 3
ph2=0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3
ph30=0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3
ph31=0 0 0 0 1 1 1 1 2 2 2 2 3 3 3 3


;pl1 : f1 channel - power level for pulse (default)
;pl2 : not used (set to 120)
;pl12 : 1H decoupling
;p1 : control 90 degree pulse at pl1
;p2 : 180 degree pulse at pl1
;p30 : control inversion 180 degree pulse at pl1
;p29 : experimental 90 degree pulse at pl1
;d1 : relaxation delay; 1-5 * T1
;NS : 1 * n, total number of scans: NS * TD0
;td1 : # spectra to take


;$Id: zg,v 1.7 2002/06/12 09:05:19 ber Exp $
```

inv_38111_wctl.sam

```
# 1 "/opt/topspin/exp/stan/nmr/lists/pp/user/inv_38111_wctl.sam"
;inv_38111_wctl.sam
;Composite pulse inversion varied across frequency offsets
```

```
; implements 38x, 111-x, 159x, 250-x
;2D sequence
;2008 May 25
; Added initial 180 to beginning 2008 July 7

# 1 "/opt/topspin/exp/stan/nmr/lists/pp/user/Avance.incl" 1
;Avance2.incl
;   for 1
;
;avance-version (06/02/20)
;
;$CLASS=HighRes Incl
;$COMMENT=

;$Id: Avance.incl,v 1.2 2006/09/13 12:12:04 chjo Exp $
# 8 "/opt/topspin/exp/stan/nmr/lists/pp/user/inv_38111_wctl.sam" 2

define list<frequency> foffs = <$FQ1LIST>
"p2=p1*38/90"
"p3=p1*111/90"
"p4=p1*159/90"
"p5=p1*250/90"
"p10=p1*2"

;$COMMENT=Composite inversion pulse sequence from Claridge pp 344
;$CLASS=Solids
;$DIM=2D
;$TYPE=inversion
;$SUBTYPE=Composite
;$OWNER=SethMcNeill

define loopcounter nfoffs
"nfoffs=td1-1"

# 1 "mc_line 26 file /opt/topspin/exp/stan/nmr/lists/pp/user/inv_38111_wctl.sam dc-measurement
inserted automatically"
    dccorr
# 26 "/opt/topspin/exp/stan/nmr/lists/pp/user/inv_38111_wctl.sam"
1 ze
  d11 pl12:f2
2 30m do:f2
  d1
  p10 ph1
  p1 ph30
  go=2 ph31 cw:f2
  30m  do:f2 wr #0 if #0  zd
  1m

  d11 pl12:f2
3 30m do:f2
  d1 foffs:f1     ; recycle delay and reset frequency
  p2 ph1          ; inversion pulses
  p3 ph2          ; inversion pulses
  p4 ph1          ; inversion pulses
  p5 ph2          ; inversion pulses
  p1 ph30 fq=0:f1  ; 90 pulse at 0 Hz offset
  go=3 ph31 cw:f2
  30m  do:f2 wr #0 if #0  zd
  foffs.inc
  lo to 3 times nfoffs  ; number of spectra to take
exit

ph1 =  0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3
ph2 =  2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1
ph30 = 0 0 0 0 1 1 1 1 2 2 2 2 3 3 3 3
ph31 = 0 0 0 0 1 1 1 1 2 2 2 2 3 3 3 3


;pl1 : f1 channel - power level for pulse (default)
;p1 : 90 degree pulse at pl1
;p2 : 38 degree pulse at pl1
```

```
;p3 : 111 degree pulse at pl1
;p4 : 159 degree pulse at pl1
;p5 : 250 degree pulse at pl1
;p10 : 180 degree pulse at pl1
;d1 : relaxation delay; 1-5 * T1
;NS : 1 * n, total number of scans: NS * TD0
;td1 : # spectra to take
```

inv0.911.sam

```
# 1 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/inv0.9511.sam"
; inv0.9511.sam
; Automatically created by /Users/mcnese/Seth/bin/write_bruker750_inv
; Created at 2008 December 31, 14:17:19
; Composite Inversion sequence from:
; /scratch/ufhpc/mcnese/inversion/20081204_7p_2/data/inv_-0.9511_7_20081207.mat
; File at:
; /Users/mcnese/Seth/PulseOptimization/spinevOpt/inversion/HPC/20081204_7p_2/data/inv_-
0.9511_7_20081207.mat
; Pulse lengths implemented (in degrees) are:
; 58.95 129.91 79.23 169.55 281.98 73.02 262.48
; Phases implemented in this file in degrees are:
; 77.56 77.42 254.88 74.96 253.65 72.38 251.08

;$COMMENT=Composite inversion pulse sequence optimized using spinev
;$CLASS=Solids
;$DIM=2D
;$TYPE=inversion
;$SUBTYPE=OptimizedComposite
;$OWNER=SethMcNeill

# 1 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/Avance.incl" 1
;Avance2.incl
;   for 1
;
;avance-version (06/02/20)
;
;$CLASS=HighRes Incl
;$COMMENT=

;$Id: Avance.incl,v 1.2 2006/09/13 12:12:04 chjo Exp $
# 20 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/inv0.9511.sam" 2

define list<frequency> foffs = <$FQ1LIST>
"p2=p29*59/90"
"p3=p29*130/90"
"p4=p29*79/90"
"p5=p29*170/90"
"p6=p29*282/90"
"p7=p29*73/90"
"p8=p29*262/90"
"p30=p1*2"

define loopcounter nfoffs
"nfoffs=td1-1"

# 1 "mc_line 34 file /opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/inv0.9511.sam dc-measurement
inserted automatically"
    dccorr
# 34 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/inv0.9511.sam"
1 ze
  d11 pl12:f2
2 30m do:f2  ; CONTROL EXPERIMENT
  d1         ; recycle delay
  p30 ph30   ; control inversion 180
  p1 ph1     ; 90 degree flip into X-Y plane
  go=2 ph31 cw:f2
  30m  do:f2 wr #0 if #0  zd
  1m do:f2
```

```
  d11 pl12:f2
3 30m do:f2
  d1 foffs:f1     ; recycle delay and reset frequency
  p2 ph2          ; inversion pulses
  p3 ph3          ; inversion pulses
  p4 ph4          ; inversion pulses
  p5 ph5          ; inversion pulses
  p6 ph6          ; inversion pulses
  p7 ph7          ; inversion pulses
  p8 ph8          ; inversion pulses
  p1 ph1 fq=0:f1  ; 90 pulse at 0 Hz offset
  go=3 ph31 cw:f2
  30m  do:f2 wr #0 if #0  zd
  foffs.inc
  lo to 3 times nfoffs  ; number of spectra to take
exit

ph2 = (65536) { 14119 30503 46887 63271 14119 30503 46887 63271 14119 30503 46887 63271 14119
30503 46887 63271 } ; phases for p2
ph3 = (65536) { 14095 30479 46863 63247 14095 30479 46863 63247 14095 30479 46863 63247 14095
30479 46863 63247 } ; phases for p3
ph4 = (65536) { 46399 62783 13631 30015 46399 62783 13631 30015 46399 62783 13631 30015 46399
62783 13631 30015 } ; phases for p4
ph5 = (65536) { 13646 30030 46414 62798 13646 30030 46414 62798 13646 30030 46414 62798 13646
30030 46414 62798 } ; phases for p5
ph6 = (65536) { 46176 62560 13408 29792 46176 62560 13408 29792 46176 62560 13408 29792 46176
62560 13408 29792 } ; phases for p6
ph7 = (65536) { 13176 29560 45944 62328 13176 29560 45944 62328 13176 29560 45944 62328 13176
29560 45944 62328 } ; phases for p7
ph8 = (65536) { 45707 62091 12939 29323 45707 62091 12939 29323 45707 62091 12939 29323 45707
62091 12939 29323 } ; phases for p8
ph30 = 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3
ph1  = 0 0 0 0 1 1 1 1 2 2 2 2 3 3 3 3
ph31 = 0 0 0 0 1 1 1 1 2 2 2 2 3 3 3 3

;pl1 : f1 channel - power level for pulse (default)
;pl2 : not used (set to 120)
;pl12 : 1H decoupling
;p1 : control 90 degree pulse at pl1
;p29 : experimental 90 degree pulse at pl1
;p30 : control 180 degree pulse at pl1
;d1 : relaxation delay; 1-5 * T1
;NS : 1 * n, total number of scans: NS * TD0
;td1 : # spectra to take
```

Refocusing Pulse Sequences Figures 4-20 to 4-22

refc_180.sam

```
# 1 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/refc_180.sam"
; ref180.sam
; Automatically created by /Users/mcnese/Seth/bin/write_bruker750_refocus
; Created at 2009 January 01, 21:20:50
; Composite refocusing sequence from:
; 180.mat
; File at:
; ~/Seth/PulseOptimization/spinevOpt/refocusing/180.mat
; Pulse lengths implemented (in degrees) are:
; 180.00
; Phases implemented in this file in degrees are:
; -0.00

;$COMMENT=Composite refocusing pulse sequence optimized using spinev
;$CLASS=Solids
;$DIM=2D
;$TYPE=refocusing
;$SUBTYPE=OptimizedComposite
;$OWNER=SethMcNeill
```

```
# 1 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/Avance.incl" 1
;Avance2.incl
;   for 1
;
;avance-version (06/02/20)
;
;$CLASS=HighRes Incl
;$COMMENT=

;$Id: Avance.incl,v 1.2 2006/09/13 12:12:04 chjo Exp $
# 20 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/refc_180.sam" 2

define list<frequency> foffs = <$FQ1LIST>
"p2=p29*180/90"
"p30=p1*2"

define loopcounter nfoffs
"nfoffs=td1-1"
define delay tau
"tau = (1s/cnst31)-(p2)/2"
define delay tau30
"tau30 = (1s/cnst31)-p30/2"

# 1 "mc_line 32 file /opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/refc_180.sam dc-measurement
inserted automatically"
    dccorr
# 32 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/refc_180.sam"
1 ze
  d11 pl12:f2
2 30m do:f2  ; CONTROL EXPERIMENT
  d1         ; recycle delay
  p1 ph1     ; 90 pulse
  tau30      ; first tau
  p30 ph30   ; refocusing pulse
  tau30      ; second tau
  go=2 ph31 cw:f2 ; acquisition and decoupling
  30m  do:f2 wr #0 if #0  zd

  d1 do:f2   ; delay between experiments

  d11 pl12:f2
3 30m do:f2       ; TEST EXPERIMENT
  d1 fq=0:f1      ; recycle delay
  p1 ph1          ; 90 pulse at 0 Hz offset
  tau foffs:f1    ; set frequency offset, first tau
  p2 ph2          ; refocusing pulses
  tau             ; second tau
  go=3 ph31 cw:f2 ; acquisition and offset
  30m  do:f2 wr #0 if #0  zd
  foffs.inc
  lo to 3 times nfoffs  ; number of spectra to take
exit


ph2= (65536) { 49152 16384 0 32768 16384 49152 32768 0 } ; phases for p2
ph1  = 0 0 1 1 2 2 3 3
ph30 = 3 1 0 2 1 3 2 0
ph31 = ph1

;pl1 : f1 channel - power level for pulse (default)
;pl2 : not used (set to 120)
;pl12 : 1H decoupling
;p1 : control 90 degree pulse at pl1
;p2 : experimental refocusing 180 degree pulse at pl1
;p30 : control refocusing 180 degree pulse at pl1
;p29 : experimental 90 degree pulse at pl1
;d1 : relaxation delay; 1-5 * T1
;NS : 1 * n, total number of scans: NS * TD0
;td1 : # spectra to take
;cnst31 : MAS rate
```

refc_151342.sam

```
# 1 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/refc_151342.sam"
; refc_151342.sam
; Automatically created by /Users/mcnese/Seth/bin/write_bruker750_refocus
; Created at 2009 January 01, 16:05:10
; Composite refocusing sequence from:
; Claridge pp. 344 (MAS1208)
; File at:
; 151342.mat
; Pulse lengths implemented (in degrees) are:
; 151.00 342.00 180.00 342.00 151.00
; Phases implemented in this file in degrees are:
; 247.00 182.00 320.00 182.00 247.00

;$COMMENT=Composite refocusing pulse sequence optimized using spinev
;$CLASS=Solids
;$DIM=2D
;$TYPE=refocusing
;$SUBTYPE=OptimizedComposite
;$OWNER=SethMcNeill

# 1 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/Avance.incl" 1
;Avance2.incl
;   for 1
;
;avance-version (06/02/20)
;
;$CLASS=HighRes Incl
;$COMMENT=

;$Id: Avance.incl,v 1.2 2006/09/13 12:12:04 chjo Exp $
# 20 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/refc_151342.sam" 2

define list<frequency> foffs = <$FQ1LIST>
"p2=p29*151/90"
"p3=p29*342/90"
"p4=p29*180/90"
"p5=p29*342/90"
"p6=p29*151/90"
"p30=p1*2"

define loopcounter nfoffs
"nfoffs=td1-1"
define delay tau
"tau = (1s/cnst31)-(p2+p3+p4+p5+p6)/2"
define delay tau30
"tau30 = (1s/cnst31)-p30/2"

# 1 "mc_line 36 file /opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/refc_151342.sam dc-measurement
inserted automatically"
    dccorr
# 36 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/refc_151342.sam"
1 ze
  d11 pl12:f2
2 30m do:f2  ; CONTROL EXPERIMENT
  d1         ; recycle delay
  p1 ph1     ; 90 pulse
  tau30      ; first tau
  p30 ph30   ; refocusing pulse
  tau30      ; second tau
  go=2 ph31 cw:f2 ; acquisition and decoupling
  30m  do:f2 wr #0 if #0  zd

  d1 do:f2   ; delay between experiments

  d11 pl12:f2
3 30m do:f2       ; TEST EXPERIMENT
  d1 fq=0:f1      ; recycle delay
```

```
  p1 ph1          ; 90 pulse at 0 Hz offset
  tau foffs:f1    ; set frequency offset, first tau
  p2 ph2          ; refocusing pulses
  p3 ph3          ; refocusing pulses
  p4 ph4          ; refocusing pulses
  p5 ph5          ; refocusing pulses
  p6 ph6          ; refocusing pulses
  tau             ; second tau
  go=3 ph31 cw:f2 ; acquisition and offset
  30m  do:f2 wr #0 if #0  zd
  foffs.inc
  lo to 3 times nfoffs  ; number of spectra to take
exit


ph2= (65536) { 28581 61349 44965 12197 61349 28581 12197 44965 } ; phases for p2
ph3= (65536) { 16748 49516 33132 364 49516 16748 364 33132 } ; phases for p3
ph4= (65536) { 41870 9102 58254 25486 9102 41870 25486 58254 } ; phases for p4
ph5= (65536) { 16748 49516 33132 364 49516 16748 364 33132 } ; phases for p5
ph6= (65536) { 28581 61349 44965 12197 61349 28581 12197 44965 } ; phases for p6
ph1  = 0 0 1 1 2 2 3 3
ph30 = 3 1 0 2 1 3 2 0
ph31 = ph1

;pl1 : f1 channel - power level for pulse (default)
;pl2 : not used (set to 120)
;pl12 : 1H decoupling
;p1 : control 90 degree pulse at pl1
;p2-6 : experimental refocusing 180 degree pulse at pl1
;p30 : control refocusing 180 degree pulse at pl1
;p29 : experimental 90 degree pulse at pl1
;d1 : relaxation delay; 1-5 * T1
;NS : 1 * n, total number of scans: NS * TD0
;td1 : # spectra to take
;cnst31 : MAS rate
```

ref2w0.8262.sam

```
# 1 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/ref2w0.8262.sam"
; ref2w0.8262.sam
; Automatically created by /Users/mcnese/Seth/bin/write_bruker750_refocus
; Created at 2009 January 04, 19:22:44
; Composite refocusing sequence from:
; /scratch/ufhpc/mcnese/refocus/ref2w20090102_8p_15/data/ref2w_-0.8262_8_20090102.mat
; File at:
; /Users/mcnese/Seth/PulseOptimization/spinevOpt/refocusing/HPC/ref2w20090102_8p_15/data/ref2w_-
0.8262_8_20090102.mat
; Pulse lengths implemented (in degrees) are:
; 166.20 35.82 165.33 280.40 122.35 153.10 66.78 233.77
; Phases implemented in this file in degrees are:
; 20.26 113.99 167.39 15.81 178.95 18.98 20.90 148.95

;$COMMENT=Composite refocusing pulse sequence optimized using spinev
;$CLASS=Solids
;$DIM=2D
;$TYPE=refocusing
;$SUBTYPE=OptimizedComposite
;$OWNER=SethMcNeill

# 1 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/Avance.incl" 1
;Avance2.incl
;   for 1
;
;avance-version (06/02/20)
;
;$CLASS=HighRes Incl
;$COMMENT=

;$Id: Avance.incl,v 1.2 2006/09/13 12:12:04 chjo Exp $
# 20 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/ref2w0.8262.sam" 2
```

```
define list<frequency> foffs = <$FQ1LIST>
"p2=p29*166/90"
"p3=p29*36/90"
"p4=p29*165/90"
"p5=p29*280/90"
"p6=p29*122/90"
"p7=p29*153/90"
"p8=p29*67/90"
"p9=p29*234/90"
"p30=p1*2"

define loopcounter nfoffs
"nfoffs=td1-1"
define delay tau
"tau = (1s/cnst31)-(p2+p3+p4+p5+p6+p7+p8+p9)/2"
define delay tau30
"tau30 = (1s/cnst31)-p30/2"

# 1 "mc_line 39 file /opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/ref2w0.8262.sam dc-measurement
inserted automatically"
    dccorr
# 39 "/opt/topspin2.1.2/exp/stan/nmr/lists/pp/user/ref2w0.8262.sam"
1 ze
  d11 pl12:f2
2 30m do:f2  ; CONTROL EXPERIMENT
  d1         ; recycle delay
  p1 ph1     ; 90 pulse
  tau30      ; first tau
  p30 ph30   ; refocusing pulse
  tau30      ; second tau
  go=2 ph31 cw:f2 ; acquisition and decoupling
  30m  do:f2 wr #0 if #0  zd

  d1 do:f2   ; delay between experiments

  d11 pl12:f2
3 30m do:f2       ; TEST EXPERIMENT
  d1 fq=0:f1      ; recycle delay
  p1 ph1          ; 90 pulse at 0 Hz offset
  tau foffs:f1    ; set frequency offset, first tau
  p2 ph2          ; refocusing pulses
  p3 ph3          ; refocusing pulses
  p4 ph4          ; refocusing pulses
  p5 ph5          ; refocusing pulses
  p6 ph6          ; refocusing pulses
  p7 ph7          ; refocusing pulses
  p8 ph8          ; refocusing pulses
  p9 ph9          ; refocusing pulses
  tau             ; second tau
  go=3 ph31 cw:f2 ; acquisition and offset
  30m  do:f2 wr #0 if #0  zd
  foffs.inc
  lo to 3 times nfoffs  ; number of spectra to take
exit


ph2= (65536) { 52840 20072 3688 36456 20072 52840 36456 3688 } ; phases for p2
ph3= (65536) { 4368 37136 20752 53520 37136 4368 53520 20752 } ; phases for p3
ph4= (65536) { 14089 46857 30473 63241 46857 14089 63241 30473 } ; phases for p4
ph5= (65536) { 52030 19262 2878 35646 19262 52030 35646 2878 } ; phases for p5
ph6= (65536) { 16194 48962 32578 65346 48962 16194 65346 32578 } ; phases for p6
ph7= (65536) { 52608 19840 3456 36224 19840 52608 36224 3456 } ; phases for p7
ph8= (65536) { 52956 20188 3804 36572 20188 52956 36572 3804 } ; phases for p8
ph9= (65536) { 10732 43500 27116 59884 43500 10732 59884 27116 } ; phases for p9
ph1  = 0 0 1 1 2 2 3 3
ph30 = 3 1 0 2 1 3 2 0
ph31 = ph1

;pl1 : f1 channel - power level for pulse (default)
;pl2 : not used (set to 120)
```

```
;pl12 : 1H decoupling
;p1 : control 90 degree pulse at pl1
;p2-9 : experimental refocusing 180 degree pulse at pl1
;p30 : control refocusing 180 degree pulse at pl1
;p29 : experimental 90 degree pulse at pl1
;d1 : relaxation delay; 1-5 * T1
;NS : 1 * n, total number of scans: NS * TD0
;td1 : # spectra to take
;cnst31 : MAS rate
```

DRAWS Pulse Sequences Figure 4-28

draws2d_uneqCP.2.1.b.sam

```
# 1 "/opt/topspin/exp/stan/nmr/lists/pp/user/draws2d_uneqCP.2.1.b.sam"
; DRAWS_uneq.sam, based on cpdraws1.dg and cpdraws1.jrl
; Automatically created by write_bruker750.m
; Created at 2007 March 05, 16:28:14
; phase sequence from:
; Regular DRAWS
; Phases implemented in this file in degrees are:
; 90.00 270.00 -0.00 90.00 270.00 270.00 90.00 -0.00 270.00 90.00
;
; This program is for measuring buildup of DQ coherence via DRAWS
; using the f2 channel for the decoupler
;set:
;d1 : recycle delay
;pl1 = X power during contact
;pl2 = H power during contact
;pl11 = x power during draws, 8.5 times the spin rate, adjust for maximum signal
;pl12 =  H power for excitation, decoupling, and acquisition, use spinal64 as cpdprg2
;spnam0 = ramp.64, ramp from 100% down to 50%, 64 points
;cnst31 = spin rate
;p2 = X 180 at power level pl1=contact power level
;p3 =proton 90 excitation pulse at pl12
;p4 = X 90 at power level pl1=contact power level
;p15 = contact time
;p31 = spinal64 proton 165 pulse at pl12
;l0 number of draws cycles, adjust for maximum signal
;make sure you stay clear off HH during draws, i.e. spin rate
;limited by decoupling power (proton decouple > 3 x carbon DRAWS power)

define pulse d2pi ; 360 degree pulse length
"d2pi=(2s/cnst31)/17"
define pulse dninety ; 90 degree pulse length
"dninety=(1s/cnst31)/34"

define pulse dpulse ; fraction of a rotor period for each pulse
"dpulse = (1s/cnst31)/10"

define delay tau ;
"tau = (1s/cnst31)-p2/2"

define loopcounter tdless1
"tdless1=td1-1"

;"blktr2=1u"
;"blktr1=2u"

# 1 "mc_line 45 file /opt/topspin/exp/stan/nmr/lists/pp/user/draws2d_uneqCP.2.1.b.sam dc-
measurement inserted automatically"
    dccorr
# 45 "/opt/topspin/exp/stan/nmr/lists/pp/user/draws2d_uneqCP.2.1.b.sam"
1 ze
2 d1 do:f2  ;recycle delay, increment receiver phase

  10u pl1:f1     ;preselect pl1 drive power level for F1
  10u pl12:f2    ;preselect pl12 drive power level for F2
  p3:f2 ph1      ;proton 90 pulse
```

```
  ;0.3u pl2:f2
  3u
  (p15 ph2):f1 (p15:sp0 ph30):f2 ;generate ramp shape with 100% to 50% down,
                                 ;64 points
  3u
  2u pl11:f1 pl12:f2 cpds2:f2
  goscnp ph29
  1m do:f2
  1m ipp29
  lo to 2 times ns
  30m wr #0 if #0  zd
  1m iu0 ; increment l0

11 d1 do:f2 ;recycle delay

  10u pl1:f1     ;preselect pl1 drive power level for F1
  10u pl12:f2    ;preselect pl12 drive power level for F2
  p3:f2 ph1      ;proton 90 pulse
  ;0.3u pl2:f2
  3u
  (p15 ph2):f1 (p15:sp0 ph30):f2 ;generate ramp shape with 100% to 50% down,
                                 ;64 points
  3u
  2u pl11:f1 pl12:f2 cw:f2

3 d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  dninety:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  dninety:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 rpp20

  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  dninety:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  dninety:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 rpp21

  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  dninety:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  dninety:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 rpp21

  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  dninety:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  dninety:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 rpp20

  lo to 3 times l0 ; number of DRAWS cycles. That's L, not 1
```

```
  (dninety ph9):f1
  (dninety ph10):f1

4 d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  dninety:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  dninety:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 rpp22

  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  dninety:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  dninety:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 rpp23

  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  dninety:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  dninety:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 rpp23

  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  dninety:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  dninety:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 rpp22

  lo to 4 times l0 ; number of DRAWS cycles. That's L, not 1

;  tau:f2 pl1:f1
;  (p2 ph15):f1
;  tau:f2
  goscnp ph31 cpds2:f2
  1m do:f2
  1m ip22*16384
  1m ip23*16384
  1m ipp10
  1m ipp15
  1m ipp31
  lo to 11 times ns
  ;30m wr #0
  30m wr #0 if #0  zd
  1m iu0
  lo to 11 times tdless1  ; number of spectra to take
exit

ph30= 0 ; sp0 phase (1H ramp)
ph1=  1 ; Proton 90 pulse phase
ph2=  0 ; p15 phase (13C ramp)
ph15= 0 1 2 3 ; p2 phase (final echo)
```

```
ph9=  1  ; First 90 phase cycle
ph10= 1 2 3 0  ; Second 90 phase cycle
ph31= 0 3 2 1  ; receiver phase

ph20= (65536) { 16384 49152 0 16384 49152 49152 16384 0 49152 16384 } ; R in
ph21= (65536) { 49152 16384 32768 49152 16384 16384 49152 32768 16384 49152 } ; Rbar in
ph22= (65536) { 16384 49152 0 16384 49152 49152 16384 0 49152 16384 } ; R out
ph23= (65536) { 49152 16384 32768 49152 16384 16384 49152 32768 16384 49152 } ; Rbar out

ph29= 0 ; receiver phase for CP at l0 = 0
```

spinev2d_755a.2.1.b.sam

```
# 1 "/opt/topspin/exp/stan/nmr/lists/pp/user/spinev2d_755a.2.1.b.sam"
; spinev2d_755a.2.1.b.sam with CP 2007 Aug 22
; draws2d_bCP.sam, based on cpdraws1.dg and cpdraws1.jrl
; Automatically created by write_bruker750.m
; Created at 2007 February 02, 16:09:33
; phase sequence from:
; ; Phases implemented in this file in degrees are:
; 90.00 270.00 -0.00 90.00 270.00 270.00 90.00 -0.00 270.00 90.00
;
; Modified from the computer generated version to have the correct
; R groups (d2pi and dninety rather than dpulse).
;
; This program is for measuring buildup of DQ coherence via DRAWS
; using the f2 channel for the decoupler
;set:
;d1 : recycle delay
;pl1 = X power during contact
;pl2 = H power during contact
;pl11 = x power during draws, 8.5 times the spin rate, adjust for maximum signal
;pl12 =  H power for excitation, decoupling, and acquisition, use spinal64 as cpdprg2
;spnam0 = ramp.64, ramp from 100% down to 50%, 64 points
;cnst31 = spin rate
;p2 = X 180 at power level pl1=contact power level
;p3 =proton 90 excitation pulse at pl12
;p4 = X 90 at power level pl1=contact power level
;p15 = contact time
;p31 = spinal64 proton 165 pulse at pl12
;l0 number of draws cycles, adjust for maximum signal
;make sure you stay clear off HH during draws, i.e. spin rate
;limited by decoupling power (proton decouple > 3 x carbon DRAWS power)

define pulse dninety ; 90 degree pulse length
"dninety=(1s/cnst31)/34"

define pulse d2pi
"d2pi=(2s/cnst31)/17"

define delay tau ;
"tau = (1s/cnst31)-p2/2"

define loopcounter tdless1
"tdless1 = td1 - 1"

;"blktr2=1u"
;"blktr1=2u"

# 1 "mc_line 46 file /opt/topspin/exp/stan/nmr/lists/pp/user/spinev2d_755a.2.1.b.sam dc-
measurement inserted automatically"
    dccorr
# 46 "/opt/topspin/exp/stan/nmr/lists/pp/user/spinev2d_755a.2.1.b.sam"
1 ze
2 d1 do:f2  ;recycle delay, increment receiver phase

  10u pl1:f1     ;preselect pl1 drive power level for F1
  10u pl12:f2    ;preselect pl12 drive power level for F2
```

```
  p3:f2 ph1      ;proton 90 pulse
  ;0.3u pl2:f2
  3u
  (p15 ph2):f1 (p15:sp0 ph30):f2 ;generate ramp shape with 100% to 50% down,
                                 ;64 points
  3u
  2u pl11:f1 pl12:f2 cpds2:f2
  goscnp ph29
  1m do:f2
  1m ipp29
  lo to 2 times ns
  30m wr #0 if #0  zd
  1m iu0 ; increment l0

11 d1 do:f2 ;recycle delay

  10u pl1:f1     ;preselect pl1 drive power level for F1
  10u pl12:f2    ;preselect pl12 drive power level for F2
  p3:f2 ph1      ;proton 90 pulse
  ;0.3u pl2:f2
  3u
  (p15 ph2):f1 (p15:sp0 ph30):f2 ;generate ramp shape with 100% to 50% down,
                                 ;64 points
  3u
  2u pl11:f1 pl12:f2 cw:f2

3 d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  dninety:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  dninety:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 rpp20

  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  dninety:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  dninety:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 rpp21

  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  dninety:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  dninety:f1 ph21 ipp21
  d2pi:f1 ph21 ipp21
  d2pi:f1 ph21 rpp21

  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  dninety:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  dninety:f1 ph20 ipp20
  d2pi:f1 ph20 ipp20
  d2pi:f1 ph20 rpp20
  lo to 3 times l0 ; number of DRAWS cycles. That's L, not 1
```

```
  (dninety ph9):f1
  (dninety ph10):f1

4 d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  dninety:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  dninety:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 rpp22

  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  dninety:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  dninety:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 rpp23

  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  dninety:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  dninety:f1 ph23 ipp23
  d2pi:f1 ph23 ipp23
  d2pi:f1 ph23 rpp23

  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  dninety:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  dninety:f1 ph22 ipp22
  d2pi:f1 ph22 ipp22
  d2pi:f1 ph22 rpp22
  lo to 4 times l0 ; number of DRAWS cycles. That's L, not 1

;  tau:f2 pl1:f1
;  (p2 ph15):f1
;  tau:f2
  goscnp ph31 cpds2:f2
  1m do:f2
  1m ip22*16384
  1m ip23*16384
  1m ipp10
  1m ipp15
  1m ipp31
  lo to 11 times ns
  ;30m wr #0
  30m wr #0 if #0  zd
  1m iu0
  lo to 11 times tdless1  ; number of spectra to take
exit

ph30= 0 ; sp0 phase (1H ramp)
ph1=  1 ; Proton 90 pulse phase
ph2=  0 ; p15 phase (13C ramp)
ph15= 0 1 2 3 ; p2 phase (final echo)

ph9=  1  ; First 90 phase cycle
```

```
ph10= 1 2 3 0  ; Second 90 phase cycle
ph31= 0 3 2 1  ; receiver phase

;ph20= (65536) { 16384 49152 0 16384 49152 49152 16384 0 49152 16384 } ; R in
;ph21= (65536) { 49152 16384 32768 49152 16384 16384 49152 32768 16384 49152 } ; Rbar in
;ph22= (65536) { 16384 49152 0 16384 49152 49152 16384 0 49152 16384 } ; R out
;ph23= (65536) { 49152 16384 32768 49152 16384 16384 49152 32768 16384 49152 } ; Rbar out

ph20= (65536) { 18088 17309 34612 17062 14680 14680 17062 34612 17309 18088 } ; R in
ph21= (65536) { 50856 50077 1844 49830 47448 47448 49830 1844 50077 50856 } ; Rbar in
ph22= (65536) { 18088 17309 34612 17062 14680 14680 17062 34612 17309 18088 } ; R out
ph23= (65536) { 50856 50077 1844 49830 47448 47448 49830 1844 50077 50856 } ; Rbar out

ph29= 0 ; receiver phase for CP at l0 = 0
```

APPENDIX D
FITTING PHI AND PSI

Running the Simulations

run_sims.sh

```
#!/bin/bash
# run_sims.sh
#
# This file should write a submitable script to run a tripeptide simulation
#
# INPUTS:
#     data_filename - file name of the TP file
#     n_nodes - number of nodes to use
#
# Seth McNeill
# 2007 December 03

if [ $# -lt 2 ] # check to make sure there are at least 2 commandline inputs
then
   echo "Useage: run_sims.sh data_filename n_nodes"
   exit 1
fi

# Generate the directory for everything to run in
dir_base="$UFHPC_SCRATCH/$USER/tripep"
date_str=`date "+%Y%b%d"` # Creates a string YYYYMMMDD
data_basename=`echo $1 | cut -d"." -f1`  # Grab everything before the first period in the
filename

if [ -d $dir_base ]
then
   echo "$dir_base already exists"
else
   echo "Creating $dir_base"
   mkdir $dir_base
fi

dir_main="$dir_base/${date_str}_$data_basename" # name of main directory to work out of
if [ -d $dir_main ] # if the main directory already exists, create append a number until it
doesn't exist
then
   ii=2
   while [ -d $dir_main ]
   do
      dir_main="$dir_base/${date_str}_${data_basename}_${ii}"
      ii=`expr $ii + 1`
   done
fi
echo "Creating $dir_main"
mkdir $dir_main

cp $1 $dir_main

exec 3<> $dir_main/q_run  # open a file named q_run for reading and writing

now_str=`date "+%Y %b %d %H:%M:%S"` # date string for file creation

# Write the script
echo "#!/bin/bash" >&3
echo "### $dir_main/q_run" >&3
echo "### Automatically generated $now_str" >&3
echo >&3
echo "### Set the error out directory" >&3
echo "#PBS -e $dir_main" >&3
echo  >&3
echo "### Set the standard output directory" >&3
```

```
echo "#PBS -o $dir_main" >&3
echo  >&3
echo "### Set the cpu time, number of cpus, and memory requirements" >&3
echo "#PBS -l walltime=0:29:00" >&3
echo "#PBS -l nodes=1:ppn=1:gige" >&3
echo "#PBS -l mem=500mb" >&3
echo "#PBS -q testq" >&3
echo "" >&3
echo "### Set the report email address" >&3
echo "#PBS -M mcnese@mbi.ufl.edu" >&3
echo "" >&3
echo "### Set the report email options" >&3
echo "#PBS -m ae" >&3
echo "### Make PBS look in the directory where the command was submitted from:" >&3
echo "" >&3
echo "### execute the program:" >&3
echo 'echo $PBS_NODEFILE' >&3
#echo "source ~/.profile" >&3
#echo "matlab -r \"chimap_clust($2, '$dir_main/', '$data_basename', '$dir_main/out/',
'~/bin');exit\" > stdout.txt" >&3
echo "run_chimap_clust.sh ~/bin/mcr/v78/v78 $2 $dir_main/ $data_basename $dir_main/out/ ~/bin "
>&3
echo "" >&3
echo "### End the script cleanly:" >&3
echo "exit 0" >&3

exec 3>&- # close file director 3

qsub $dir_main/q_run

exit 0
```

chimap_clust.m

run_sims.sh starts a compiled version of chimap_clust.m.

```
function chimap_clust(nnodes_txt, data_dir, TP_mfile, out_dir, path_dir)
% function chimap_clust(nnodes, data_dir, TP_mfile, out_dir, path_dir)
%
% where phis and psis are vectors [min max], fit is the matrix of fits, simx
% is a 3-D matrix of the simulation axis at each simulation point and
% simdat is the data at each simulation point.
% SAM 23 May 2005
%
%  !!! WARNING: THIS SCRIPT DELETES ALL FILES IN THE OUTPUT DIRECTORY !!!
%
% Modified to not be compiled 2007 Nov 07 SAM
% Modified to have phi/psi data in the TP file 2007 Nov 13 SAM
% Modified to (hopefully) work as either compiled or not compiled
% (isdeployed) 2008 Jun 25 SAM

start_time = clock;
format compact

numbered_dirs = 1; % If no local temp available on node machines
avg_calc_time = 35; % minutes
report_email = 'mcnese@mbi.ufl.edu';
if(isdeployed) % use filenames if the function is compiled
    execute_mname = [path_dir '/run_chimap_mfile_compiled.sh'];
    cleanup_mname = [path_dir '/run_chimap_cleanup_compiled.sh'];
else % use function names if running in Matlab
    execute_mname = ['chimap_mfile_compiled'];
    cleanup_mname = ['chimap_cleanup_compiled'];
end
% Make sure that nnodes is a number, not a character
if(ischar(nnodes_txt))
    nnodes = str2double(nnodes_txt);
else
    nnodes = nnodes_txt;
```

```
end

start_dir = pwd;
TP = loaddata(data_dir, TP_mfile);
TP.max_mem = 500; %% MB
TP.TP_mfile = TP_mfile;
disp('TP = ')
disp(TP)

if(~exist(out_dir, 'dir'))
    [out_s, out_msg] = mkdir(out_dir);
    if(~out_s)
        error(['Failed to create ' out_dir '. ' out_msg]);
    end
end

system(['rm -r ' out_dir '*']); % Delete all contents of the output directory

if(length(TP.phi_limits) ~= 2)
    error('TP.phi_limits must contain exactly 2 numbers')
end
if(length(TP.psi_limits) ~= 2)
    error('TP.psi_limits must contain exactly 2 numbers')
end

phimin = min(TP.phi_limits);
phimax = max(TP.phi_limits);
psimin = min(TP.psi_limits);
psimax = max(TP.psi_limits);

if(length(TP.stepsize) > 1)
   phi_axis = phimin:TP.stepsize(1):phimax;
   psi_axis = psimin:TP.stepsize(2):psimax;
else
   phi_axis = phimin:TP.stepsize:phimax;
   psi_axis = psimin:TP.stepsize:psimax;
end

nphi = length(phi_axis);
npsi = length(psi_axis);
ncalcs = nphi*npsi; % number of calculations to be done
TP.walltime = ceil(ncalcs*avg_calc_time/nnodes/60); %% hours
disp('_____________________________________________')
disp(['Setting wall time to ' num2str(TP.walltime) ' hours for ' num2str(nnodes) ' nodes'])
disp('Press ctrl-c if this seems incorrect!')
disp('---------------------------------------------')
pause(0.1)

pt_list = zeros(ncalcs, 2);

% Generate list of points to evaluate
pt = 1;
for j = 1:nphi
    for k = 1:npsi
       pt_list(pt,:) = [phi_axis(j) psi_axis(k)];
       pt = pt + 1;
    end
end

% This section divides the number of calculations among the nodes so that
% no node does more than one more calculation than the others
base_calcs_per_node = floor(ncalcs/nnodes);
extras = mod(ncalcs, nnodes);
nnode_calcs = base_calcs_per_node*ones(1,nnodes);
for i = 1:extras
    nnode_calcs(i) = nnode_calcs(i) + 1;
end

% Generate the programs, scripts, and data for each node
last_pt = 0;
jids = [];
```

```
for j = 1:nnodes
    if(numbered_dirs)
        node_work_dir = [out_dir num2str(j) '/'];
    end
    if(~exist(node_work_dir, 'dir')) % check to make sure the needed directory exists
        [SUCCESS] = mkdir(node_work_dir);
        if(~SUCCESS)
            error(['Failed to create ' node_work_dir]);
        end
    end

    start_pt = last_pt + 1;
    stop_pt = last_pt + nnode_calcs(j);
    pts = pt_list(start_pt:stop_pt,:);
    pts_filename = [node_work_dir 'pts_' num2str(j) '.dat'];
    eval(['save ' pts_filename ' pts -ascii']);
    last_pt = stop_pt;

    qsub_script{j} = write_chimap_qsub_script_compiled(j, pts_filename, data_dir, TP, ...
        execute_mname, node_work_dir, out_dir, path_dir, numbered_dirs);
    [stat, process{j}] = system(['qsub ' qsub_script{j}]); % submit script
    jids = [jids process{j}(1:end-1) ':']; % store list of jobs for cleanup script
    disp([qsub_script{j} ' has been submitted as process ' process{j}]);
    disp('Pausing for 10 s so that all the scripts don''t go for the common start files at
once');
    pause(10); % this is added so that they don't all try for the common start files at once.
end

jids = jids(1:end-1); % get rid of extra comma at the end
disp(jids)

cleanup_sname = write_chimap_cleanup_script_compiled(cleanup_mname, report_email, data_dir, TP,
out_dir);

[stat, response] = system(['qsub -W depend=afterany:' jids ' ' cleanup_sname]);
disp(['chimap_cleanup_script job #: ' response])

exit
```

chimap_clust.m is dependant on three sub-files.

chimap_mfile_compiled.m

```
function chimap_mfile_compiled(i_txt, pts_filename, data_dir, TP_mfile, node_work_dir, out_dir,
path_dir)
% chimap_mfile(i, npts, data_dir, data_mfile, node_work_dir,
% out_dir, path_dir)
%
% This function gets executed on each node for finding the chi squared fit.
%
% Seth McNeill
% 17 June 2005

start_time = clock;

if(isstr(i_txt))
    i = str2num(i_txt);
else
    i = i_txt;
    i_txt = num2str(i);
end
info_filename = [node_work_dir 'info' i_txt '.txt'];
% addpath(genpath(path_dir));
start_dir = pwd;
disp(['The present working directory (from Matlab) is ' start_dir]);
cd(data_dir)
% eval(['TP = ' TP_mfile ';']);   % Load in tripeptide data
TP = loaddata(data_dir, TP_mfile); % Load in tripeptide data
pts = load(pts_filename);
```

```
npts = size(pts, 1);
cd(start_dir);

for j = 1:npts
    cycle_start_time = clock;

    [fit(j), simx(j,:), simd, expx, expdat] = peptide_fit_compiled(pts(j,:), TP, node_work_dir);
    ppf(j,:) = [pts(j,1) pts(j,2) fit(j) simd'];
    phipsisimx(j,:) = [pts(j,1) pts(j,2) simx(j,:)];
    save([out_dir 'phipsifit_' TP.name '_' num2str(i) '.dat'], 'ppf');
    save([out_dir 'phipsisimx_' TP.name '_' num2str(i) '.dat'], 'phipsisimx');

    calcs_left = npts - j;
    cycle_time(j) = etime(clock, cycle_start_time);
    avg_cycle_time = etime(clock, start_time)/j;

    time_left = calcs_left*avg_cycle_time;
    finish_date_num = now + time_left/(3600*24); % must convert seconds to days
    disp(['Current cycle time = ' sec2str(cycle_time(j)) ...
        ', avg cycle time = ' sec2str(avg_cycle_time)])
    disp(['Total time = ' sec2str(etime(clock, start_time)) ...
        ', est. finish time: ' datestr(finish_date_num)])
    disp(['Total cycles = ' num2str(npts) ', completed cycles = ' ...
        num2str(j) ', cycles left = ' ...
        num2str(calcs_left)])
    body = sprintf('\n%s\n[phi, psi] = [%.2f, %.2f], Est. Time Left: %s\n', ...
        datestr(now), pts(j,1), pts(j,2), sec2str(time_left));
    body = sprintf('%sCurrent cycle time = %s, ', body, sec2str(cycle_time(j)));
    body = sprintf('%sAverage cycle time = %s\n', body, sec2str(avg_cycle_time));
    body = sprintf('%sAccumulated time = %s, ', body, sec2str(etime(clock, start_time)));
    body = sprintf('%sest. finish time: %s\n', body, datestr(finish_date_num));
    body = sprintf('%sest. total time: %s\n', body, sec2str(etime(clock, start_time) +
time_left));
    body = sprintf('%sTotal cycles = %.0f, completed cycles = %.0f, cycles left = %.0f\n', body,
npts, j, calcs_left);
    info_fid = fopen(info_filename, 'a');
    fprintf(info_fid, '%s', body);
    fclose(info_fid);
end

exit
```

chimap_cleanup_compiled.m

```
function ppf_filename = chimap_cleanup_compiled(report_email, data_dir, TP_mfile, out_dir)
% function cleanup_mname = chimap_cleanup(report_email, data_dir, TP_mfile,
% out_dir)
%
% This function writes the script that cleans up and reorganizes the data
% from the parallel processing. It also sends an email to the address in
% report_email saying that it is done.
%
% Seth McNeill
% 17 June 2005
format compact

pfile_base = 'phipsifit_';
simxfile_base = 'phipsisimx_';

start_dir = pwd;
TP = loaddata(data_dir, TP_mfile); % Load in tripeptide data

pfiles = dir([out_dir pfile_base '*']);
npfiles = length(pfiles);
simxfiles = dir([simxfile_base '*']);
nsimxfiles = length(simxfiles);

ppf = [];
for i = 1:npfiles
    newdatastruct = load([out_dir pfile_base TP.name '_' num2str(i) '.dat'], '-mat');
```

```
    fname = fieldnames(newdatastruct);
    newdata = eval(['newdatastruct.' fname{1}]);
    ppf = [ppf; newdata];
end

pps = [];
for i = 1:npfiles
    newdatastruct = load([out_dir simxfile_base TP.name '_' num2str(i) '.dat'], '-mat');
    fname = fieldnames(newdatastruct);
    newdata = eval(['newdatastruct.' fname{1}]);
    pps = [pps; newdata];
end

phimin = min(ppf(:,1));
phimax = max(ppf(:,1));
psimin = min(ppf(:,2));
psimax = max(ppf(:,2));
ppf_filename = sprintf('%s%s_ppf_%g_%g_%g_%g.dat', out_dir, TP.name, phimin, ...
    phimax, psimin, psimax);
ppf_filename_v6 = sprintf('%s%s_ppf_%g_%g_%g_%g-v6.dat', out_dir, TP.name, ...
    phimin, phimax, psimin, psimax);
pps_filename = sprintf('%s%s_simx_%g_%g_%g_%g.dat', out_dir, TP.name, phimin, ...
    phimax, psimin, psimax);

save(ppf_filename, 'ppf');
save(ppf_filename_v6, 'ppf','-v6');
save(pps_filename, 'pps');

[minimum, row] = min(ppf(:,3));
min_phi = ppf(row,1);
min_psi = ppf(row,2);

body = sprintf('phimin = %f\nphimax = %f\npsimin = %f\npsimax = %f\n', phimin, ...
    phimax, psimin, psimax);
body = sprintf('%s\nMinimum of %f at [%f, %f]\n', body, minimum, min_phi, min_psi);
body = sprintf('%sCrystal file: %s\nNumber of Gamma angles: %d\n', body, ...
    TP.crystal_file, TP.gamma_angles);
body = sprintf('%s\n%s\n', body, datestr(now));
disp(body)

exit
```

write_chimap_qsub_script_compiled.m

```
function qsub_script_name = write_chimap_qsub_script_compiled(j, pts_filename, data_dir, TP,
execute_mname, node_work_dir, out_dir, path_dir, numbered_dirs)
% qsub_script_name = write_chimap_qsub_script(j, pts_filename, data_dir,
% TP_mfile, execute_mname, node_work_dir, out_dir, path_dir, numbered_dirs)
%
% This function writes the script to be passed to qsub for chi map
% creations.
%
% Seth McNeill
% 17 June 2005

start_dir = pwd;
disp(['write_chimap_qsub_script_compiled start_dir: ' start_dir]);

report_email = 'mcnese@mbi.ufl.edu';

qsub_script_dir = node_work_dir;
if(exist(qsub_script_dir) ~= 7) % check to make sure the needed directory exists
    mkdir(qsub_script_dir);
end
qsub_script_name = [qsub_script_dir TP.name '_' num2str(j) '.sh'];

[fid, msg] = fopen(qsub_script_name, 'w');
if(fid < 0)
    disp(msg);
    error(['Unable to open: ' qsub_script_name]);
```

```
end

fprintf(fid, '#!/bin/bash\n\n');
fprintf(fid, '### %s\n', qsub_script_name);
fprintf(fid, '### This script automatically generated by %s\n', mfilename());
fprintf(fid, '### %s\n\n', datestr(now));
fprintf(fid, '### Set the error out directory\n');
fprintf(fid, '#PBS -e %s\n\n', qsub_script_dir);
fprintf(fid, '### Set the standard output directory\n');
fprintf(fid, '#PBS -o %s\n\n', qsub_script_dir);

fprintf(fid, '### Set the cpu time, number of cpus, and memory requirements\n');
fprintf(fid, '#PBS -l walltime=%.0f:00:00\n', TP.walltime);
fprintf(fid, '#PBS -l mem=%.0fmb\n', TP.max_mem);
% fprintf(fid, '#PBS -l select=1:ncpus=1:mem=%.0fmb\n', max_mem);
fprintf(fid, '#PBS -l nodes=1:ppn=1:gige\n');
% fprintf(fid, '#PBS -l place=free\n');

fprintf(fid, '\n### Set the report email address\n');
fprintf(fid, '#PBS -M %s\n', report_email);
fprintf(fid, '\n### Set the report email options\n');
fprintf(fid, '#PBS -m a\n');

fprintf(fid, '### Make PBS look in the directory where the command was submitted from:\n');
fprintf(fid, 'cd %s\n\n', start_dir);
fprintf(fid, 'echo %s pwd is $PWD\n', qsub_script_name);
fprintf(fid, '\n### execute the program:\n');
if(isdeployed)
    fprintf(fid, '%s %s %d %s %s %s %s %s %s\n\n', execute_mname, TP.MCRroot, j, ...
        pts_filename, data_dir, TP.TP_mfile, node_work_dir, out_dir, path_dir);
else
    fprintf(fid, 'matlab -r "%s(%d, ''%s'', ''%s'', ''%s'', ''%s'', ''%s'', ''%s'');exit"\n\n',
execute_mname, j, ...
        pts_filename, data_dir, TP.TP_mfile, node_work_dir, out_dir, path_dir);
end
fprintf(fid, '### End the script cleanly:\n');
fprintf(fid, 'exit 0\n');

fclose(fid);
```

GGV TP File

A TP file specifies the spin system to be simulated and the regions to simulate over. The GGV TP file shown is a typical one.

```
function data = GGVdata()
% This function returns a data structure with all the experimental GAV
% data.
%
% Seth McNeill
% 30 June 2005

%Data needed for Simpson simulations
data.name = 'GGV_750';
data.MCRroot = '~/bin/mcr/v78/v78';
data.B0 = 750e6;            % B0 magnetic field
data.spin_angle = 54.735610;% Spinning axis degrees
data.T1_DW = 5e-6;      % T1 dwell time
data.T1_SW = 1/data.T1_DW;  % T1 spectral width
data.T2_DW = 2e-5;          % T2 dwell time
data.T2_SW = 1/data.T2_DW;  % T2 spectral width
data.TRot = 200e-6;       % rotation time
data.np = 1;                % Second dimension number of points
data.ni = 47;               % First dimension number of points

data.NDRAWS = 4;            % Number of DRAWS cycles to use
data.crystal_file = 'rep144'; % Crystal file to use
```

```
data.gamma_angles = 14;     % number of gamma angles to use
data.use_cluster = 0;       % 1 = use simpson clustering, 0 = don't use it

data.nspins = 2;            % number of spins being looked at
data.nucleus{1} = 'C13';    % Note the cell array to deal with different
data.nucleus{2} = 'C13';    % lengths of names
data.offset = 0.15;
data.split = 0.5;
data.tensor(1,:) = [73.2 6.7 -79.9]; % ppm delta11, delta22, delta33 where delta33 > delta22 >
delta11 and sum to ~0
data.tensor(2,:) = [73.2 6.7 -79.9]; % ppm delta11, delta22, delta33 where delta33 > delta22 >
delta11 and sum to ~0
data.reference(1) = 0; % ppm
data.reference(2) = 0; % ppm
data.iso(1) = data.offset - data.split;
data.iso(2) = data.offset + data.split;
data.sigma11 = 42; % 42 for alpha helix, 37 for beta sheet; angle between sigma11 and CN bond
data.phi_limits = [-95 -55];
data.psi_limits = [-180 180];
data.stepsize = [5 2];
%data.phi = -95 to -55 in 5 steps;
%data.psi = -180 to -180 in 2 steps;

% put in which data points in t1 were actually sampled experimentally
data.ind = [0:44];

%exp =GGV_t1 data from Manish;
data.exp_r=[0.838628571
0.670742857
0.506928571
0.323942857
0.1863
0.078371429
0.013742857
-0.047642857
-0.074714286
-0.090414286
-0.103714286
-0.099471429
-0.086157143
-0.100314286
-0.085042857
-0.072628571
-0.086085714
-0.0759
-0.068533333
-0.061916667
-0.07565
-0.074014286
-0.089042857
-0.094085714
-0.0953
-0.101885714
-0.075871429
-0.066085714
-0.056485714
-0.012671429
0.059585714
0.1233
0.23065
0.3677
0.532342857
0.715
0.866757143
0.994585714
0.985685714
0.938957143
0.798342857
0.643328571
0.479928571
0.318157143
```

```
0.175542857];
%0.086142857
%0.011716667];

data.exp_i=[0.0179
-0.03495
-0.025
-0.06175
-0.0579
-0.0727
-0.09815
-0.0825
-0.10165
-0.1043
-0.0932
-0.087
-0.1124
-0.10425
-0.0742
-0.0776
-0.04565
-0.01315
0.0118
0.0454
0.06265
0.06685
0.10305
0.1067
0.0878
0.0827
0.083
0.07425
0.06825
0.05715
0.0354
0.01635
0.00925
-0.01775
-0.00865
-0.01575
-0.02075
-0.0173
-0.02655
-0.0157
-0.03245
-0.0204
-0.0393
-0.0562
-0.07815];
%-0.07885
%-0.0877];

data.std = (0.02 + 0.02i)*ones(size(data.exp_r));

data.std_r=[0.047036571
0.038695859
0.021744173
0.026218687
0.022902765
0.027615437
0.016744139
0.021110965
0.016091346
0.01281398
0.01742024
0.016217862
0.016228869
0.011051761
0.013649525
0.025786541
0.025926655
```

```
0.020743433
0.016225741
0.010065668
0.008825135
0.018662478
0.016354496
0.020527334
0.024617
0.012567475
0.018683122
0.019232042
0.013215323
0.024050344
0.014659289
0.025977298
0.010817532
0.017353155
0.022587006
0.026732502
0.014315825
0.008236186
0.021136337
0.015793338
0.012921024
0.018565803
0.027602519
0.023006076
0.015794499];
%0.023279595
%0.028061676];

data.std_i=[0.033658283
0.000636396
0.012162237
0.00417193
0.024465895
0.013293607
0.015909903
0.010323759
0.010535891
0.015839192
0.002404163
0.013152186
0.001414214
0.040941483
0.020647518
0.007212489
0.005868986
0.008838835
0.023193102
0.016263456
0.013930004
0.010818734
0.003040559
0.014990664
0.003676955
0.002262742
0.002687006
0.004313351
0.001343503
0.003747666
0.012303658
0.004313351
0.009687363
0.021283914
7.07107E-05
0.024112341
0.024960869
0.049073211
0.015768481
0.009758074
```

```
0.010677312
0.027860007
0.005939697
0.004808326
0.016475588];
%0.006717514
%0.011313708];

data.std = data.std_r + i*data.std_i;

%normalize the data
normalization = 1/(max(data.exp_r));
data.exp = normalization*(real(data.exp_r)+i*real(data.exp_i));
data.std = normalization*data.std;
```

Miscellaneous Functions

The functions and scripts in this section demonstrate some fitting routines for the data generated by the scripts in this appendix. These functions are not guaranteed correct, but demonstrate the idea of how to go about fitting the data.

2Dfittingnotes.m

```
function [fits_ppf] = dqDRAWSfitting(dat_fname, TP_fname)
% function [fits_ppf] = dqDRAWSfitting(dat_fname, TP_fname)
% 2Dfittingnotes.m

% Seth McNeill
% 2008 July 23

load('-mat', dat_fname); % loads in matrix named ppf
eval(['TP = ' TP_fname]);  % loads TP structure
t1_fits_real_ppf = ppf_t1_fit(ppf, TP, 'real',1);
fits_real_ppf = ppf_fourier_fit(ppf, TP, 'real',1);

fits_ppf=ppf_fourier_fit(ppf, TP,[],1);
[min_chi2, min_ind] = min(fits_ppf(:,3));
disp(['Min of ' num2str(min_chi2) ' found at (' num2str(fits_ppf(min_ind, 1)) ...
    ', ' num2str(fits_ppf(min_ind, 2)) ')']);
% find(fits_ppf(:,1)==-60 & fits_ppf(:,2)==-65) % replaced by min_ind line
% above

% fits_ppf([544:724],3) % to plot a line of fit?

simd=fits_ppf(min_ind,[TP.ind + 4])'; % + 4 because ppf's first 3 lines are phi, psi, fit
fit_data_fourier(simd, TP, 'real', 'YES', [TP.ind + 1]) % +1 because ind is zero referenced
```

do_fits_script.m

```
phi = -65;
psi = -100;

% dat_fname = '2008Jul16_GFGdata_750/out/GFG_750_ppf_-145_-105_-180_180.dat';
% TP_fname = '2008Jul16_GFGdata_750/GFGdata_750';
dat_fname = '2008Jul16_GAVdata_750/out/GAV_750_ppf_-85_-45_-180_180.dat';
TP_fname = '2008Jul16_GAVdata_750/GAVdata_750';
load('-mat', dat_fname); % loads in matrix named ppf
[pathstr, TPname, TPext] = fileparts(TP_fname); % this is to check to see if the TP file is in a
different directory
if(~isempty(pathstr))
    pd = pwd;
    cd(pathstr)
    eval(['TP = ' TPname]);  % loads TP structure
```

```
    cd(pd)
else
    eval(['TP = ' TP_fname]);  % loads TP structure
end

sim_ind = ppf(:,1) == phi & ppf(:,2) == psi;


TP.exp = ppf(sim_ind, 4:end)';
TP.exp_r = real(TP.exp);
TP.exp_i = imag(TP.exp);
TP.std = TP.std(1)*ones(size(TP.exp));

GFG750data = dqDRAWSfitsims(ppf, TP);
TPGFG750 = TP;


% dat_fname = '2008Jul21_GFGdata_600/out/GFG_600_ppf_-145_-105_-180_180.dat';
% TP_fname = '2008Jul21_GFGdata_600/GFGdata_600';
dat_fname = '2008Jul21_GAVdata_600/out/GAV_600_ppf_-85_-45_-180_180.dat';
TP_fname = '2008Jul21_GAVdata_600/GAVdata_600';
load('-mat', dat_fname); % loads in matrix named ppf
[pathstr, TPname, TPext] = fileparts(TP_fname); % this is to check to see if the TP file is in a
different directory
if(~isempty(pathstr))
    pd = pwd;
    cd(pathstr)
    eval(['TP = ' TPname]);  % loads TP structure
    cd(pd)
else
    eval(['TP = ' TP_fname]);  % loads TP structure
end

sim_ind = ppf(:,1) == phi & ppf(:,2) == psi;

TP.exp_r = real(ppf(sim_ind, 4:end))';
TP.exp_i = imag(ppf(sim_ind, 4:end))';
TP.exp = TP.exp_r + i*TP.exp_i;
TP.ind = 0:(length(TP.exp)-1);
TP.std = TP.std(1)*ones(size(TP.exp));

GFG600data = dqDRAWSfitsims(ppf, TP);
TPGFG600 = TP;

figure('name','chi squared along psi')
orient landscape
hold all
grid on
plot(GFG750data.ftreal_psi(:,2), GFG750data.ftreal_psi(:,3), 'linewidth', 2)
plot(GFG750data.ft_psi(:,2), GFG750data.ft_psi(:,3), 'linewidth', 2)
plot(GFG600data.ftreal_psi(:,2), GFG600data.ftreal_psi(:,3), 'linewidth', 2)
plot(GFG600data.ft_psi(:,2), GFG600data.ft_psi(:,3), 'linewidth', 2)
xlabel('\psi (degrees)', 'fontsize', 20)
ylabel('Relative \chi^2', 'fontsize', 20)
legend(['GFG750 Real \phi=' num2str(GFG750data.ftreal_psi(1,1))], ...
    ['GFG750 \phi=' num2str(GFG750data.ft_psi(1,1))], ...
    ['GFG600 Real \phi=' num2str(GFG600data.ftreal_psi(1,1))],...
    ['GFG600 \phi=' num2str(GFG750data.ft_psi(1,1))])
legend boxoff
set(gca,'fontsize', 18)


correct_data600 = real(GFG600data.fits_real_ppf(GFG600data.fits_real_ppf(:,1) == phi,
3))/min(real(GFG600data.fits_real_ppf(GFG600data.fits_real_ppf(:,1) == phi,
3)))*length(TPGFG600.ind);
correct_data750 = real(GFG750data.fits_real_ppf(GFG750data.fits_real_ppf(:,1) == phi,
3))/min(real(GFG750data.fits_real_ppf(GFG750data.fits_real_ppf(:,1) == phi,
3)))*length(TPGFG750.ind);
figure('name','chi squared along correct psi')
orient landscape
hold all
```

```
grid on
plot(GFG600data.fits_real_ppf(GFG600data.fits_real_ppf(:,1) == phi, 2), correct_data600,
'linewidth', 2)
plot(GFG750data.fits_real_ppf(GFG750data.fits_real_ppf(:,1) == phi, 2), correct_data750,
'linewidth', 2)
xlabel('\psi (degrees)', 'fontsize', 20)
ylabel('Relative \chi^2', 'fontsize', 20)
legend(['GFG750 Real \phi=' num2str(phi)], ...
    ['GFG600 Real \phi=' num2str(phi)]);
legend boxoff
set(gca,'fontsize', 18)


figure('name','Bar graphs of exp fourier components')
orient landscape
hold all
grid on
barh = bar(GFG600data.comps, [GFG600data.coeff_exp' GFG750data.coeff_exp']);
legend('600†MHz','750 MHz')
legend boxoff
set(gca, 'fontsize', 36)
xlabel('MAS Sideband Number', 'fontsize', 42)
set(barh(1), 'FaceColor', [1 0.5 0])
set(barh(2), 'FaceColor', [0 0 1])
set(barh, 'EdgeColor', 'none', 'barwidth', 1)
a = axis;
axis([-6 6 a(3) a(4)])

figure('name','Bar graphs of sim fourier components')
orient landscape
hold all
grid on
barh = bar(GFG600data.comps, [GFG600data.coeff_sim' GFG750data.coeff_sim']);
legend('600†MHz','750 MHz')
legend boxoff
set(gca, 'fontsize', 36)
xlabel('MAS Sideband Number', 'fontsize', 42)
set(barh(1), 'FaceColor', [1 0.5 0])
set(barh(2), 'FaceColor', [0 0 1])
set(barh, 'EdgeColor', 'none', 'barwidth', 1)
a = axis;
axis([-6 6 a(3) a(4)])
```

dqDRAWSfitsims.m

```
function data = dqDRAWSfitsims(ppf, TP)
% function [fits_ppf] = dqDRAWSfitting(dat_fname, TP_fname)
%
% dat_fname: file name for dat file containing a ppf matrix
% TP_fname: file name for structure of corresponding TP file

% Seth McNeill
% 2008 July 23

% t1_fits_real_ppf = ppf_t1_fit(ppf, TP, 'real',1);
fits_real_ppf = ppf_fourier_fit(ppf, TP, 'real',1);
    [ftreal_min_chi2, ftreal_min_ind] = min(fits_real_ppf(:,3));
    figure('name','fourier real data/sim')
    orient landscape
    hold all
    grid on
    plot(TP.exp_r, 'linewidth', 2)
    plot(real(fits_real_ppf(ftreal_min_ind, 4:end)), 'linewidth', 2)
    xlabel('Data point', 'fontsize', 18)
    legend('Exp', ['Sim (' num2str(fits_real_ppf(ftreal_min_ind,1)) ', '
num2str(fits_real_ppf(ftreal_min_ind,2)) ')'])
    legend boxoff
    set(gca, 'fontsize', 18)
    % title({['\fontsize{20}' regexprep(dat_fname, '_', '\\_')], ['\fontsize{18}'
regexprep(TP_fname, '_', '\\_')]})
```

```
    disp(['FT Real min of ' num2str(ftreal_min_chi2) ' found at ('
num2str(fits_real_ppf(ftreal_min_ind, 1)) ...
        ', ' num2str(fits_real_ppf(ftreal_min_ind, 2)) ')']);
    ftreal_psi_inds = fits_real_ppf(:,1) == fits_real_ppf(ftreal_min_ind, 1);
    ftreal_psi(:,1) = fits_real_ppf(ftreal_psi_inds,1);
    ftreal_psi(:,2) = fits_real_ppf(ftreal_psi_inds,2);
    ftreal_psi(:,3) =
fits_real_ppf(ftreal_psi_inds,3)/min(fits_real_ppf(ftreal_psi_inds,3))*length(TP.ind);
    psichi_f = figure('name','Chi Squared along psi');
    orient landscape
    hold all
    grid on
    plot(ftreal_psi(:,2), ftreal_psi(:,3), 'linewidth', 2)
    xlabel('\psi (degrees)', 'fontsize', 20)
    ylabel('\chi^2', 'fontsize', 20)
    set(gca, 'fontsize', 18)

%     keyboard


fits_ppf=ppf_fourier_fit(ppf, TP,[],1);
    [ft_min_chi2, ft_min_ind] = min(fits_ppf(:,3));
    figure('name','fourier all data/sim')
    orient landscape
    hold all
    grid on
    plot(TP.exp_r, 'linewidth', 2)
    plot(real(fits_ppf(ft_min_ind, 4:end)), 'linewidth', 2)
    xlabel('Data point', 'fontsize', 18)
    legend('Exp', ['Sim (' num2str(fits_ppf(ft_min_ind,1)) ', ' num2str(fits_ppf(ft_min_ind,2))
')'])
    legend boxoff
    set(gca, 'fontsize', 18)
    % title({['\fontsize{20}' regexprep(dat_fname, '_', '\\_')], ['\fontsize{18}'
regexprep(TP_fname, '_', '\\_')]})
    disp(['Min of ' num2str(ft_min_chi2) ' found at (' num2str(fits_ppf(ft_min_ind, 1)) ...
        ', ' num2str(fits_ppf(ft_min_ind, 2)) ')']);
    ft_psi_inds = fits_ppf(:,1) == fits_ppf(ft_min_ind, 1);
    ft_psi(:,1) = fits_ppf(ft_psi_inds,1);
    ft_psi(:,2) = fits_ppf(ft_psi_inds,2);
    ft_psi(:,3) = fits_ppf(ft_psi_inds,3)/min(fits_ppf(ft_psi_inds,3))*length(TP.ind);
    figure(psichi_f)
    plot(ft_psi(:,2), ft_psi(:,3), 'linewidth', 2)
    legend(['Real Fourier Fit at ' num2str(ft_psi(1,1))], ['Fourier Fit '
num2str(ftreal_psi(1,1))])
% title({['\fontsize{20}' regexprep(dat_fname, '_', '\\_')], ['\fontsize{18}' regexprep(TP_fname,
'_', '\\_')]})
% find(fits_ppf(:,1)==-60 & fits_ppf(:,2)==-65) % replaced by min_ind line
% above

% fits_ppf([544:724],3) % to plot a line of fit?

data.simd=fits_ppf(ft_min_ind,[TP.ind + 4])'; % + 4 because ppf's first 3 lines are phi, psi, fit
[data.fit_param, data.comps, data.coeff_exp, data.coeff_sim] = fit_data_fourier(data.simd, TP,
'real', 'NO', [TP.ind + 1]); % +1 because ind is zero referenced

data.fits_real_ppf = fits_real_ppf;
data.fits_ppf = fits_ppf;
data.ftreal_psi = ftreal_psi;
data.ft_psi = ft_psi;

% keyboard
```

LIST OF REFERENCES

[1] M.H. Levitt, *spin dynamics*, John Wiley & Sons, Ltd., 2001.

[2] S. McNeill, P. Gor'kov, J. Struppe, W. Brey, and J. Long, "Optimizing ssNMR Experiments for Dilute Proteins in Heterogeneous Mixtures at High Magnetic Fields," *Magnetic Resonance in Chemistry*, vol. 45, Dec. 2007, pp. S209-S220.

[3] A.G. Webb, "Radiofrequency Microcoils in Magnetic Resonance," *Progress in Nuclear Magnetic Resonance Spectroscopy*, vol. 31, Jul. 1997, pp. 1-42.

[4] D.I. Hoult and R.E. Richards, "The Signal-to-Noise Ratio of the Nuclear Magnetic Resonance Experiment," *Journal of Magnetic Resonance (1969)*, vol. 24, Oct. 1976, pp. 71-85.

[5] E. Paulson, R. Martin, and K. Zilm, "Cross Polarization, Radio Frequency Field Homogeneity, and Circuit Balancing in High Field Solid State NMR Probes," *Journal of Magnetic Resonance*, vol. 171, Dec. 2004, pp. 314-323.

[6] R. Martin, E. Paulson, and K. Zilm, "Design of a Triple Resonance Magic Angle Sample Spinning Probe for High Field Solid State Nuclear Magnetic Resonance," *Review of Scientific Instruments*, vol. 74, Jun. 2003, pp. 3045-3061.

[7] A. De Angelis and S. Opella, "Bicelle Samples for Solid-State Nmr of Membrane Proteins," *Nature Protocols*, vol. 2, 2007, pp. 2332-2338.

[8] J.A. Stringer, C.E. Bronnimann, C.G. Mullen, D.H. Zhou, S.A. Stellfox, Y. Li, E.H. Williams, and C.M. Rienstra, "Reduction of RF-Induced Sample Heating with a Scroll Coil Resonator Structure for Solid-State NMR Probes," *Journal of Magnetic Resonance*, vol. 173, Mar. 2005, pp. 40-48.

[9] F. Doty, J. Kulkarni, C. Turner, G. Entzminger, and A. Bielecki, "Using a Cross-Coil to Reduce Rf Heating by an Order of Magnitude in Triple-Resonance Multinuclear Mas at High Fields," *Journal of Magnetic Resonance*, vol. 182, Oct. 2006, pp. 239-253.

[10] C.V. Grant, S. Sit, A.A. De Angelis, K.S. Khuong, C.H. Wu, L.A. Plesniak, and S.J. Opella, "An Efficient 1H/31P Double-Resonance Solid-State NMR Probe That Utilizes a Scroll Coil," *Journal of Magnetic Resonance*, vol. 188, Oct. 2007, pp. 279-284.

[11] B. Dillmann, K. Elbayed, H. Zeiger, M. Weingertner, M. Plotto, and F. Engelke, "A Novel Low-E Field Coil to Minimize Heating of Biological Samples in Solid-State Multinuclear Nmr Experiments," *Journal of Magnetic Resonance*, vol. 187, Jul. 2007, pp. 10-18.

[12] A. Krahn, U. Priller, L. Emsley, and F. Engelke, "Resonator with Reduced Sample Heating and Increased Homogeneity for Solid-State NMR," *Journal of Magnetic Resonance*, vol. 191, Mar. 2008, pp. 78-92.

[13] J. de Lacaillerie, B. Jarry, O. Pascui, and D. Reichert, “"cooking the Sample": Radiofrequency Induced Heating During Solid-State Nmr Experiments,” *Solid State Nuclear Magnetic Resonance*, vol. 28, Sep. 2005, pp. 225-232.

[14] C. Li, Y. Mo, J. Hu, E. Chekmenev, C. Tian, F. Gao, R. Fu, P. Gor'kov, W. Brey, and T. Cross, “Analysis of RF Heating and Sample Stability in Aligned Static Solid-State NMR Spectroscopy,” *Journal of Magnetic Resonance*, vol. 180, May. 2006, pp. 51-57.

[15] S. Dvinskikh, V. Castro, and D. Sandstrom, “Heating Caused by Radiofrequency Irradiation and Sample Rotation in C-13 Magic Angle Spinning Nmr Studies of Lipid Membranes,” *Magnetic Resonance in Chemistry*, vol. 42, Oct. 2004, pp. 875-881.

[16] P. Gor'kov, R. Witter, E. Chekmenev, F. Nozirov, R. Fu, and W. Brey, “Low-E Probe for F-19-H-1 NMR of Dilute Biological Solids,” *Journal of Magnetic Resonance*, vol. 189, Dec. 2007, pp. 182-189.

[17] P.L. Gor'kov, E.Y. Chekmenev, R. Fu, J. Hu, T.A. Cross, M. Cotten, and W.W. Brey, “A Large Volume Flat Coil Probe for Oriented Membrane Proteins,” *Journal of Magnetic Resonance*, vol. 181, Jul. 2006, pp. 9-20.

[18] P.L. Gor'kov, E.Y. Chekmenev, C. Li, M. Cotten, J.J. Buffy, N.J. Traaseth, G. Veglia, and W.W. Brey, “Using Low-E Resonators to Reduce RF Heating in Biological Samples for Static Solid-State NMR up to 900 MHz,” *Journal of Magnetic Resonance*, vol. 185, Mar. 2007, pp. 77-93.

[19] S. Grant, L. Murphy, R. Magin, and G. Friedman, “Analysis of Multilayer Radio Frequency Microcoils for Nuclear Magnetic Resonance Spectroscopy,” *Magnetics, IEEE Transactions on*, vol. 37, 2001, pp. 2989-2998.

[20] W. Froncisz and J. Hyde, “The Loop-Gap Resonator - a New Microwave Lumped Circuit Electron-Spin-Resonance Sample Structure,” *Journal of Magnetic Resonance*, vol. 47, 1982, pp. 515-521.

[21] L.D. Hall, T. Marcus, C. Neale, B. Powell, J. Sallos, and S.L. Talagala, “A Modified Split-Ring Resonator Probe for NMR Imaging at High Field Strengths,” *Journal of Magnetic Resonance (1969)*, vol. 62, May. 1985, pp. 525-528.

[22] A. Bax, “A Simple Method for the Calibration of the Decoupler Radiofrequency Field Strength,” *Journal of Magnetic Resonance (1969)*, vol. 52, Mar. 1983, pp. 76-80.

[23] C.S. Zuo, K.R. Metz, Y. Sun, and A.D. Sherry, “NMR Temperature Measurements Using a Paramagnetic Lanthanide Complex,” *Journal of Magnetic Resonance*, vol. 133, Jul. 1998, pp. 53-60.

[24] P.L. Gor'kov, C. Qjan, J.A. Kitchen, M. Sharma, T. Cross, and W.W. Brey, Pacific Grove, CA: 49th Experimental Nuclear Magnetic Resonance Conference, 2008.

[25] C. Dybowski and G. Neue, "Solid State Pb-207 NMR Spectroscopy," *PROGRESS IN NUCLEAR MAGNETIC RESONANCE SPECTROSCOPY*, vol. 41, Dec. 2002, pp. 153-170.

[26] B.M. Fung, A.K. Khitrin, and K. Ermolaev, "An Improved Broadband Decoupling Sequence for Liquid Crystals and Solids," *Journal of Magnetic Resonance*, vol. 142, Jan. 2000, pp. 97-101.

[27] A. Pines, M.G. Gibby, and J.S. Waugh, "Proton-Enhanced NMR of Dilute Spins in Solids," *The Journal of Chemical Physics*, vol. 59, Jul. 1973, pp. 569-590.

[28] E. Stejskal, J. Schaefer, and J. Waugh, "Magic-Angle Spinning and Polarization Transfer in Proton-Enhanced Nmr," *Journal of Magnetic Resonance*, vol. 28, 1977, pp. 105-112.

[29] G. Metz, X. Wu, and S. Smith, "Ramped-Amplitude Cross-Polarization in Magic-Angle-Spinning NMR," *Journal of Magnetic Resonance Series A*, vol. 110, Oct. 1994, pp. 219-227.

[30] D.M. Gregory, D.J. Mitchell, J.A. Stringer, S. Kiihne, J.C. Shiels, J. Callahan, M.A. Mehta, and G.P. Drobny, "Windowless Dipolar Recoupling: The Detection of Weak Dipolar Couplings Between Spin 1/2 Nuclei with Large Chemical Shift Anisotropies," *Chemical Physics Letters*, vol. 246, Dec. 1995, pp. 654-663.

[31] M. Mehta, M. Eddy, S. McNeill, F. Mills, and J. Long, "Determination of Peptide Backbone Torsion Angles Using Double-Quantum Dipolar Recoupling Solid-State NMR Spectroscopy," *Journal of the American Chemical Society*, vol. 130, Feb. 2008, pp. 2202-2212.

[32] T. Karlsson, J. Popham, J. Long, N. Oyler, and G. Drobny, "A Study of Homonuclear Dipolar Recoupling Pulse Sequences in Solid-State Nuclear Magnetic Resonance," *Journal of the American Chemical Society*, vol. 125, Jun. 2003, pp. 7394-7407.

[33] P. Beckmann and C. Dybowski, "A Thermometer for Nonspinning Solid-State Nmr Spectroscopy," *Journal of Magnetic Resonance*, vol. 146, Oct. 2000, pp. 379-380.

[34] J. Cavanagh, W.J. Fairbrother, A.G. Palmer, and N.J. Skelton, *Protein NMR Spectroscopy: Principles and Practice*, San Diega, California: Academic Press, 1996.

[35] T.D.W. Claridge, *High-Resolution NMR Techniques in Organic Chemistry*, Oxford, UK: Elsevier, 1999.

[36] M. Munowitz, *Coherence and NMR*, New York: John Wiley & Sons, Inc., 1988.

[37] M. Mehring and V.A. Weberrub, *Object-Oriented Magnetic Resonance*, San Diega, California: Academic Press, 2001.

[38] M.J. Duer, *Introduction to Solid-State NMR Spectroscopy*, Wiley-Blackwell, 2005.

[39] J.J. Craig, *Introduction to Robotics Mechanics and Control*, Reading, Massachusetts: Addison-Wesley Publishing Company, Inc., 1989.

[40] G. De Paëpe, P. Hodgkinson, and L. Emsley, "Improved Heteronuclear Decoupling Schemes for Solid-State Magic Angle Spinning Nmr by Direct Spectral Optimization," *Chemical Physics Letters*, vol. 376, Jul. 2003, pp. 259-267.

[41] S.A. Smith, T.O. Levante, B.H. Meier, and R.R. Ernst, "Computer Simulations in Magnetic Resonance. An Object-Oriented Programming Approach," *Journal of Magnetic Resonance, Series A*, vol. 106, Jan. 1994, pp. 75-105.

[42] M. Bak, J.T. Rasmussen, and N.C. Nielsen, "Simpson: A General Simulation Program for Solid-State Nmr Spectroscopy," *Journal of Magnetic Resonance*, vol. 147, Dec. 2000, pp. 296-330.

[43] M. Veshtort and R.G. Griffin, "SPINEVOLUTION: A Powerful Tool for the Simulation of Solid and Liquid State NMR Experiments," *Journal of Magnetic Resonance*, vol. 178, Feb. 2006, pp. 248-282.

[44] J.A. Nelder and R. Mead, "A Simplex Method for Function Minimization," *The Computer Journal*, vol. 7, Jan. 1965, pp. 308-313.

[45] J.C. Lagarias, J.A. Reeds, M.H. Wright, and P.E. Wright, "Convergence Properties of the Nelder--Mead Simplex Method in Low Dimensions," *SIAM J. on Optimization*, vol. 9, 1998, pp. 112-147.

[46] R. Freeman and H.D.W. Hill, "Phase and Intensity Anomalies in Fourier Transform NMR," *Journal of Magnetic Resonance (1969)*, vol. 4, Jun. 1971, pp. 366-383.

[47] R. Freeman, *Spin Choreography: Basic Steps in High Resolution NMR*, Oxford University Press, USA, 1998.

[48] M.H. Levitt and R. Freeman, "NMR Population Inversion Using a Composite Pulse," *Journal of Magnetic Resonance (1969)*, vol. 33, Feb. 1979, pp. 473-476.

[49] R. Freeman, S.P. Kempsell, and M.H. Levitt, "Radiofrequency Pulse Sequences Which Compensate Their Own Imperfections," *Journal of Magnetic Resonance (1969)*, vol. 38, Mar. 1980, pp. 453-479.

[50] M.H. Levitt and R. Freeman, "Compensation for Pulse Imperfections in NMR Spin-Echo Experiments," *Journal of Magnetic Resonance (1969)*, vol. 43, Apr. 1981, pp. 65-80.

[51] A.J. Shaka and R. Freeman, "Composite Pulses with Dual Compensation," *Journal of Magnetic Resonance (1969)*, vol. 55, Dec. 1983, pp. 487-493.

[52] R. Tycko, "Broadband Population Inversion," *Physical Review Letters*, vol. 51, 1983, p. 775.

[53] R. Tycko, H.M. Cho, E. Schneider, and A. Pines, "Composite Pulses Without Phase Distortion," *Journal of Magnetic Resonance (1969)*, vol. 61, Jan. 1985, pp. 90-101.

[54] D. Sakellariou, A. Lesage, P. Hodgkinson, and L. Emsley, "Homonuclear Dipolar Decoupling in Solid-State NMR Using Continuous Phase Modulation," *Chemical Physics Letters*, vol. 319, Mar. 2000, pp. 253-260.

[55] D.P. Burum, M. Linder, and R.R. Ernst, "Low-power multipulse line narrowing in solid-state NMR," *Journal of Magnetic Resonance (1969)*, vol. 44, Jul. 1981, pp. 173-188.

[56] G. De Paëpe, D. Sakellariou, P. Hodgkinson, S. Hediger, and L. Emsley, "Heteronuclear Decoupling in Nmr of Liquid Crystals Using Continuous Phase Modulation," *Chemical Physics Letters*, vol. 368, Jan. 2003, pp. 511-522.

[57] S. Conolly, D. Nishimura, and A. Macovski, "Optimal Control Solutions to the Magnetic Resonance Selective Excitation Problem," *Medical Imaging, IEEE Transactions on*, vol. 5, 1986, pp. 106-115.

[58] J. Mao, T.H. Mareci, K.N. Scott, and E.R. Andrew, "Selective Inversion Radiofrequency Pulses by Optimal Control," *Journal of Magnetic Resonance (1969)*, vol. 70, Nov. 1986, pp. 310-318.

[59] D. Rosenfeld and Y. Zur, "Design of Adiabatic Selective Pulses Using Optimal Control Theory," *Magnetic Resonance in Medicine*, vol. 36, 1996, pp. 401-409.

[60] T.E. Skinner, T.O. Reiss, B. Luy, N. Khaneja, and S.J. Glaser, "Application of Optimal Control Theory to the Design of Broadband Excitation Pulses for High-Resolution NMR," *Journal of Magnetic Resonance*, vol. 163, Jul. 2003, pp. 8-15.

[61] N. Khaneja, T. Reiss, C. Kehlet, T. Schulte-Herbr$\ddot{u}$ggen, and S.J. Glaser, "Optimal Control of Coupled Spin Dynamics: Design of NMR Pulse Sequences by Gradient Ascent Algorithms," *Journal of Magnetic Resonance*, vol. 172, Feb. 2005, pp. 296-305.

[62] C.T. Kehlet, A.C. Sivertsen, M. Bjerring, T.O. Reiss, N. Khaneja, S.J. Glaser, and N.C. Nielsen, "Improving Solid-State NMR Dipolar Recoupling by Optimal Control," *Journal of the American Chemical Society*, vol. 126, 2004, pp. 10202-10203.

[63] M. Veshtort and R.G. Griffin, "High-Performance Selective Excitation Pulses for Solid- and Liquid-State NMR Spectroscopy," *ChemPhysChem*, vol. 5, 2004, pp. 834-850.

[64] M.A. Keniry and B.C. Sanctuary, "Experimental Verification of Composite Inversion Pulses Obtained by Series Expansion of the Offset Angle," *Journal of Magnetic Resonance (1969)*, vol. 97, Apr. 1992, pp. 382-384.

[65] N.S. Bai, M. Ramakrishna, and R. Ramachandran, "Design of Phase-Distortionless Broadband Composite-Pulse Sequences Via Simulated Annealing," *Journal of Magnetic Resonance, Series A*, vol. 102, Apr. 1993, pp. 235-240.

[66] R. Ramachandran, “Erratum: Volume 102, Number 2 (1993), Series A, in the article "Design of Phase-Distortionless Broadband Composite-Pulse Sequences via Simulated Annealing" by N. Sunitha Bai, M. Ramakrishna, and R. Ramachandran, pages 235-240,” *Journal of Magnetic Resonance, Series A*, vol. 105, Dec. 1993, pp. 328-329.

[67] F.D. Mills, V.C. Antharam, O.K. Ganesh, D.W. Elliott, S.A. McNeill, and J.R. Long, “The Helical Structure of Surfactant Peptide KL4 When Bound to POPC: POPG Lipid Vesicles,” *Biochemistry*, vol. 47, Aug. 2008, pp. 8292-8300.

[68] B.C. Sanctuary, D.J. Isbister, X. Wang, L. Lu, and X. Yang, “Series Expansion for Composite Pulses in NMR,” *Journal of Magnetic Resonance (1969)*, vol. 96, Feb. 1992, pp. 229-236.

[69] Z. Tosner, T. Vosegaard, C.T. Kehlet, N. Khaneja, S.J. Glaser, and N.C. Nielsen, “Optimal Control in NMR Spectroscopy: Numerical implementation in SIMPSON,” *Journal of Magnetic Resonance*, vol. doi:10.1016/j.jmr.2008.11.020 , 2008.

BIOGRAPHICAL SKETCH

Back in 1978, Seth McNeill was born third of four boys to James and Margaret McNeill. His first 15 years were spent in southern California halfway between Los Angeles and Palm Springs. He was home schooled through third grade, and went to the school where his older brothers were already in fourth grade. The teacher was worried that Seth would not sit still through a whole day of school, but somehow she managed to get him to do so. After his freshman year of high school, his family moved to Tri-Cities in southeastern Washington state. Here Seth went to two more high schools before graduating from Upper Columbia Academy. Somewhere along the way it had become obvious that electrical engineering would be his major of pursuit having learned electronics from his older brother (now a mechanical engineer) who had learned it from his father. In his freshman year of college Seth started working for Ralph Stirling, the projects engineer, at Walla Walla College—a job he continued throughout his college career, including some summers. This job provided good training in embedded systems. He enjoyed reading the datasheets of the parts he was supposed to enter into the Mentor Graphics database. He also learned UNIX systems and was forced to learn vi, which is still his preferred UNIX editor. About March of his final year (he squeezed his 4 year degree down to 5 years somehow), he realized that to achieve his goal of teaching electrical engineer at the collegiate level he would have to go to graduate school. It was already past application deadlines, so he asked the company he had interned and done his senior project for (Cadwell Laboratories) if they needed another electrical engineer for a year. They agreed to it and Seth enjoyed a year living at home and working on low power, embedded system design for wearable, distributed medical monitoring. That job taught him the importance of understanding digital signal processing. When applying for graduate schools, Seth looked east with the thought that he could try out eastern living for a while and move back west when he was done. He applied to a few schools by

January, but sometime in February he saw the micro air vehicle project at the University of Florida advertised in the "propaganda" sheet that had been sent to his dad, who had done a fellowship at UF. That sparked his interest since he had been tinkering with robotics since seventh or eighth grade. Further exploration showed that UF did not have a hard closing date for applications. He sent one to them late and with his most outlandish statement of purpose—and he got in! Not only that, UF was the only school who offered any kind of financial support starting his first year. After visiting and finding the faculty friendly and helpful and the church more so, he decided to attend UF. Seth did a non-thesis master's focusing on DSP, networking, and primarily artificial intelligence, pattern recognition, and image processing. When he left for an internship at Caltech's Jet Propulsion Lab the summer after he graduated with his master's, he planned to start his PhD under Dr. Michael Nechyba. Upon his return from his summer internships (he also worked a couple weeks at an aggregate packing plant engineering firm where he had worked the previous summer), he found that Dr. Nechyba had decided to move to industry. Dr. Arroyo was kind enough to take Seth on if he could find funding. Seth spent that fall semester visiting labs around campus that were working on projects that were interesting. Out of the blue one day, his friend OJ from church suggested that Seth take a short introductory course on nuclear magnetic resonance. Not knowing much more than that it involved large super conducting magnets and cryogens Seth agreed. During the last lab of the class, Dr. Long's postdoc, Dr. Mini Samuel-Landtiser, asked what he did. Upon hearing that he was a programmer since coming to graduate school, she said that Dr. Long might be able to use his skills. An hour later he had an offer to join the lab where he completed his PhD. It has been a surprising path, but God has always been behind it making it work for the best.