UNIVERSITY OF OXFORD

Coherent Control of Spin Systems for Quantum Information Processing

by

Ben Rowland

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the
Department of Physics

July 2012

Declaration of Authorship

I, BEN ROWLAND, declare that this thesis titled, 'Coherent Control of Spin Systems for Quantum Information Processing' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

"A hypothesis or theory is clear, decisive, and positive, but it is believed by no one but the man who created it. Experimental findings, on the other hand, are messy, inexact things, which are believed by everyone except the man who did that work."

Harlow Shapley (1885-1972)

UNIVERSITY OF OXFORD

Abstract

Department of Physics

Doctor of Philosophy

by Ben Rowland

Over the last few years, the GRAPE algorithm has become of central importance in the development of general purpose control sequences for several branches of Nuclear Magnetic Resonance. In this thesis, the application of the GRAPE algorithm to quantum information processing tasks is considered. First the theory underlying the algorithm is reviewed in detail, then a number of extensions and improvements to the core technique are developed. An implementation of the GRAPE algorithm using GPUs is presented and compared with a standard CPU based implementation.

A variety of experimental results are presented covering many different aspects of the practical use of GRAPE. This includes an evaluation of some of the errors that can affect the performance of GRAPE sequences in real experiments, and their relative importance. The strengths and weaknesses of GRAPE compared to the other possible techniques are assessed, and some suggestions made regarding potential developments in this direction.

Pseudo-pure states are a crucial component of any NMR based quantum computer, but many current methods for preparing them sacrifice purity in exchange for simplicity in the preparation sequence. This thesis also presents a new method for robustly generating pseudo-pure states with the maximum possible purity.

Acknowledgements

First and foremost I must thank my supervisor, Jonathan Jones, for all the help he has given me throughout this project. It was Jonathan who first inspired an interest in the world of quantum information in me while I was still an undergraduate, and he has continued to provide support and wisdom throughout my doctoral studies. No issue has proved too large or small for his guidance, and I leave every meeting, if not always with a solution to a problem, then certainly with renewed confidence and a fresh arsenal of techniques to try.

Minaru Kawamura, initially a distant collaborator in Japan, also provided a useful set of fresh eyes during his visit to Oxford. His very different perspective on NMR and QC, combined with his bubbling enthusiasm, helped several new ideas take flight.

Sharing a laboratory in the chemistry department left me indebted to a whole range of great people: the head of the lab, Peter Hore, was a wonderful first point of call with any lab issues, and equally important in making sense of anything no-one else could understand. John Adams did a fantastic job of keeping the spectrometer running through the various equipment failures and Pete Biggs was also very helpful in solving our computer issues. I must also thank Shyam Masakapalli and Alice Bowen for sharing the responsibility of performing liquid nitrogen fills, and Paul Mitchell for filling the dewar for us.

Mike Giles at the OERC was kind enough to give some of his expertise on GPU programming, and Nick Soffe was an invaluable source of information on issues regarding the spectrometer, as well as lending various components. David Lucas was also kind enough to lend me a digital oscilloscope.

Finally I must thank my wife, Tracy, who has been the very model of understanding and supportiveness over the last few years. I am also enormously grateful for the proofreading she and Jonathan have done, they have made this thesis rather more readable than it might otherwise have been.

Contents

Abbreviations

CG	**C**onjugate **G**radients
CUDA	**C**ompute **U**nified **D**evice **A**rchitecture
DDS	**D**irect **D**igital **S**ynthesis
FID	**F**ree **I**nduction **D**ecay
FWHM	**F**ull **W**idth at **H**alf **M**aximum
GPU	**G**raphics **P**rocessing **U**nit
GRAPE	**GR**adient **A**scent **P**ulse **E**ngineering
HPC	**H**igh **P**erformance **C**omputing
NMR	**N**uclear **M**agnetic **R**esonance
ORE	**O**ff **R**esonance **E**rrors
PLE	**P**ulse **L**ength **E**rrors
RF	**R**adio **F**requency
SM	**S**treaming **M**ultiprocessor
SMCP	**S**trongly **M**odulated **C**omposite **P**ulses

Chapter 1

Quantum Computing

Computers today are hugely widespread, and an integral part of our modern life. Since their earliest inception, the technology underpinning them has changed beyond belief, from the humble beginnings of the punched card in Charles Babbage's "Analytical Engine" [1], through the thermionic valves of Alan Turing's "Colossus" machines during the second world war [2] to the exponential trends of today's microprocessors [3]. However, all these technologies are based around the same concept, the manipulation of classical binary data.

All conventional computers have a data store consisting of a set of binary digits, or bits. Each bit is a two level system, which can only store either the value zero or one. Strings of these zeroes and ones then make up the data for this system, which is given meaning by an encoding which specifies how to interpret it. The computer then operates by manipulating the data in its store according to its programming.

A quantum computer is created when the memory changes from being a classical two level system, restricted to only being in one of its two possible states at a time, to a quantum two level system [4]. In the quantum case, it is possible to put the system not only into its two principal states, but also into any superposition of them [5]. Correspondingly a group of such objects can also adopt a superposition of all possible states, so the number of states that can be stored (and therefore operated upon) at the same time in a quantum computer grows exponentially with the number of quantum bits, or "qubits" [6].

Various algorithms have been developed to take advantage of these superposition states, enabling certain types of problem to be solved more efficiently using a quantum computer than with any classical computer [7]. These include Grover's search algorithm and quantum counting [8] and most famously, Shor's factoring algorithm [9], which enables the solution of both the discrete logarithm and integer factorisation problems in polynomial time, exponentially faster than the most efficient known classical approach. As the difficulty of integer factorisation is the key to various widely used public-key encryption techniques such as RSA [10], Shor's algorithm has been a powerful motivator in the desire to achieve practical implementations of quantum computers.

A second, equally important potential application for quantum computers is the simulation of other quantum systems [7]. Simulating a quantum system with a classical computer is an exponential problem, because the Hilbert space of such a system grows exponentially with the number of particles. However, a quantum computer can be used to simulate a large range of quantum systems directly, while potentially allowing easier readout of results and enhancing understanding of the original quantum system [11].

There are many candidate two level quantum systems which have been proposed as potential quantum computing platforms, including the polarisation states of photons [12], vibrations of atoms in an egg-box lattice [13], atomic energy levels in trapped ions [14, 15] and charge states in quantum dots [16]. One of the most successful approaches for a variety of small scale quantum computing experiments is liquid state Nuclear Magnetic Resonance (NMR) [17], where the qubits use spin states of spin-$\frac{1}{2}$ atomic nuclei. Although it has several key flaws which greatly reduce its long term viability [18], the wide range of uses for liquid state NMR means that the equipment and many general purpose techniques are already well established. This makes it an ideal test environment for experimenting with key ideas on a small scale. Because of the similarity in both the theory and several experimental techniques between NMR and other quantum computing approaches, these ideas can then be transferred to other systems as well [17].

I begin this thesis with a brief explanation of quantum computing in liquid state NMR, including how the qubits are created, how we can manipulate the values stored in them, and how to ultimately read out the results of the computation. Then in Chapter 2 I discuss in more detail how to initialise NMR quantum computers before they can be used, and present an entirely new method that provides an optimal initialisation routine.

In Chapter 3 I introduce the GRAPE algorithm, which allows the efficient calculation of new designs for quantum logic gates in spin systems, and present my implementation of the algorithm, including various novel features to perform a much faster search for optimal sequences. Chapter 4 then covers the innovations I have introduced to improve on the core GRAPE algorithm, such as improving its robustness to various types of experimental error, and demonstrates some of the theoretical results of working with GRAPE.

Chapter 5 presents the details of my development of a new version of the GRAPE algorithm designed to take advantage of the massively parallel vector computations possible in high performance GPUs, which provides an enormous reduction in calculation time relative to standard computing performance.

Finally in Chapter 6 I present the bulk of my experimental work with implementing the GRAPE algorithm, which includes the things that GRAPE can do better than previous traditional implementations, the experimental errors and flaws which still affect its performance, and how these issues might be best tackled in the future.

1.1 Qubits

The traditional place to commence any discussion on a quantum computing implementation is with the DiVincenzo criteria [19], five key principles required for quantum computation:

1. a scalable physical system with well characterised qubits;
2. the ability to initialise the state of the qubits to a simple fiducial (reference) state, such as $|0\cdots0\rangle$;
3. long relevant decoherence times, much longer than the gate operation time;
4. a "universal" set of quantum gates;
5. a qubit-specific measurement capability.

Traditionally the two energy eigenstates of a two level quantum system are labelled $|0\rangle$ and $|1\rangle$, by analogy with their binary equivalents, and this basis is known as the

"computational" basis. Usually the labelling is such as to make $|0\rangle$ the state with lower energy, so that the energy ground state corresponds to the $|00\cdots0\rangle$ state, and will have the highest thermal population, but this is not necessary. A qubit may then exist in any general superposition state of the two basis states:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{1.1}$$

Because this form contains an ambiguous "global" phase which cannot be observed by any experiment, it is conventional to choose the global phase to make the amplitude of the $|0\rangle$ state real and positive [20]. Normalisation also requires that $|\alpha|^2 + |\beta|^2 = 1$, allowing us to rephrase the state of the qubit in a somewhat different form

$$|\psi\rangle = \cos(\theta/2)\,|0\rangle + \sin(\theta/2)\mathrm{e}^{\mathrm{i}\phi}\,|1\rangle \tag{1.2}$$

The description of the system has now been reduced to only two real numbers, θ and ϕ, rather than the two complex numbers we started out with, but it is when these numbers are interpreted as polar co-ordinates that the power of this formulation really becomes apparent. The "Bloch" sphere allows the visualisation of any qubit state as a point on the surface of the sphere, with the two poles representing the eigenstates of the computational basis. The Bloch picture allows the relationship between different states to be visualised easily, and also provides an effective and accessible way to understand all the possible transformations that may be performed to move between states.

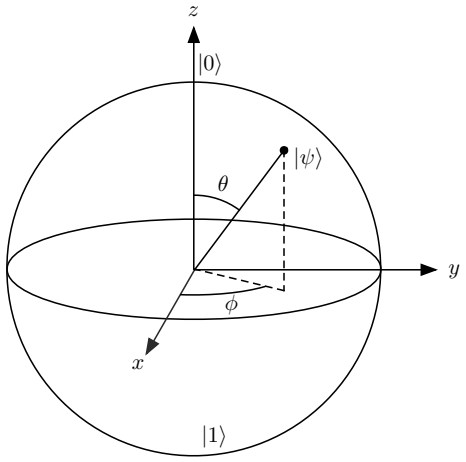


FIGURE 1.1: The Bloch sphere is a useful tool for visualising many features of a single qubit system. It is also widely used in conventional NMR to describe single spins.

1.1.1 The Qubit in NMR

An atomic nucleus has an intrinsic spin angular momentum $\vec{I}$, which gives it an associated magnetic moment

$$\vec{\mu} = \gamma\hbar\vec{I} \tag{1.3}$$

where γ is the gyromagnetic ratio of the nucleus and is determined by the nuclear type. It is customary in NMR to work in natural units and so drop the $\hbar$ terms for convenience. When placed inside a region of magnetic field $\vec{B}_0$, the nucleus experiences an interaction described by the Zeeman Hamiltonian

$$\mathcal{H} = -\vec{\mu} \cdot \vec{B}_0 \tag{1.4}$$

By standard convention the axes are chosen so as to align the z-axis along the magnetic field direction, so that

$$\mathcal{H} = -\mu_z B_0 = -\gamma B_0 I_z \tag{1.5}$$

This splits the energy levels of the nuclear spin according to their projection along the z-axis. Clearly then a spin-$\frac{1}{2}$ nucleus will fulfil our requirements for a two level quantum system to make up a qubit. In this case the two levels will be split by an energy gap of γB_0, the resonance frequency of the transition also known as the Larmor frequency ω_L. The largest magnetic fields which can be reasonably generated and maintained at precise levels are in the range of 15–20 T, and the gyromagnetic ratios of the principal spin-$\frac{1}{2}$ nuclei are given in Table 1.1 [21].

Table 1.1: Gyromagnetic Ratios of Principal spin-$\frac{1}{2}$ NMR Nuclei

Nucleus	$\gamma/2\pi$ MHz T^{-1}
^{1}H	42.576
^{13}C	10.705
^{15}N	-4.316
^{19}F	40.053
^{31}P	17.235

A spin-$\frac{1}{2}$ nucleus is an excellent implementation of a qubit, as it is a true two level system with none of the approximations that have to be used in some other implementations. Unfortunately, even for ^{1}H, which has the highest gyromagnetic ratio among stable nuclei, the energy gap is at most only a few μeV, far too low to allow detection

of individual photons. Instead it is necessary to use an ensemble of identical but independent nuclei, which may be achieved by using a fairly dilute solution of the molecule in an inert solvent.

A second consequence of this small energy gap is that the corresponding wavelengths of control fields are very long, on the order of 1m, which means that it is impossible to discriminate between individual nuclei according to their position in space. Instead it is necessary to use molecules where the different qubits have different transition frequencies. Using different nuclear species is the simplest way to achieve this, as the different gyromagnetic ratios will produce a wide range of Larmor frequencies. However, the short supply of different spin-$\frac{1}{2}$ nuclear types means that the problem of using multiple nuclei of the same type as separate qubits must be addressed. Fortunately the location of a nucleus within the molecule will have an effect on the precise magnetic field it experiences, as the local electron cloud will produce some shielding of the nucleus from the main field. Thus as long as molecules are chosen where the spin-$\frac{1}{2}$ nuclei are not in identical environments, they can be distinguished by this effect, known as the "chemical shift", although the shift will only be on the order of 1 kHz, rather than the order of 100 MHz achieved with different nuclear types.

The basic internal Hamiltonian, that is the constant Hamiltonian of the system without any control fields being applied, for a multi-qubit molecule in a magnetic field is of the form

$$\mathcal{H}_i = \frac{1}{2}\sum_j \omega_j \sigma_z^j + \frac{1}{4}\sum_{j<k} \omega_{jk}\vec{\sigma}^j \cdot \vec{\sigma}^k \tag{1.6}$$

where $\frac{1}{2}\vec{\sigma}^j$ and $\frac{1}{2}\sigma_z^j$ are the nuclear spin, and its projection onto the z-axis respectively, of the jth qubit, so that the first term is simply the Zeeman splitting discussed above, both due to the main field and the chemical shift, and the second term describes the coupling terms between each magnetic moment of all the individual pairs of qubits. The $\frac{1}{2}$ term appears in the nuclear spin operator as the nuclei are spin-$\frac{1}{2}$. Correspondingly the inter–qubit coupling is the scalar product of two $\frac{1}{2}\vec{\sigma}$ vectors, giving the overall factor of $\frac{1}{4}$. Because the molecules are in solution at room temperature, the coupling terms between spins on different molecules are time averaged to zero by rapid molecular tumbling [22].

The strength of the "spin–spin" coupling is generally much weaker than the Zeeman effect, at least for large B_0, as long as $|\omega_{12}| \ll |\omega_1 - \omega_2|$, the full Heisenberg form for the

coupling may be replaced by the considerably more tractable Ising form

$$\mathcal{H}_i = \frac{1}{2}\sum_j \omega_j \sigma_z^j + \frac{1}{4}\sum_{j<k} \omega_{jk}\sigma_z^j\sigma_z^k \tag{1.7}$$

where the truncation of the coupling by the Zeeman terms corresponds to first order perturbation theory.

The two qubit system is an important one in quantum computing, as it is the smallest system for which non-trivial quantum effects can be observed. It is also often possible to decompose larger systems into a pair of active qubits and the remaining set of qubits waiting to be operated on. For this reason, the two qubit system has its own notation which is worth introducing here. Traditionally the two qubits are labelled I and S, and the strength of the coupling between them is denoted by J, so the Hamiltonian is written

$$\mathcal{H}_i = 2\pi\nu_I I_z + 2\pi\nu_S S_z + 2\pi J I_z S_z \tag{1.8}$$

In this equation, the angular Larmor frequencies are replaced by the ordinary Larmor frequencies ν_I and ν_S to match the spin–spin coupling J, typically given in Hz. An alternative form is also used to express the Hamiltonian matrices, known as "product operator notation" [23]. This is a widely used notation in NMR which uses products of the single spin angular momentum operators to form a basis of the Hilbert space of the system. It is a useful approach in NMR because both logic gates and density matrices (which will be described later) can be expressed in terms of this basis. Arbitrary logic gates and density matrices may be deconstructed into product operator components, and a small set of simple rules can be used to work out the effect of each logic gate component on each density matrix component, before linearity is used to recombine the results to give a final answer, which can simplify understanding of the action of a complicated logic gate.

1.2 Initialisation

The ability to initialise the registers to some known initial state is a crucial part of any computer's design. It is rarely considered in the classical case, as there it has a trivial implementation, but in the quantum arena the problem is considerably more involved.

Because of the unitary nature of quantum mechanical evolution, all operations should be reversible [24]. This means that any quantum logic gate can always be run in reverse to reconstruct the original inputs from the outputs. However, the initialisation problem requires that the output should always be the same, irrespective of the input values.

This means that initialisation requires some additional processes in order to work, that reduce the isolation of the quantum computer from its environment. The two generally preferred methods are to either cool the system into its ground state, initialising it to $|00\cdots0\rangle$, or to perform a projective measurement and use the results of that to transform the state into the desired one [25]. Unfortunately, the ensemble techniques used in NMR quantum computing do not permit projective measurements to be made, as will be explained later in Section 1.5, so this technique is not useable in this system.

Although cooling the system is a theoretical possibility, even the largest energy separation between the qubit's energy levels is only a few μeV, as described above. When this is compared to the room temperature energy of 25 meV, it is clear that the populations of the different levels at this temperature will be almost the same, with only a very slightly higher occupancy of the ground state. In order to force nearly all the spins in the ensemble into the ground state, a temperature of around 50 mK would be required. Although such temperatures can be reached for some systems such as ^{3}He [26], they are not practical for a system which uses samples consisting of liquid solutions of small molecules.

1.2.1 Mixed States

Instead it is necessary to be somewhat more flexible in dealing with the ensemble. It is convenient to move to the density matrix description of the qubits. The density matrix is obtained by taking the outer product of a qubit state with itself

$$\rho = |\psi\rangle\langle\psi| = \begin{pmatrix} \alpha\alpha^* & \alpha\beta^* \\ \beta\alpha^* & \beta\beta^* \end{pmatrix} \tag{1.9}$$

Although such a formulation contains a certain redundancy when dealing with qubits in so called "pure" states, which can be in a coherent superposition of the eigenstates as described above, its strength lies in its ability to represent "mixed" states, which are a

statistical combination of multiple pure states in the form

$$\rho = \sum_i p_i \left|\psi_i\right\rangle \left\langle\psi_i\right| \tag{1.10}$$

The density matrix picture thus provides a compact description of any state that the system can be in, coherent or otherwise, containing only the physically observable information for that state.

It is also possible to visualise density matrix states on the Bloch sphere. Expressing the density matrix using the Pauli basis gives

$$\rho = \frac{1}{2}\left(s_0\sigma_0 + s_x\sigma_x + s_y\sigma_y + s_z\sigma_z\right) \tag{1.11}$$

and as all the matrices on both sides are Hermitian, all s coefficients must be real. The normalisation constraint means that $\mathrm{Tr}\rho = 1$, so as the three Pauli operators are all traceless, s_0 must equal 1, and for pure states, the three components s_x, s_y, s_z form a unit length "Bloch" vector, which points to the same point on the Bloch sphere as described by the θ and ϕ picture. It is then clear that mixed states are formed by a weighted sum of pure state Bloch vectors, and so correspond to points inside the sphere, with the maximally mixed state, represented by the identity matrix $\mathbb{I}/2$, at the centre of the sphere.

For a one qubit system, a mixed state can always be decomposed into a mixture of the maximally mixed state and one other pure state, with a Bloch vector with length less than 1. Fortunately, the observables in NMR are all traceless, so the maximally mixed state contributes no signal to the final output. It is also invariant under unitary transformations, so no quantum logic gates will cause the maximally mixed component to generate any signal. It is thus common in NMR to use the "deviation density matrix" picture [27], which permits multiples of the identity matrix to be added or subtracted from the density matrix. To examine the power of this description, consider the thermal state of a single qubit system. In thermal equilibrium with the Zeeman σ_z Hamiltonian s_x and s_y must be zero, so the thermal state is of the form $\rho = \frac{1}{2}(\sigma_0 + \delta\sigma_z)$, where δ is a complicated function of the resonance energy and temperature representing the

polarisation of the spin.

$$\begin{aligned}\rho = \frac{1}{2}(\sigma_0 + \delta\sigma_z) &= \frac{1}{2}\delta\left(\sigma_0 + \sigma_z\right) + \frac{1}{2}\left(1-\delta\right)\sigma_0 \\ &= \frac{1}{2}\delta\begin{pmatrix} 1+1 & 0 \\ 0 & 1-1 \end{pmatrix} + \frac{1}{2}\left(1-\delta\right)\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\end{aligned} \tag{1.12}$$

Applying the deviation density matrix approach and dropping the identity matrix term gives

$$\rho = \delta\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \tag{1.13}$$

In this picture therefore, the thermal state can be considered equivalent to the pure ground state $|0\rangle\langle 0|$, albeit with a reduction in the size of the signal arising from the δ term.

Mixtures of this form consisting mostly of the maximally mixed state with a small component of a single pure state are called "pseudo-pure" states [28], or sometime "effective pure states" [29], because during a quantum computation consisting of unitary transformations, the pure component will transform as a true pure state would, and the maximally mixed component is unchanged, so the experimental behaviour of the state is the same as for a pure state, but with a reduced signal size dependent on the purity of the state. In systems with more than one qubit, the Bloch sphere picture is no longer valid, but the general form of the pseudo-pure state remains the same

$$\rho = (1-\epsilon)\frac{\mathbb{I}}{2^n} + \epsilon\,|\psi\rangle\langle\psi| \tag{1.14}$$

Unfortunately in multiple qubit systems the thermal state no longer corresponds to the pseudo-pure ground state. Instead there are several techniques which must be used to generate pseudo-pure states from the thermal state. These will be covered in detail in Chapter 2.

Although this example of applying the deviation density matrix picture shows how the standard expression for a pure state can be extracted from an NMR density matrix, it is in fact the conventional approach in NMR to use this technique to extract the traceless part of the governing Hamiltonian. This is useful because it corresponds directly to the observables of the NMR system. Although a traceless Hamiltonian can never describe a

genuine physical system, which must of course have a trace of one, the deviation density matrix approach shows that the measured signal in each case will be the same.

1.3 Decoherence Times

No real quantum system is entirely isolated from its environment, and interactions between the qubits and external systems cause them to undergo uncontrolled non-unitary evolution called decoherence or relaxation [30]. Quantum computation relies on the qubits remaining in coherent superposition, so the duration of any quantum algorithm is restricted by the length of time decoherence can be kept at bay [31].

Although the actual processes causing decoherence can be extremely complex, it is usually sufficient to merely attempt to describe the overall statistical effect of the decoherence processes by assuming each qubit relaxes independently, creating an exponential decay of un-decohered spins [32]. Decoherence processes in NMR are usually divided into two principal categories, longitudinal or T_1 decoherence and transverse or T_2 decoherence.

1.3.1 Spin–Spin Decoherence

T_2 decoherence is also known as "spin–spin" decoherence, and represents the loss of phase information on the qubit [33]. In NMR this is generally caused by variations in the local environment of spins on different molecules. Variations in the magnetic field across the sample, caused either by imperfections in the main magnetic field or by nearby paramagnetic contaminant atoms, will vary the Larmor frequency of qubits, causing them to gain phase at different rates. Over time this will cause the phases of a single qubit across different molecules to diverge, until they average to zero and the phase information for the ensemble is lost. The behaviour of spins in magnetic fields will be discussed further in Section 1.4, and a more mathematical treatment of T_2 decoherence will be given in Section 4.5.

In this kind of decoherence, it is not the phases on the individual spins which are lost, just the phase coherence between different members of the ensemble. Remarkably, it

is possible to reverse a significant proportion of this decoherence and recover the phase information using a technique known as the "spin echo".

In the spin echo technique, after a period of decoherence, where each spin in the ensemble acquires a unique phase based on its localised Larmor frequency, a logic gate is applied to the qubit which has the effect of negating the phases of the individual spins. The system is then allowed to evolve for a further period of the same duration, during which time the spins each gain the same amount of phase, so that when combined with the negative phase from the first period, the phases of all the spins making up the qubit come back into alignment, reversing the decoherence and generating a second "echo" signal.

Of course, even using a spin echo, it is impossible to recover all the phase information across the entire set of qubits. As the molecules move about in their solution, their local environments will change and so will the rate at which they each gain phase relative to one another. Thus the phase gained by some spins in the first part of the echo sequence will not match exactly the phase in the second part, and so some information is irrevocably lost. It is thus necessary to introduce two "spin–spin" decay time constants. The entire phase decoherence rate is characterised by the constant T_2^*, which represents the rate at which the raw signal decays. The usually rather longer constant representing the irreversible loss of phase information is then called simply T_2, and this represents the rate at which the height of the echo signal decays. This is illustrated in Figure 1.2.

The Bloch sphere has already been demonstrated as an excellent aid in visualising the relative positions of qubit states, and even mixed states. It is also possible to understand the effects of relaxation by looking at how different types distort the Bloch sphere's surface. Phase damping acts by removing phase information from a state while keeping the same probabilities in each state. This is equivalent to condensing the Bloch sphere in the x and y directions, while leaving the z-axis invariant. This is shown in Figure 1.3.

1.3.2 Spin–Lattice Relaxation

The second decoherence mechanism, T_1 decoherence, is also known as "spin-lattice" relaxation, as it is caused by interactions between the qubit spins and their surrounding environment [33]. While computation is being performed on the qubit atoms, the other

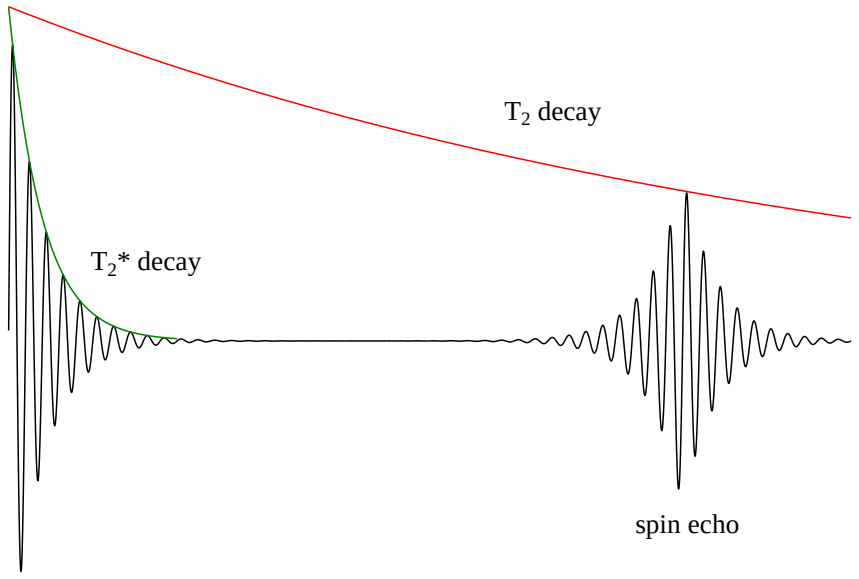


FIGURE 1.2: The observed transverse magnetic moment, normally referred to as the "Free Induction Decay" (FID), decays with the time constant T_2^*. Negating the phases after a period $\frac{\tau}{2}$ causes them to re-converge after a total time τ, producing an echo signal. However, some irreversible decay will occur, causing the height of the echo signal to decay with the time constant T_2.

components of the sample remain in thermal states, and over time random interactions with these will cause the system to rethermalise to the state

$$\rho_{\text{th}} = \frac{1}{2^n}\sigma_0 + \sum_i \epsilon_i \frac{1}{2}\sigma_z^i \tag{1.15}$$

This process is again more fully mathematically described in Section 4.5, but it is also possible to visualise the effect of this kind of relaxation on the states of the Bloch sphere, as a convenient and intuitive way to describe the situation in at least the single spin case. As time increases, the entire sphere contracts toward the thermal state, that is the north pole on the Bloch sphere. This is shown in Figure 1.4.

Typical values for T_2 of spin-$\frac{1}{2}$ nuclei in dilute solution are generally in the range of a few seconds, although it must always be appreciated that measurements of T_2 relaxation will inevitably include some contribution from T_1 processes. T_1 values are usually somewhat longer, ranging from several seconds up to a few minutes, although some exotic

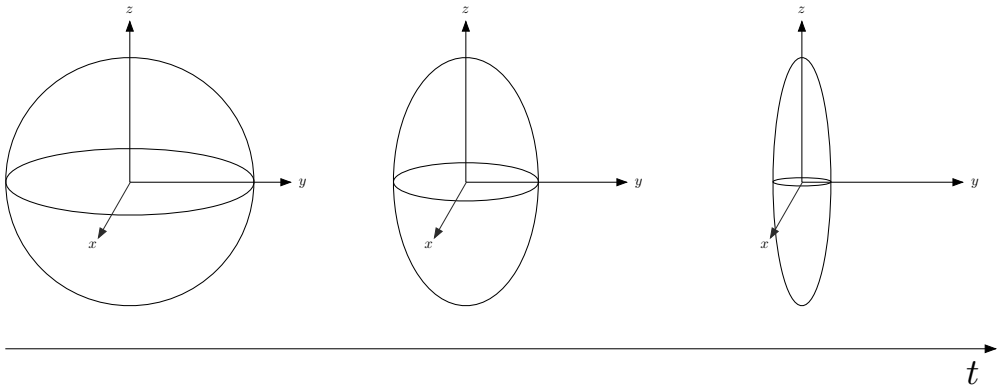


FIGURE 1.3: Phase damping decoherence causes the Bloch sphere to collapse in towards the z-axis. These are states with no relative phases, classical states.

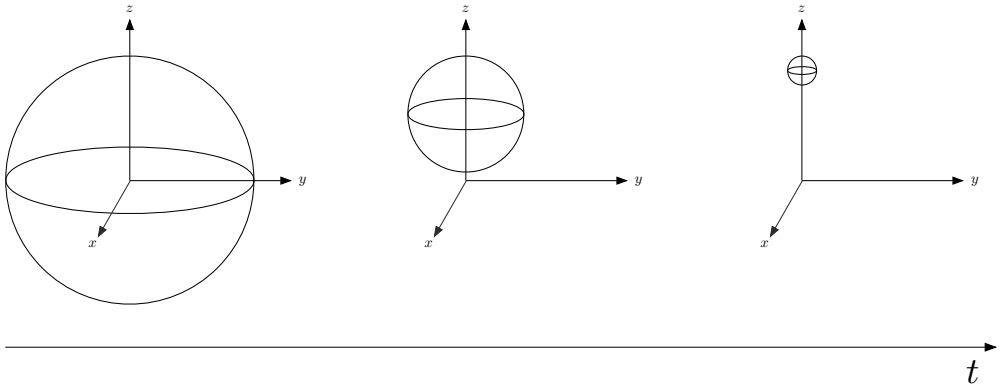


FIGURE 1.4: Amplitude damping decoherence causes the Bloch sphere to collapse in towards the thermal state.

samples can have much longer relaxation times. These timescales are far longer than those achievable in most other quantum computing implementations, mainly because the energy gap between the qubits' levels is much smaller than in most other systems. However, the most relevant issue with decoherence is not the absolute length of coherent computation, but the ratio of the decoherence time to individual gate duration, which is also much longer in NMR, again because of the relatively low energy scales. A single quantum logic gate generally takes between 5–10 μs for a simple short pulse, and up to around 100 ms for the more complicated multiple qubit gates, allowing a reasonable number of gates to be performed before the system decoheres too much.

It should also be noted that relaxation to some well defined state (in this case the thermal

state) is an absolutely essential component of initialisation. In the absence of a projective measurement, as discussed in Section 1.5, there is no other way to put the system into a known state in order to begin operations. As T_2 is generally the faster decoherence mechanism, a long T_1 is usually more inconvenient in practical terms, increasing the delay between experiments.

1.4 Logic Gates

So far the discussion of quantum computers has focussed entirely on the quantum register: how to construct the qubits, how to initialise them into a desired state, and for how long they will be useable. However, equally important to the task of constructing a quantum computer is the ability to perform logic gates on the qubits. For a complete quantum computer, it is necessary to be able to perform any possible transformation of the system.

This section begins by reviewing the basics of how control may be exerted on nuclear spin qubits, then goes on to consider the requirements of quantum computation and whether NMR is able to meet them.

Transitions between the qubit's states are electric dipole forbidden, but may of course be induced by applied magnetic fields. The effect of the main magnetic field, acting along the z-axis, was described in Section 1.1.1, and it is now logical to consider the effect of a field in the transverse plane, initially restricting the discussion to a single qubit for simplicity. It may be assumed without loss of generality that the field acts along the x-axis, as all possible axes in the transverse plane differ only by a phase transformation.

The Hamiltonian in this case takes the form

$$\mathcal{H} = -\frac{1}{2}\omega_L\sigma_z + V\sigma_x = \begin{pmatrix} -\frac{1}{2}\omega_L & V \\ V & \frac{1}{2}\omega_L \end{pmatrix} \tag{1.16}$$

where V describes the strength of the transverse field and is in units of $\mathrm{rad\,s^{-1}}$, being analogous to the Larmor frequency. The factor of two difference will be explained shortly. It is clear that the evolution of this state will be somewhat complicated if ω_L and V are of comparable magnitudes, and as B_0 (which governs the size of the Larmor frequency)

is generally the largest static field that can be achieved, it is generally impossible to make a static V much larger than ω_L.

Rather than a static field, it is much easier to generate oscillating magnetic fields in the transverse plane, simply using the magnetic component of electromagnetic radiation. The analysis of such oscillatory behaviour is easier using a basis transformation to a more convenient frame of reference, making this an appropriate moment to review the principles of such transformations.

All basis-state transformations are unitary in order to preserve the inner product in the transformed basis. Thus the transformation between wavefunctions in different frames can be written as

$$|\psi\rangle = U\left|\tilde{\psi}\right\rangle \qquad \left|\tilde{\psi}\right\rangle = U^\dagger|\psi\rangle \tag{1.17}$$

A change of basis will also cause a transformation of the Hamiltonian. Fortunately, the time-dependent Schrödinger equation gives us a relationship between the transformed state and the transformed Hamiltonian

$$\begin{aligned}
\tilde{\mathcal{H}}\left|\tilde{\psi}\right\rangle &= \mathrm{i}\frac{\mathrm{d}}{\mathrm{d}t}\left|\tilde{\psi}\right\rangle && (1.18)\\
&= \mathrm{i}\frac{\mathrm{d}}{\mathrm{d}t}U^\dagger|\psi\rangle && (1.19)\\
&= \mathrm{i}\left(U^\dagger\frac{\mathrm{d}|\psi\rangle}{\mathrm{d}t} + \frac{\mathrm{d}U^\dagger}{\mathrm{d}t}|\psi\rangle\right) && (1.20)\\
&= \left(U^\dagger\mathcal{H} + \mathrm{i}\frac{\mathrm{d}U^\dagger}{\mathrm{d}t}\right)|\psi\rangle && (1.21)\\
&= \left(U^\dagger\mathcal{H}U + \mathrm{i}\frac{\mathrm{d}U^\dagger}{\mathrm{d}t}U\right)\left|\tilde{\psi}\right\rangle && (1.22)
\end{aligned}$$

The first term is the transformation one might expect to apply to the Hamiltonian, while the second term, which only appears for non-fixed transformations, is a fictitious energy arising from the use of an accelerating frame. This, of course, is just the kind of frame needed to treat the oscillatory electromagnetic field.

The most effective way to analyse the magnetic field is to decompose it into its two circularly polarised components, then consider each component in turn. In the laboratory frame, a rotating wave will present a constant magnitude Hamiltonian with a smoothly

increasing phase

$$\mathcal{H}_{\text{rot.wave}} \propto \cos(\omega t)\sigma_x + \sin(\omega t)\sigma_y = \begin{pmatrix} 0 & \mathrm{e}^{-\mathrm{i}\omega t} \\ e^{\mathrm{i}\omega t} & 0 \end{pmatrix} \tag{1.23}$$

and it is clear that in the frame rotating with the wave, this Hamiltonian must transform to remove the time dependence. The transformation that will achieve this is

$$U = \begin{pmatrix} \mathrm{e}^{\mathrm{i}\omega t/2} & 0 \\ 0 & \mathrm{e}^{-\mathrm{i}\omega t/2} \end{pmatrix} \tag{1.24}$$

Having identified the form of the transformation to move into a rotating frame, it is now time to apply it to the complete Hamiltonian from Equation 1.16

$$\mathcal{H} = -\frac{1}{2}\omega_L\sigma_z + V\cos(\omega t)\sigma_x = \begin{pmatrix} -\frac{1}{2}\omega_L & V\cos\omega t \\ V\cos\omega t & \frac{1}{2}\omega_L \end{pmatrix} \tag{1.25}$$

$$\tilde{\mathcal{H}} = \begin{pmatrix} \frac{1}{2}(\omega - \omega_L) & V\cos(\omega t)\mathrm{e}^{-\mathrm{i}\omega t} \\ V\cos(\omega t)\mathrm{e}^{\mathrm{i}\omega t} & -\frac{1}{2}(\omega - \omega_L) \end{pmatrix} \tag{1.26}$$

The fictitious energy introduced by the transformation into the rotating frame takes the form of a magnetic field oriented along the z-axis and opposed to the main physical field. This is not that surprising as the form of the transformation is precisely equivalent to the propagator of the main field Hamiltonian. Thus by choosing the frequency of the oscillating magnetic field to be equal to the Larmor frequency, the Hamiltonian in the rotating frame can be greatly simplified.

$$\tilde{\mathcal{H}} = \frac{1}{2}V\begin{pmatrix} 0 & 1 + \mathrm{e}^{-2\mathrm{i}\omega_L t} \\ 1 + \mathrm{e}^{2\mathrm{i}\omega_L t} & 0 \end{pmatrix} \tag{1.27}$$

At this point, it is convenient to make the "rotating wave" approximation. Because the second circular component of the oscillating magnetic field is oscillating at twice the Larmor frequency from this rotating frame, it is assumed that it changes too rapidly, and so only presents a time averaged zero field thus:

$$\tilde{\mathcal{H}} = \frac{1}{2}V\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \frac{1}{2}V\sigma_x \tag{1.28}$$

This assumption is not quite perfect, as the Larmor frequencies of many nuclear species are not that large, and more careful calculations indicate that this second component produces a small shift in the frequencies, called the Bloch–Siegert shift [34].

Having disposed of both the troublesome main field component and the time dependence in the Hamiltonian by making the transformation into the rotating frame, it is now simple to calculate the evolution of a general state under this kind of Hamiltonian using the time dependent Schrödinger equation

$$\mathcal{H}\left|\psi\right\rangle = \mathrm{i}\frac{\mathrm{d}}{\mathrm{d}t}\left|\psi\right\rangle \tag{1.29}$$

which has the solution

$$\left|\psi(t)\right\rangle = U(t)\left|\psi(0)\right\rangle \tag{1.30}$$

where

$$U(t) = \mathrm{e}^{-\mathrm{i}\mathcal{H}t} = \mathrm{e}^{-\mathrm{i}Vt\sigma_x/2} \tag{1.31}$$

where U(t) is called the propagator and represents a basis transformation forward or backwards in time. The exponential of a matrix may be calculated either by diagonalising it, in which case the matrix exponential is merely the exponential of the diagonal components, or by applying the standard Taylor series for the exponential. The result for the propagator is

$$U = \begin{pmatrix} \cos(Vt/2) & -\mathrm{i}\sin(Vt/2) \\ -\mathrm{i}\sin(Vt/2) & \cos(Vt/2) \end{pmatrix} \tag{1.32}$$

and the effect of this on the ground state $\left|0\right\rangle$ is

$$\left|\psi(t)\right\rangle = U(t)\left|0\right\rangle = \begin{pmatrix} \cos(Vt/2) & -\mathrm{i}\sin(Vt/2) \\ -\mathrm{i}\sin(Vt/2) & \cos(Vt/2) \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \cos(Vt/2) \\ -\mathrm{i}\sin(Vt/2) \end{pmatrix} \tag{1.33}$$

Thus the effect of the oscillating magnetic field is to drive oscillations between the two eigenstates of the system, a process known as "Rabi flopping" [35]. A more interesting observation, however, is the effect on a general superposition state. This is best visualised using the Bloch sphere, where logic gates (propagators) can be visualised as rotations of the Bloch vector, moving the state over the surface of the sphere. All unitary transformations are length preserving, so propagators will always produce pure rotations, never bringing the state closer to or further from the origin, unlike decoherence processes.

The effect of this propagator on the Bloch sphere is to perform a rotation through an angle $\theta = Vt$ about the x-axis of the sphere. It is worth noting that if $\theta = 180°$, this propagator acts to interchange the amplitudes of the computational basis states, and this gate is equivalent to the classical NOT gate. Furthermore, this quantum NOT gate also has the effect of negating the phase of any state it is applied to, and this is the gate used in the spin echo sequence described earlier in Section 1.3.1.

When moving into the rotating frame, the transformation was carefully chosen so that in the rotating frame, the rotating component of the magnetic oscillation became a static component aligned along the x-axis. However, it is clear that a second rotating magnetic field with the same frequency, but at a different phase from the first, would also appear as a static field, but at a different phase from the original component. The Hamiltonian for such a field would take the form

$$\mathcal{H} = \begin{pmatrix} \frac{1}{2}(\omega - \omega_L) & V\cos(\omega t + \phi)\mathrm{e}^{-\mathrm{i}\omega t} \\ V\cos(\omega t + \phi)\mathrm{e}^{\mathrm{i}\omega t} & -\frac{1}{2}(\omega - \omega_L) \end{pmatrix} \tag{1.34}$$

and using the resonance condition $\omega = \omega_L$ and the rotating wave approximation this simplifies to

$$\mathcal{H} = \begin{pmatrix} 0 & \frac{1}{2}V\mathrm{e}^{\mathrm{i}\phi} \\ \frac{1}{2}V\mathrm{e}^{-\mathrm{i}\phi} & 0 \end{pmatrix} = \frac{1}{2}V\left(\sigma_x\cos\phi + \sigma_y\sin\phi\right) \tag{1.35}$$

Thus the transformation behaves as one might expect: by varying the phase of the resonant magnetic field, it is possible to change the axis of rotation in the transverse plane. Rotations about the z-axis can be achieved either by a sequence of x and y rotations, or simply by absorbing them into the rotating frame, allowing full three-axis control over the spin.

Visualising logic gates as rotations over the Bloch sphere gives rise to a convenient notation for describing certain common logic gates. The gate θ_i represents a rotation of an angle θ clockwise around the Bloch sphere, where $i \in x, -x, y, -y, z, -z$ represents one of the principal axes of the system. This is a useful shorthand for a set of gates which are frequently used in quantum information processing. The notation θ_ϕ is also used to describe a rotation about a general axis in the transverse plane at an angle ϕ to the positive x-axis.

Another convenient shorthand notation is the use of the symbols X, Y and Z to represent rotations of 180° about the appropriate axis, which are equivalent to the Pauli operators σ_x, σ_y and σ_z respectively. The X gate is also referred to as the NOT gate, although any 180°_ϕ gate would be able to perform the classical NOT exchange of $|0\rangle$ and $|1\rangle$ eigenstates, when we consider superposition states only the X gate does not introduce additional undesirable phases between the states.

It is also interesting to consider what happens if the radiation is not quite in resonance with the transition frequency. In this case the Hamiltonian must be left in its general form from Equation 1.25, with the difference in angular frequencies being some small quantity δ:

$$\tilde{\mathcal{H}} = \begin{pmatrix} \frac{1}{2}\delta & V\cos(\omega t)\mathrm{e}^{-\mathrm{i}\omega t} \\ V\cos(\omega t)\mathrm{e}^{\mathrm{i}\omega t} & -\frac{1}{2}\delta \end{pmatrix} \tag{1.36}$$

Making the standard rotating wave approximation once again removes the time dependence of the Hamiltonian. The propagator can then be calculated using the exponential:

$$U(t) = \exp\left(-\mathrm{i}t \times \begin{pmatrix} \frac{\delta}{2} & \frac{V}{2} \\ \frac{V}{2} & -\frac{\delta}{2} \end{pmatrix}\right) \tag{1.37}$$

to give the significantly more complex off resonance propagator

$$U = \begin{pmatrix} \cos(\Omega t/2) - \mathrm{i}\frac{\delta}{\Omega}\sin(\Omega t/2) & -\mathrm{i}\frac{V}{\Omega}\sin(\Omega t/2) \\ -\mathrm{i}\frac{V}{\Omega}\sin(\Omega t/2) & \cos(\Omega t/2) + \mathrm{i}\frac{\delta}{\Omega}\sin(\Omega t/2) \end{pmatrix} \tag{1.38}$$

where $\Omega = \sqrt{V^2 + \delta^2}$. It is clear that the on-resonance result may be recovered simply by setting δ to zero. Under off-resonance conditions, the frequency of the Rabi oscillations increases relative to the on-resonance case, but the system no longer oscillates between $|0\rangle$ and $|1\rangle$. This can be understood best in the Bloch sphere picture. If the precession rate of the spin is slightly different from that of the rotating frame, not all of the main magnetic field will be cancelled by the transformation. Thus the total magnetic field around which the spin will precess will be the sum of the radiation component and the residual main field. This total field will have a greater magnitude than the radiation field alone, but the axis will be perturbed from the xy plane, so the rotation will no longer be able to interconvert the two basis states.

1.4.1 The Vector Model

The vector model is an entirely classical picture inspired by the Bloch sphere of a single qubit system, which gives an alternative and more intuitive picture of how such a system behaves. The core concept is to replace the spin by a classical magnetic moment, oriented along the same direction as the corresponding Bloch vector [36].

When such a magnetic moment is placed in a magnetic field, it will precess around the field axis at a rate determined by the size of the magnetic field, in an exactly analogous way to the way in which the basis state $|1\rangle$ picks up a phase relative to the ground state $|0\rangle$.

The transformation into the rotating frame is also easily understood using this picture. Obviously in a frame rotating at the same rate as the spin, it must appear as a purely static magnetic moment. Since the spin is not precessing about the z-axis, the main field must be zero in this frame. This is a simple but effective way of understanding the gauge transformation into the rotating frame and the corresponding fictitious magnetic field which is generated to cancel out the main field. In the rotating frame, the synchronised rotating component of the oscillating magnetic field will also be static, and at an angle determined by the phase, and the magnetic moment will precess around that in a precisely similar way to the rotation about the main field in the lab frame. These are just a few examples of how the vector model can help to give a more intuitive understanding for the behaviour of a single qubit system. It must be remembered, however, that it will not give accurate results for larger systems, and so must be used with care.

It should also be noted that the version of the vector model most widely used [37] in conventional NMR uses a non-standard convention for the direction of rotations. I will avoid potential issues with this by consistently using the same convention for all rotations.

1.4.2 Universal Gates

In classical computing, both the binary NAND and NOR gates are called “universal” gates, because it is possible to construct any conceivable classical logic circuit using only combinations of either of these two gates [24]. Having such a universal gate for the

quantum computer would be extremely useful [38]. Unfortunately, neither the NAND or NOR gates are suitable for quantum computing, as they are not reversible, but more complex two qubit quantum gates can be used.

The ability of a quantum computer to occupy superposition states also means that a qubit can have an infinite number of different states, which must therefore require an infinite number of combinations of the universal gate to be able to generate them all, some of which will require an infinite sequence of gates. Consequently, any universal gate can only be used to generate circuits to perform logic gates to an arbitrary accuracy, with the number of logic gates increasing with the desired precision [39, 40]. Remarkably, it turns out that almost any non trivial two-qubit gate can be used as a universal gate in this way [41].

The key to this construction of gates from a universal set is the Solovay–Kitaev theorem [42], which states that given a set of gates which act as generators of the $\mathrm{SU}(2^n)$ group for an n qubit system, any arbitrary logic gate can be approximated using a sequence of gates drawn from the universal set, and with length of $O(\log_c(\frac{1}{\epsilon}))$, where c is a constant and ϵ is a measure of the deviation of the approximation from the target gate. This theorem is important because it shows that not only can any arbitrary gate be generated from the universal set, but that it can be done in polylogarithmic time. If there were no limits on the length of each approximation, the potential benefits of many quantum algorithms over their classical counterparts would be lost.

The Solovay–Kitaev theorem can be expressed as a simple recursive algorithm [43] whereby at each level of the recursion some of the error terms are cancelled to provide an improved approximation. In order to work efficiently, the algorithm must have a reasonably large pool of basic gates at the bottom of the recursion. These are normally generated by brute force by considering all possible combinations of the basic gates up to some maximum number. Although there are some individual multi-qubit gates which can act as universal sets on their own, it is clear that the larger the set of basic gates used, the faster and more efficient the approximation will be.

One example of a commonly used universal set [17] consists of the two single qubit gates, the Hadamard (H) and fourth root of Z gate (T)

$$H = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \quad T = \begin{pmatrix} 1 & 0 \\ 0 & \mathrm{e}^{\mathrm{i}\pi/4} \end{pmatrix} \tag{1.39}$$

and the two qubit controlled-NOT gate (c-NOT)

$$\text{c-NOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{1.40}$$

The Hadamard gate is responsible for interconverting the computational basis and the important $|\pm\rangle$ basis, defined as

$$|+\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \quad |-\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right) \tag{1.41}$$

which lie along the $\pm x$ axes of the Bloch sphere. This basis is important because the $|+\rangle$ state is the equally weighted superposition state used to exploit quantum parallelism in many quantum algorithms. The action of the Hadamard gate can be modelled on the Bloch sphere as a rotation of 180° about the line $x = z$. The fourth root of Z gate is a rotation around the z-axis, just like the Z gate, but only through an angle of 45°, so that T^4 produces a complete Z gate rotation of 180°.

The controlled-NOT gate is a very important gate in quantum computing, as it allows conditional logic, where the final state of one of the qubits is dependent on the input states of both qubits. In this case, a NOT gate is applied to the second qubit (also called the target), if and only if the first qubit (called the control) is in the state $|1\rangle$

Of course, the c-NOT gate is not alone in having these qualities and there are in fact an infinite number of other two qubit gates with the same properties which may be interconverted using single qubit gates. The c-NOT is the traditional gate used in theoretical descriptions, but of more practical interest in NMR is the controlled-Z gate,

which has the matrix form

$$\text{c-Z} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \tag{1.42}$$

which is related to the controlled-NOT gate by Hadamard single qubit gates using the network shown in Figure 1.5 [44].

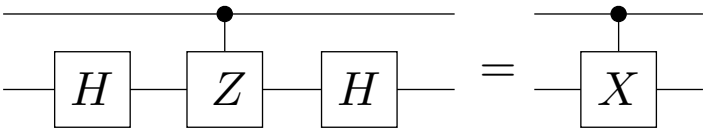


FIGURE 1.5: The controlled-NOT gate (also known as a controlled-X gate), can be constructed from the controlled-phase gate (also known as a controlled-Z gate) by applying two Hadamard gates to the control qubit line either side of the main gate. In this notation, the dot and connecting line represents a controlling qubit, so the gate is only performed on the target qubit if the control qubit is in state $|1\rangle$.

The c-Z gate is more important in NMR because it is symmetric with respect to the two qubits, just like the Ising coupling between spins. In fact, it turns out that the c-Z gate can be implemented in a two qubit system simply with a period of evolution under the Ising Hamiltonian of duration 1/2J followed by a simultaneous 90_{-z} gate [37].

In larger systems, the Hamiltonian will contain coupling terms between several pairs of spins. The couplings between pairs apart from the target pair must be suppressed to prevent incorrect evolution of those parts of the system. This can be achieved using spin-echo techniques, as described in Section 1.3.1.

As described previously, the spin-echo for one qubit works by applying a NOT gate halfway through a period of evolution, negating the phase so that after the complete evolution the sum of the first negated phase and the second phase is zero, effectively removing any Zeeman-like coupling terms. An alternative picture is to consider that as the NOT gate works to interchange the states $|0\rangle$ and $|1\rangle$, the phase during the second period of evolution is going in the opposite way to the original phase, so they cancel out. This is a useful way of thinking about the multiple qubit case. In this case whenever a NOT gate is applied to a single qubit, its couplings to the other qubits will start acting in reverse. Thus by always applying NOT gates to two target qubits together,

and to the other qubits separately, and ensuring that each qubit experiences an equal duration in each direction, all undesired couplings can be reversed [45]. This is obviously a complicated problem in large, fully-coupled systems, and will usually be done most efficiently by a computer sequence compiler [46]. A sample circuit for three qubits, suppressing couplings between all pairs except qubits 1 and 2 is shown in Figure 1.6.

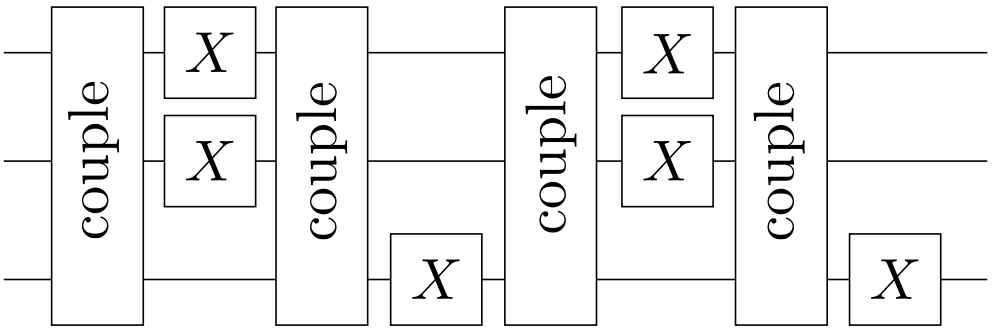


FIGURE 1.6: Spin echo techniques can be used to suppress evolution under the Zeeman Hamiltonian by using a pair of NOT gates to reverse the spin precession direction for half the required period. If two qubits have the NOT gates applied at the same times, they will continue to evolve under the spin–spin coupling, but these couplings can also be suppressed by offsetting their respective reverse periods. In this figure, the top two qubits will experience a period of coupling under only their spin–spin interaction, while all the Zeeman terms and all couplings with the bottom qubit have been suppressed.

While the Solovay–Kitaev theorem allows the generation of any arbitrary quantum logic gate using sequences of only these three gates, the larger the number of different gates which may be implemented directly as components of the universal set, the more efficient and accurate the generation will be [47, 48]. This is particularly important experimentally, where decoherence processes place a hard limit on the length of a quantum algorithm. As no gate can be implemented absolutely perfectly, it is also the case that the more gates that are used in a sequence, the greater the accumulated errors will generally become.

1.5 Measurement

As already discussed, the small energy gap between the two levels of an NMR qubit makes detecting a single photon emission from the upper level almost impossible, necessitating the use of an ensemble containing a very large number of identical copies of

the relevant molecule. Furthermore the average lifetime of excited states against spontaneous emission is typically on the order of 10^9 years, making detecting fluorescence an entirely impractical method of determining the state.

Instead, measurements of NMR qubit states are performed in precisely the same way as other NMR experimental data acquisitions, by acquiring an NMR spectrum. This is a largely classical operation, which involves measuring the net transverse magnetisation of the sample, which will be precessing about the main magnetic field direction, as described by the vector model [49]. By measuring its x and y components over some acquisition time, usually on the order of a few seconds, and then taking the Fourier transform, the component frequencies of the spectrum, corresponding to the Larmor frequencies of the constituent qubits, are identified.

Measuring both the x and y components of a spin's magnetic moment ought to be impossible, as they are non-commuting operators. However, by only measuring the combined, classical magnetisation of a large ensemble, the measurement is not a true projective measurement on any individual molecule, and so does not in fact cause superposition states to collapse [21]. It is in fact impossible to perform a projective measurement in an NMR system, and this is one of the key limitations with the NMR approach to quantum computing.

1.5.1 NMR Spectra: Single Spin

The simplest example case to consider is for a single qubit system. In the thermal state, the qubit will be in a pseudo-pure state, aligned along the main magnetic field. There will therefore be no precession and so correspondingly no measured signal. It is therefore customary in traditional NMR to apply a 90° rotation to the qubit to align it along the x-axis immediately before measuring so that during the acquisition period the transverse (precessing) magnetisation is at a maximum.

This precession will generate a signal at the Larmor frequency of the qubit. The natural linewidth of the peak is very narrow because of the low energy and correspondingly long lifetime of the excited state. In general, the principal cause of broadening is due to the non-uniformity of the main magnetic field across the sample volume. Differences in the magnetic field will produce different Larmor frequencies at different parts of the

volume, which can be a significant concern as not only will the signal line be broadened and possibly misshapen, but some parts of the sample will no longer be perfectly on-resonance with any control fields. These off-resonance effects and the errors they can produce will be examined further in Chapter 4.

To ensure that the magnetic field is as uniform as possible, secondary "shimming" coils can be used to adjust the field. Usually, the field is adjusted to give the sharpest response possible, although shimming is also used to try and obtain a good, symmetric lineshape.

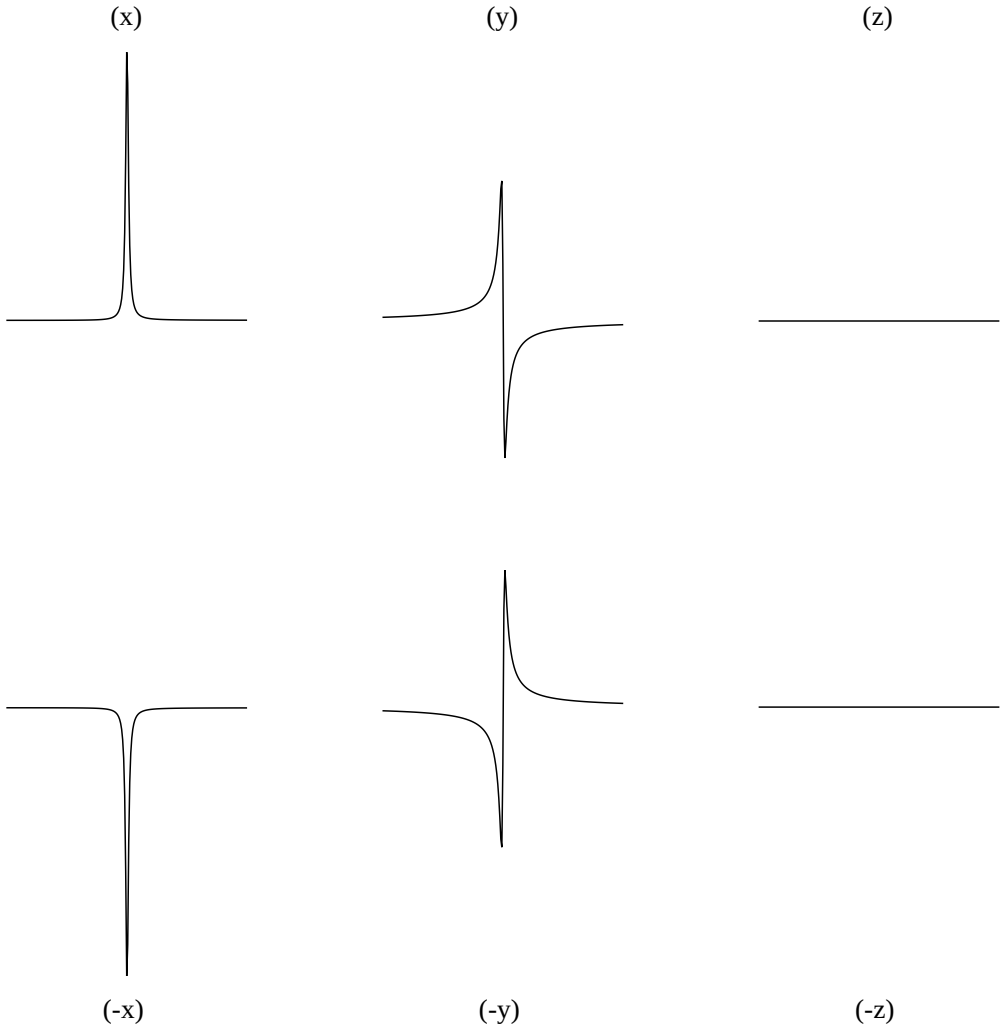


FIGURE 1.7: Stylised NMR spectra showing the expected signal from a single spin system aligned along each of the six primary Cartesian axes.

Figure 1.7 shows stylised spectra for the principal states of a single spin system. This is an appropriate moment to mention some of the conventional terms in NMR spectroscopy which will be used when describing spectra throughout this thesis. As described above,

observing states aligned along the z-axis gives no signal at all, while any component of magnetisation in the transverse plane will produce some signal.

The standard convention is to define the x-axis so that spectra from x aligned qubits have a standard Lorentzian lineshape. The direction of rotation of a spin under RF radiation means that the conventional axis along which the radiation is applied is $-y$. By analogy with other spectroscopy techniques, the spectrum obtained from an x-axis aligned spin is described as "absorptive", while the inverted spectrum from when the spin is aligned along $-x$ is called an "emissive" lineshape.

Signal obtained from y-axis aligned spins differs from the x aligned spins only in an offset phase of $\frac{\pi}{2}$, transforming them from even functions to odd functions, and producing the characteristic "dispersive" lineshapes shown in Figure 1.7. As such it is straightforward to re-phase spectra simply by multiplying by any pure phase complex number. Spins in the transverse plane lying between the Cartesian axes will produce spectra which are partially dispersive, and usually the best way of identifying their exact angle in the transverse plane is to determine the required degree of re-phasing to produce a fully absorptive lineshape.

In general, the most effective measurement technique is usually highly experiment specific. In some cases it will be possible to gain useful information from a direct measurement. Other times, the phase damping will be artificially increased (see Section 2.4) to contract the Bloch sphere onto the z-axis as described in Section 1.3.1, then a standard 90° observe pulse will enable the relative populations of $|0\rangle$ and $|1\rangle$ to be measured from the amplitude of the signal. It is important that the phase damping is performed first so that the axis of rotation is perpendicular to the Bloch vector.

This produces a signal whose size will vary from maximally absorptive to maximally emissive depending on the relative phases of the states $|0\rangle$ and $|1\rangle$. The phase of any individual NMR spectrum has no meaning, but by first acquiring a reference spectrum defining a known state such as $|+\rangle$, other spectra can be assigned a relative phase from that. The absolute height of an NMR spectrum also has no precise meaning, because of the nature of ensemble computing, many things can affect the polarisation of the spin system, and the intrinsic sensitivity of the NMR spectrometer. The only two factors of interest are the relative heights of the lines compared to a simple reference spectrum, and

the signal to noise ratio. For this reason, most experimentally acquired NMR spectra are plotted without a y-axis.

1.5.2 NMR Spectra: Multiple Spins

The second simplest sample spectrum is for two uncoupled spins. Although this is not a useful system for quantum computing, where there must always be some chain of couplings linking all the qubits in use, the situation often arises when using deuterated water as a solvent: although ^{2}H is a spin-1 nucleus with a quite different Larmor frequency from ^{1}H, even a small quantity of only partially deuterated HOD contaminating a sample will introduce an additional, isolated qubit into the spectrum. A second uncoupled qubit will produce an additional independent line in the spectrum.

The range of frequencies that can be detected by a single acquisition is somewhat limited, so usually only one nuclear species may be observed in a single experiment. When both qubits are of the same species, a so called "homonuclear" system, both lines will be visible in the acquired spectrum, while if the system is "heteronuclear" with two distinct nuclear species, only one line can in general be observed per experiment, and multiple repeats must be performed in order to observe the complete system.

The situation becomes more complicated when the spins are coupled, as then the energy gap for each spin becomes dependent on the state of the other spin. This has the effect of splitting the line into a "doublet", with a separation given by the J-coupling constant ω_{12}. This is the system described in Equation 1.8. For each qubit, in thermal equilibrium the other qubit will have almost equal populations in the two states, so an observation of either qubit (or both in a homonuclear system) will give both lines equal intensity. This is represented in Figure 1.8.

When the two qubits are not in thermal equilibrium, but instead in some state resulting from a computation, more information can be obtained. The simplest analysis arises when the system is in one of its four computational eigenstates. In a heteronuclear system, exciting one of the spins will only produce a single line, corresponding to the value of the other spin, while the phase of the line (absorptive or emissive) indicates the value of the observed qubit. In a homonuclear system, both qubits will be excited by the 90° pulse, meaning that both lines are observed for each qubit, and it is simply the

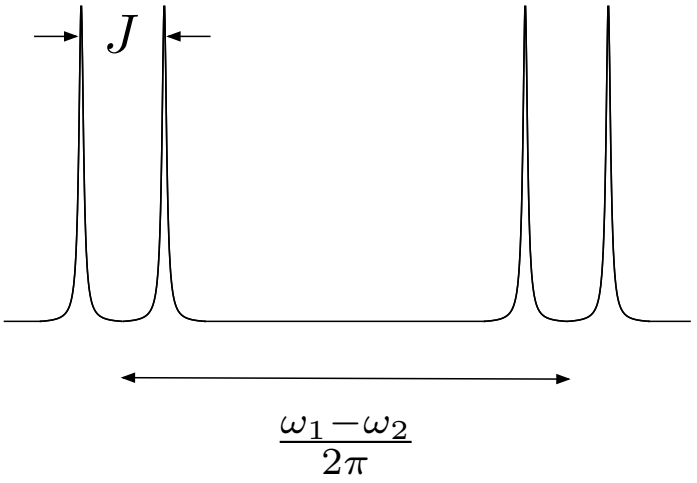


FIGURE 1.8: In a two qubit system, the signal from each qubit is split into a doublet because of the interaction with the other spin. The centres of each doublet are separated by the difference in resonance frequency between the two nuclei.

phase of each doublet that identifies the state of that qubit. Stylised spectra depicting the results for each of these cases are shown in Figure 1.9. In general the analysis is more difficult for complicated states, and often the only way to identify the state is to run a simulation of the algorithm and then compare the simulated results with the experimental ones.

All these ideas can be generalised to larger spin systems, although the analysis does become more complex. As mentioned previously, it is essential that there exists some kind of coupling chain between all qubits in a system, thus the lines will be split again for each coupling that the qubit experiences, producing more complicated "multiplet" line groupings.

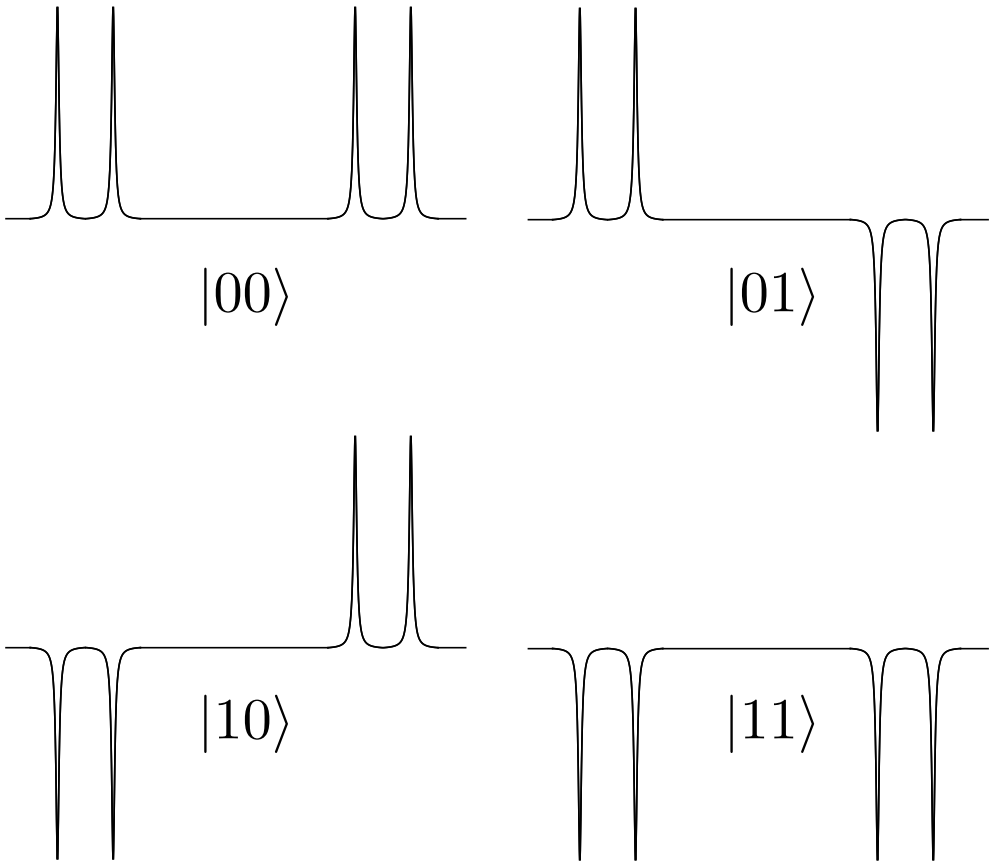


FIGURE 1.9: Stylised NMR spectra showing the four computational basis states of a two spin system.

Chapter 2

Generating Pseudo-Pure States

As described in Section 1.2, it is unfeasible in practice to produce a pure ground state in an ensemble NMR quantum computer, but it is possible to purify a small proportion of the ensemble, while leaving the rest unobservable and contributing no signal to the result [18, 50]. This chapter will give more details of the theory underpinning these pseudo-pure states, review some of the approaches that may be used to prepare them, and finally present an experimental implementation of a novel technique originally conceived by Kawamura, which produces the purest possible state [51].

2.1 The Thermal State

When treating an ensemble system such as an NMR sample, the thermal equilibrium state is given by the partition function [52]

$$\begin{gathered} \rho = \frac{\mathrm{e}^{-\beta\mathcal{H}}}{\mathcal{Z}} \\ \beta = \frac{1}{kT} \qquad Z = \mathrm{tr}\left(\mathrm{e}^{-\beta\mathcal{H}}\right) \end{gathered} \tag{2.1}$$

At room temperature, the high temperature approximation allows the exponential to be truncated to just the first two terms.

$$\rho = \frac{\mathbb{I} - \beta\mathcal{H}}{\mathcal{Z}} \tag{2.2}$$

It is also clear in this high temperature limit that the normalisation function $\mathcal{Z}$ must be equal to 2^n for an n qubit system so that the trace of ρ is one ($\mathrm{tr}(\mathbb{I}) = 2^n$ and $\mathrm{tr}(\mathcal{H}) = 0$).

Because the spin–spin couplings are generally very small compared to the dominant Zeeman splitting, it is acceptable to ignore their contribution to first order, so that the general thermal state takes the form

$$\rho \approx \frac{1}{2^n}\left(\mathbb{I} - \frac{1}{2kT}\sum_j \omega_j \sigma_z^j\right) \tag{2.3}$$

As explained in Section 1.2, the convention in NMR is to drop multiples of the maximally mixed state using the deviation density matrix picture, and to be somewhat informal about normalisation considerations. This means that in effect the thermal density matrix can be expressed as

$$\rho \propto \sum_j \omega_j \sigma_z^j \tag{2.4}$$

In the single qubit case this has a particularly simple representation which turns out to be equivalent to the pseudo-pure state. However the situation is considerably more complicated when there are multiple qubits in the system, and particularly when they are of different nuclear species, as then the Larmor frequencies ω_j are substantially different.

2.2 The Pseudo-Pure State

A pure state density matrix freshly initialised to the state $|0\cdots 0\rangle$, described in the computational basis, consists of a 1 in the top left element, and zeroes everywhere else. The equivalent pseudo-pure density matrix has some relatively small fraction of this pure state, with the rest made up from the maximally mixed state.

As an Hermitian matrix, the density matrix can always be decomposed into a sum of eigenvector matrices

$$\rho = \sum_i a_i \left|i\right\rangle \left\langle i\right| \tag{2.5}$$

where $\sum_i a_i = 1$ to preserve the trace. In the single qubit case, this is particularly simple, as there are only two eigenvectors

$$\rho_{1\text{qubit}} = a\left|\psi\right\rangle\left\langle\psi\right| + (1-a)\left|\psi^{\perp}\right\rangle\left\langle\psi^{\perp}\right| \tag{2.6}$$

representing opposite points on the Bloch sphere (although these quantum states are orthogonal to one another, it should be noted that their vectors on the Bloch sphere are at 180°). Because of the orthogonality of these matrices, the density matrix can be re-expressed using the maximally mixed state

$$\begin{aligned}\rho_{1\text{qubit}} &= (1-a)\left(\left|\psi^{\perp}\right\rangle\left\langle\psi^{\perp}\right| + \left|\psi\right\rangle\left\langle\psi\right|\right) + (2a-1)\left|\psi\right\rangle\left\langle\psi\right| \\ &= (1-a)\mathbb{I} + (2a-1)\left|\psi\right\rangle\left\langle\psi\right|\end{aligned} \tag{2.7}$$

This shows that for a single qubit system, all mixed states are pseudo-pure states. The situation is unfortunately much more complicated in larger systems. A general n qubit system will require 2^n eigenvectors to fully describe it. Using the same tactics as for the single qubit system, the maximally mixed state may be introduced

$$\begin{aligned}\rho = \sum_i a_i\left|i\right\rangle\left\langle i\right| &= \sum_i (a_i-\lambda)\left|i\right\rangle\left\langle i\right| + \lambda\sum_i \left|i\right\rangle\left\langle i\right| \\ &= \sum_i (a_i-\lambda)\left|i\right\rangle\left\langle i\right| + \lambda\mathbb{I}\end{aligned} \tag{2.8}$$

By applying the deviation density matrix approach, the $\lambda\mathbb{I}$ term may then be dropped. Now the condition for the density matrix to describe a pseudo-pure initial state becomes clear

$$\sum_i (a_i-\lambda)\left|i\right\rangle\left\langle i\right| = \epsilon\left|0\right\rangle\left\langle 0\right| \Rightarrow (a_i-\lambda) = \delta_{i0} \tag{2.9}$$

Therefore all eigenvalues of the density matrix except one must be equal to one another. A pseudo-pure state is thus a special case of a density matrix where all but one of the eigenvalues are degenerate. This is the same as for a pure state, except that in that case, all eigenvalues apart from one are zero, whereas for a pseudo-pure state the value of the degenerate eigenvalues λ may be obtained from the invariance of the trace of the

density matrix

$$a_0 + (2^n - 1)\lambda = 1 \tag{2.10}$$

$$\epsilon + (2^n)\lambda = 1 \tag{2.11}$$

$$\lambda = \frac{1-\epsilon}{2^n} \tag{2.12}$$

$$\rho = \frac{1-\epsilon}{2^n}\mathbb{I} + \epsilon \left|0\cdots0\right\rangle \left\langle 0\cdots0\right| \tag{2.13}$$

A pseudo-pure state may thus be obtained by averaging the populations of all but one of the computational eigenstates. However, unitary transformations obviously leave the eigenvalues invariant, so it is impossible to convert a mixed state into a pseudo-pure one through quantum logic transformations. For the same reason it is extremely difficult to increase the purity of the state ϵ [53], and this is generally limited by the thermal polarisation of the sample [54].

There are several non-unitary techniques which can be applied to produce pseudo-pure states, these may be roughly divided into two schools, temporal averaging [55] and spatial averaging [21]. A third quite different approach, logical labelling [56], has not found widespread experimental use, and is not considered further here.

2.3 Temporal Averaging

To characterise the principle behind temporal averaging, the two qubit case is considered in detail, from which the general case may be achieved simply by scaling up the concepts. The thermal state is a diagonal matrix in the computational basis

$$\rho_{\text{th}} = \begin{pmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \end{pmatrix} \tag{2.14}$$

and by permuting various populations, it is possible to transform this state into the following two additional states

$$\rho_{\text{th2}} = \begin{pmatrix} a & 0 & 0 & 0 \\ 0 & c & 0 & 0 \\ 0 & 0 & d & 0 \\ 0 & 0 & 0 & b \end{pmatrix} \qquad \rho_{\text{th3}} = \begin{pmatrix} a & 0 & 0 & 0 \\ 0 & d & 0 & 0 \\ 0 & 0 & b & 0 \\ 0 & 0 & 0 & c \end{pmatrix} \tag{2.15}$$

A linear superposition of these three density matrices would clearly then produce a pseudo-pure state with the populations of all but the $|00\rangle$ state averaged

$$\rho_{\text{av}} = \begin{pmatrix} a & 0 & 0 & 0 \\ 0 & \frac{b+c+d}{3} & 0 & 0 \\ 0 & 0 & \frac{b+c+d}{3} & 0 \\ 0 & 0 & 0 & \frac{b+c+d}{3} \end{pmatrix} \tag{2.16}$$

Because of the linearity of all quantum logic gates, it is possible to combine these states after an algorithm has been performed and obtain an equivalent result to combining them before computing

$$A(\rho_{\text{th}} + \rho_{\text{th2}} + \rho_{\text{th3}}) = A(\rho_{\text{th}}) + A(\rho_{\text{th2}}) + A(\rho_{\text{th3}}) \tag{2.17}$$

Temporal averaging thus performs a sequence of experiments using different starting states, and then combines the results to produce a result equivalent to that which would be achieved by producing the combined state and then performing the algorithm upon it.

In a larger n qubit system, there will be $2^n - 1$ states to be averaged over, so the most conceptually simple approach is to simply prepare $2^n - 1$ permuted states and average them

$$\rho_{\text{PP}} = \frac{1}{2^n - 1} \sum_{j=0}^{2^n-2} P_j \rho_{\text{th}} P_j^\dagger \tag{2.18}$$

where the P_j are the various permutation operations. This approach, known as exhaustive averaging, is extremely general and will work for any size of spin system and any set of initial populations [21]. Unfortunately, the number of permutations rises exponentially with the size of the system, cancelling out the potential improvement

in processing arising from the exponential growth in superposition states. This form of temporal averaging is however highly parallelisable, as each experiment can be run entirely independently on any number of individual quantum computers.

In the meantime, attempts have been made to identify more practical methods of temporal averaging, using non-cyclic permutations with uneven weightings. For example, a method for preparing a pseudo-pure state in a four qubit system has been described requiring only five permutations [57], rather than the fifteen necessary for exhaustive averaging.

It can also be constructive to decompose the pseudo-pure state density matrix in terms of product operators such as in the two spin case which gives

$$\rho_{\mathrm{PP}} = I_z + S_z + 2I_zS_z \tag{2.19}$$

Although $2^n - 1$ product operators will in general be required to fully characterise the pseudo-pure state, which might lead one to presume the same number of experiments would be required as for exhaustive temporal averaging, this description often makes clear how certain experiments can be combined. For example, in the two state case above, the first two product operators taken together form the thermal state for the system, so combining them into a single experiment is in fact easier than performing the separate experiments. It is also sometimes possible to identify certain product operators which cannot lead to any NMR signal, and these may then be neglected from the calculation [58], although this may not be applied in the general case.

2.4 Spatial Averaging

Spatial averaging techniques [59] aim to produce a single sample in a pseudo-pure state, as opposed to the multiple experiments of temporal averaging. While this is obviously desirable, both to conserve resources and conceptually, spatial averaging has been generally less popular in implementation, partly because of the relative complexity of the preparation sequences, and partly because additional control equipment is generally required.

As described previously, in order to produce a pseudo-pure state from a thermal state density matrix, the system must undergo some form of non-unitary evolution. The most obvious form of non-unitary behaviour to use is z-axis dephasing, removing all off-diagonal elements from the density matrix [30]. This has the principal advantage that it will not effect the population in the $|0\cdots0\rangle$ state, and is also relatively easy to achieve.

Provided that a system is chosen in which T_2 is much shorter than T_1, dephasing can in principle be achieved simply by waiting for this decoherence process to act. However, it is usually desirable to obtain a somewhat higher degree of control over the dephasing. This is generally achieved with "crush gradients". So called because they will remove transverse magnetisation and so rapidly "crush" the observable NMR signal, the crush gradients apply an additional magnetic field to the sample which is position dependent. Just as in T_2^* decoherence, this difference in environment across the sample causes the individual spins making up each qubit to precess at different rates while the gradient is turned on. As they get out of phase, the combined transverse magnetisation component decays away very rapidly [60]. It is this deliberate spoiling of the spatial phase coherence of the transverse magnetisation component which is described by the name "spatial" averaging.

The original spatial averaging sequence, designed for a two qubit system, is due to Cory *et al.* [28] and can be described most efficiently using product operator notation

$$\begin{aligned}
I_z + S_z &\xrightarrow{60^\circ S_x} I_z + \frac{1}{2}S_z - \frac{\sqrt{3}}{2}S_y \\
&\xrightarrow{\text{crush}} I_z + \frac{1}{2}S_z \\
&\xrightarrow{45^\circ I_x} \frac{1}{\sqrt{2}}I_z - \frac{1}{\sqrt{2}}I_y + \frac{1}{2}S_z \\
&\xrightarrow{\text{couple}} \frac{1}{\sqrt{2}}I_z - \frac{1}{\sqrt{2}}2I_xS_z + \frac{1}{2}S_z \\
&\xrightarrow{45^\circ I_{-y}} \frac{1}{2}I_z - \frac{1}{2}I_x + \frac{1}{2}2I_xS_z + \frac{1}{2}S_z + \frac{1}{2}2I_zS_z \\
&\xrightarrow{\text{crush}} \frac{1}{2}I_z + \frac{1}{2}S_z + \frac{1}{2}2I_zS_z
\end{aligned} \tag{2.20}$$

The "couple" operation denotes a period of evolution under only the spin–spin coupling for a period of $\frac{1}{2J}$, equivalent to the controlled-Z gate as described in Section 1.4.

In any sequence involving more than one crush gradient application, great care needs to be taken to avoid any unwanted "gradient echoes". These occur when logic gates between the two crushes cause the two dephasings to partially cancel out, refocussing some or all of the signal. This is precisely analogous to the spin-echo described previously, where two periods of evolution under dephasing decoherence factors can be at least partially cancelled out with a NOT gate.

When using crush gradients in homonuclear systems, it is also important to avoid any intermediate state containing any zero-quantum coherences, as these are off-diagonal elements of the density matrix which are not removed by crush gradients [61]. In heteronuclear systems, this is not an issue, and so a simpler preparation sequence may be used to prepare the pseudo-pure state [62]

$$\begin{aligned} I_z + S_z &\xrightarrow{45^\circ(I_x+S_x)} \\ &\xrightarrow{\text{couple}} \\ &\xrightarrow{30^\circ(I_{-y}+S_{-y})} \\ &\xrightarrow{\text{crush}} \sqrt{\frac{3}{8}}\left(I_z + S_z + 2I_zS_z\right) \end{aligned} \tag{2.21}$$

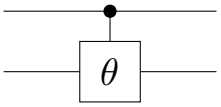
However, this sequence requires the initial state of the two spins to be $I_z + S_z$, whereas for a heteronuclear system, the ratio of I_z to S_z in the thermal state will be given by the ratio of the Larmor frequencies. Before this sequence can be performed, the polarisations of the two nuclear species must be equalised. The simplest solution to this problem is simply to perform a rotation of the appropriate angle on the spin with higher polarisation, followed by a crush to remove the transverse component and leave a reduced z polarisation, but this will result in what is potentially a substantial loss of available polarisation. More complicated sequences may instead be used to balance the polarisations without waste [21].

One obvious drawback of both of these preparation sequences is that they prepare pseudo-pure states with a somewhat diminished purity, which will produce a reduced measurement intensity compared with exhaustive temporal averaging, which always prepares a pseudo-pure state of the highest possible purity. It has historically been the case that spatial averaging techniques have sacrificed some state purity to keep the preparation sequences comparatively simple. There is however an exception to this general rule, the method of controlled-transfer gates [51].

2.5 Controlled-Transfer Gates

The concept of a controlled gate was introduced in Section 1.4.2, they are conditional gates where a single qubit logic gate is either performed or not performed on a target qubit, based on the state of a control qubit. A controlled-transfer gate is slightly more complicated, as it consists of a controlled-rotation gate, with arbitrary angle θ, followed by a crush gradient. This crush gradient will remove all the off-diagonal elements of the density matrix and, so long as the density matrix prior to the rotation was also diagonal, will not generate any zero-quantum coherences.

It is customary to use the symbol



to represent the controlled-transfer gate, with the following crush gradient being implicit. Because of the immediately following crush, the precise phase of the controlled-rotation is not important.

The action of the controlled-transfer gate is clearly to mix the populations of two of the diagonal components of the density matrix. Precisely which two states will be mixed depends on the choice of control and target qubits, although the ground state $|0\cdots0\rangle$ will be unaffected by any controlled-transfer gate. As an example, consider the two-qubit system. For the gate shown above, the controlled transfer gate will perform the transformation

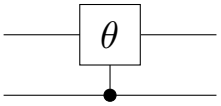
$$\begin{pmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \end{pmatrix} \rightarrow \begin{pmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c' & 0 \\ 0 & 0 & 0 & d' \end{pmatrix} \tag{2.22}$$

with

$$c' = \frac{c + d + (c - d)\cos\theta}{2} \tag{2.23}$$

$$d' = \frac{c + d - (c - d)\cos\theta}{2} \tag{2.24}$$

Its companion gate, with the second qubit as the control and the first qubit as the target, is represented by the network

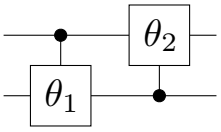


and will mix the populations b and d

$$\begin{pmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d \end{pmatrix} \rightarrow \begin{pmatrix} a & 0 & 0 & 0 \\ 0 & b' & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & d' \end{pmatrix} \tag{2.25}$$

In the special case of a system where the polarisation of one spin is twice that of the other spin, a single controlled-transfer gate with a rotation angle of $\frac{\pi}{2}$ applied to the more highly polarised spin will immediately generate a pseudo-pure state. This is a well known result that has been applied to homonuclear systems where the polarisation of one spin is halved with an initial preparation step [63], and also in heteronuclear systems where the excess polarisation of one spin is reduced to be twice that of the other spin [64, 65]. However, both these results require some polarisation reduction technique before the application of the controlled-transfer gate, so they do not produce states with the highest possible purity. The use of multiple controlled-transfer gates does allow the production of maximally pure states, developed from an original idea by Kawamura to the first experimental implementation described in this chapter.

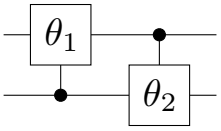
A sequence of the two gates together will serve to mix all three states b, c and d and



careful choice of the two angles will allow the preparation of a pseudo-pure state

$$\theta_1 = \arccos\left(\frac{2b-(c+d)}{3(c-d)}\right) \qquad \theta_2 = \frac{\pi}{2} \tag{2.26}$$

where the gates here are applied from left to right as usual. If the order of the gates is interchanged to give the network



then the values for b and c must be interchanged in Equation 2.26 to obtain the pseudo-pure state.

This network will average the populations of all the computational eigenstates except $|00\rangle$, which is left untouched. As long as the state $|00\rangle$ is chosen to be the one with the largest population at thermal equilibrium, not a difficult choice to make, this method will clearly generate a pseudo-pure state with the highest purity which is theoretically possible. The technique is equivalent to exhaustive temporal averaging techniques, but achieved in a single experiment.

The controlled-transfer technique described here is easiest to apply for homonuclear systems, for in this case the polarisations of the two nuclei are equal, so $d = -a$ and $b = c = 0$, giving $\theta_1 = \arccos(1/3)$ for all homonuclear systems. However, no assumptions are made about the initial populations of the density matrix, so it can be applied both to heteronuclear systems and to systems not starting at thermal equilibrium.

Some care must be taken because in some cases only one of the two networks may be used, as the other generates an impossible value for θ_1. In general for heteronuclear two-qubit systems, a solution always exists if the target of the first controlled-transfer gate is the qubit with the larger polarisation.

Controlled-transfer gates are in fact even more important in heteronuclear systems, as the conventional spatial averaging techniques in heteronuclear systems often begin with an equalising sequence to balance the polarisation of the two qubits. This is generally done simply by reducing the polarisation of the more highly polarised spins, sacrificing a significant quantity of the available polarisation in the process [62].

2.6 Asymptotic Controlled-Transfer Gates

To apply the controlled-transfer gate approach just described, precise knowledge of the initial state is required to determine the correct angles for the controlled rotations, in order to achieve the desired pseudo-pure state. This can be a limitation in samples with a very long T_1, for example, where it might be convenient to run a sequence of experiments without waiting for the sample to fully relax each time.

Applying a single controlled-transfer gate with angle 90° will serve to equalise the populations of two of the three states, precisely which two depending on which qubit is the control and which the target. By applying a sequence of these 90° controlled-transfer gates, alternating control and target qubits each time, the state will be driven asymptotically toward the pseudo-pure state.

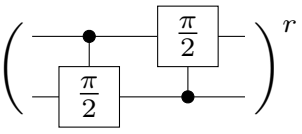
This method has the advantage that only one controlled-transfer gate is required, with a simple angle of rotation, and that it will work for any initial state. If the initial state is known in an individual case, it is always possible to achieve a faster convergence using precisely tailored gates, but this approach can be applied to any system, homonuclear or heteronuclear, in thermal equilibrium or some other initial state.

In order to measure the way in which this asymptotic method approaches the pseudo-pure state, it is necessary to define some measurement of the quality of the intermediate states. Conventional fidelity definitions depend only on the population of the ground state [66], so are not affected by controlled-transfer gates, and clearly cannot help here. Instead, it makes sense to consider the RMS deviation of the intermediate density matrix ρ from the target pseudo-pure state ρ_0, divided by the corresponding value for the initial (usually thermal) state ρ_i

$$\epsilon(\rho) = \sqrt{\frac{\mathrm{tr}[(\rho - \rho_0)^2]}{\mathrm{tr}[(\rho_\mathrm{i} - \rho_0)^2]}} = \frac{||\rho - \rho_0||_F}{||\rho_\mathrm{i} - \rho_0||_F} \tag{2.27}$$

where $||M||_F$ is the Frobenius norm of M, and the desired pseudo-pure state has $\epsilon = 0$ [51].

It is now possible to apply this measurement criterion to the intermediate states of the asymptotic approach. The circuit for this operation is



(2.28)

where r denotes the number of iterations of the sequence performed. Once again the analysis is more straightforward in the thermal homonuclear case, where the form of the density matrix elements are all known. In this case, each subsequent iteration of the asymptotic sequence will reduce ϵ by a factor of four, resulting in an extremely close approximation to the pseudo-pure state after only a few iterations. To analyse heteronuclear systems, or those beginning from states other than thermal equilibrium, precise knowledge of the initial state is required to perform the calculation. However, it is clear that for a heteronuclear system, the sequence will converge more rapidly if the first gate in each pair is applied with the more highly polarised qubit as the target.

2.7 Experimental Implementation

When preparing pseudo-pure states, it is often the simplicity of the preparation sequence which is of greatest interest to the experimenter, rather than achieving the maximum possible purity in the final state. Controlled-transfer methods offer a potentially straightforward, general purpose approach which is also able to offer the highest possible purity of pseudo-pure state. It is therefore useful to test the process to ensure it works as well in practice as in theory.

The NMR molecule chosen to perform the test was the two-qubit heteronuclear spin system, consisting of ^{1}H and ^{13}C, provided by dissolving ^{13}C-labelled sodium formate in D_2O [67]. Experiments were performed at 20°C, using a Varian INOVA spectrometer with a nominal ^{1}H frequency of 600 MHz, giving a ^{13}C frequency at around 125 MHz. Both spins were placed on resonance in their own rotating frames, reducing the internal Hamiltonian to only the spin–spin coupling term, with a measured strength $J = 194$ Hz.

The observed relaxation times were $T_1 = 14.7$ s and $T_2 = 0.35$ s for the ^{13}C qubit, and $T_1 = 5.5$ s and $T_2 = 0.61$ s for the ^{1}H qubit.

A controlled-X rotation by an angle θ could be implemented by performing the corresponding controlled-Z rotation using the spin–spin coupling, then converting this into an X rotation using a pair of Hadamard gates on the target qubit. Note that the Hadamard gates do not need to be controlled: this can be easily seen from writing out the circuit

$$U = \mathbb{I} \otimes \mathrm{H} \cdot \big(|0\rangle \langle 0| \otimes \mathbb{I} + |1\rangle \langle 1| \otimes \theta_z \big) \mathbb{I} \otimes \mathrm{H} = |0\rangle \langle 0| \otimes \mathrm{H}^2 + |1\rangle \langle 1| \otimes \mathrm{H}\theta_z\mathrm{H} \tag{2.29}$$

and as the Hadamard is a 180° rotation gate, $\mathrm{H}^2 = \mathbb{I}$, giving

$$U = |0\rangle \langle 0| \otimes \mathbb{I} + |1\rangle \langle 1| \otimes \theta_x \tag{2.30}$$

a controlled-θ_x rotation gate as desired.

The Hadamard gate is actually somewhat complicated to perform experimentally. Because its axis of rotation is not in the transverse plane, it cannot be performed using a standard control field. To achieve the correct rotation directly, the RF pulse must be applied off-resonance. Alternatively, the Hadamard can be constructed as a sequence of rotations about the principal axes of the system, for example the sequences $180_z 90_y$ and $90_{-y} 180_z$ both perform the Hadamard gate [44].

In this case, it is possible to simplify the Hadamard implementation still further. Before each controlled-transfer gate is applied, the system is guaranteed to be entirely in Z states, with no transverse magnetisation. Thus it will be unaffected by any Z rotation gates, which can simply be dropped from the sequence, so a Hadamard at the beginning of the controlled-transfer gate, implemented as $180_z 90_y$ can be replaced by a pseudo-Hadamard, just a 90_y gate. By the same argument, after the final Hadamard, the system will be crushed to a purely diagonal state, so a Z gate immediately before the gradient pulse can also be dropped, and a final Hadamard implemented as $90_{-y} 180_z$ can also be replaced with a second pseudo-Hadamard, the 90_{-y} gate [44].

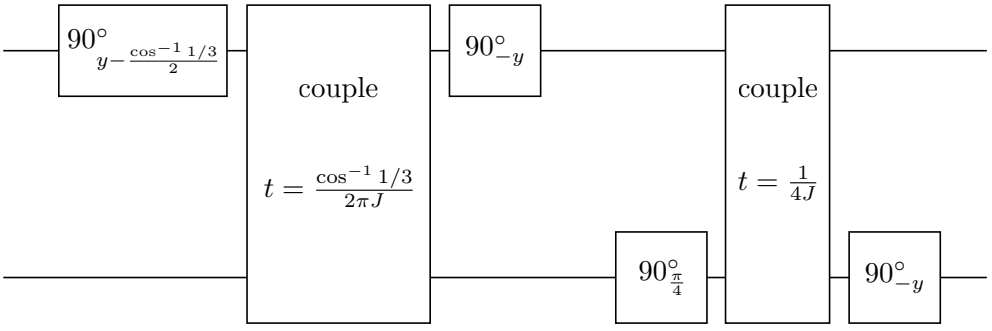
The controlled-θ_z gate may be implemented by a period of evolution under only the spin–spin coupling followed by a bilateral Z rotation. During the evolution under the spin–spin coupling, a phase difference is generated between the $|00\rangle$ and $|11\rangle$ states (where the spins are aligned) and the $|01\rangle$ and $|10\rangle$ states (with anti-aligned spins). The uniform Z rotation then applies opposite phase gains to the $|00\rangle$ and $|11\rangle$ states, while leaving the other two states unchanged. By balancing the two angles correctly, the two

phase gains for $|00\rangle$ can be made to cancel out, giving the states $|00\rangle$, $|01\rangle$ and $|10\rangle$ the same phase, while $|11\rangle$ gains a total relative phase of 2θ, where $\theta = \frac{\pi t}{J}$ is given by the coupling period t. As mentioned above, in a heteronuclear system where both spins are in individual rotating frames, there are no Zeeman couplings to be refocussed, and so the spin–spin evolution can be achieved simply by a period of delay.

However, this sequence may also be simplified. A universal Z rotation is simply equivalent to changing the coordinate system used to define the axes. It can therefore be implemented simply by changing the angle of all the gates performed either before or after it, according to the relationship

$$\gamma_z \theta_\phi = \theta_{\phi - \gamma} \gamma_z \qquad \theta_\phi \gamma_z = \gamma_z \theta_{\phi + \gamma} \tag{2.31}$$

In this way a Z rotation may be propagated either forward or backward through the gate sequence. What is more, it is only important to move it to either the thermal state, or to any gradient crush pulse, as at both of these locations, the system is guaranteed to be invariant under such a Z rotation, so they can be dropped. This is a useful example of how manipulating the reference frame can reduce the number of gates required. As each gate physically performed will inevitably contribute some kind of error to the system, keeping the number of gates required in a sequence as low as possible by these kind of manipulations is a very good habit to get in to. The gate sequence has now been reduced as much as possible and looks like



As described above, a sequence which uses multiple gradient pulses must take care to avoid accidental refocussing of any components of the signal. This can most easily be dealt with by applying gradients along different axes each time, which greatly reduces

the likelihood that any significant volumes in the sample will be refocussed. However, many NMR probes only offer a single gradient direction, usually along the z-axis. In this case, it is particularly important that the integrated strength and duration of the gradient pulse (which will determine the level of artificial phase decoherence) is different each time, while being large enough to ensure complete decoherence on each application.

In systems without gradients, it is still possible to simulate these spatial averaging techniques. During the course of my research in this area, the gradient amplifier on the spectrometer described above suffered significant component failure, rendering it highly unsuitable for publication quality data collection. For this reason, it was necessary to develop an alternative technique for performing a non-unitary crush gate.

As described in Section 1.3.1 with regard to T_2 decoherence processes (and more fully explained in 4.5.1), the process of a group of spins decohering by a gradual separation of phase vectors is exactly equivalent to the process of individual spins undergoing randomised phase flip gates. The complete distribution of an ensemble of spins across the entire transverse plane, producing a net zero magnetisation, is entirely equivalent to a population of spins, half of which have experienced phase flip gates. The decoherence process can thus be expressed by the transformation

$$\rho \rightarrow \frac{1}{2}\rho + \frac{1}{2}\mathrm{Z}\rho\mathrm{Z} \tag{2.32}$$

This approach allows the conversion of the spatial averaging technique into a form of temporal averaging. Using the linearity of the system once again, the effect of the crush gradient can be simulated by using two experiments, one of which features a Z rotation at the location of the crush pulse, and simply averaging the results. Furthermore, these Z rotations can of course be performed by simply redefining the axes of subsequent gates as described earlier in this section. For a single gradient replacement, this will be entirely familiar to NMR spectroscopists as a version of phase cycling [36]. It is also clear that this pseudo-spatial averaging technique is just an alternative form of temporal averaging.

When replacing several gradient pulses in the same sequence, as becomes necessary in particular for asymptotic sequences, the situation becomes somewhat more complicated. Obviously the number of separate experiments required increases exponentially with the

number of gradients used, and for each experiment, a different set of phases will need to be determined for each logic gate, depending on its position relative to the various "gradient" pulses.

Because of the large number of individual experiments required, going up for example to 256 for the asymptotic sequence with $r = 4$, this approach without physical gradients is unlikely to have any significant practical use, compared to the equivalent sequences using gradients, or even exhaustive temporal averaging. However, it is a useful way of demonstrating the value that might be achieved using a gradient capable apparatus, even if one is not currently available. It does also remove any danger of gradient echoes, and produces high quality spectra with good signal-to-noise ratio.

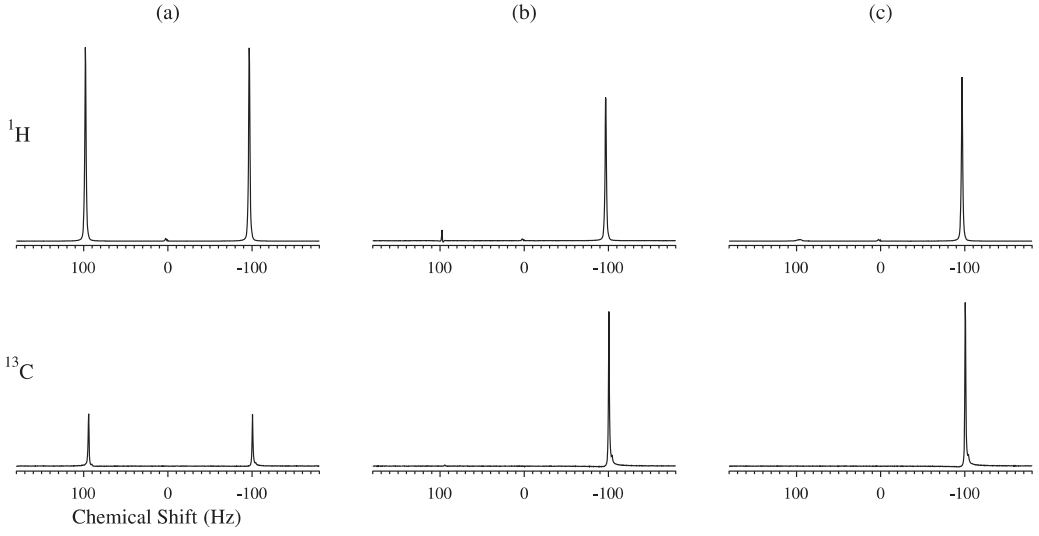


FIGURE 2.1: The controlled-transfer gate method of producing pseudo-pure states is compared with the standard spatial averaging sequence by Cory et al. Each experiment is performed twice, once being followed by a simple hard 90° observe pulse on the hydrogen nuclei, and once on the carbon nuclei. The spectrum obtained in each case is for the same nucleus targeted with the observe pulse: no signal is visible on the un-targeted spins, as the preceding crush puts them into a longitudinal state (data not shown). Following NMR conventions, spectra are plotted with chemical shift increasing from right to left. Column (a) shows the reference spectra obtained by only applying the observe pulse Column (b) shows the Cory pseudo-pure state and Column (c) shows the controlled-transfer pseudo-pure state. There is an issue of how to vertically scale spectra acquired from different nuclei as there is no absolute scaling in NMR. The approach adopted here is to plot the two spectra in Column (c) with the same height and use the same scaling for all other figures. Happily this creates the correct ratio between the spectra in Column (a), where we expect the relative heights to be in the ratio 1:4, the same as the ratio of Larmor frequencies.

Figure 2.1 shows the pseudo-pure state generated by this approach, and contrasts it to the traditional sequence due to Cory et al. described in Eq. 2.20. The thermal state is also shown for reference. All spectra are shown after applying a 90° excitation pulse to the spin to be observed in that experiment; no signal is visible without an excitation pulse, or if the unobserved spin is excited (this data, a collection of noisy flat lines, is not shown). For a true pseudo-pure state, the spectrum should show a single line on the right hand component of the doublet and no signal on the other component. The height of the line indicates the purity of the pseudo-pure state, but the absolute height is in arbitrary units, so comparisons may only be drawn between the relative heights of the lines in different spectra. As expected, the controlled-transfer approach displays the expected purity increase over the Cory sequence, and also shows a smaller error on the left-hand component.

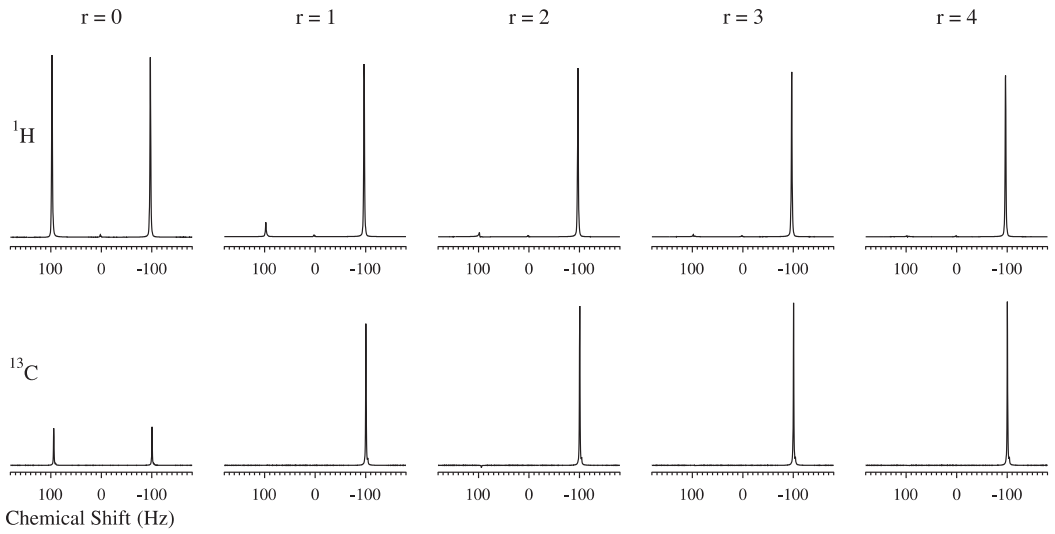


FIGURE 2.2: The asymptotic method of generating pseudo-pure states using the method of controlled-transfer gates is demonstrated. The same pulse and observe approach as described in Figure 2.1 is used to obtain the spectra. Once again the left-most spectra are the reference pair produced by doing nothing but the observation pulse, then moving to the right the number of iterations of the asymptotic method increments each time.

The results for the asymptotic controlled-transfer technique are shown in Figure 2.2. Spectra are shown with between $r = 0$ and $r = 4$ repeats of the preparation sequence. Because of the large number of individual experiments required for this data collection, experiments were only separated by 18 s, only slightly longer than the T_1 time for the ^{13}C

spin, so the initial state for each experiment was not precisely the thermal equilibrium state. This is also useful as the asymptotic approach is expected to be robust against all possible starting states. As the figure shows, the asymptotic approach causes rapid convergence to an essentially perfect pseudo-pure state, as it should.

2.8 Conclusions and Extensions

Pseudo-pure states are a vital component of NMR computing, being necessary to fulfil the second of the diVincenzo criteria. Exhaustive temporal averaging can be used to obtain pseudo-pure states with the highest possible purity, but requires a large number of separate physical experiments to obtain the data for a single calculation. Conventional spatial averaging techniques require complex sequences, which are usually simplified at the cost of final state purity.

Controlled-transfer gates offer a much simpler way to prepare a pseudo-pure state in a single experiment, and are guaranteed to generate the purest possible states. Gate networks can be precisely calculated to produce the pseudo-pure state in a fixed number of gates if the initial state is known, but it is also possible to use an asymptotic approach which requires more gates, but is more robust against a variety of issues.

Theory and experimental results have been presented here for a two qubit implementation. The controlled-transfer gate technique can also be extended to larger systems. Networks have been derived for three spin, homonuclear systems for both the direct approach and the asymptotic one [51]. It is also possible to implement with a chain topology, where only nearest neighbour interactions are present. Further work is being done by my coworkers to extend this to larger systems.

Chapter 3

Developing Control Sequences

The interaction of nuclear spins with external, radio frequency, radiation, was described in detail in Section 1.4. This is the mechanism by which control of the spins is achieved, and quantum logic gates performed upon them. Some simple gates have already been examined in the previous chapters. However, as mentioned previously, it is often extremely useful to have the ability to generate arbitrary gates, rather than relying on long, complex combinations of a universal set.

It is impossible to totally eradicate errors from experimental implementations of logic gates, and over a sequence of several gates, even very small errors can accumulate to produce a potentially serious error signal. Decoherence of various forms can also be an issue if the sequence is long. However, calculating a single gate to perform the same operations as a whole sequence of universal gates will only introduce one implementation error, and will generally be much quicker to implement, reducing these issues considerably.

3.1 Heteronuclear Systems

In a fully heteronuclear system, that is one in which each qubit is represented by a different nuclear species, the resonance frequencies of each spin will be sufficiently far away from each other that control radiation at the frequency of one will have essentially no effect on the others. This permits the use of the multiply rotating frame [68], in which each spin is considered in a separate frame rotating at its Larmor frequency, so

that all the Zeeman interaction terms disappear from the main Hamiltonian, leaving just the spin–spin coupling components.

As the strength of these interactions is generally a few hundred Hz or less, compared to a maximum power for the control radiation of around 40 kHz, it is convenient to divide an NMR quantum computation into a sequence of "pulses" and "delays". During a pulse, control radiation is applied to one or more qubits for a short period, and the spin–spin interaction is ignored during this time. In the delays between pulses, there is no control radiation, and the spins interact only through the spin–spin coupling. In reality, delay periods will often contain spin echo pulses to refocus some of the coupling terms, leaving only the desired terms to act during the delay period.

Because each control frequency only affects one qubit, it is possible in a heteronuclear system to selectively address each qubit, despite the lack of spatial resolution of the RF field. Furthermore, since during pulses the interaction between spins is ignored, each qubit may be considered as an isolated single qubit system, and the Bloch sphere picture applied without risk. All conceivable single qubit logic gates are therefore simply rotations on the surface of the sphere, and the necessary duration and phase of the radiation to achieve a particular rotation are fairly straightforward to work out analytically.

In some cases, it is useful to use somewhat more complicated implementations of a logic gate than the naive single rotation about the specified axis. Systematic errors, which will be discussed in detail in the next chapter, can often be guarded against by the replacement of the simple pulse with a "composite pulse", where a short set of simple pulses are applied back to back, with the effect of largely cancelling the error terms in each pulse [69]. However, these pulses are also relatively simple to design, as they are still only single qubit logic gates, and no consideration need be given to the rest of the system when designing them. It is of course possible to use the same gate pulse on any qubit in the system, simply by changing the frequency of the radiation. Furthermore, a number of methods have been developed for generating composite pulses to perform arbitrary rotation gates [70, 71].

For these reasons, working with heteronuclear systems makes many aspects of designing control sequences relatively simple, allowing many small scale quantum algorithms to be performed. However, because the number of spin-$\frac{1}{2}$ nuclear species available is limited,

it is clear that to obtain a system with more qubits in it, it becomes necessary to move to at least partially homonuclear systems.

3.2 Homonuclear Systems

In a homonuclear system, there are multiple qubits of the same nuclear species. Their resonance frequencies will all be slightly different depending on their positions in the molecule, the "chemical shift", but these differences will generally be much smaller than the differences between different nuclear species [72]. This means that multiple qubits will respond to the radiation from one control field at the same time (there are some exceptions to this rule, particularly in ^{19}F, where the chemical shift can often be many kHz).

A further consequence of this general response to radiation is that it is no longer possible to keep the qubits in individually rotating frames, as the phase of radiation for one qubit is no longer independent of that of the other qubits of the same nuclear type. Instead, the complete system must be considered together, and the Zeeman terms will remain in the Hamiltonian.

Pulse selectivity can be achieved, at least in some cases, by using "soft" pulses [73, 74]. By reducing the power of a pulse, and increasing the duration to produce an equivalent rotation on resonance, it is possible to reduce the frequency spread of a pulse, reducing the range of "on resonance" frequencies [73]. These pulses are also usually "shaped". Shaped pulses are an extension of the concept of composite pulses, where a pulse is divided into a large number of subpulses (typically much larger than for composite pulses), so that the power of the control field can be varied in a smooth way to fit a particular function. This is used to achieve a smoother excitation profile in the frequency domain, the most common choices being Gaussian [75], Hermite [76] and BURP [77] pulses. These pulses can excite one line and leave another line 500 Hz away unexcited, but there are numerous problems with this approach.

The principal concern with using weak pulses is the length of time they take to implement. For a very weak and long pulse, the spin–spin coupling can no longer be ignored, and it is impossible to treat a pulse as a simple rotation on the Bloch sphere, as there will be additional evolution based on the states of the other qubits in the system.

It is also necessary to consider the evolution of those other spins during such a pulse. During a standard hard pulse of a few μs, there is essentially no effect on the other spins and their state is not disturbed. However, in a long, weak pulse, perhaps lasting several ms, there will be substantial interaction due to the spin–spin coupling on all spins, and it is impossible to correct for this using spin echo techniques, partly because of the driven evolution of the coupled spin, and partly because the control transmitter is already generating the weak pulse. This means that the logic gate being applied is not truly selective, as there will be some extremely complex secondary gate being applied to the other qubits in the system at the same time.

A second issue is that it is impossible to use composite pulse sequences, as described in the section on heteronuclear systems, as these increase the length of a pulse by around an order of magnitude, which is acceptable for short hard pulses but would be totally impossible for long weak pulses. As these composite pulses are often vital to counteract systematic errors and other experimental issues, these problems will be much more severe in homonuclear systems.

In some systems the effects of decoherence may also come into play. Decoherence and relaxation processes place a limit on the maximum duration of a computation. Although some techniques exist for prolonging the working life of the qubits [78], these are difficult to combine with many low power logic pulses, and cannot prevent the eventual return of the system to the thermal state. Using very long pulses will greatly reduce the number of logic gates which may be performed before relaxation, limiting the complexity of the sequences available.

This very simple form of selectivity does not allow multiple qubits to be affected simultaneously, either in the same way, or by applying different simultaneous gates. This means that to apply a gate to two qubits in a three qubit system, for example, requires applying the gate twice in sequence, once for each of the two qubits to be affected. This obviously extends still further the duration of what is a very simple simultaneous logic gate in heteronuclear systems, increasing the sources of error already discussed. This makes the use of such logic gates, which are extremely important in spin–spin coupling refocussing for example, much less effective.

For all of these reasons, an alternative technique for calculating logic gates is required. The solution is to further extend the concept of shaped pulses. Most conventional

NMR shaped pulses have subpulses which only vary in amplitude, not in phase, and the desired shapes are chosen using simple Fourier arguments to relate the time dependent pulse shape with frequency excitation [79]. Relaxing both of these restrictions, letting phase vary over a pulse, and considering arbitrary amplitude shaping, allows for a much finer degree of control over the system, and transformations which would otherwise be impossible.

Choosing the correct shape for a pulse is of course extremely difficult to do by hand. Instead, by using a computer to simulate the evolution of the quantum system, it is possible to calculate the transformation applied by any given shaped sequence of control field radiation. Standard optimisation techniques may then be used to identify a sequence which will give the desired evolution for a chosen logic gate. As the simulation is capable of including effects such as the spin–spin coupling, off-resonance Zeeman coupling, systematic errors and decoherence effects, it can produce a sequence taking all of these factors into account, which an analytical approach to pulse sequence design could not hope to do. This rest of this chapter will focus on these joint practices of simulation and optimisation.

3.3 Simulating Quantum Evolution

To describe the behaviour of a quantum system, it is necessary to turn once again to the time dependent Schrödinger equation.

$$\mathrm{i}\frac{\partial}{\partial t}\left|\psi\right\rangle = \mathcal{H}\left|\psi\right\rangle \tag{3.1}$$

It is customary to divide the Hamiltonian into two distinct components, a fixed internal Hamiltonian $\mathcal{H}_i$, describing the background evolution of the system, and a set of control Hamiltonians $\mathcal{H}_c$ characterising the interaction of the various applied control fields with the system and applied with time varying strengths $u_c(t)$

$$\mathcal{H}(t) = \mathcal{H}_i + \sum_c u_c(t)\mathcal{H}_c \tag{3.2}$$

A formal solution to Equation 3.1 can be written as

$$\left|\psi(t)\right\rangle = U(t)\left|\psi(0)\right\rangle \tag{3.3}$$

where the operator $U(t)$ is called the propagator and has the form

$$U(t) = \mathcal{T} \exp\left(-\mathrm{i} \int_0^t \mathcal{H}(t')dt'\right) \tag{3.4}$$

and $\mathcal{T}$ is the Dyson time-ordering operator [80]. Of course, although formally correct, this solution is entirely impractical as it requires calculating time-ordered products of non-commuting operators. However, by making the Hamiltonian piecewise constant, it is possible to do away with all these issues and immediately write down

$$U(t) = \prod_j U_j \qquad U_j = \mathrm{e}^{-\mathrm{i}\mathcal{H}_j \tau_j} \tag{3.5}$$

where U_j is called the jth subpropagator of U, and $\mathcal{H}_j$ and τ_j are the Hamiltonian and duration respectively of each constant piece of the overall Hamiltonian sequence. The continuous control field strengths must also become discrete vectors u_n, so that

$$\mathcal{H}_j = \mathcal{H}_i + \sum_c u_{cj} \mathcal{H}_c \tag{3.6}$$

Any operation is then entirely described by its set of control vectors u_{cj}, and it will be this set of vectors, hereafter referred to as a "shaped pulse sequence", which will be supplied to the spectrometer to generate the relevant RF fields. Thus it is finding the correct shaped pulse sequence which is the key to designing new logic gates.

It is worth noting that this piecewise requirement does not imply significant loss of generality over the previous continuous form, as it is possible to simply discretise any continuously varying Hamiltonian sequence into a suitably fine grained piecewise constant version. As this will be done by the software implementing the experimental control sequences anyway, any discretisation at this time scale or shorter will not affect the quality of the simulation in any significant way.

This simulation step allows the calculation of a total propagator matrix from a pulse sequence. The next step is to define a quality metric which will determine how a particular pulse sequence's matrix compares to the target logic matrix the sequence is attempting to produce.

The obvious metric to use is the Hilbert–Schmidt inner product, defined as

$$\langle U_t | U_f \rangle = \frac{\mathrm{tr}(U_t^\dagger U_f)}{N} \tag{3.7}$$

where U_t represents the total propagator implemented by the current pulse sequence, U_f is the final propagator for the desired logic gate and N is the dimension of the system. However, it is also useful for the metric to be independent of any global phase difference between the two propagators, so instead the square modulus of the inner product is used

$$\Phi = \langle U_t | U_f \rangle \langle U_f | U_t \rangle \tag{3.8}$$

The quantity Φ is known as the "fidelity" and is a simple scalar value with a range between 0.0 and 1.0 describing the quality of a given pulse sequence, with higher values corresponding to superior sequences. The quality of a pulse sequence may thus be improved by seeking to maximise Φ as a function of the control vectors u_{cj}. Note also there is a corresponding quantity known as the "infidelity", given by $1 - \Phi$, which is sometimes more convenient to refer to.

Although there are some techniques which do not require it, it is generally acknowledged that the most efficient methods for finding maxima require the calculation of the local gradient to guide the search direction along the path of steepest ascent. The simplest technique for numerically calculating a gradient for a complicated function of many variables is to use finite difference methods, where the fidelity is calculated a short but finite distance along each variable in turn, and the difference in the fidelities, divided by this difference, gives an approximation to the gradient. Various more advanced finite difference methods exist, which give a more accurate approximation by using multiple evaluations in each direction, but these of course also take longer to compute. Given an n dimensional system, finite difference methods will require $\mathrm{O}(n)$ fidelity evaluations to compute the gradient.

Unfortunately, there are $\mathrm{O}(n)$ subpropagator matrices to be calculated for each fidelity evaluation, so overall calculating the gradient by finite difference methods will be an $\mathrm{O}(n^2)$ operation. This means that it is impractical to use this method with large numbers of control parameters, as the gradient calculation just takes too long.

3.4 Strongly Modulated Composite Pulses

In order to try to work around this issue, Cory *et al* [81–83] adopted a hybrid approach somewhere between traditional composite pulse techniques and the fully shaped pulse method called "Strongly Modulated Composite Pulses" (SMCP). In this approach, the number of steps in the pulse sequence is kept fairly low (generally between three and thirty) to allow the rapid evaluation of the fidelity and gradient functions. The parameters affecting each timestep are its duration and the power, starting phase and frequency of each control field.

The form of the control Hamiltonian for RF radiation at a power γ and phase ϕ in a homonuclear system (the extension to a mixed hetero/homonuclear system is straightforward) is

$$\mathcal{H}_c = 2\pi\gamma \left(\cos\phi\, X + \sin\phi\, Y\right) \tag{3.9}$$

where $X = \sum_i \frac{1}{2}\sigma_x^i$ and $Y = \sum_i \frac{1}{2}\sigma_y^i$ are the x and y rotation operators for all the spins in the system, sometimes written as F_x and F_y in conventional NMR spectroscopy.

The internal Hamiltonian is slightly more complicated. The first choice is which frame to choose as the shared rotating frame for the system. An obvious candidate is the frequency midway between the two outermost resonance frequencies, to minimise the Zeeman terms in the Hamiltonian. Unfortunately, in experimental NMR an artefact known as a "quadrature image", caused by an imbalance in the x and y FID detection circuits, can generate mirror images of peaks in the spectrum, 90° out of phase with the original signal. If the spectrum is centred between two qubit signals, each will overlap with the quadrature image of the other, thus potentially creating distortion in the measurements.

Of course, the precise choice of frame is entirely unimportant, as it is always possible to interconvert pulse sequences considered in different frames through a process known as "phase ramping" [84]. A constant phase pulse in one frame, viewed from another, simply appears to have a smoothly increasing (or decreasing) phase, which may be approximated by simply dividing the pulse into many small steps and incrementing the phase of each one.

For this reason, I have generally found it most convenient to choose the largest or sharpest line on a spectrum to define my rotating frame frequency, simply to make the process of reliably returning to the same frequency between experiments more straightforward.

Once a frame is selected, it is possible to determine the Zeeman terms in the internal Hamiltonian, simply by measuring the frequency difference between the peak and the chosen centre frequency. The spin–spin coupling may then be determined by the splitting of each peak, as described in Section 1.5.

This gives the base internal Hamiltonian $\mathcal{H}_0$ in the reference frame. Because SMCP steps are allowed to vary in frequency, the internal Hamiltonian for a particular step will be modified by the addition of the additional Zeeman terms due to the difference in rotation speed

$$\mathcal{H}_i = \mathcal{H}_0 - 2\pi\delta\nu Z \tag{3.10}$$

where $\delta\nu$ is the frequency difference between the step frequency and the reference frequency.

These two Hamiltonians allow us to calculate the propagator matrix in the frame of the timestep

$$U_{\text{timestep}} = \mathrm{e}^{-\mathrm{i}t(\mathcal{H}_0 - 2\pi\delta\nu Z + 2\pi\gamma(\cos\phi\, X + \sin\phi\, Y))} \tag{3.11}$$

Finally, the axes of the two frames will diverge over the duration of the step due to this frequency difference. Therefore to obtain the propagator in the reference frame it is necessary to apply a final Z rotation of $2\pi\delta\nu t$ to the logic gate in the timestep's frame.

$$U = \mathrm{e}^{-\mathrm{i}t2\pi\delta\nu Z}\mathrm{e}^{-\mathrm{i}t(\mathcal{H}_0 - 2\pi\delta\nu Z + 2\pi\gamma(\cos\phi\, X + \sin\phi\, Y))} \tag{3.12}$$

As the operators in the second exponential do not all commute with Z, the two cannot be simply combined, but in the absence of any pulse ($\gamma = 0$), the propagator returns to the simple case of coupling under $\mathcal{H}_0$ for all frequencies, as required.

Using this expression for the step subpropagators it is now possible to calculate the combined propagator, and from that the fidelity for any given SMCP sequence. Because the number of control parameters is generally fewer than a hundred, evaluations of the fidelity are kept fairly short, allowing simple brute force maximum finding routines such

as the simplex method [85] to be quite effective. It is also possible to further improve sequence generation efficiency by starting with a very small number of individual steps, only increasing the total when no further improvements are possible at the current number.

Strongly modulated composite pulses are very effective at identifying sequences of good fidelity, around 0.995 and above, in small systems, and particularly for simpler gates such as the selective rotations discussed in Section 3.2 above. However, they can struggle to achieve good results in larger systems, or in the implementation of more complicated gates such as the controlled-NOT. Also, experimentally the sharp discontinuities between pulses can cause transient effects, both in pulse generation and in the probe circuitry, thus reducing the quality of their experimental performance [86].

For this reason it remains desirable to develop methods for simulating true shaped pulse sequences, with many hundreds of individual steps. In particular, it is important to be able to evaluate the gradient in reasonable time. Such a technique was recently introduced by Khaneja et al [87], which they named "GRadient Ascent Pulse Engineering" (GRAPE).

3.5 Faster Gradient Calculation

The key observation in the GRAPE approach is that, although each evaluation of the fidelity for a given control sequence requires evaluating all the subpropagators, if a single parameter in the control sequence is changed, as when calculating the gradient along that parameter, only the individual subpropagator characterised by that parameter will actually undergo any change. Thus, by storing all the subpropagators during the first calculation of the fidelity, each element of the gradient can be calculated in constant time, so the overall gradient calculation goes from being $\mathrm{O}(n^2)$ to $\mathrm{O}(n)$.

The procedure is as follows: when calculating the fidelity function, store both the forward and backward combined propagators for each timestep

$$\begin{aligned} X_j &= U_j U_{j-1} \cdots U_2 U_1 \\ P_j &= U_{j+1}^\dagger \cdots U_N^\dagger U_{\text{target}} \end{aligned} \tag{3.13}$$

where U_{target} is the propagator for the desired logic gate. This formulation has the useful property that any given pair X_j and P_j can be used to calculate the fidelity

$$\Phi = \langle P_j | X_j \rangle \langle X_j | P_j \rangle \tag{3.14}$$

From this definition of the fidelity it is easy to see how the fidelity changes as the jth step of the cth control field u_{cj} is varied

$$\frac{\partial \Phi}{\partial u_{kc}} = \frac{\partial}{\partial u_{cj}} \left(\langle P_j | U_j X_{j-1} \rangle \langle U_j X_{j-1} | P_j \rangle \right) \tag{3.15}$$

Since only U_j is dependent on u_{cj}, this can be alternately re-expressed as

$$\begin{aligned} \frac{\partial \Phi}{\partial u_{kc}} = & \left\langle P_j | \frac{\partial U_j}{\partial u_{kj}} X_{j-1} \right\rangle \langle X_j | P_j \rangle + \\ & \langle P_j | X_j \rangle \left\langle \frac{\partial U_j}{\partial u_{cj}} X_{j-1} | P_j \right\rangle \end{aligned} \tag{3.16}$$

This technique is a classic example of the well established trade off between calculation speed and memory usage. For the small systems currently being explored, more than enough memory is generally available on modern computer systems, so this is an entirely practical technique to apply, effectively reducing the computational burden by several orders of magnitude.

So far, the GRAPE algorithm as described is extremely general, capable of being applied to any form of subpropagator, so long as the partial derivatives may be calculated, either analytically or numerically. However, the most common practical implementation for GRAPE sequences keeps all timesteps the same length, and uses a fixed frequency, so that the only parameters for each timestep are the power and phase of the RF field. It is usually more convenient to express these as the strengths of the x and y Hamiltonians directly so that

$$\mathcal{H}_j = \mathcal{H}_0 + \sum_c u_{cj} \mathcal{H}_c \tag{3.17}$$

where the $\mathcal{H}_c$ are the different rotation Hamiltonians (x and y) and the u_{cj} give the strength each Hamiltonian interacts during timestep j. The propagator takes the form

$$U_j = \mathrm{e}^{-\mathrm{i}\delta t \left(\mathcal{H}_0 + \sum_c u_{cj} \mathcal{H}_c \right)} \tag{3.18}$$

The convenient feature of these restrictions to the form of the propagator is that it is

straightforward to take the partial derivatives of this propagator, and they all have the same form.

To first order in δu_{kj} the change in subpropagator U_j is given by

$$\delta U_j = -\mathrm{i}\delta t \delta u_{cj} \bar{\mathcal{H}}_c U_j \tag{3.19}$$

where $\bar{\mathcal{H}}_c$ is the control Hamiltonian viewed from the stationary basis of the total Hamiltonian during that timestep

$$\bar{\mathcal{H}}_c \delta t = \int_0^{\delta t} U_j(\tau) \mathcal{H}_c U_j^\dagger(\tau) \tag{3.20}$$

and $U_j(\tau)$ is of course just the partial subpropagator up to a time τ through the timestep

$$U_j(\tau) = \mathrm{e}^{-\mathrm{i}\tau\left(\mathcal{H}_0 + \sum_c u_{cj}\mathcal{H}_c\right)} \tag{3.21}$$

Of course, for very short timesteps ($\delta t \ll ||\mathcal{H}_0 + \sum_c u_{cj}\mathcal{H}_c||$), $U_j(\tau)$ becomes very close to the identity, so the approximation $\bar{\mathcal{H}}_c \approx \mathcal{H}_c$ can be made, and this leaves the simple result

$$\frac{\partial U_j}{\partial u_{cj}} = -\mathrm{i}\delta t \delta \mathcal{H}_c U_j \tag{3.22}$$

and inserting this into Equation 3.16 further simplifies the result

$$\begin{aligned} \frac{\partial \Phi}{\partial u_{cj}} = &- \delta t\big(\langle P_j|\mathrm{i}\mathcal{H}_c X_j\rangle \langle X_j|P_j\rangle + \\ &\langle P_j|X_j\rangle \langle \mathrm{i}\mathcal{H}_c X_j|P_j\rangle \big) \end{aligned} \tag{3.23}$$

As these two expressions are complex conjugates of one another, it is clear that their sum will be real, so the expression, as well as the final stage of the calculation, may be simplified to

$$\frac{\partial \Phi}{\partial u_{cj}} = -2\mathrm{Re}\left(\delta t\big(\langle P_j|\mathrm{i}\mathcal{H}_c X_j\rangle \langle X_j|P_j\rangle\big)\right) \tag{3.24}$$

Although each timestep of this format is less flexible than the timesteps used in SMCP techniques, this allows the GRAPE technique to calculate gradients for sequences consisting of several hundred, or even several thousand, timesteps, orders of magnitude faster than was previously possible. The next step is to identify an efficient technique for using these gradients to efficiently identify high quality sequences.

3.6 Optimising along the Gradient

Knowing the local gradient allows the fidelity of the sequence to be improved, simply by moving the sequence in the direction of that gradient. A more interesting question is exactly how far along this direction to move the sequence to obtain the maximum improvement. Viewed along the gradient direction, the fidelity function will be a curve, guaranteed to be increasing as steeply as is possible from the starting position. It is obvious that to obtain the greatest possible improvement along the search direction, it is necessary to identify the nearest local maximum along this line, and move the sequence to this point, from where the next improvement direction may be calculated.

It is important that the procedure should not take too long to identify the nearest local maximum, otherwise it might be simpler just to move a fixed distance along the gradient and not bother looking for the correct maximum, relying on an increased number of steps to achieve convergence to a final maximum. However, for reasons which will be explained further in the next section, being able to find the maximum accurately allows more advanced search techniques to be employed, so it is worth spending some processing time searching for it.

It is clearly unfeasible to sweep smoothly along the gradient direction evaluating the gradient at each point to look for a maximum. Instead, the most sensible approach is to begin by identifying a length along the search direction within which a maximum must exist, a process known as bracketing the maximum. Once the bracket is established, it may gradually be contracted down to find the maximum.

A bracket can be simply identified by finding three points along the gradient direction, where the middle point has the highest fidelity. To efficiently establish a bracket, a search should expand out from the starting point exponentially so that a range of length scales can be considered. First of all a point is selected a short distance from the starting position. The third point is generated by moving $\Phi \times d$ beyond the second point, where d is the separation of the first two points and Φ is the golden ratio. If the three points describe a continuously increasing line section, the first point is discarded, the second and third points become the new first and second points and a new third point is generated, the process being repeated until the third point has a lower fidelity than the second point for the first time. The reason for using the golden ratio is a fairly

minor one, it is important that the length scale should not grow too fast with each iteration of the bracketing process, in case the local maximum is missed and the bracket considers too large a section of the curve, but it is also useful that it should not grow too slowly for efficiency reasons. While Φ is not the only number that would satisfy these requirements, it turns out to be convenient when contracting the bracket to have the three points arranged in this ratio.

In order to contract the bracket, a fourth point must be added, somewhere between the two end points. One of the end points is then discarded to leave a new triple covering a smaller area. The simplest way to choose the location of the fourth point is to make no assumptions about the shape of the curve approximated by the points, and choose the fourth point purely on the locations of the other three. By creating the fourth point in the larger interval, and so that the ratio of the two new intervals is also Φ, the two possible new triples both have the same size, so whichever is chosen, the bracket shrinks by a constant factor of Φ each iteration, a technique known as the "golden section search" [88].

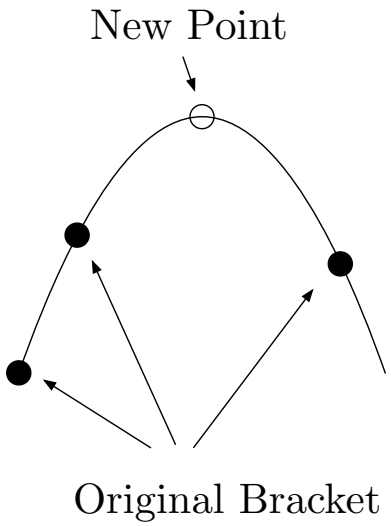


FIGURE 3.1: By assuming the curve being optimised is broadly quadratic, Brent's method can achieve much faster convergence than a simple golden section search, at least if the assumption holds. Each iteration, a quadratic is fitted through the three points of the original bracket and a fourth point is generated at the apex of this parabola. A new bracket is then constructed using the best three points out of the four.

It is possible to improve on this linear convergence using a more complex technique known as "Brent's method" [89]. This uses the observation from Taylor series that, at

least when close to a stationary point, most curves may be approximated as quadratics. Thus Brent's method fits a quadratic polynomial through the three points and uses the location of the maximum of that approximation as the fourth point, as shown in Figure 3.1. This allows an extremely fast convergence when the curve is close to a quadratic, but the method also includes an additional check on the quality of the quadratic approach, and can fall back to the golden section search for a given step if the quadratic is struggling.

3.7 Optimising the Search Direction

By using Brent's method or a comparable technique, the control sequence can be moved along its local gradient to the nearest local maximum in a single step. By definition the local gradient at the new point will be orthogonal to the previous local gradient. However, this new search direction is not usually the optimal one. Figure 3.2 shows a simple example where the steepest ascent technique requires a whole sequence of steps to converge. It is most convenient to begin considering this problem using a simple quadratic maximum. Although a non-linear function such as the GRAPE fidelity will not generally give such a simple shape to the surface, the fact that it is a smooth surface means that we can use a Taylor series to approximate the fidelity as such, provided we are sufficiently close to the maximum.

For a quadratic maximum, the surface can be represented using a matrix A as

$$f = x^T A x \tag{3.25}$$

The displacement from a starting point to the target maximum can be expressed in terms of a set of n orthogonal vectors, where n is the dimension of the system. It is clear therefore that a sequence of n steps, one in each of these orthogonal directions, and each of the correct length, would arrive at the maximum. However, it is impossible in general to calculate the correct length of these vectors. The alternative option is to use A-orthogonal vectors, which are defined by

$$x_j^T A x_i = \delta_{ij} \tag{3.26}$$

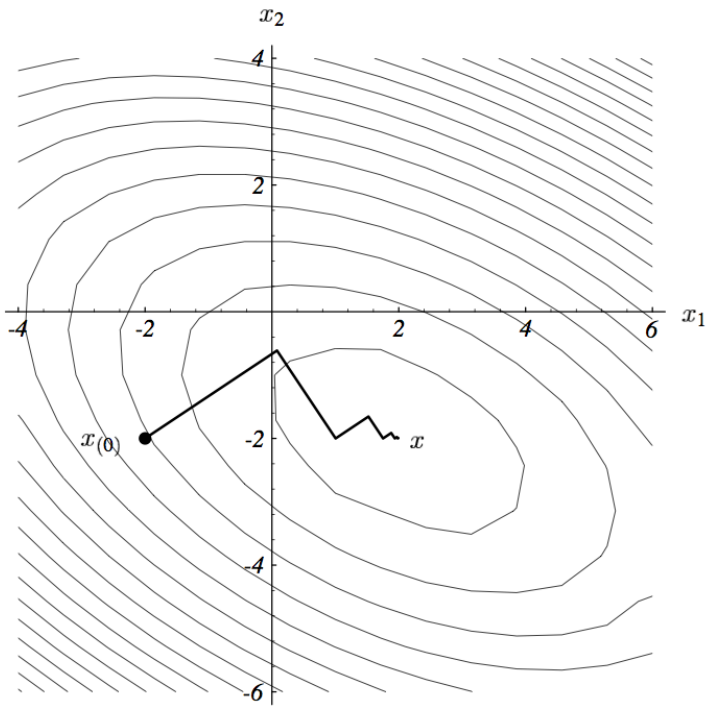


FIGURE 3.2: Even for a relatively simple maximum, it is possible for the method of steepest ascent to require a large number of steps to reach a maximum. In this case, the initial gradient is at almost 45° degrees to the eigenvectors of the maximum, and so each new search direction must also be at a similar angle, meaning that each new search direction is nearly parallel to the search direction before the previous one. This results in a highly inefficient search.

These A-orthogonal vectors will be at right angles in a co-ordinate system in which A is diagonal. Thus it is also possible to decompose the displacement of a starting point as a sum of these A-orthogonal vectors, and it turns out it is possible to correctly identify the size of each component simply by finding the maximum along that search direction. Thus given a set of "conjugate" vectors, the maximum can be reached in only n steps. Furthermore, such a set can be calculated step by step using the local gradient at each step and the Gram–Schmidt procedure. This technique, known as the "Conjugate Gradient" (CG) method, is an extremely effective maxima finding approach [90]. Figure 3.3 shows how CG can find the same maximum as Figure 3.2 in only two steps.

Of course, as mentioned previously, these techniques will not be as successful for the non-linear GRAPE as they are in the linear domain. Rather than being guaranteed to converge in precisely n steps, even CG methods struggle with some fidelity surfaces. However, using this approach considerably improves the rate at which points converge to high fidelity points, and CG is particularly important near the maxima, as here the

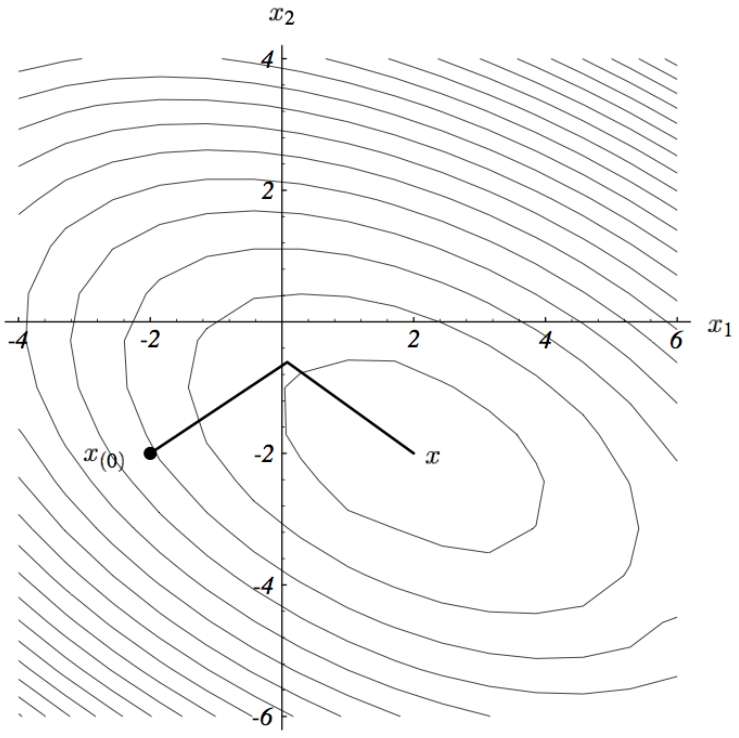


FIGURE 3.3: Starting from the same initial position as the steepest ascent approach above, the conjugate gradient technique makes exactly the same first step as the steepest ascent approach. However, because the next search direction is generated to be conjugate to, rather than orthogonal to the first search direction, it only takes two steps to find the maximum of this system.

surface often becomes very difficult to optimise using steepest ascent techniques, but the closer the sequence gets to the maximum, the better the linear approximation becomes, and the more superior the convergence.

When applying conjugate gradient convergence techniques, the sequence should be reset every n steps or so, and a new set of conjugate vectors generated from the local gradient. This prevents the build up of rounding errors, and allows the algorithm to forget sections of the fidelity surface a long way from the current area, which are less likely to be linearly related to the immediate vicinity.

The use of conjugate gradients made a very significant impact on the calculation time for many sequences, reducing the number of steps required to achieve a given fidelity by at least an order of magnitude. This demonstrates that despite the nonlinearity of the simulation, some techniques developed for use in linear systems can still be used to achieve performance improvements, even if not to the same degree as is possible in the systems they are designed for.

3.8 Other Applications of the GRAPE algorithm

The GRAPE algorithm was originally developed to expedite the generation of pulses for conventional NMR experiments. The requirements in this area are generally for point-to-point style transformations, where the initial state is well known and the goal is to bring the system to a specific final state, as quickly as possible and taking into account the effects of relaxation. In this case, an alternative fidelity can be considered which considers the overlap between final and target density matrices.

These considerations are significantly different to those required for generating quantum logic gates. A quantum logic gate is defined by a unitary transformation, and this has to be correctly performed for every possible initial state, rather than being able to rely on a particular initial state.

One of the most interesting extensions for the GRAPE algorithm to be developed is the extension of the timestep derivative terms beyond first order in the timestep length δt [91]. This is particularly important near peaks in the fidelity function, where the first order term becomes smaller faster than some of the higher order terms, so that the standard GRAPE gradient becomes less accurate close to the target positions it is trying to reach.

In this work, only first order gradient terms were considered as this was found to be sufficient to produce target pulses with sufficiently high fidelity for the small spin systems being considered. In any future work, potentially considering larger spin systems, it would certainly be sensible to investigate the improvements possible using higher order derivative components, although it should be remembered that there will be a commensurate increase in calculating time to compute these gradient terms.

When dealing with larger spin systems calculating gradients become ever more unwieldy, as the size of density matrices, propagators and Hamiltonians increases by a factor of four with each additional spin in the system. For this reason it is often necessary to make certain simplifications to the system so that sparsity can be generated and exploited in the matrix operations. This can include dynamic cutoffs, so that a small change in a parameter may result in the inclusion or exclusion of a particular term, generating discrete changes in the simulation [92]. The result of this is that these terms may no longer be differentiable using numerical techniques because of these step changes.

Instead it may be necessary to use analytical derivatives [92]. It is not yet clear whether these kind of simplifying steps are acceptable in quantum computation calculations, but for the size of spin systems considered in this work it has not been necessary to consider them.

Chapter 4

Taking Things Further

Chapter 3 discussed the idea of using a simulation of quantum evolution to develop pulse sequences to perform arbitrary logic gates with high fidelity, and presented two methods for doing this efficiently: Strongly Modulated Composite Pulses, and Gradient Ascent Pulse Engineering. However, in the naive form so far described, neither of these approaches is entirely suitable for experimental use. The inability of any experimental setup to provide the idealised quantum system considered in the previous chapters means that various additions and extensions to the core algorithms must be incorporated to provide experimentally useful sequences.

In this chapter a range of sources of error are presented, together with some techniques for mitigating their effects. Although these errors are generally very minor, when trying to achieve gate fidelities of 0.999 and above, any source of error becomes significant.

4.1 Discretised Control Strengths and Rounding Errors

It is important when using these sequence generation algorithms to use full double precision floating point accuracy for the calculations. Often a single improvement step will only increase the fidelity by around 10^{-7} or so, particularly when the current fidelity is above 0.995, and it is only a large number of such improvement steps which leads to significant improvement in the fidelity. For this reason it is important that the precision is as high as possible to prevent the individual small improvements from being lost.

Of course, it may well not be possible for experimental hardware to actually reproduce pulse sequences to this level of accuracy. Because it makes use of RF radiation, which has been well developed for use in a wide range of applications, NMR generally has extremely good reproduction of control pulses, but there are still limits. The spectrometer used for acquiring experimental data for this thesis discretises power into 1024 discrete levels between zero and maximum, and the phase is discretised to a quarter of a degree, providing a much coarser level of control than the double precision accuracy available to the simulator.

For a system with only a handful of possible control field strengths, it might make sense to simply exhaustively search through them for a sequence, rather than use the generation techniques described here. However, for any larger search space, the inability to accurately compute gradients means that the only sensible solution is to perform the search on the fine grained simulation, and then coarsen the final sequence for experimental use.

There is no easy way to force the generating algorithm to find sequences which are robust against this coarsening process. The only practical thing to do is deliberately coarsen the final sequence and test it in the simulator to check how robust it is against this process. For a given sequence it is possible to plot the fidelity as a function of the number of control strength levels available to determine how sensitive it is to this kind of treatment. It may be useful to examine a somewhat more severe coarsening than will be imposed experimentally to ensure the sequence will perform as expected.

Another issue that needs to be addressed is the time resolution of the steps. In many NMR systems this will be of the same order as the timesteps, so the length of each timestep must be chosen carefully to be an exact multiple of the timebase of the spectrometer. For GRAPE, this is not a major problem as the timestep length is constant and set by the user, so can simply be chosen to fit in with this restriction, but for SMCP where the duration of each step is variable, there will inevitably be some effect. Fortunately the steps in SMCP are usually much longer than in GRAPE, and there are fewer of them, so the rounding errors are of less concern. However it is still worthwhile to force the truncation of timesteps and check the effect in the simulator on a given sequence, rather than simply assuming that all will be well.

4.2 Extra Delays

When implementing a control sequence, there will often be additional fixed delays introduced around the actual sequence. In NMR, for example, there is generally a fixed period before and after an RF pulse to allow amplifier gating which will vary depending on individual equipment. During these periods the system will evolve under its internal Hamiltonian, which will subtly affect the effect of the pulse. Although in general these delays will only be on the order of 10μs, so that the effect of each delay is fairly small, in complicated algorithms involving many gates, the errors will grow with each delay until they can potentially cause problems. As this is an extremely simple problem to solve, there is no need to allow this form of error to creep in at all.

This problem can be resolved in two entirely equivalent ways, depending on the preference of the individual user. One solution is to calculate the propagators for the delays and use their inverse forms to adjust the target propagator demanded from the sequence generator. This method treats the delays as additions to the control sequence, and has the advantage of clarity, as each component is kept separate and the user must be aware of all the different elements, and flexibility, as it is easy to change the propagator to allow for different delay periods.

The second option is to incorporate the propagators for the delays into the simulation step, so that the propagator generated by the simulator includes the subpropagators for the delays as well as the control field section. This method has the advantage of simplicity, as an end user doesn't have to worry about the delay propagators at all, and can simply continue to use the same standard gates as before. It also allows multiple different gates to be requested without having to apply the same delay propagators to them, so although more complex to implement, it does provide a more convenient interface subsequently.

4.3 Transient Control Signals

The basic simulation steps for both SMCP and GRAPE assume that the control field properties will change instantaneously between each timestep, jumping from one set of values to the other discontinuously and remaining totally fixed at the correct value for

the whole of each step. In reality, of course, this is not feasible. There will inevitably be some period between timesteps during which the fields adjust themselves, consisting of some kind of transient decay from the old values to the new.

The exact shape that these transient signals will take is highly dependent on the design of the individual system, but a reasonable model to adopt would be some kind of oscillatory behaviour damped by an exponential decay. The precise balance between these two components will depend on the system, and may most easily be determined by simply measuring the control field.

During these transient periods the real system will not be subject to the correct control fields as assumed for the simulation, which causes a divergence of the two systems, leading to errors. In conventional shaped pulses where only the amplitude of the RF is varied, these transients are less of an issue, as the effect of transients before and after each pulse will largely cancel out. However, in GRAPE sequences, which change both amplitude and phase parameters at each timestep, the transient effects from each change will not necessarily be balanced, causing more severe error components.

The issue of transients is not a straightforward one to address. While it is possible to simulate their effects using the simple model described above, this would require a much higher number of subpropagators, as each timestep would need to be divided much more finely. This would increase the calculation time by orders of magnitude, making it essentially impractical. Furthermore, the different potential phase relationships between each control field transient would mean that it would be impossible to predict with any certainty exactly how the transients would behave.

Probably the best way to deal with transients is to reduce their effect by using a smaller number of longer timesteps. The decay of each transient is likely to be fairly constant, so using longer timesteps means that the proportion of the step affected by the transient will decrease, while there will also be fewer transients in a given duration. The best way to determine the correct duration for timesteps is by experiment. An entire family of sequences can be generated from one another, starting with a sequence with only a few timesteps and iteratively doubling the number of steps and re-optimising. The simulated performance should improve with the number of timesteps, but as transients become more of an issue experimentally, the corresponding laboratory performance will begin to decrease above a certain threshold.

Transients are likely to remain an issue in sequence generation for some time, although the precise degree to which they interfere is likely to be highly sequence and equipment dependent. Ultimately, the desire to use long timesteps to defeat transients will come into conflict with the necessity of using short timesteps to gain high fidelity sequences. I will return to examining transients in Chapter 6 which looks at them experimentally.

4.4 Control Field Limits

In any experimental setup, there will be physical limits to the strength with which the control Hamiltonians can be applied. In NMR, this is the maximum power of the RF control field, which on our system is generally restricted to about 45 kHz, particularly for long sequences. This restriction is important because too much RF power can heat the sample, which will affect the internal Hamiltonian and introduce errors into the experimental performance. Very high RF powers can even damage the probe and other equipment.

In order to encourage the sequence generation algorithm to prefer low power sequences, a penalty function may be introduced which acts to reduce the "fidelity" as the power goes up. If the penalty function is an additive one, then linearity means that the gradient function will also be simply increased by the derivative of the penalty function. A simple penalty function such as

$$\phi_{\mathrm{RF}} = -\alpha \sum_j \sum_c (u_{cj})^2 \tag{4.1}$$

reduces the fidelity according to the total RF power over the pulse sequence, weighted by the arbitrary paramter α. The corresponding gradient component is equally straightforward

$$\frac{\partial \phi_{\mathrm{RF}}}{\partial u_{cj}} = -2\alpha u_{cj} \tag{4.2}$$

In order to allow the algorithm to search over a reasonable range of RF powers, the weighting for such a function has to be fairly weak, providing gentle encouragement rather than strict demands for low power sequences. This means that it alone will not be sufficient to prevent the power from exceeding its maximum permitted level, while allowing powers close to the maximum to be examined.

One option to solve this problem is simply to return any power which exceeds the maximum permitted power back to the maximum at the end of every iteration. However, this causes considerable problems with the next iteration, as the gradient will still point into forbidden areas of the search space, Brent's method will not find a true maximum along the effective search direction and the calculated conjugate gradients will no longer be truly conjugate.

A more complex but more correct method is to introduce a much sharper penalty function into the algorithm which provides a sharper fall off at the maximum power line. This allows the penalty function to prevent sequences above the maximum power, but have little discouraging effect on ones just below. However, such a penalty function still introduces its own problems.

Because of the sharpness of the penalty function, a very narrow ridge may form along the maximum power boundary. This kind of ridge is typically very difficult for steepest ascent and even conjugate gradient methods to traverse, as usually only very small steps can be taken along the ridge. Even more importantly, if the local high fidelity maxima are all just over the maximum power boundary, the algorithm will get trapped on these local maxima created by the penalty function.

In general, for an isolated incident where the power increases beyond the maximum, it is generally worth simply resetting the starting position to a new, low power, seed value and running the algorithm again. If it repeatedly generates excessively high power sequences, it is almost certainly more constructive to examine the other parameters of the sequence first, rather than try and use a penalty function to force a high fidelity sequence where one may not exist, for example if the total duration of the sequence is insufficient for the desired evolution. In nearly all cases, I have found that the GRAPE algorithm in particular has not caused any problems of this type when the duration of the sequence and the initial seed were correctly chosen.

4.5 Decoherence

A basic description of decoherence processes in NMR was given in Section 1.3, but to include these processes in the quantum simulation, a more mathematical description

is required. The standard way to treat decoherence is using the "operator sum representation" [33]. This approach transforms a density matrix into a weighted mixture of transformed versions of itself

$$\rho \to \sum_k E_k \rho E_k^\dagger \tag{4.3}$$

The operators E_k are not required to be unitary, the only requirement being

$$\sum_k E_k^\dagger E_k = \mathbb{I} \tag{4.4}$$

in order to preserve the trace of the density matrix. Use of the operator sum representation will become clearer by observing how it is applied to decoherence processes in NMR.

4.5.1 T_2 Decoherence

Although T_2 decoherence is caused by the gradual dephasing of spins precessing at different rates due to changes in their local environments, this picture is actually entirely equivalent to the related "phase-flip" decoherence, where each spin has a constant probability in time that its phase will be flipped by 180°.

This process creates an exponential decay on un-flipped spins, tending towards the steady state of half the spins flipped and half un-flipped, leaving the average phase at zero. The proportion of un-inverted spins is given by

$$\lambda = \frac{1}{2}\left(1 + \mathrm{e}^{-t/\mathrm{T}_2}\right) \tag{4.5}$$

Thus the partially decohered state can be written in the form

$$\rho = \lambda\rho_0 + (1-\lambda) Z \rho_0 Z \tag{4.6}$$

It is immediately clear that this is precisely the form required for the operator sum representation with the operators

$$E_0 = \sqrt{\lambda}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad E_1 = \sqrt{1-\lambda}\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{4.7}$$

4.5.2 T_1 Decoherence

It is somewhat more difficult to conceptualise the precise processes occurring in T_1 decoherence, because of the interactions with external systems. The relaxation towards a particular state, in this case the thermal state, is described as "generalised amplitude damping" and can be represented in the operator sum notation with the following operators

$$E_0 = \sqrt{p}\begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-\gamma} \end{pmatrix} \tag{4.8}$$

$$E_1 = \sqrt{p}\begin{pmatrix} 0 & \sqrt{\gamma} \\ 0 & 0 \end{pmatrix} \tag{4.9}$$

$$E_2 = \sqrt{1-p}\begin{pmatrix} \sqrt{1-\gamma} & 0 \\ 0 & 1 \end{pmatrix} \tag{4.10}$$

$$E_3 = \sqrt{1-p}\begin{pmatrix} 0 & 0 \\ \sqrt{\gamma} & 0 \end{pmatrix} \tag{4.11}$$

where $\gamma = 1 - \mathrm{e}^{-t/T_1}$ sets the characteristic rate of relaxation and $p = \frac{1}{2}(1+\epsilon)$ measures how far from the maximally mixed state the thermal state is. These operators are somewhat more complicated than the phase damping set. For one thing, there are four operators necessary here, when only two were required for phase damping.

To understand this, it is best to consider the operators as two pairs: E_0 and E_1 represent damping towards the ground state $|0\rangle$ and E_2 and E_3 represent approach towards the excited state $|1\rangle$. In each pair there is one operator (E_1 and E_3) which describes flipping of the qubit state, these are non unitary operators as they describe one-way flips, losing or gaining a quantum of energy from the thermal energy of the environment. The other two operators (E_0 and E_2) are more difficult to visualise, in each case for small values of t they are close to the identity, but as time increases they act to decrease the population of one of the levels. This represents the fact that as the probability of a flip due to the other operator increases over time, if there is no flipping of the qubit state after a certain time, it becomes more likely that the qubit is already in the target state.

It is the conflicting contributions between these two pairs of operators that allows the system to relax back towards a particular mixed state. In this case, the state which is

left unchanged by the action of these operators is

$$\rho = \begin{pmatrix} p & 0 \\ 0 & 1-p \end{pmatrix} \tag{4.12}$$

which is precisely the thermal state, as required.

The combined effects of both forms of relaxation can be considered by applying both sets of operator sums sequentially. In turn, this description of relaxation may be scaled up to a larger system of qubits by simply applying the operator sums to each qubit one after the other, however, in a larger system more care must be taken with the amplitude damping coefficients, in order to ensure that the system relaxes to the thermal state, rather than to a pseudo-pure state. In the single qubit system, there is no difference, which makes it easy to become careless about such matters.

It is important to note that this decoherence model assumes uncorrelated relaxation, which is not a correct assumption. However, while the study of correlated relaxation continues to be of considerable interest in conventional NMR[93–95], it is too complex to include a complete correlated relaxation model into these quantum information processes, so instead the simpler model is used.

The fact that these decoherence processes must be described by the operator sum representation, which acts on density matrices and is non-linear, rather than the propagators used in the main simulation which are linear, means that it is impossible to consider the effect of decoherence purely on a logic gate. Instead, the initial state of the system is also required, as the effect of decoherence on the logic gate will be different depending on the initial state.

This means that decoherence cannot be included in the standard simulation and taken into account when designing new sequences. Its effects cannot thus be minimised by the optimisation routines. However, it is possible at least to use a second simulation step which can evaluate the final state when a logic gate is applied in the presence of decoherence, which will give an indication of whether to expect any significant problems to arise, even if it cannot directly solve any that it does predict.

The simplest way to incorporate the decoherence processes into the simulation is to apply the subpropagator matrices to the quantum state density matrix one by one, and

in between each one to apply the operator sum routines to decohere the density matrix. It is clear that decoherence is a continuous process, but its operators do not commute with the subpropagator matrices. Thus depending on the length of the timesteps, it may be necessary to subdivide them into multiple parts, and apply decoherence effects in between each part. The best way to find out the correct interval at which to apply the decoherence processes is simply to experiment with a few different values, the results should approach the correct result asymptotically as the interval decreases. This is because matrices close to the identity, such as the small evolutions considered here, almost commute.

An interesting side point is that, although these decoherence processes require the density matrix picture and so cannot be factored in for the calculation of unitary transformations, they can be included in simulations which only seek to find point-to-point sequences which transform a system from a known initial state into a particular final state. Although these sequences are of less interest in quantum computation, they are important for many problems in conventional NMR [87].

4.6 Systematic Errors

While systematic errors exist in many quantum computing implementations, they are not usually a major cause for concern, as any reproducible error can generally be calibrated out. It is generally simple to include the effects of a systematic error into the simulation for a sequence generator, so its effects can be taken into account and the correct logic gate still achieved. It is in ensemble quantum computers such as those using NMR that systematic errors become a real issue.

With a large number of identical quantum computers performing the same calculation, and the measured result as some kind of average value of the individual spin results, it is important that every spin should experience the same environment and control fields as all its fellows, otherwise their evolutions will diverge, reducing the final signal and introducing errors into the result. However, spatially varying systematic errors will provoke just such a divergence of evolution within the sample. There are two principal forms of spatially varying systematic errors in NMR, Pulse Length Errors (PLEs) and Off Resonance Errors (OREs).

Pulse length errors, which would be more accurately described by the name pulse *strength* errors, occur when the size of a control field deviates from its intended value in a reproducible way. This means that the rotation angle of each pulse is either increased or decreased in a linear way.

This is the major source of systematic error in NMR quantum computing, as the RF field is not sufficiently uniform across the sample volume. The simplest way to determine the uniformity of the field is to observe Rabi flopping (nutation) of a single qubit system. This may be done using a series of experiments driving the spin through a sequence of different rotation angles and observing (acquiring a spectrum). For a uniform field, all the spins should Rabi flop at the same frequency, producing a simple sine oscillation in the data. However, in a non-uniform field, spins in different locations will experience different strength fields and rotate at different frequencies. Fourier transforming the flopping data will give the distribution of field strengths. Figure 4.1 shows a sample of Rabi flopping I collected for my spectrometer, together with the corresponding Fourier transform data showing a rough distribution of pulse length errors. This shape is somewhat characteristic of a typical probe response, with a fairly wide, flattish central peak with a FWHM of about 10%. There is then some small amount of sample experiencing frequencies in the range from 50% up to 90%.

There is an interesting set of artefacts in the transformed data, mainly visible at 50% and 150%, with further smaller distortions at higher relative frequencies. These are caused by oscillations in the temperature of the room causing frequency modulation artefacts with variations in the amplifier power. It is important to remember that, although the Rabi flopping data closely resembles a FID dataset, it is in fact a collation of several thousand individual experimental runs, with each one contributing a single data point on the curve. This number of experiments takes several hours to run, during which time conditions in the laboratory can fluctuate considerably on timescales which are irrelevant during an individual data run. The spectrometer lab is air conditioned, preventing any significant deviation in temperature, but there is a small oscillation with a period of about fifteen minutes due to the set up of the thermostat.

Because of this wide range of control field strengths present in a single sample, a pulse sequence which is designed assuming the desired field strength will always be applied to all spins identically cannot perform correctly for a very significant proportion of the

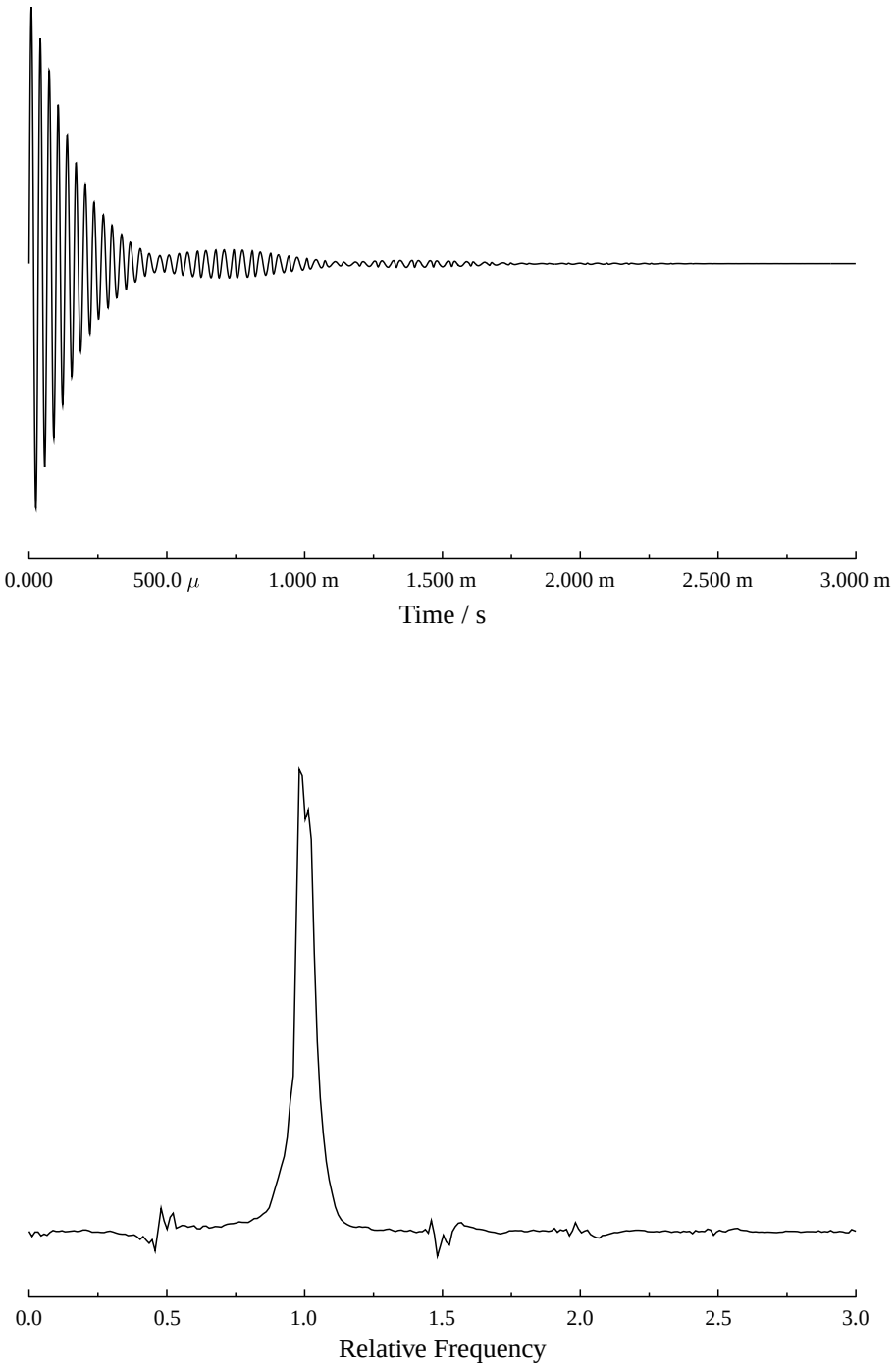


FIGURE 4.1: The top figure shows transverse excitation in a single spin system under Rabi flopping by a hard pulse on resonance. The longest excitation time is 3 ms during which time the signal size drops to below 1% of its initial value. The signal is then apodised using a Hamming function to bring the signal to zero by the end of the data to avoid any truncation artefacts. The lower figure shows the Fourier transform of this data showing the collection of frequencies at which different parts of the sample are being driven, and thus the range of pulse length errors present in the spectrometer.

sample. Instead a sequence is desired which will produce the correct response for a range of PLEs.

This can be achieved by using a composite pulse sequence [69] for which the rotation errors in each timestep cancel out, rather than adding up. In conventional NMR most composite pulses have been designed for explicitly point-to-point logic gates, which will only perform the correct transformation on certain types of initial state. Instead, general rotor or Class A composite pulses must be used which will affect every initial state correctly [69]. In heteronuclear systems where each spin can be pulsed independently, several sequences have been designed to perform these arbitrary rotations correctly in the presence of pulse length errors [71].

Early examples of these include the somewhat morbidly named CORPSE [96] and SCROFULOUS [70] families developed by Jones and Cummins from an original idea by Tycko, but probably the most successful sequence is the BB1 [97] sequence which suppresses both first and second order errors with a simple three subpulse correcting pulse, and performs very nearly as well experimentally as in theory [98]. BB1 has also been extended by Brown *et al.* to show how each order of error terms can be removed in sequence [99, 100]. However, these all use hard pulses which cannot be applied selectively in a homonuclear system. Instead GRAPE and SMCP need to be modified to generate more robust pulses.

The effects of off resonance radiation were examined in Section 1.4. Off resonance errors occur when these effects are produced when undesired. This is usually caused by deviations in the main magnetic field which causes spins in different locations in the sample to have different resonance frequencies. However they can also be caused by fluctuations in other factors affecting the Zeeman Hamiltonians, such as the temperature of the sample.

Off resonance errors are usually somewhat less significant than pulse length errors, as the frequencies concerned are unlikely to change by more than a few Hz. For many pulse sequences, this kind of shift will have little effect, but there are some sequences, such as some controlled gates, or gates designed to select only a single line in a multiplet, which require extremely sharp frequency selectivity. These sequences often exploit very narrow features in the frequency response of the spins, and only a very small shift is enough to create a significant error signal. Searching for sequences which are effective

for a range of frequencies, even a fairly small range, forces the generation of much more robust pulse sequences.

The effect of a pulse length error can be introduced into the quantum simulation by adjusting the strengths of the control Hamiltonians by a constant factor

$$\mathcal{H}_j = \mathcal{H}_i + \sum_c u_{cj}(1+f)\mathcal{H}_c \tag{4.13}$$

where f is the fractional pulse length error. Similarly, the effect of an off resonance error can be introduced by adding an extra Zeeman Hamiltonian corresponding to the frequency shift for each spin.

$$\mathcal{H}_j = \mathcal{H}_i + 2\pi g \sum_n \frac{1}{2}\sigma_z^n + \sum_c u_{cj}\mathcal{H}_c \tag{4.14}$$

where g is the frequency shift in Hz. Incorporating these terms into the simulation allows a sequence generation algorithm to allow for them in the derivation of a pulse sequence, so a particular error can be negated. By running a sequence through the simulator with several different values for the error magnitudes it is possible to test the sequence's behaviour for a range of errors. Going one step further, the simple fidelity metric can be replaced by a composite fidelity formed from a weighted average of fidelities at different errors. This composite fidelity gives a measure of the performance of the sequence over the defined range of errors. The choice of error values to be used, and the relative weightings to be assigned to each is an interesting one: the computation length will increase linearly with the number of error values, so a balance must be struck between width and flatness of error tolerance, and calculation time.

Because of the linear nature of the composite fidelity, the composite gradient can be likewise formed simply using the same weighted average as for the fidelity. This "composite search space" technique allows sequences to be identified which perform with high fidelity over a set of different error values, and for which the calculation time grows only linearly with the number of error values being considered.

Although this technique only truly optimises the fidelity for a discrete set of error values, providing they are chosen sufficiently close together the fidelity for error values in between will also be high, meaning that the sequence can be truly robust against a complete range of systematic errors. It is also important to choose the set of error values

carefully so there is no periodic behaviour in their distribution, as this can lead to the identification of sequences which display highly oscillatory fidelity distributions.

4.6.1 Pulse Length Error Tolerant Gates

Figures 4.2 and 4.3 show the theoretical performance of some example GRAPE sequences allowing for a range of pulse length errors. Figure 4.2 shows two implementations of a Hadamard gate using GRAPE, and compares them to the simple pulse. Meanwhile Figure 4.3 compares a GRAPE sequence performing a NOT gate not only with the simple pulse, but also with BB1.

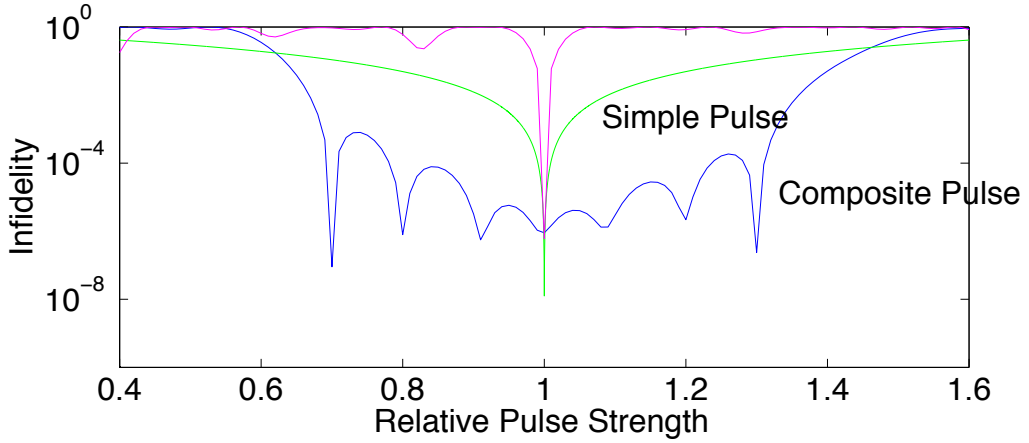


FIGURE 4.2: Two GRAPE sequences for performing a Hadamard gate are compared with the simple pulse version for a range of relative pulse strengths. The purple GRAPE line is optimised only in the absence of PLEs, and drops off even more rapidly than the simple pulse as they are introduced. The blue GRAPE line is optimised using the composite search spaces technique with strength values every 10% between 70% and 130% and displays a convincingly superior performance relative to the simple pulse.

In both of these figures it is possible to see the characteristic shape of a GRAPE PLE curve, with distinct sharp peaks of high fidelity and curving valleys of lower fidelity in between. It should come as no surprise that these peaks correspond to the specific pulse length error values used to optimise the sequences. Thus GRAPE calculated fidelity of a sequence is not generally the average or minimum fidelity experienced for any PLE in the range, but rather represents the highest values of fidelity. This means that an inspection of this kind of graph is imperative to ensure that the valley fidelities do not drop to unacceptable values.

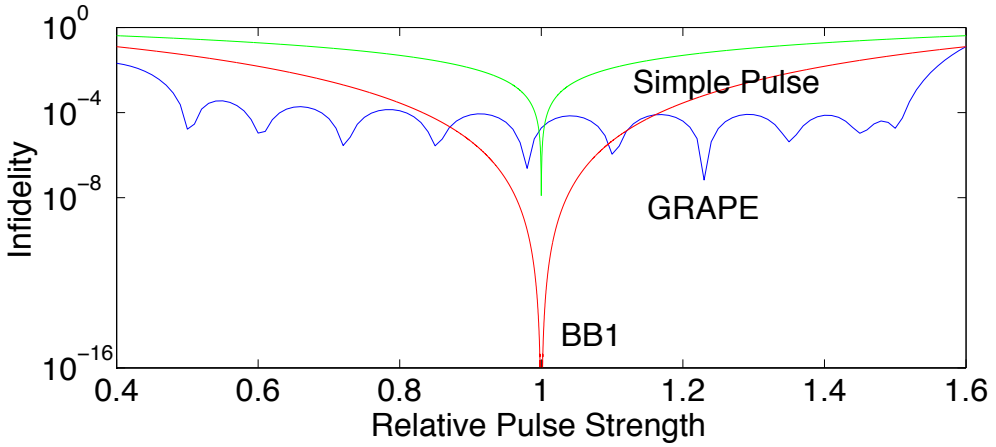


FIGURE 4.3: A GRAPE sequence for performing a NOT gate is compared with both the simple pulse version and BB1 for a range of relative pulse strengths. The GRAPE sequence is optimised using composite search spaces distributed unevenly across the range 50%-150% to avoid oscillatory behaviour. The GRAPE sequence is able to give a fidelity of better than 0.999 across this entire range, but it is unable to match the extremely good performance of the BB1 sequence with pulse length errors below 10%.

The drop between peaks can of course be reduced by bringing the peaks closer together. However, this must either compromise overall tolerance width, or increase calculation time. It is also often useful to use aperiodic intervals between optimisation values, as it is possible on occasions for the GRAPE algorithm to exploit certain periodic relationships in fidelity and produce what appears to be an extremely high fidelity sequence, which in fact drops to totally unacceptable values in between the peaks. An example of this is shown in Figure 4.4. Although this does not happen on every occasion with evenly spaced error values, as can be seen in Figures 4.2 and 4.3.

4.6.2 Combined Errors

While pulse length errors may be the dominant systematic error in most conventional NMR experiments, they are not the only one. It is also necessary to consider the tolerance of pulse sequences to off-resonance errors as well. This is also useful in testing GRAPE's ability to deliver multiply resistant error sequences, which may be more important in other systems.

Figure 4.5 shows an example of how GRAPE can generate sequences which are tolerant of both pulse length errors and off resonance errors simultaneously, considerably improving performance over the naive pulse implementation.

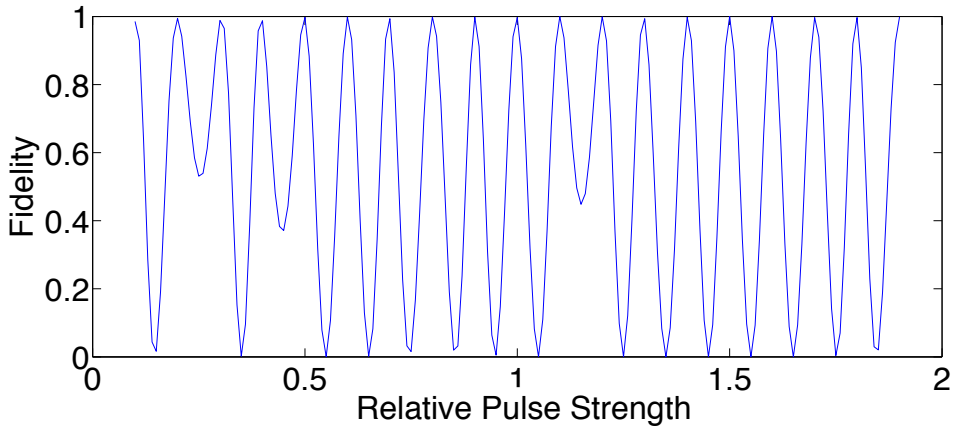


FIGURE 4.4: A GRAPE sequence for performing a NOT gate generated with a large number of equally spaced PLE control points. The calculated fidelity for this gate at the control points is better than 0.999999, but the performance outside of these highly specific regions is extremely poor. These problems can generally be countered by distributing the control points in a non-uniform manner.

This discussion of systematic errors has focussed on the spatially varying systematic errors prevalent in NMR quantum computers. However, the composite search space technique can be applied to other forms of systematic error equally well, including those which may occur in other quantum computing implementations. The other principal form of error is time varying, where the effect of a control field may vary on a timescale somewhat longer than an individual gate, but short compared to a computation or sequence of computations, such as might be caused by temperature fluctuations in a laboratory or piece of equipment. The range of variation is likely to be somewhat smaller than that typically found in NMR pulse length errors, but may still be significant enough to make it worth developing robust pulse sequences.

For many conventional shaped pulses in NMR, insight may be gained into how they achieve their results by examining plots of the amplitude and phase of the pulse. Unfortunately, for numerically designed pulses such as those derived using GRAPE techniques, there is often no clear pattern to be discerned, as the simulation is taking into account too many factors for us to consider directly. Example amplitude–phase timing plots are shown in Figures 4.6 and reffig:Ix90. Experimental implementations of both these sequences are also presented in Chapter 6, Figures 6.1 and Figure 6.9.

4.7 Suppression of Contaminant Spins

It is an altogether too common occurrence in NMR that a sample is contaminated by a small quantity of an additional molecule containing spin-$\frac{1}{2}$ nuclei of the same species as the qubits, most notably water, with its two protons. This is particularly prevalent in samples dissolved in D_2O, in which there is invariably some contamination with only partially deuterated HOD. Furthermore, unless a sample is extremely well sealed, this is also a situation which is only going to deteriorate over time.

One of the most commonly addressed issues across the field of NMR is therefore the suppression of these signals from the final spectrum [101]. As they are on different

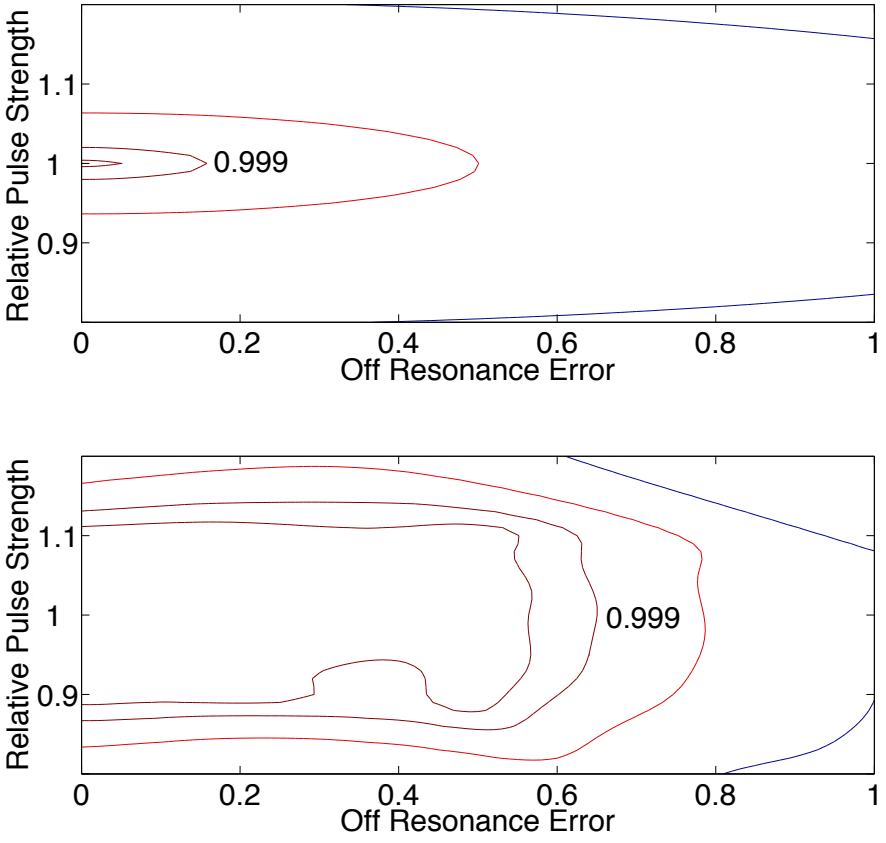


FIGURE 4.5: The top contour plot shows the fidelity of a naive hard pulse NOT gate as a function of relative pulse strengths and increasing off resonance errors. The ORE is the frequency offset in Hz. Fidelity is plotted on a log scale so each contour represents an extra 9 on the fidelity, from 0.9 to 0.9999. It is clear that the window of pulse acceptability (fidelity > 0.999) is very limited for this pulse. The lower plot shows the equivalent data for a GRAPE derived sequence, which is able to expand the highest fidelity region by more than an order of magnitude in each direction.

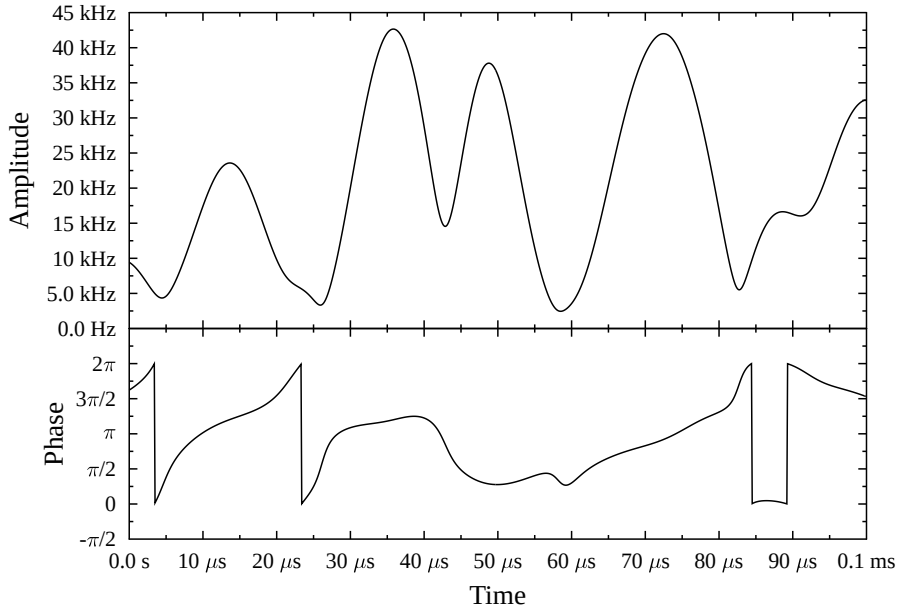


FIGURE 4.6: Amplitude–Phase timing plots for a sample GRAPE pulse. This 100 μs pulse implements a hard 90° pulse, with a tolerance to pulse length errors of 50% and a fidelity of 0.999999.

molecules from the qubits, there is no direct coupling between them and they do not interfere during the computation stage. However, the contaminating signal can be much larger than that of the qubits, and in general the action of the quantum algorithm may have placed it in any arbitrary state. An emissive or absorptive peak is unlikely to cause too much problem more than a couple of Hz away from its centre frequency, but if the line should happen to be fully dispersive, there may be significant baseline distortion more than a kHz away.

Of course, one way this problem can be corrected is in post-processing. If the contaminant peak frequency is well known, it is possible to fit a lineshape to it around that frequency and then subtract that lineshape from the spectrum. This is a reasonably acceptable technique if the qubit frequencies are sufficiently distant from the contaminant frequency that the shape of the distortion is simple and relatively minor, but will not work as well on closely grouped lines.

A second method that may be used is to selectively excite the contaminant frequency and then apply a crush gradient to the system. As the qubit spins are left in their

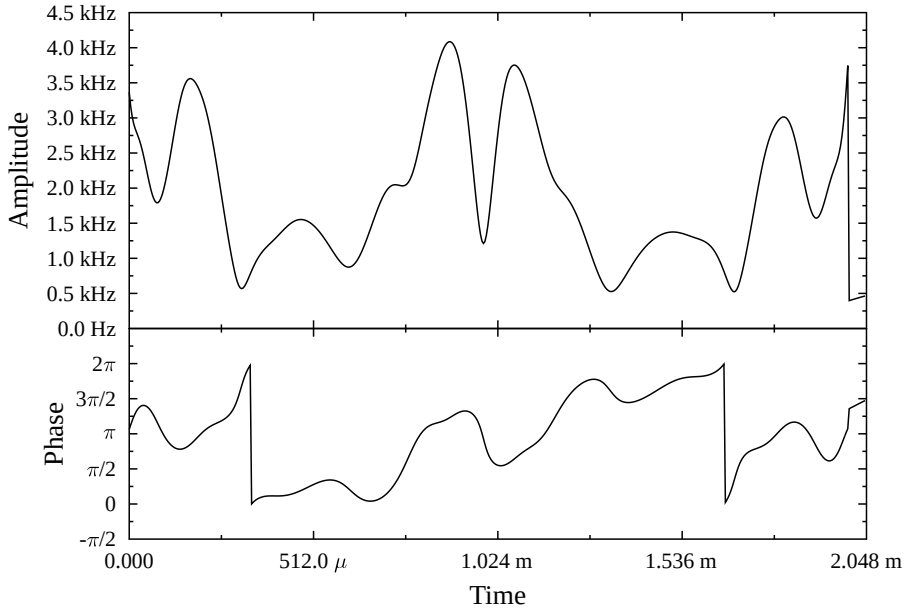


FIGURE 4.7: Amplitude–Phase timing plots for a sample GRAPE pulse. This 2 ms pulse implements a selective 90° pulse, targeting one spin in a two spin system and leaving the other spin untouched with a tolerance to pulse length errors of 30% and a fidelity of 0.9999.

thermal Z states by the selective excitation, they are also unchanged by the crush pulse, while the contaminant spins are fully decohered and contribute no average signal. Again this is much easier to achieve when the qubit frequencies are far from the contaminant spin frequency, but it is a more precise way of removing all the effects of the contaminant spin. Its principal disadvantages are that the application of further gradients may cause echo effects, resurrecting parts of the unwanted signal, and of course that it requires gradient generators as additional equipment, which not all spectrometers have available.

Ultimately, the best way to deal with this problem is to incorporate the contaminant spin into the sequence generation process. The obvious way to do this is to increase the size of the spin system by one. The Hamiltonian for the enlarged system will be very straightforward, as there is no coupling between the extra spin and the original system, only an additional Zeeman term is needed. It is then possible to use this enlarged system to generate logic gates incorporating an identity gate for the contaminant spin together with the original logic for the system.

This approach will effectively remove all signal coming from the contaminating molecule,

but introduces a significant additional computational burden. Increasing the system by one spin quadruples the size of the Hamiltonian and propagator matrices. This greatly increases the time taken to perform the necessary matrix algebra involved in generating sequences, while the additional complexity of the logic gates being generated will also have an effect on calculation times.

Instead, a better approach is to use the composite search space principles described above. Because the two systems are uncoupled, it is possible to simulate the contaminant and the computer molecule entirely separately, and generate a propagator for each system. These may then be compared to the desired gates in each case and the overall fidelity calculated as some kind of average of the individual fidelities. The additional computation required to simulate a single spin system is relatively trivial, so this allows the final result to be significantly improved for remarkably little computational effort.

4.8 Alternative Timesteps

As discussed in Section 3.5, the form of the timesteps used in the GRAPE algorithm is very rigid, only allowing the X and Y control Hamiltonian strengths to vary. This has the advantage that the gradient components are all calculated identically, simplifying the optimisation steps, but it does restrict some aspects of the sequence. In particular, the timescale of variation for the sequence is always going to be fixed by the choice of length for the timesteps. Because this is generally chosen to be short, it is possible to construct longer features using blocks of timesteps, but this behaviour is always going to be competing with short scale components.

The conventional sequence for a controlled-NOT gate involves selective gates lasting around 1 ms and a delay that can be more than fifty times longer. This kind of pulse shape is extremely difficult for the GRAPE algorithm to describe, because of the small spin–spin coupling strength and correspondingly long minimum sequence duration. When practically limited to only simulating sequences of up to 1000 timesteps or so, the minimum length of each timestep is a great deal longer than that possible when trying to generate sequences for much shorter gates. This time resolution may be too long to allow the best possible generation of the necessary selective gate components.

What is needed is a way of varying the timescale of different components of the sequence. Equation 3.24 shows that the gradient may be calculated efficiently for any form of timestep provided it is simple to calculate the gradient for each subpropagator. The standard GRAPE timestep is extremely useful as it parallels precisely the form of the control files used by the spectrometer, however it is possible to include new and alternative forms of timestep in order to allow greater flexibility in sequence generation.

The most obvious auxiliary timestep is a simple delay, of varying duration. The Hamiltonian in this case will simply be the internal one

$$\mathcal{H} = \mathcal{H}_i \tag{4.15}$$

and the propagator is equally straightforward

$$U = \mathrm{e}^{-\mathrm{i}\mathcal{H}_i\tau} \tag{4.16}$$

Finally the gradient may be calculated as

$$\frac{\mathrm{d}U}{\mathrm{d}t} = \lim_{\delta t \to 0} \left(\frac{\mathrm{e}^{-\mathrm{i}\mathcal{H}_i \delta t} - \mathbb{I}}{\delta t} \right) U \tag{4.17}$$

and expanding the exponential to first order

$$\frac{\mathrm{d}U}{\mathrm{d}t} = \left(\frac{-\mathrm{i}\mathcal{H}_i \delta t}{\delta t} \right) U = -\mathrm{i}\mathcal{H}_i U \tag{4.18}$$

Timesteps of this form can be included within the sequence, which will allow the sequence to use different timescales to achieve multiple forms of evolution and perform much longer sequences without having to greatly increase the number of timesteps.

This chapter has outlined some of the techniques which may be used to enhance and expand the core sequence generation techniques that were previously described. The next chapter will now move on to look at some computational optimisation techniques which can be used to run these algorithms as quickly and efficiently as possible.

Chapter 5

GPU Optimisation

Even with the optimisations and techniques suggested in Chapter 4, the search for a high fidelity control sequence using iterative methods can require a very large number of iterations. For two qubit systems, some target gates can require many thousands of cycles to converge to a suitable sequence. In turn, each of these cycles will involve the application of Brent's method or some similar maxima finding approach to determine the distance to move along a particular gradient line. In the cases studied, the range of step sizes studied by Brent's method between the smallest distances it may move towards a maximum and the largest it can sometimes identify is around 10^9, taking potentially hundreds of iterations to converge to the maximum along the search direction. The process is further complicated by the inclusion of systematic error tolerance, which introduce more local maxima into the fidelity landscape, and increase the number of iterations required for high fidelity sequences.

One popular way to achieve a significant speed up in calculation is to parallelise the problem. This means identifying one or more segments of code which run multiple times during the algorithm, and then distributing the workload between multiple processors running concurrently, reducing the time taken to run the algorithm. However, not all problems are easy to run in parallel. In order for the code to run at the same time on multiple processors, each run has to be totally independent of all the others. Although the conjugate gradient technique and Brent's method involve code loops which are repeated very many times during the optimisation process, they are both iterative processes. That means that each run through the loop requires the information from

the previous run as part of its input, so the order in which they are run remains strictly fixed and they cannot be performed concurrently.

During profiling of the running of the GRAPE code, it became apparent that between 80% and 90% of total calculation duration was spent inside the matrix exponential function. This function is the basic building block of the quantum simulation, used to evaluate the sub-propagator for every timestep, every iteration of Brent's method and CG. Furthermore, each matrix exponential is independent of the results of each other matrix exponential, at least for all the exponentials involved in a single fidelity or gradient calculation. This makes it a much more tempting target for parallelisation.

In order to achieve any performance improvement using parallelism, it is important that each distributed component being run on a separate processor should have enough calculation to do that the processing time is significantly in excess of the latency associated with distributing the job to the remote machines and recombining it afterwards. Because each individual matrix exponentiation takes very little time, it might be expected that the reduction in calculation time associated with dividing the exponentiations between multiple processors would be dwarfed by the overheads of communication. However, there is one auxiliary processor many computers have access to with extremely high bandwidth, and specifically designed to perform tasks with a great deal of parallelisation: the Graphics Processing Unit (GPU).

5.1 The Graphics Processing Unit

The GPU is a processor designed principally to offer hardware acceleration of many of the rendering operations used to generate images for display on a screen, allowing more complex scenes to be constructed in real time. In particular, the demands of the video game industry for ever increasing 3D graphics rendering capabilities has been the main driver in the development of modern GPUs.

Rendering 3D graphics primitives onto a 2D screen begins with a rasterisation operation, where the 3-dimensional shape is converted into a set of pixels (also known as "fragments") in the screen space. Once this set has been calculated, a "per-fragment" program is run for each pixel to calculate the final colour that should be drawn on screen.

This program can be used to apply various graphics operations such as lighting calculations, texturing and procedural effects. This process is ideally suited for parallelisation because the calculation for each pixel does not need any information about the results for any other pixel. Modern GPUs are therefore designed around the goal of optimising these fragment programs with a parallel approach.

The way a GPU achieves this is highly efficient. Because the same program is run for each pixel, the GPU is able to operate on a Single Instruction, Multiple Data (SIMD) [102] model called Stream processing [103], where multiple hardware pipelines execute the same program on their individual registers, synched to the same clock signal. This means that a single execution thread controls all operations on all datasets allowing much greater throughput of calculations than a single processing unit, while avoiding the overheads and complexities of dividing the calculation between multiple distinct processors.

Over the last couple of years the principal GPU companies have been looking to expand their market by attracting the scientific community with the development of General Purpose GPUs (GPGPUs). Considerable improvements in the sophistication of the code that can be run on the GPU, including conditional and iterative constructs, have been combined with hardware improvements such as full double precision floating point arithmetic and significantly increased number of registers and size of low level cache, to produce so called "personal supercomputers". The most high-end GPGPUs no longer actually have a connector capable of outputting video to a monitor for display.

At the present time, the most well developed GPGPU programming environment is nVidia's [104] proprietary Compute Unified Device Architecture (CUDA) system [105], which is a recursion-free subset of the C language. The other principal competitors in the GPGPU arena are Microsoft's DirectCompute [106, 107] and the open standard OpenCL [108] but they do not yet have as many supported features or as much stability as CUDA.

All GPU work for this project was carried out on the Oxford Supercomputing Centre's (OSC) [109] SKYNET cluster using the nVidia Tesla M2050 GPU. This is therefore an appropriate moment to examine the more important features of this particular GPU to highlight the potential it offers and challenges it faces.

5.2 The Tesla M2050 GPU

The Tesla M2050 is based on the CUDA architecture called "Fermi", and offers a peak performance of 515 Gigaflops at double precision, giving it approximately 10x the calculation horsepower of a standard quad-core x86 CPU [110]. At the time of writing, this processor is the most powerful GPU available for High Performance Computing (HPC) applications.

As with all CUDA GPUs, the M2050 is divided into a set of independent "streaming multiprocessor"s (SM). Each identical SM then consists of 32 hardware cores, each with 1024 registers. The SM also contains 48kb of "shared" memory, extremely low latency memory which can be accessed by all the cores on the SM but not any other SM, 16kb of L1 cache to reduce the latency of accessing the main device memory and an 8kb cache explicitly for storing constant values. The SM will support up to 48 threads running concurrently on each hardware core.

The M2050 has 14 SMs, giving a total availability of 448 hardware cores and more than 21,500 threads running at the same time. It also has a total device memory shared between the SMs of 3Gb, with a bandwidth of 140Gb/s.

Within each SM, all 32 hardware cores execute the same instruction in synch with each other, using their individual datasets. This means that any program running on the GPU requires a minimum of 32 threads running at the same time to avoid wasting the extra capacity. When more threads are required, they are divided into blocks of 32 called "warps", and execution then alternates between "active" warps, with some warps becoming "inactive" while they are waiting for data or previous instructions to complete.

It is obvious that using fewer than 32 threads at a time wastes resources as some hardware cores are not operating on real data, but it is somewhat less intuitive that the key to achieving the highest possible efficiency on the GPU is to create as many threads as possible. Many operations take several clock cycles to complete, and a given warp may be stalled waiting for a single operation to complete for all of that time. If multiple warps are available, the processor can overlap their execution so while one warp is waiting for a result, the next one can start processing. This is even more important in the case of memory access from the main device memory. This can take several hundred cycles to complete, potentially an enormous waste of resources. However, with 40 warps running

on the SM, this delay only corresponds to around 10 cycles per warp, so in most cases the warps have enough to do between reads to hide this latency as well.

It is important to note that there are some limiting factors to take into account. Each hardware core only has access to 1024 registers, which must be shared between the threads, so to get 48 warps of threads, each thread would only have access to 21 registers, which may be insufficient for some programs. Shared memory is also limited, 48 warps of threads would only get access to 32 bytes of shared memory each, enough to store only 4 double precision floating point numbers.

5.3 The Matrix Exponential on the GPU

Much has been written on the varying techniques which may be used to numerically approximate the matrix exponential. The standard diagonalisation approach used analytically suffers from several drawbacks when used numerically, it is quite slow and not at all robust for certain types of matrix.

The two most generally applied techniques are both iterative, the Padé and Taylor series. Of these, the Padé series converges in fewer iterations, but requires much more calculation per step. The Padé technique is also more robust in exponentiating general matrices. As with all iterative techniques, the first few terms provide best agreement close to the zero point of the approximation and each new term expands the region of agreement. Thus, the closer to the zero matrix the Hamiltonian is, the fewer iterations will be required to achieve a good approximation to the exponential. Fortunately, one of the requirements for the GRAPE algorithm to work correctly is that the timesteps be short compared to the control field strength, so the Hamiltonian matrices are already quite small.

On top of this, it is possible to make use of the so-called "scale and square" technique. This makes use of the fact that

$$e^{H} = e^{H/2} \times e^{H/2}$$

so it is straightforward to make the matrix to be exponentiated closer to the zero matrix by dividing it by two several times, and then simply squaring the result a corresponding

number of times to obtain the final answer. The simple scalar division and single matrix multiplication of a "scale and square" is somewhat cheaper than calculating an additional term in the expansion for either Padé or Taylor series, so it is useful to experiment and identify the most appropriate ratio of operations.

By assigning one thread on the GPU to each matrix to be exponentiated, it is possible to exponentiate up to 448 matrices (the number of individual hardware cores on the GPU) in the same time as exponentiating one. This sounds like an extremely impressive performance boost, but there are several factors which mean that this speed up is not achievable in practice, and that other techniques must be used to exploit the full potential of the GPU.

5.3.1 Limiting Factors on the GPU: Core Count vs. Flops

The first thing to note is that the clock frequency on a modern GPU is much lower than that of an equivalent CPU. The principle reason for this is power consumption. Every clock tick, resistance in the processor circuitry causes heat to be deposited in the chip. This heat must be dissipated at the same rate it is being produced in order to prevent the chip from overheating and damaging itself. This is what limits the clock frequency on both the CPU and GPU, but because the GPU contains so many more cores, the clock must run slower. This means that although the GPU should in principle be able to perform 448 exponentiations as fast as it can perform one, it could not perform that one operation nearly as fast as the CPU could.

Instead, the standard metric to compare the processing capability of a GPU and a CPU is the maximum number of floating point operations per second (flops) performed by each device. This gives a more accurate representation of the relative time two processors are likely to take to perform a given task. As noted above, the M2050 GPU can produce 515 Gigaflops at double precision, giving it approximately ten times the calculation horsepower of a standard quad-core x86 CPU. This means that running the exponentiations on the GPU the same way as they are run on the CPU, without any further concessions to the parallel architecture, we would expect a performance improvement of no more than about an order of magnitude.

5.3.2 Limiting Factors on the GPU: Unused Capacity

The GPU may be able to exponentiate 448 matrices at the same time, but the number of timesteps in a particular GRAPE sequence may not be precisely 448. Many sequences will use far fewer timesteps to reduce the search space. When acting on a smaller number of steps, the remaining processing capacity of the GPU will be unused.

The best way to reduce this effect is to create more than one thread per exponentiation. The more threads that are used in total, the larger the fraction of them that will be running with a fully occupied GPU. As an example, consider a GRAPE sequence with 128 timesteps. Using one core per exponentiation, the fraction of available capacity being used is only $128/448 \approx 28.5\%$.

However, when using series approximations to find the exponential, each step consists of simple linear operations such as multiplication and addition. These can be effectively parallelised by distributing each exponentiation across 16 cores, one core per element of the matrix. In this case the total number of threads will be $16 \times 128 = 2048$. The first 1792 of these threads will be run with the GPU at full capacity, with only the final 256 being run on a partially unused processor. This means that the efficiency of the GPU in this case will be $2048/2240 \approx 91.4\%$.

On top of this enormous improvement in efficiency (and therefore in performance), increasing the number of threads running at any one time will also improve the performance by allowing the compiler to hide pipeline stalls and delays by interleaving steps for all the threads running on each core. For example, the compiler may be able to use multiple components of its Arithmetic Logic Unit (ALU) at the same time, with one warp's threads multiplying two numbers while another's are adding two numbers.

5.3.3 A Truly Parallel Matrix Exponential Implementation: Compute Limited or Bandwidth Limited

Probably the most important thing to consider when designing a parallel algorithm is whether it will be compute limited, i.e. the speed of execution is determined by the number of flops available, or bandwidth limited, where the speed of execution is determined by the time taken to transfer necessary data to and from the GPU.

The GPU is naturally well suited to problems which are compute limited, as it offers a very high flop rate, so as long as the algorithm is well designed and uses all available flops, the GPU will give substantially superior performance to a CPU. However, partially due to expense, and partially due to the relatively recent development of GPGPU concepts, the currently available GPUs do not have the same sophisticated cache level design as modern CPUs, so algorithms with a large number of memory reads will usually be limited by stalls while waiting for fetches from memory. Furthermore, the GPU is generally connected to the CPU and main memory by a PCIe bus or something similar, which is far slower than a read from RAM, so for problems which require a large quantity of data to be transferred either to or from the GPU, this can introduce a serious delay.

In the case of a two qubit quantum system, the Hamiltonian will be a 4×4 complex matrix, requiring $4 \times 4 \times 2 \times 8 = 256$ bytes per matrix. We might thus expect that transferring say 512 timestep Hamiltonians onto the GPU to be exponentiated would require sending more than 131kb of data to the GPU. However, because our Hamiltonians are constructed from a linear combination of the constant X and Y control Hamiltonians and the internal $\mathcal{H}_0$, we can store these three matrices on the device ahead of time, and then only send the X and Y control strengths to the GPU, reconstructing the Hamiltonians on arrival. This reduces the necessary data transfer, and therefore the duration of this component of the operation, by a factor of 16.

5.3.4 A Truly Parallel Matrix Exponential Implementation: Thread Count

The next step in designing a parallel algorithm is to determine the number of threads we can divide the operation between. As discussed above, simply assigning one thread to each matrix to be exponentiated does not make nearly as much use of the potential of the device as we would like. Instead, for a 4×4 matrix the logical first choice is to use 16 threads, one to each component of the matrix. Assigning one core to each element of an array is generally good practice as it avoids the possibility of multiple cores trying to write to the same element of the array at the same time, meaning that no locks or other multithreaded precautions are required. Furthermore, it is particularly efficient when working on a synchronised processor like a GPU, which supports a simultaneous

write across all threads in a warp when the target memory locations are in a continuous block.

Because the Hamiltonians are complex, there is also a case to be made for using a full 32 threads per matrix, with a real and imaginary thread for each component. However, although this would potentially increase the core occupancy by ∼1-2%, there would also have to be some duplication of data between real and imaginary threads, wasting memory resources as well as necessitating some duplicated memory reads, so an overall increase in performance is unlikely, while the code required would become substantially more complex, increasing the number of bugs and development time.

One consequence of the single operation memory write to aligned memory is that, as each core operates first on the real part of its matrix element, then on the imaginary part, it is essential that the matrices be stored in a structure of arrays format, where the real components form one continuous block and the imaginary components form a second, rather than the more common array of structures, which stores each real and imaginary component as a neighbouring pair.

5.3.5 A Truly Parallel Matrix Exponential Implementation: The Final Exponential Algorithm

As discussed at the beginning of this section, a choice must be made between the Taylor and Padé approximations in calculating the matrix exponential. For small matrices such as the ones involved in the GRAPE algorithm, the Taylor series provides a stable approximation and is somewhat quicker and simpler to calculate than the Padé series. For this reason, an exponential approximation using seven terms of the Taylor series and two "scale and square" operations was used, as this was enough to give results correct to full double precision with the maximum possible power for the longest considered timestep duration. It would be possible to implement a more flexible routine varying the number of terms based on the size of the matrix or the size of each term as they drop to zero, but the conditional branching and inter-thread communication necessary would greatly add to the overheads of the operation, thus being unlikely to produce any overall speed-up in the running.

Both series require the calculation of a whole sequence of matrices, which must be kept in memory at the same time. For each of these intermediate products, the same core is responsible for writing the value at a particular element as for the final matrix. Although each core is responsible for writing to only a single data element in each matrix, various matrix operations such as multiplication require that the cores be able to read the values of other components from the other intermediate matrices. To enable this communication between threads, the GPU makes use of *shared memory*.

Shared memory is a particularly low-level block of memory that sits alongside the L1 cache in each SM and offers an extremely short read and write time. The full set of 32 threads in a warp can access 32 adjacent (and in-order) memory elements in only a few clock cycles, roughly the same latency as the registers. It is also possible, although somewhat slower, for each thread to access random elements of shared memory, allowing communication between the threads. Because operations are synchronised between all cores in the SM, there will not be any race conditions between reading and writing threads.

5.4 Computing products of propagators

Calculating the fidelity of a control sequence requires more than just exponentiating the Hamiltonian matrices for the subpropagators. It is also necessary to compute the combined in-order product of all the subpropagators in order to obtain the final logic matrix. To calculate the gradient the complete set of forward and backward partial propagators are also required.

Working out a sequence of matrix products like this initially seems like an obviously serial problem, not at all suited for parallelisation, as each product in the chain requires the previous product to have been calculated already. However, by exploiting the associativity of matrix multiplication, the problem can be made at least partially parallelisable.

5.4.1 The Parallel Reduction

The conventional approach to calculating the combined product of a sequence of matrices is to group them in the form

$$(U_n(U_{n-1}(\cdots(U_3(U_2U_1)\cdots)$$

This sequential approach uses n steps, with one matrix multiplication in each step. However, by changing the grouping of the brackets like this

$$(U_nU_{n-1})\cdots(U_4U_3)(U_2U_1)$$

then more than one matrix multiplication may be performed concurrently, rather than having to wait for the previous result to be completed. This approach allows the use of only $\log_2 n$ steps, with the number of multiplications per step beginning at n/2 and halving each step, down to the final step $U_f = U_{n\cdots n/2+1}U_{n/2\cdots 1}$. On the GPU, all the multiplications in one step can be performed in sync with each other, so the overall duration of the scan operation is reduced by a factor of $\frac{n}{\log_2 n}$.

5.4.2 The Parallel Reduction: Implementation Techniques

In the matrix exponentiation function, each set of 16 threads are only responsible for one matrix (and associated intermediate variables) so each thread only needs to share data with the others in its set. However, in the reduction operation, after the first step, each set is accessing the results of calculations carried out by a different thread in the previous step. By the final step, the last thread block will have had to access the results from all other threads.

As described above, the fastest way to handle inter-thread communication like this is using shared memory, but this is only shared between threads on a single SM. In order to make use of all 14 SMs at the same time, some communication has to be done through the main device memory between the different SMs. However, this takes much longer to do, a typical device memory fetch is around two orders of magnitude slower than from shared memory.

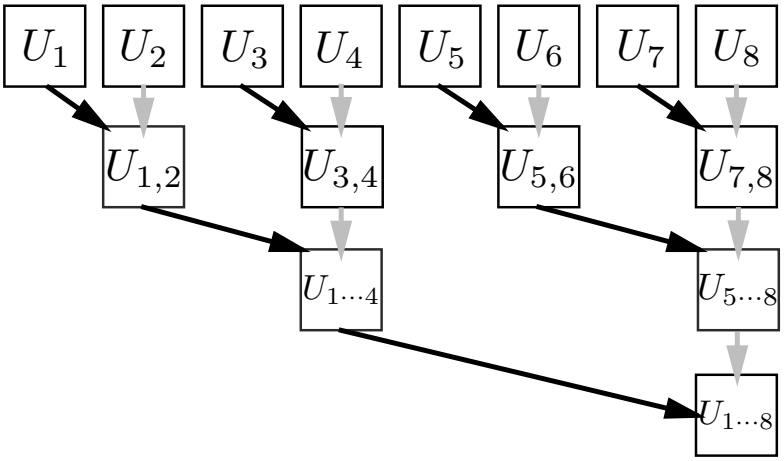


FIGURE 5.1: The reduction operation calculates products of pairs of matrices, then the products of pairs of the products, and so on until it has the complete product of the entire set. Each black arrow here represents a multiplication operation being carried out and each row of boxes represents one step in the calculation. By operating in parallel, the GPU can perform the seven multiplications required to calculate $U_{1\cdots8}$ in only three steps, less than half the time that would be taken using a sequential approach. The last step of the calculation is to read the result from the previous block of threads (U_0) and calculate the combined product $U_{0\cdots8}$ so that this operation can be spread over multiple SMs, but the results of the calculation can be easily combined.

The solution to these two problems is to use a hybrid approach. As each SM launches, it takes a block of matrices to reduce, storing all results in shared memory for maximum efficiency. When it has calculated the combined product of its block, it then waits for the SM calculating the previous block to finish its calculation and post the result back to main memory, from where it can access it. It multiplies the previous running total by its own block result, then returns the updated total to main memory and signals to the next SM that it can make use of the new result. While this may at first glance seem to keep all the SMs waiting in a queue for all the previous calculations to finish, thereby doing away with the parallel potential, the vast majority of each SMs calculations can be performed before the waiting, so once an SM receives the signal that the previous SM has finished, it need only perform a single memory read of the previous calculation matrix, one matrix multiplication, and a write back of the result.

The first task for a block of threads launching on an SM is to generate a unique ID

number to decide which group of matrices it will reduce. This will ensure that there is no duplication of calculations between SMs. It is also important that the ID numbers should be assigned in launch order, so the first SM to launch takes the first block of matrices. This increases the probability that the blocks will also finish in order, so reducing the waiting delays, and also prevents the possibility that the blocks running on the SMs are not waiting for the results from a block which has not yet launched.

The simplest way to do this is to maintain a global static variable, with an initial value of zero, counting the number of launched blocks of threads. Each time a new block launches, it takes the current value as its ID, then increments the total. In a multi-threaded environment, however, it is somewhat more complicated, as there is a danger that multiple threads will read the same value before any of them update it, resulting in them all taking the same ID number. To avoid these "race conditions", the CUDA environment provides a set of "atomic" operations, which perform a memory read, manipulation and write-back which are guaranteed to happen uninterrupted by any other threads. This approach will enforce serialisation of operations on that memory, which can have a significant impact on performance, so it must be used sparingly, but it is very important for risk free inter-SM communication.

The second inter-SM communication which needs to occur is the signal each SM must send when it has finished calculating, signifying that it has updated the matrix in global memory to contain the combined product up to the end of its block. Once again, the simplest way to do this is to maintain a global counter variable which keeps track of how many thread blocks have so far performed this operation. At the end of its program, when the intra-block reduction is completed, a thread block goes into a waiting loop, continually reading this value until it reads the same as its own ID number. When this loop exits, it knows it can safely read the stored value in device memory, and calculate the combined product, then return this value to device memory. Once this operation is completed, it can safely increment the counter, causing the SM with the next ID to exit its holding loop.

5.4.3 The Parallel Scan

The reduction operation is ideal for calculating the final propagator for a particular GRAPE sequence, which is all that is required for calculating the fidelity of that sequence. However, in order to use GRAPE to calculate gradients, the complete set of forward and backward partial propagators are necessary. While the traditional sequential procedure calculates these values as intermediate products on the way to the final result, the parallel reduction produces a completely different set of intermediate products which are of no immediate use, as shown in Figure 5.1.

Fortunately there is a second operation which can be performed after the reduction to obtain the results required, turning the procedure into a full scan operation. The reduction is a downward pass over the data, updating every other product of the previous step each step, halving the number of elements. To complete the scan, a second, upward pass is performed, where the number of elements being calculated is doubled each time, and the previously abandoned elements are updated to the correct values.

This process is most clearly explained diagrammatically. Figure 5.2 shows the upward part of the scan for an SM dealing with a group of matrices in the centre of a sequence. When each SM launches a block of threads, it first performs the standard interior reduction operation and then waits for the previous SM to supply it with the combined product to the start of its block (represented in Figure 5.2 as U_0). Once it has calculated the combined product to the end of its block, and passed that on to allow the next thread to work on it, the upward scan component begins, propagating U_0 all through the set of matrices and calculating all the correct partial propagators. A very similar process is performed interleaved with these calculations to calculate the backward partial propagators as well.

5.5 Gradient Calculation

Once the complete set of forward and backward partial propagators have been calculated, it is a relatively straightforward matter to calculate the individual components of the

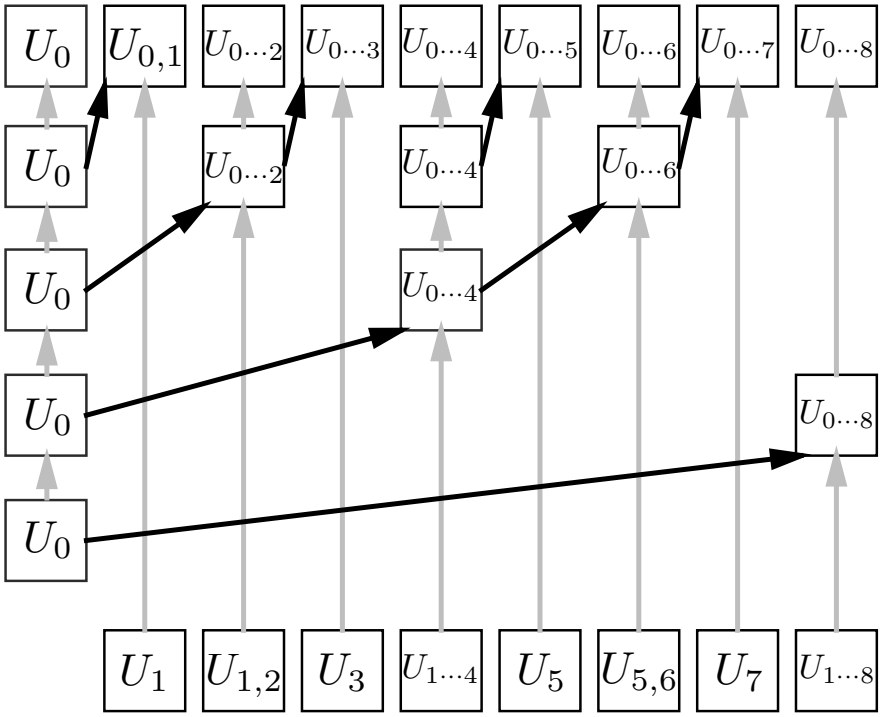


FIGURE 5.2: The scan operation begins at the end of the reduction operation, with the incomplete partial matrix products as shown here on the bottom line. The product to start U_0 is read in from shared memory, then for each step in the process each updated matrix multiplies a previously unmodified matrix, propagating the new values through the system. The final result is shown on the top line, where all partial matrix products have been calculated correctly.

gradient. Equation 3.24 gives us the formula, and it is reproduced here for reference:

$$\frac{\partial \Phi}{\partial u_{cj}} = -2\mathrm{Re}\left(\mathrm{i}\delta t\big(\langle P_j|\mathcal{H}_c X_j\rangle \langle X_j|P_j\rangle\big)\right) \tag{5.1}$$

It is of course straightforward in practice to simplify this yet further. $\langle X_j|P_j\rangle$ is simply the fidelity, and so can be calculated once by the first group of threads for one matrix and then reused for all subsequent thread blocks. It is also simple to take the imaginary part of $\delta t\big(\langle P_j|\mathcal{H}_c X_j\rangle \langle X_j|P_j\rangle\big)$ rather than multiplying by i and taking the negative of the real part. The principal calculations required are multiplying the appropriate Hamiltonian matrix by the forward partial propagator X_j, then multiplying the backward partial propagator P_j by this product. The trace of the imaginary part of this matrix is performed with a reduction operation, where the diagonal components are added in

pairs then the sums are themselves added to give the trace. This is then multiplied by $2\delta t$ to give the gradient component as desired.

5.6 Benchmarks and Scaling

It is always extremely difficult to quantify exactly how much more efficient code written to run on the GPU is compared to the CPU code it is replacing, because the hardware being used is different. It is clear that both programs can be made to run faster, simply by upgrading the hardware they are running on. As each program cannot be run on the other's hardware, how can their relative performance be assessed?

Of course, from a practical point of view, this issue of relative performance given equal hardware is not important. Of far greater interest is the actually measured relative performance given the available hardware. This can be considered from two angles, either by asking, what is the current maximum achievable by the best hardware available on each side or, comparing pound for pound spent, which approach offers the most cost effective processing.

As previously discussed, this problem is not well suited for traditional parallelisation and execution across multiple machines in a grid based supercomputer, because the overheads outweigh each calculation step. For this reason, the performance of each technique is restricted by the achievable processing power in a single top of the range card.

At the current time, one of the highest performing GPUs available is the Tesla M2050 from nVidia, described above and the GPU used throughout this work. There are in fact only a small number of GPUs so far available which meet the Compute Capability (CC) 2.0 standard, which is required for this code because of a number of features not available in earlier versions of the CC standard such as atomic instructions. The M2050 is also one of the first chips to have the enlarged 48kb shared memory which is extremely important for this kind of matrix manipulation heavy computation. For the moment therefore, this GPU is the only one sensible to consider.

The choice of CPU is of course far wider, with an enormous range capable of performing the necessary computations. The most powerful machine available to me for performing

calculations was a quad-core Mac Pro with a processor speed of 3.2 GHz and 4 Gb of RAM, so that was what was used for comparisons. While about a year older than the GPU card, it is still an extremely powerful machine. The cost of the GPU card is approximately equal to the cost of an entire high end workstation, so to actually build a system to make use of the GPU would cost roughly twice as much as simply running the code on a CPU based system.

In order to test the relative performance of the two systems, a single starting seed sequence was chosen of 512 timesteps, each lasting 4 μs, and 500 iterations of GRAPE were performed upon it to optimise it to a fidelity of about 0.97. Extensive Quality Assurance testing had already been carried out to ensure that the individual components of each step of the calculation produced identical results using each technique, so both methods would produce the same final answer.

The Mac Pro took on average 45.2 s to perform 500 iterations of the GRAPE algorithm, performing 11 iterations per second. This is certainly an extremely usable performance, and optimising sequences to high fidelity can usually be achieved in just a few hours. Running the code on the GPU however took only 0.55 s, representing a performance improvement of just over 80 times. This is a fantastic improvement, as not only does it represent an enormous increase in my personal potential to generate new pulse sequences, but it also suggests that the GPU approach would be superior to even the latest and most expensive CPUs, even if they were significantly superior to my test CPU case.

Using the GPU approach allows sequences that might previously have taken a day to generate using the CPU to be produced in only a few minutes. However, its chief strength is not in reducing the time taken to produce these sequences, but to bring much more complicated sequences into the realms of possibility. Some sequences rely on long periods of coupling under the internal Hamiltonian of the system, but also require complicated and fine-grained pulse shaping at the same time. With the CPU's capability, a trade-off is generally required between the length of the pulse and the length of the timesteps within it, as only a limited number of timesteps can be simulated. However, using the GPU, it is possible to massively increase the number of timesteps in a simulation, thereby increasing the length of a sequence without sacrificing resolution.

The final question to consider within this chapter is: where will this approach lead?

This code has demonstrated that it is possible to achieve extreme improvements in performance by using a GPU based approach for two spin GRAPE calculations. However, many of the optimisations and implementation details are only valid for the case of a two spin system. The fact that current generation GPUs operate with 16 threads at once, and that a two spin system can be represented using 4×4 matrices is a fortunate coincidence, while the amount of shared memory available in each SM allows just enough timesteps to be processed in a single SM to allow the parallel reduction and scan operations to operate efficiently.

If the number of spins in the system were to be increased to three, not only would the code have to be modified somewhat to handle the change in matrix size, but many of the components would no longer be able to run as efficiently and the speed improvements would be dramatically less. This may well change in the future as the resources available to GPUs increase, but for now the two spin case remains the sweet spot to be taken advantage of.

The other potential area where GPUs might become useful is in much larger spin systems. One of the earliest areas of interest in GPU programming was in large matrix manipulation, at which they can be significantly faster than CPUs. Conventionally matrix multiplication is only worth performing on the GPU if the size of the matrix is large enough that the calculation time dwarfs the overheads of sending the data to and from the GPU. My idea in this chapter was to group several small matrices together and process them as a group, rather than processing them individually, which would have been far slower on the GPU. However, in a larger system, the matrices will become big enough that they can be individually exponentiated and multiplied *etc.* on the GPU faster than on the CPU. Although none of the techniques described in this chapter would be directly applicable to this large scale case, significant calculation speedups could be achieved here also.

GPUs are much newer than CPUs and their potential is only now beginning to be fully explored. Their current rate of improvement year on year far outstrips the CPU, and an increasing number of the world's top supercomputers are coming to be based largely on GPU technologies. As GPU clock speeds and RAM increase, and the bandwidth to the main CPU comes down, code running on GPUs instead of CPUs is only going to get relatively faster and faster, so it is definitely worth taking advantage as soon as possible.

Chapter 6

Experimental Results

So far this thesis has described the power of the GRAPE algorithm, the theoretical sequence generation performance improvements it can give relative to earlier techniques, and the powerful extensions that can be made to enhance the sequences it produces still further. However, the most important question regarding GRAPE has yet to be addressed, that is, does it produce high fidelity logic gates or not? That is the question this chapter will deal with.

6.1 Single Spin and Heteronuclear Systems

As discussed previously, heteronuclear systems are usually considered fairly simple to control using only idealised pulses (where couplings are ignored) to implement single qubit gates, and delays for spin–spin interactions. This means that many of the problems that GRAPE attempts to solve are not even present in these systems. The only potential use of GRAPE in this case is for dealing with systematic errors, usually pulse length errors, as described in Section 4.6. However, pulses such as BB1 [97], which also provide excellent tolerance of PLEs, are not only well understood, but also much simpler to implement than the more complicated shaped pulses produced by GRAPE.

In these circumstances, GRAPE seems somewhat superfluous. However, not only is the single spin system a useful starting point in developing GRAPE sequences because it is conceptually simpler and less computationally demanding than larger systems, but in some systems the systematic error behaviour may be somewhat more complex

than the PLE dominated case prevalent in current liquid state NMR. In such a system, standard PLE tolerant sequences might be insufficient in the presence of other errors, and these might be better addressed by GRAPE. For this reason, it is worth examining the performance of GRAPE in a single spin system.

A number of examples of GRAPE sequences designed to be resistant to pulse length errors and combined PLEs and OREs where shown in Chapter 4, demonstrating the ability of the technique to produce single qubit sequences with prodigious tolerance to systematic errors, rivalling the best known conventional techniques.

The next step in investigating the performance of these GRAPE sequences is to test them in a laboratory experiment, and compare their performance with those standard techniques. Figure 6.1 shows precisely this for a 90° rotation gate, comparing the naive hard pulse, a BB1 sequence and a GRAPE sequence for a range of relative pulse strengths between 0 and 200%.

It is impossible to perform a real experiment with a precisely selected pulse length error as the exact reason that it is important to be able to tolerate a range of errors is that any NMR experiment will always have some distribution of RF powers in the control radiation, largely due to inhomogeneities across the sample volume. This means that each experimental result for a given PLE is actually a weighted average of the results in the vicinity of the desired PLE, with the weighting determined by the probe's response. As this is generally asymmetric, this will cause some asymmetry in the data, as can be seen in Figure 6.1 but it is at least equally applicable to all experiments, so comparisons between the different methods can be made.

Some phase error can also be seen at the right hand edge of the GRAPE pulse spectra. This is not a measurement error but in fact the predicted performance of the GRAPE sequence at large values of PLE. In order to produce an extremely flat response with relative pulse strengths close to 100%, the GRAPE algorithm sacrifices quality at larger error values.

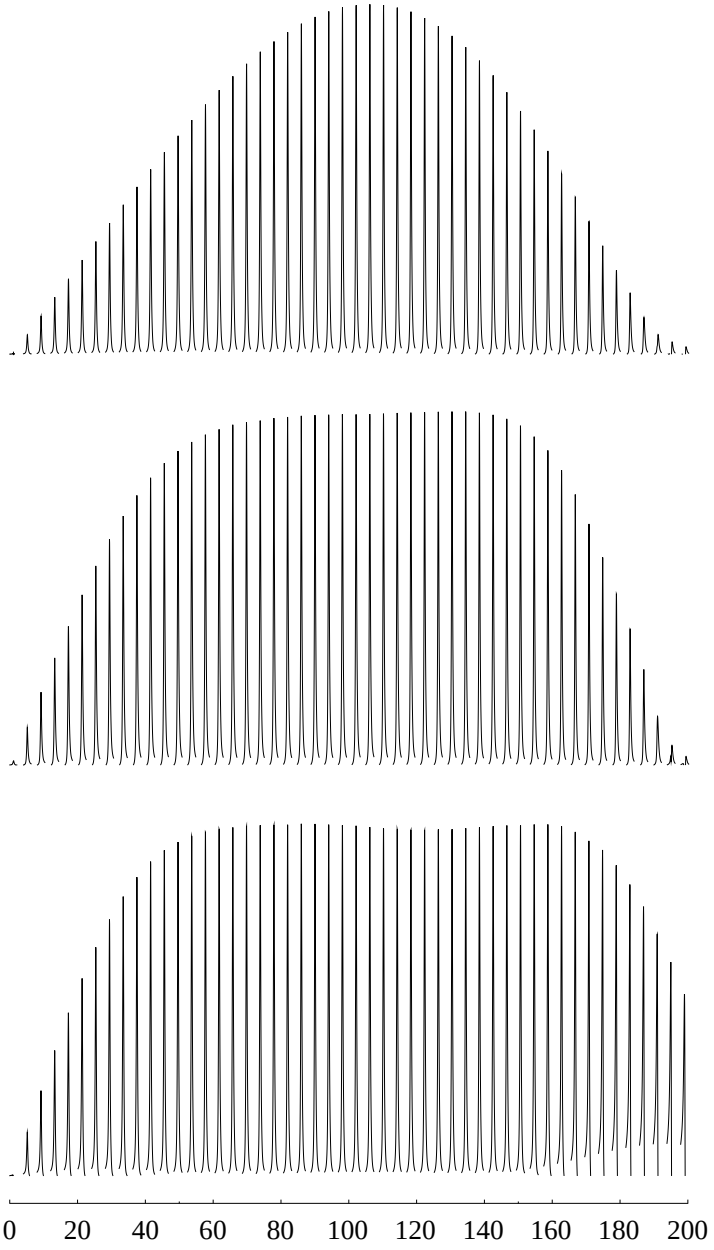


FIGURE 6.1: Each figure here shows 50 individual spectra, each acquired with a different artificially induced relative pulse strength in the range 0-200%. The top figure shows the sine wave response produced by a simple hard pulse. The middle figure uses BB1, which removes error terms up to the sixth power. The bottom figure is from a GRAPE sequence which is even wider than the BB1 version.

6.2 Larger Systems

Of course, the much more important tests of GRAPE derived sequences must come in multiple spin, homonuclear systems. The obvious place to begin is by moving up to

a two spin system, which demonstrates the principles of homonuclear GRAPE while keeping the system relatively small and fast to simulate. The principal sample used for my two spin GRAPE experiments is a 50 mM solution of the pyrimidine base cytosine in D_2O; a rapid exchange of the two amine protons and the single amide proton with the deuterated solvent leaves two remaining protons forming an isolated two spin system.

Figure 6.2 shows the cytosine molecule in the standard representation, and the corresponding NMR spectrum. On a spectrometer with a reference frequency for protons of 600 MHz, the two doublets are measured to be separated by 914.84 Hz, and each doublet has a splitting of 7.17 Hz. From Figure 6.2, the two doublets can be seen to have different heights, even though the same number of each spin is present in the sample, and they have identical gyromagnetic ratio. This is because the widths of the peaks in each doublet are not equal: the S qubit has a slightly shorter T_2 decay time and so is slightly wider than the I doublet, slightly reducing it in amplitude as well. A small amount of "roofing" can also be observed, where the heights of the lines within each doublet form a slope towards the other doublet. This effect can be understood by applying the full Heisenberg coupling interaction between the spins, rather than the simpler Ising coupling normally assumed in analytical treatments of the system. When the J coupling is much smaller than the separation of the two doublets, first order perturbation theory shows us that the eigenstates of the system do not change compared to using the Ising coupling, but there is a shift in energy levels. As the relative size of J increases however, first order perturbation theory begins to break down, and the eigenstates of the system do begin to change, leading to changes in the signal intensities. The fact that this roofing can be observed for this sample shows that this effect is important in this case, and so using something like GRAPE, which automatically includes the full interaction term in its calculations, is even more useful.

The most obvious initial gates to test the GRAPE algorithm with are spin selective gates, that is gates which perform a particular logic gate on one qubit, and leave the other qubit unaffected. The ability to construct these single qubit logic gates would enable universal quantum computing, just as for heteronuclear systems, as the controlled-Z gate can still be performed simply by a couple period of the appropriate length.

Figure 6.3 shows an early attempt to produce a selective gate applying a rotation of 90° about the $-y$-axis to the I qubit, while leaving the S qubit unaffected. Although the I

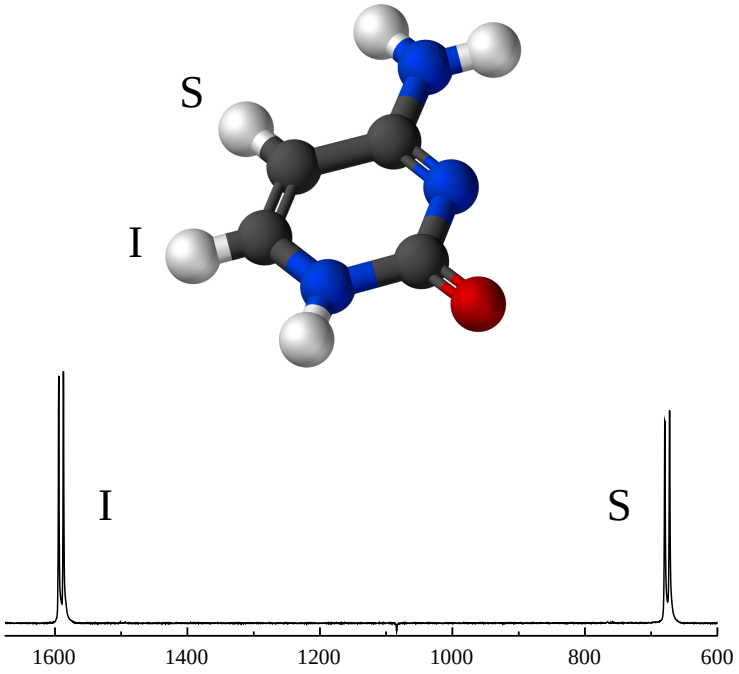


FIGURE 6.2: The pyrimidine base cytosine, with the spin-$\frac{1}{2}$ computational qubits marked as I and S [111]. The reference spectrum is obtained by applying a 90_{-y} hard rotation pulse to the system to put it into the $I_x + S_x$ state, then observing. The I qubit produces the left hand doublet at around 1600 Hz, and the S qubit produces the right hand doublet. The frequency reference here is relative to the water line caused by contaminant HDO protons, which is the largest signal and therefore used as a reference.

qubit does indeed seem to have undergone roughly the correct rotation to generate the I_x state, it is clear that the S qubit has not been left in the S_z state as desired, and the phase of the excited spin is not fully absorptive, as should be the case, but is instead at some phase angle relative to the hard pulse used as a reference phase. There is a small but significant excitation of the S doublet, and these flaws are quite enough to render the gate unacceptable for use in quantum algorithms. For the gate in question, the calculated fidelity was above 0.9999 and the size of any excitation on the S qubit should have been significantly smaller. Figure 6.4 shows the difference between the result predicted using the simulator and the experimentally measured result. As equivalent results were obtained for a variety of other derived gates, it immediately became clear that there is some feature of the experimental setup which is not accounted for in the simulator, introducing this error.

All simulated results were created using the same algorithmic engine used to model the effects of GRAPE pulses to generate the simulated final state of the system after

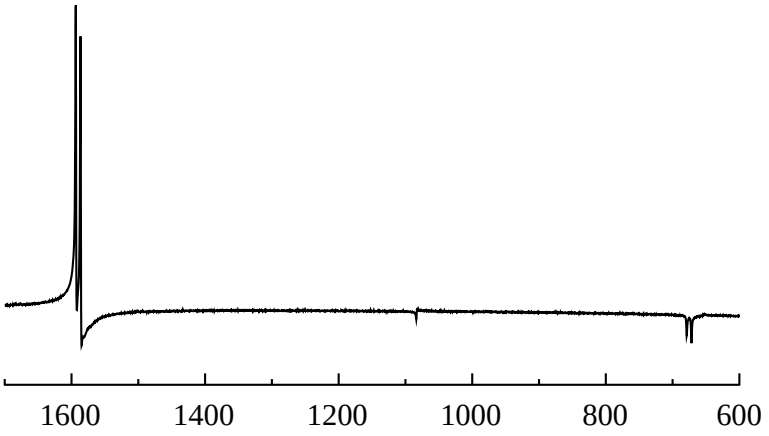


FIGURE 6.3: The spectrum obtained by applying an early version of a GRAPE derived selective 90° rotation to the I qubit. The expected spectrum should show a completely absorptive doublet on the left hand side, and no signal at all on the right.

a control pulse was applied. Then the same engine was used to simply evolve the system under the internal Hamiltonian without any control fields applied, to simulate the evolution during the free induction decay. The period of free evolution was broken down into microsecond steps to allow T_2 decoherence modelling to be performed at the end of each step, as described in Section 4.5.1, while during the simulated GRAPE pulse, decoherence processes were applied at the end of each timestep. During the observation period, at the end of each evolution and relaxation step the observable quantity $I_x + \mathrm{i}I_y$ for the state was calculated, creating a simulated FID. This simulated FID is then apodised and Fourier transformed to give the spectrum, just as experimentally measured FIDs are. This close correspondence between the methods used to generate real and simulated spectra allows an extremely good agreement between the two for reference spectra, as can be seen in Figure 6.5.

The first possibility that suggested itself was insufficient tolerance of pulse length errors by the sequences. Most sequences were designed to be resistant to errors over a range of ±30%, as this seemed to offer a suitable balance between robustness and length of development. As a result a set of pulses were designed to test the effect of changing the range of errors from between ±1% to between ±50%, as well as checking if further improving the fidelity would improve matters. However this very quickly showed that no achieved pulse, even with extremely high fidelity and even greater PLE tolerance,

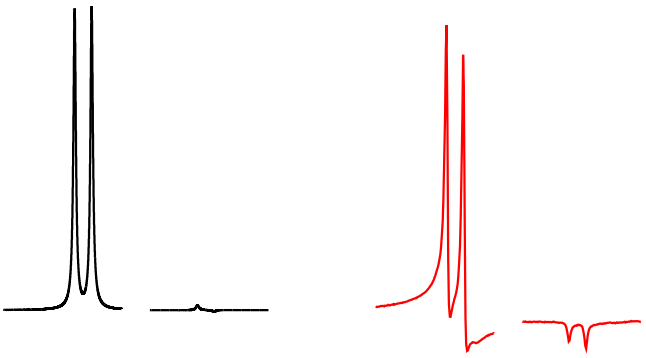


FIGURE 6.4: The same spectrum as Figure 6.3, shown in red on the right, is compared with the spectrum calculated using the simulator. In this figure, as well as many subsequent ones, a cut down form of spectrum is used showing only those parts of the frequency range within 25 Hz of each doublet, in order to allow the details of the peaks to be seen clearly. The rest of the spectrum consists solely of noise, so no data is lost using this method.

could improve on the quality of this initial result, suggesting that these were not the cause of the problem.

6.3 RF Control Field Transients

The next obvious candidate to investigate in the hunt for the experimental flaw was the circuitry responsible for generating and delivery the RF control fields to the sample. Because of the sophisticated behaviour they are designed to achieve, GRAPE pulses are necessarily highly dependent upon the precise shaping of the control fields. In the simulator these are of course delivered with perfect accuracy, whereas if the actual quality of the RF fields in the experimental runs is not good enough, significant deviations would be observed.

A simple model for the RF generator (by which I refer to the combined electronic components responsible for generating the RF and delivering it to the sample: synthesizer, mixer, amplifiers, filters and probe) is a damped oscillator. At the start of each timestep, the values of phase and amplitude for the field are changed. These changes cannot occur

instantaneously, there must instead be some relaxation towards the new value. The rate of this relaxation will be dependent upon the properties of the circuit, but it is clear there must be some period at the start of each timestep where an incorrect phase and amplitude will be delivered. The characteristic period of this relaxation will be a fixed property of the system and will be independent of the length of the timestep, so the shorter the timesteps are chosen to be, the greater the impact from this transient behaviour.

As a basic test to investigate the impact of changing the length of the timesteps, the sequence used to produce Figure 6.3 was used as the basis for a complete family of progressively coarser sequences. Starting with the 1024 timestep sequence previously generated, a 512 timestep sequence was created by averaging pairs of timesteps, generating a sequence of the same duration and with a broadly similar shape, but with timesteps twice as long as the previous sequence. This new sequence was then reoptimised in GRAPE to the highest fidelity achievable in a reasonable length of time. The same procedure was then applied to create the complete sequence of gates from 1024 timesteps right down to a single timestep.

Fig 6.5 shows the spectra obtained from this set of sequences, and compares them to the results obtained from simulating the sequences instead. The fidelities of the sequences involving fewer than around 128 timesteps are generally of very low quality and so should not be expected to give the correct selective rotation gate, but it is instructive to examine the relative performance of experiment and simulator for these sequences all the same, to observe to what extent transient behaviour is an issue at various coarseness levels.

The agreement between experiment and theory is extremely good for both amplitude and phase up to about $n = 32$, after which the rate at which the experimental performance improves is not as good as for the simulated sequences, although it does improve until $n = 512$. The most pronounced disagreement comes at $n = 1024$ where there is a sharp decrease in the quality of the final spectrum compared to the previous versions, although the simulated fidelity is highest for this sequence.

This result was born out by other corroborative experiments testing several other GRAPE sequences with timestep lengths ranging down to $2\,\mu$s, and suggests that some factor, most likely transient effects, causes significant problems with implementing sequences

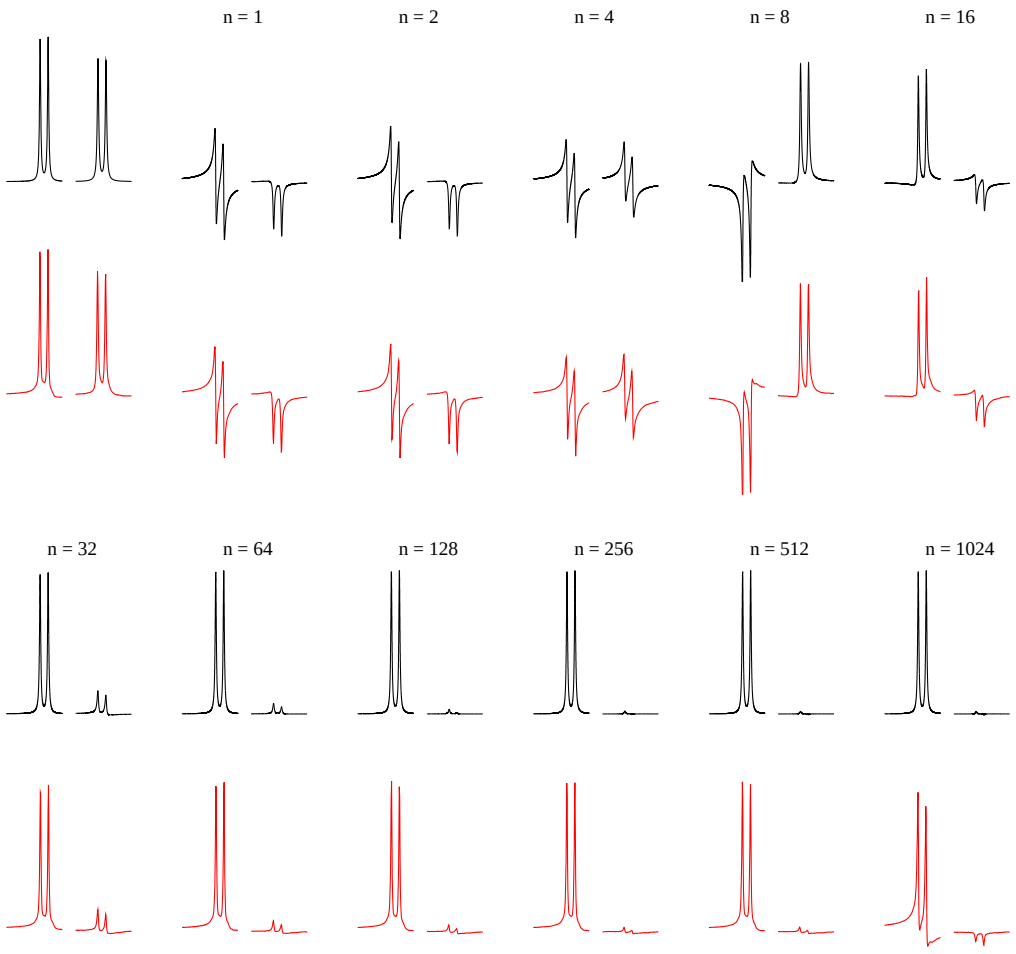


FIGURE 6.5: The spectra resulting from a group of sequences performing a selective 90° excitation pulse on the I qubit. Each sequence is labelled by the number of timesteps used to generate it, and the duration of all sequences is 2048 μs. Simulated results are shown in black on the top row, with experimental results underneath in red. A reference spectrum from a hard 90° pulse is shown at top left for comparison purposes.

when the length of the timestep decreases from 4 μs to 2 μs. Using a slightly coarser sequence immediately removes most of the error term observed in the initial pulse sequences. While this sudden decrease in quality seems quite sharp, and it should be possible to use similar techniques to those described in this section to more precisely identify the nature and position of this deterioration in quality, the fact that it can be avoided simply by using slightly longer timesteps makes it easy to work around this issue in practice, once it has been identified.

However, it is also clear that transient effects are having some smaller impact even for

sequences with much longer timesteps. Any sequence with timesteps shorter than 16 μs has a noticeably worse performance compared to simulated results than those sequences with longer timesteps. Unfortunately, it has so far proved impossible to generate sequences with this coarseness of timestep and acceptably high fidelity, so it is necessary to make a compromise between fidelity and implementation performance relative to simulator, and for my specific equipment, I found 512 steps of 4 μs each, generating 2048 μs sequences, gave the best overall performance.

The ability of GRAPE to generate sets of pulses, all of the same length, greatly adds to its ease of use over other techniques such as SMCP. This length seems to be a reasonable choice for selective sequences, obviously the shorter a pulse is the better from the point of view of decoherence and calculation time considerations, but the algorithm must be given enough time to work with if it is to identify high fidelity pulses. Correctly identifying the minimum length of time to perform an arbitrary gate, particularly while performing error suppression *etc.* is a non trivial problem, and so trying a range of potential times and choosing one which seems to work is the most effective strategy currently available.

Throughout the rest of this chapter, many facets of experimental GRAPE sequence application will be examined. In order to allow the easiest possible reference between different effects, whenever appropriate the same GRAPE pulse is used to demonstrate in each case. The sequence used for this purpose is a selective 90° rotation acting on the I qubit about the $-y$-axis, just as used in this section.

6.4 Hidden Delays

Another potential source of errors, particularly in algorithms involving multiple logic gates, are the additional delays which may be introduced by the spectrometer in between shaped pulses, for example to allow amplifier gating.

To test the effect of these delays, I applied two 90° rotation gates back to back, using the same set of progressively coarser gates as used to produce Figure 6.5, with the expected result of producing the $-I_z + S_z$ state which generates no signal. Again the experiment is simulated, assuming the two pulses are applied without any delay between the sequences, and the results are compared with the experimentally determined values which will be affected by the delays. The results of this experiment are displayed in

Figure 6.6. Just as for Figure 6.5, for small numbers of timesteps (low n) the fidelity of the logic gates is extremely poor, so the expected result is not actually the target $-I_z + S_z$ state. Results for these gates are still included in order to demonstrate the agreement between simulated results of the pulses and the corresponding experimental results.

The sharp deterioration in implementation quality when moving from $n = 512$ to $n = 1024$ is observed once again here, but it is also clear that the quality of the other experimental pulses is nothing like as good as was achieved for a single pulse, even for sequences consisting of only a few timesteps. The two possible causes of this drop in performance when moving from one sequence to two are: significant delay between the

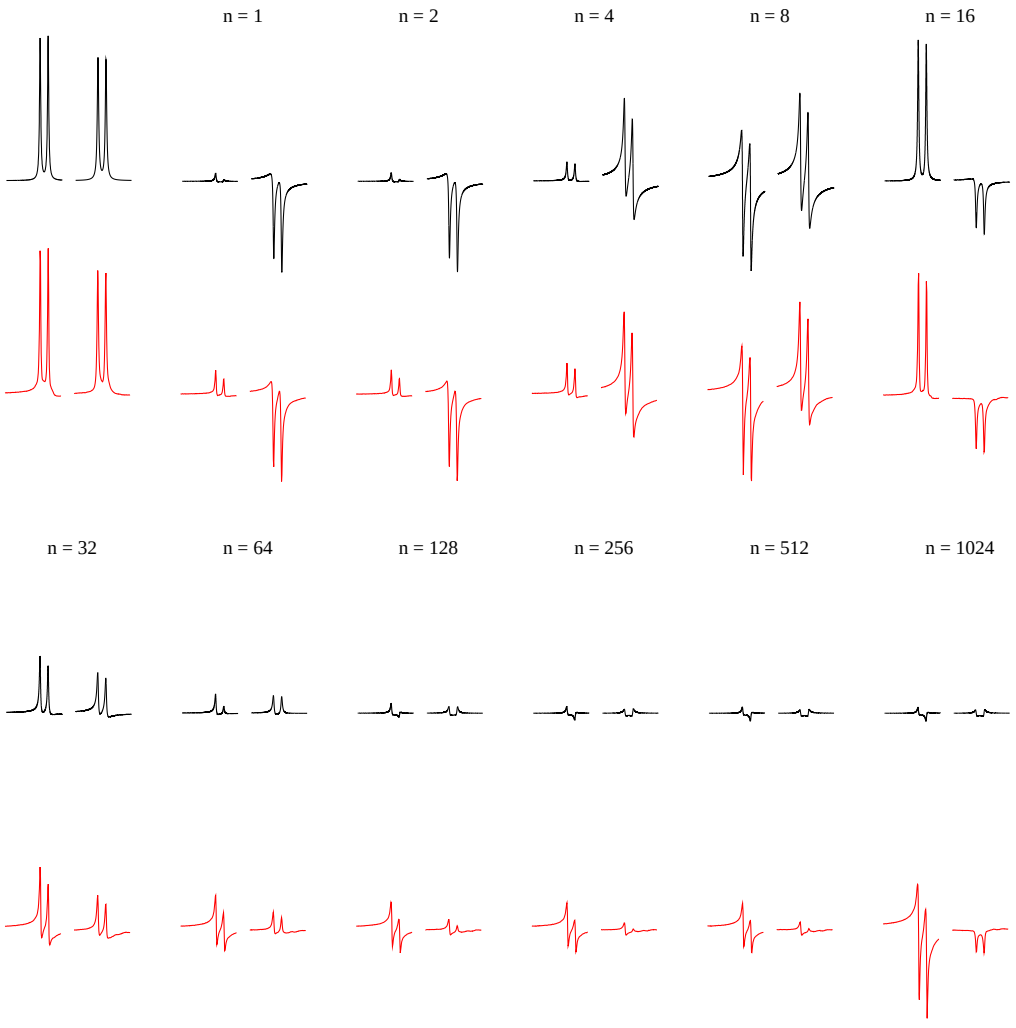


FIGURE 6.6: Each sequence initially used to produce Figure 6.5 is run twice back to back to theoretically produce a 180° rotation.

pulses causing additional evolution of the system, or a lack of reproducibility in precise phase and power for the two pulses, meaning that the angle and axis of rotation for the two gates is subtly different.

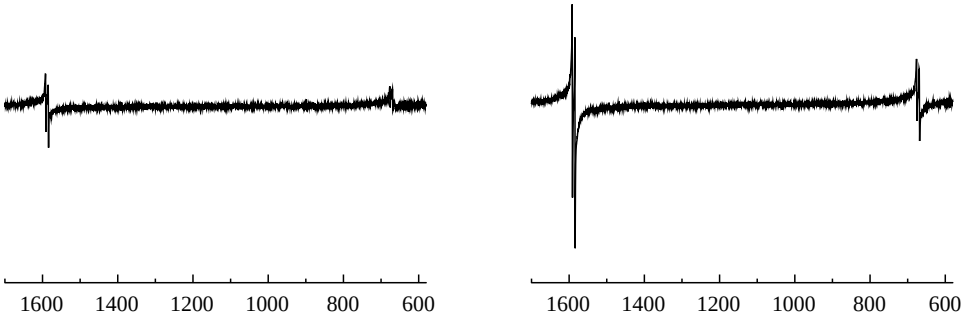


FIGURE 6.7: Creating a single 180° rotation pulse sequence by concatenating two copies of a 90° sequence and then implementing the combined sequence as one pulse (as shown in the left hand figure) produces a much higher quality of null than applying the two copies of 90° as two separate pulses, nominally back to back (shown in the right hand figure).

Creating a combination pulse sequence by concatenating the two pulse files produces a sequence which performs a much better 180° rotation pulse, an example of which can be seen in Figure 6.7. Although even the most precise NOT gate tends to display a highly visible error signal in all NMR implementations, even heteronuclear and single spin systems, the quality of the concatenated 180° gate is as good as for the 90° gate used to create it, demonstrating clearly that the poor quality displayed in Figure 6.6 must be due to the use of two separate pulses experimentally.

6.5 Pulse Length Error Tolerance

Having identified and eliminated the first dominant source of error in the experimental performance of GRAPE sequences, the tolerance of pulses to the principal source of systematic error, pulse length errors, becomes sufficiently important to warrant further investigation. Figure 6.8 shows an example of the tolerance of a GRAPE pulse to pulse

length errors. It is interesting to observe that the selective excitation of the targeted qubit seems significantly more robust against pulse length errors than the null excitation on the un-targeted qubit. This is partly due to the fact that errors in the angle of spins in the transverse plane only appear as the square root of the actual error, whilst the errors in longitudinal spin angle appear linearly. Even allowing for this however, the null is definitely the more challenging component of the logic gate, a fact which is well known to NMR spectroscopists.

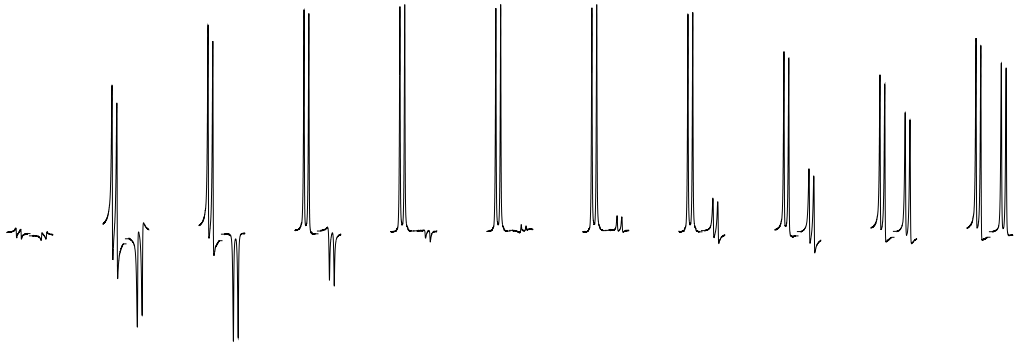


FIGURE 6.8: A selective 90° rotation gate on the I qubit, observed for a range of pulse length errors between 0% and 200% at 20% intervals. The sequence is designed to tolerate errors up to 30%.

The other important point to make when observing this kind of PLE tolerance figure is that each spectrum cannot be regarded as resulting from the precise PLE value indicated, this represents only the centre of the distribution of frequencies shown in Figure 4.1. The measured PLE tolerance curve therefore represents the convolution of the true PLE curve with the RF frequency distribution, which will distort the shape significantly.

A number of GRAPE sequences were developed to perform the same selective excitation gate with a range of PLE tolerance, to test the importance of including such behaviour in the GRAPE sequences. It became clear very quickly that although sequences with very narrow tolerances (below ±10%) performed badly compared to sequences with wider tolerance, increasing the tolerance to very wide amounts gave no improvement over more modest widths, suggesting that another, as yet unidentified, error term was the principal remaining source of error in the implementation. From this point, all GRAPE sequences were developed with ±30% PLE tolerance. Although this degree of error tolerance did

not produce visibly superior spectra to somewhat narrower tolerances, in the two spin system, achieving it was relatively straightforward, and did not add an unacceptable amount of computation time compared to narrower tolerance levels. In more complex spin systems for which good PLE tolerance is difficult to achieve, a much more careful analysis of the minimum suitable tolerance would be required.

6.6 True Unitary Transformations

Although the performance of GRAPE sequences has not yet achieved the quality predicted by simulated runs, it is still appropriate at this point to consider what can already be achieved using GRAPE. One of the most important advantages of using GRAPE to generate sequences is its ability to produce true unitary transformation matrices, which perform the correct logic gate on any initial state, rather than simply creating "point-to-point" sequences which only work to transform a single known initial state into the correct target state, as is common in conventional NMR.

The best way to completely determine the complete unitary transform matrix for a sequence is to use quantum process tomography [112]. However, this is a complex and arduous process, so for brevity I have implemented a more compact version, considering the effect of GRAPE sequences on the three states aligned with the principal axes of the system.

Demonstrations of this effect are shown in Figure 6.9 for selective 90° rotations on both qubits. The quality of the later sequences with lower time resolution and wide PLE tolerance is significantly higher than for the initial sequences previously observed such as Figure 6.3, with very little excitation visible on the untargeted qubit.

The necessary achieved fidelity for acceptable quantum computation is uncertain, but commonly suggested desirable values are around 0.9999. It is impossible to accurately determine fidelities to this degree simply by measuring a spectrum. Instead, the best way to assess the performance of these sequences is to use them in more complicated sequences, and observe the compound deterioration in overall quality as it achieves more measurable degrees.

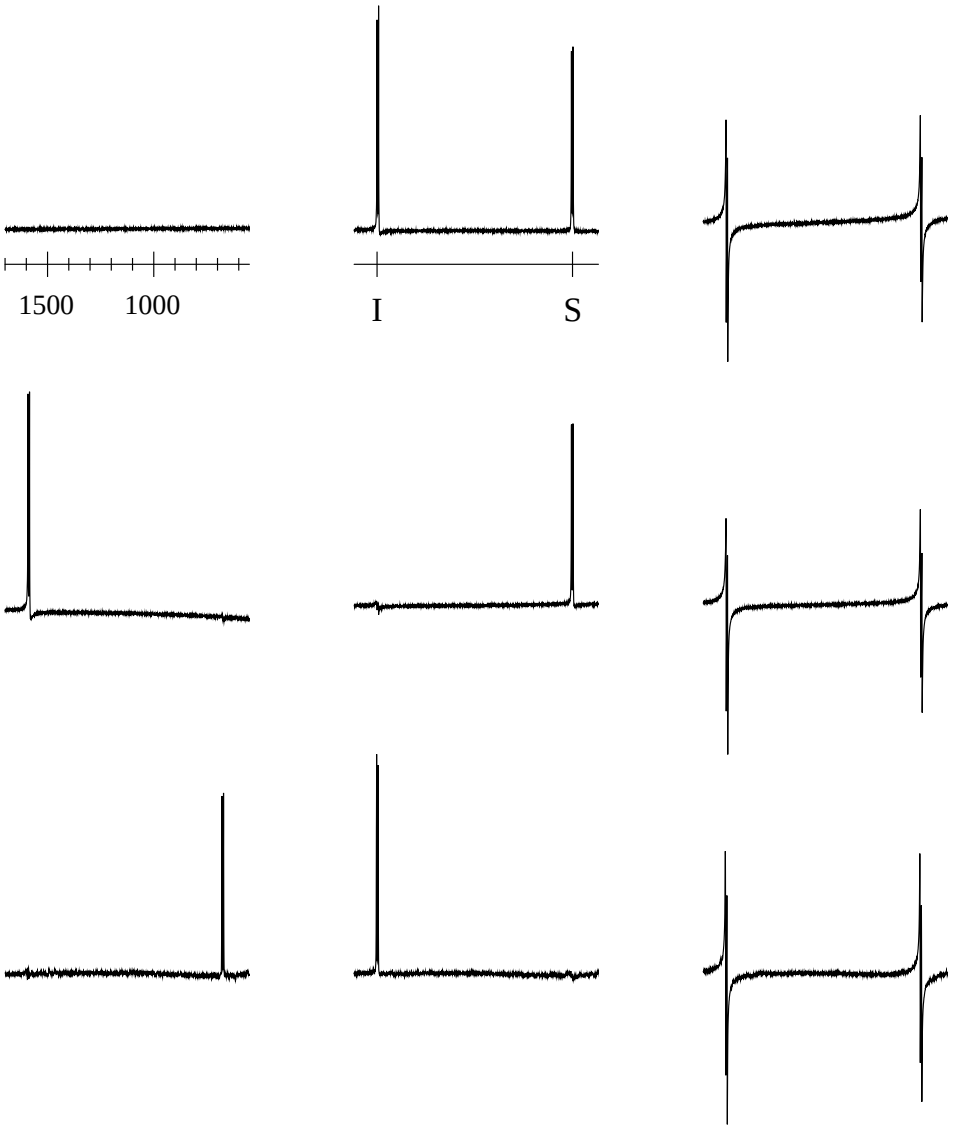


FIGURE 6.9: GRAPE derived sequences are true logic gates which are independent of starting state and are not simple "point-to-point" transformations. The top row spectra show states prepared with both qubits along the Z, X and Y major axes respectively. There is no signal from the Z state as there is no transverse magnetisation, only noise is observed. The X and Y signals are identical, simply 90° out of phase with each other. The second and third rows then show the resulting state after a GRAPE derived selective 90° rotation gate about the y-axis is applied to the I qubit (middle row) and S qubit (bottom row). As expected in all cases there is no change for the un-targeted spin. From an initial Z state, the targeted spin is excited into the X state, while starting from the X state, the spin ends up in a -Z state and no signal is visible. Rotation about the y-axis leaves the Y state unchanged, so all three Y spectra are the same.

6.7 Homonuclear Pseudo-Pure State Preparation

Using the controlled transfer gate method to produce pseudo-pure states is an excellent candidate as a semi-complex quantum algorithm to test GRAPE sequences with. Pseudo-pure state preparation is a vital component of any quantum computer implementation, so developing a good method for doing this is useful in itself, and the controlled transfer gate technique can be implemented using only selective 90° rotation gates on both qubits. Using only a small number of gates means that the potential sources of error are fewer than if a large number of gates were used, so it is clearer where the errors are.

At the same time, controlled transfer gates provide a challenging test of the quality of the pulse sequences. Controlled transfer gates are basically "jump and return" type sequences [113], specifically of the type $1\bar{1}$, which rely on a period of coupling between two equal and opposite pulses to allow a phase difference to build up between the lines in each doublet, so that the second pulse returns one line to its starting point, while moving the second line to a new location. These jump and return sequences are well known to be difficult to implement experimentally, as they are not at all robust to off resonance effects, and signals only a very small distance from the precisely correct frequency can experience markedly different excitation.

In conventional NMR a family of sequences known as binomial solvent suppression sequences [101] have been developed to provide much higher quality jump and return sequences for solvent suppression, which can also provide a broader excitation profile. Unfortunately, Bowdrey and Jones [113] showed that these sequences are far less effective for use in a quantum computing context, so cannot improve this sequence. Thus high quality controlled transfer gates remain an effective benchmark for control quality.

A sample of pseudo-pure states generated by this approach is shown in Figure 6.10. The basic result is as we would expect, with obvious good excitation of the correct state. However, there is still an unpleasantly large excitation on the untargeted qubit, as well as some population in the other line in the doublet. These errors are of the order of approximately five percent. As this is the cumulative error resulting from five selective rotation gates, one might tentatively assign a rough error per gate of about one percent. While this is an extremely imprecise estimate of the error, it is clear that the fidelity

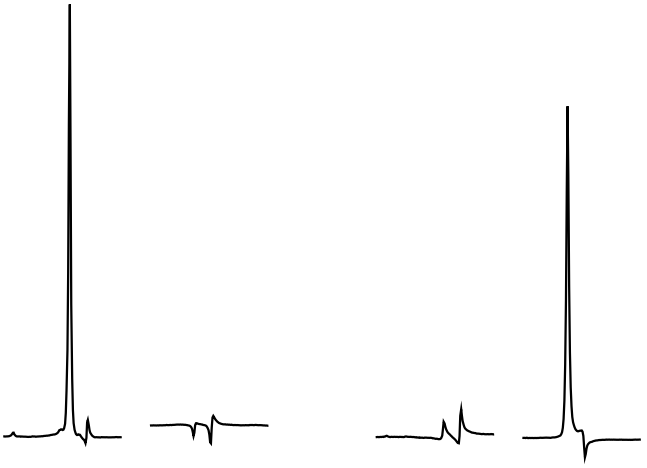


FIGURE 6.10: A pseudo pure state obtained using the controlled transfer gate method is subsequently excited by selective I (left hand figure) and S (right hand figure) 90° rotations. The expected result in each case is that the left hand line of the excited doublet (indicating the 0 state for the other qubit) should be fully absorptive (to indicate the 0 state of the targeted qubit). All other lines should be completely unexcited.

being achieved with this experimental setup at least is insufficient for the requirements of full quantum computation.

6.8 Direct RF Observation

In order to further investigate the quality of RF generated by the spectrometer, it is useful to directly observe the signal going into the probe coils. This can be achieved by tapping off the signal using a digital oscilloscope before the probe. Of course, this technique does not allow monitoring of any distortion introduced into the RF by the probe itself, but the quality of the probe is such that this is unlikely to be a problem. It would also be extremely difficult to introduce any device capable of measuring the RF output by the probe without disturbing the probe environment and distorting the measurements. For this reason it makes sense to begin by observing the rest of the system which can be done purely in the electronices. Because the signal is superimposed upon the resonance frequency at 600 MHz, observing it directly is technically challenging because a sequence of duration ~ 2 ms observed at sufficient resolution to measure the oscillations at resonance frequency requires $\sim 10^7$ data points.

Instead it is convenient to beat the signal RF with a second unshaped oscillation at the resonance frequency. This is equivalent to observing the signal in the rotating frame, and recovers the pulse shaping specified in the sequence file. This dramatically reduces the time resolution required to observe the sequence and so makes it possible to acquire the full 2 ms required for a standard GRAPE sequence. Figure 6.11 gives a schematic diagram showing the configuration of the connections to the mixer used to combine the two RF signals.

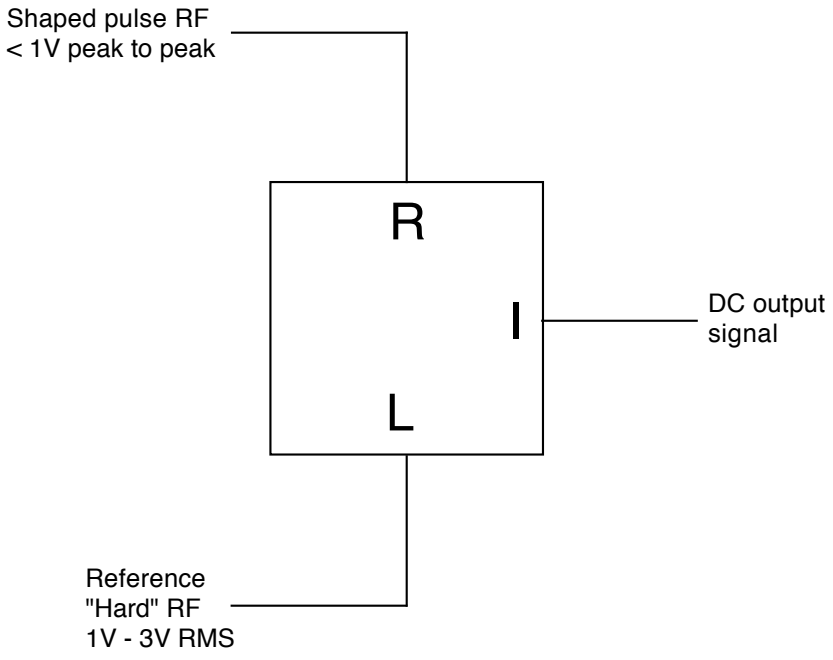


FIGURE 6.11: Schematic Diagram showing the connections for the RF mixer used to beat the shaped pulse down to a DC signal, and the necessary powers that must be supplied to the input terminals of the mixer.

The spectrometer uses a working frequency of 20 MHz to shape the RF, which is then beaten up to the required output frequency. In order to prevent this signal from "bleeding" into the mixer and interfering with the main component, band pass filters at the observe frequency must be applied to both the input signals. It is also important to provide some kind of low pass filter to the output signal, to make sure that any high frequency signal left over is not preserved in the final observation.

In order to fully characterise the measured RF, two acquisitions are required, with the phase of the secondary oscillation changed by 90° between the two acquisitions. This

allows the full complex signal observed in the rotating frame to be obtained; this can then be considered as amplitude and power components, useful for comparison with the sequence file used by the spectrometer to generate the RF, or rephased into x and y components for comparison with the sequence as originally generated by GRAPE.

When measuring the RF in this way, it is vitally important to be sure that any discrepancies measured by the oscilloscope are genuine deficiencies in the RF, rather than being due to any limitations in the measuring apparatus. This required a sequence of tests to ensure that the oscilloscope and RF mixer were operating with their rated parameters, such as ensuring that the response of the oscilloscope was faster than the transient recovery time of the RF sequence, as otherwise the genuine transient behaviour would have been lost behind the transients generated by the oscilloscope.

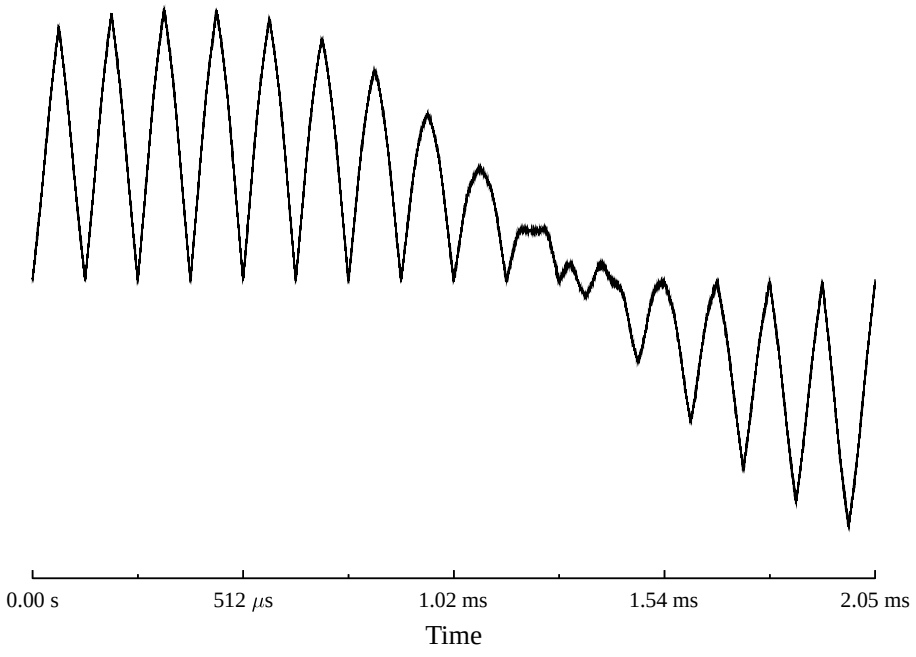


FIGURE 6.12: This pulse sequence consists of 16 identical triangular shapes. Each shape has a constant phase and the phase between each triangle ramps up by 12°. Because the signal is observed beaten against a reference frequency at a constant phase, the heights of the triangles are modulated by a sinusoidal oscillation. However there is also significant distortion of the shapes of triangles with phase differences close to $\pi/2$. This behaviour was ultimately traced to a non-linear response in the RF mixer caused by the reference frequency having insufficient power.

Figure 6.12 shows an early example of a test sequence used to test the linearity of the measurement at a range of phases. The pulse sequence used is a set of triangle

shaped pulses with phases varying from 0 to 180. The expected behaviour is to see a set of triangles with sinusoidally varying heights, but there is extreme deformation of the triangle shapes at phases that are close to pure imaginary. Fortunately this effect was found to be due to the secondary oscillation component in the mixer being supplied at too low a power, rather than a flaw in the original RF generated by the spectrometer. By simply raising the amplification of this component to the required 1-3 V RMS, the problem was solved.

One of the interesting questions in comparing this measured RF data with the desired RF is defining a common phase reference. Because there is no simple relationship between the basic phases of the main RF and the secondary resonant oscillation used to beat it down, the measured real and imaginary components are inevitably offset by some arbitrary phase from the proscribed RF. This can be seen in Figure 6.12 by the arbitrary offset into the sinusoidal oscillation of the height of each triangle. In order to calculate the correct phase adjustment to enable the real and imaginary components to be compared like for like between measured and desired RF, a simple square pulse is inserted before the main sequence at 45° to the real component. In the measured data it is then possible to calculate the phase required to give this pulse equal height in the real and imaginary components, allowing the data to be correctly phased with ease.

Figure 6.13 shows some examples of transient behaviour in simple RF pulses. Even large jumps in amplitude have comparatively minor transients, and because they are simply oscillatory, the average effect over the period the transient is evident is mostly cancelled out.

A much bigger problem is the transient spikes present in even very small phase jumps. Because their occurrence is unpredictable, it is impossible to make sequences resistant to such behaviour, while their magnitude when they do occur is much greater than for the amplitude jumps. These effects will not cancel out, and so offer the potential to affect the overall quality of the sequence if the timesteps are too short relative to the transient spikes.

Having examined the behaviour of simple pulse sequences and the transient effects introduced by changes in the RF phase or amplitude, it is time to look at how correctly whole pulse sequences are produced by the RF generation circuits. As an example pulse sequence, I will be using the same selective 90° sequence on spin I used elsewhere in this

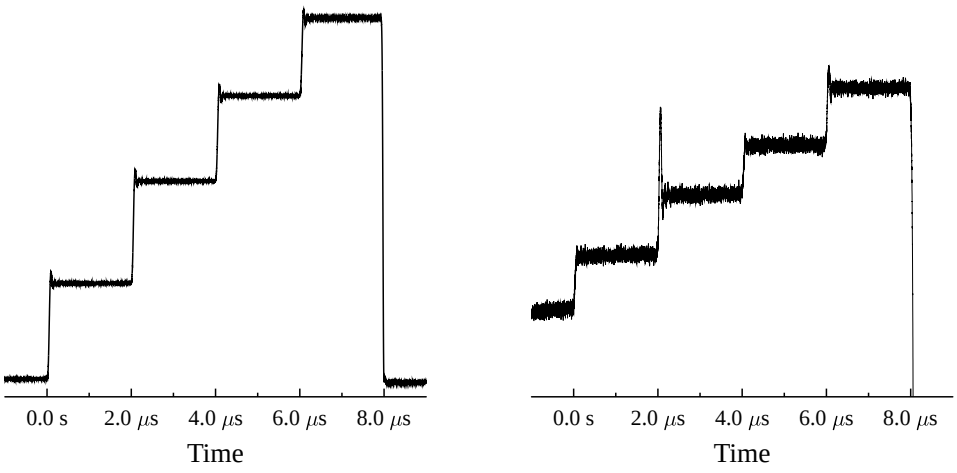


FIGURE 6.13: Transient behaviour in directly observed RF pulses. The left hand figure shows a pulse sequence consisting of amplitude jumps of 25% increasing from zero power to full power. Here the transient behaviour is precisely as expected, with a short period of oscillation recovering to the new value. The maximum deviation is about 10% of the jump and the behaviour at each jump is very regular. On the right hand side is a pulse sequence consisting of phase jumps of only 1° each, keeping the power constant. Only the real part of the data is shown, so the phase jumps are represented as amplitude jumps. Here the transient behaviour is much more unpredictable. Despite the very small size of the jumps, relative to the amplitude jumps on the left, some jumps display transient spikes larger than the size of the jump, while other jumps display no visible transient effects. Timesteps for both sequences are both 2 μs long.

Chapter. However, because of the clear differences evident in the simple test sequences between phase and amplitude jumps, a simplified version of the pulse sequence was also produced consisting only of the amplitude modulation, with all phases set to zero, to determine if this effect is significant in the more complicated GRAPE pulse sequences.

Figure 6.14 shows these two sequences captured by a high resolution digital oscilloscope with a 10 ns resolution. The degree of transient behaviour shown for the sequence involving phase modulation is clearly visible and it is easy to see why this could cause considerable problems with sequence quality if the timesteps become too short. The figure with no phase modulation is much better behaved as expected.

An alternative way to view this data is shown in Figure 6.15. Here the sequence is shown as an amplitude and phase, allowing both components of the full sequence to be visualised simultaneously. Each timestep can be seen as a tight knot of points, while the transient effects can be clearly seen as the giant loops which connect adjacent timestep blocks.

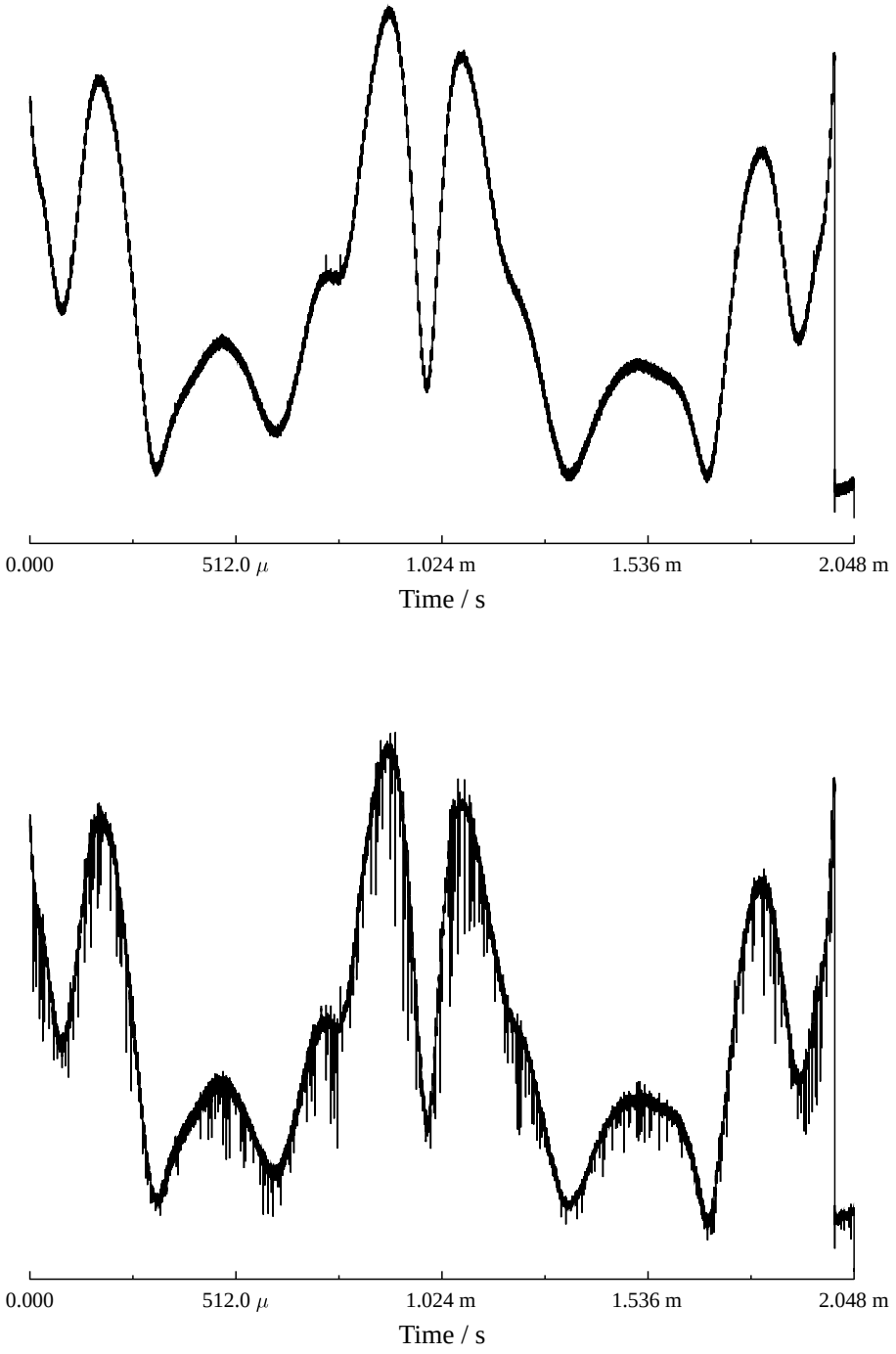


FIGURE 6.14: A comparison of the quality of a pulse sequence involving only amplitude modulation with one including phase modulation as well. The sequence for the top figure here has had the phases for each timestep set to zero so consists only of amplitude shaping. The bottom figure shows the original sequence complete with phase modulation as well. In both cases the absolute value of the data acquired is shown, displaying the amplitude only.

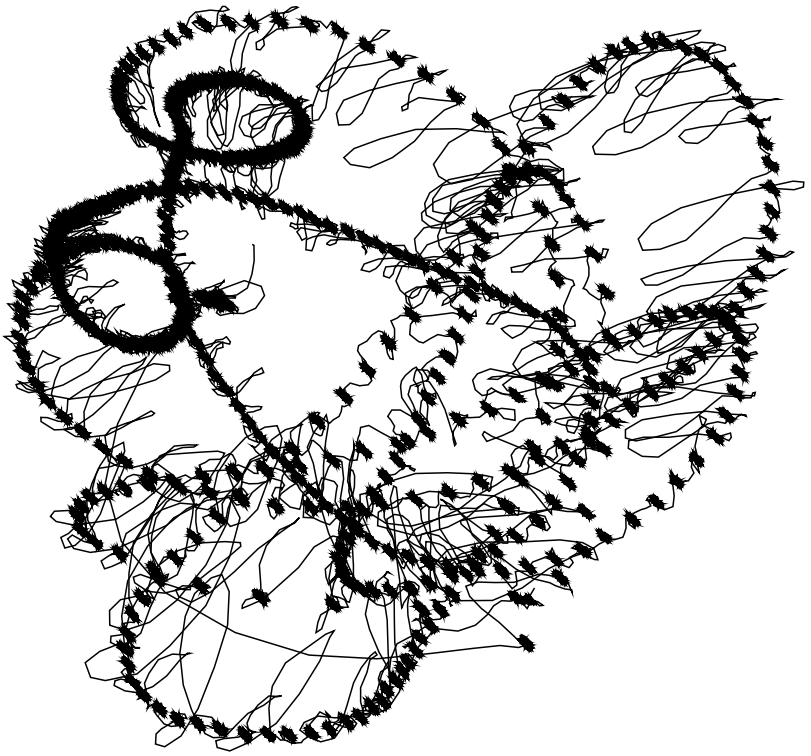


FIGURE 6.15: The same data used for the lower part of Figure 6.14 is presented as amplitude and phase data, plotted in the complex plane with power as the magnitude and phase as the argument, to present a different perspective on the structure of the sequence, and the timesteps and transients that connect them. Each timestep is shown as a tight bundle of fairly constant power and phase. We would expect the transient connecting lines to transfer smoothly between blocks, but instead there are often elaborate loops which go far away from the timesteps they are connecting.

Due to the extent of these transients, when analysing this data it is convenient to discretise the shape back into the original 512 timesteps by taking an average of the values within each timestep. As is evident from Figure 6.15, each timestep can be distinctly identified, so it is simple to average the correct groups. It is unclear whether such averaging should include the transients at the beginning of each pulse, or only consider an average of some interior range of points. In fact, for timesteps of $4\,\mu$s as for this sequence, the two averages are essentially indistinguishable, given the duration of the transient periods. This fact suggests that although transients are obviously a

significant problem for shorter timestep sequences, the other errors preventing perfect performance of logic gates for coarser sequences are not in fact transient related.

Figure 6.16 shows this timestep averaged sequence for both the phased and unphased original sequences, and compares the absolute value of each with the amplitude modulation of the sequence source file used to generate the RF. It is clear that the shape of the unphased sequence matches the target shape extremely well, the two lines are virtually indistinguishable. Unfortunately, for the real sequence, which includes phase modulation as well, it is clear that the shape is a lot less precisely matched. There are several places where the difference between the measured amplitude and the target amplitude differ by a few percent, which is quite enough to explain the errors observed in the quality of spectra.

Figure 6.17 shows the corresponding comparison between the phase of the measured data and that requested in the original pulse file, although this comparison can obviously only be made for the original, phased version of the measured data. The agreement here is very good, much better than for the amplitude data.

Examining Figure 6.16 in more detail, some slightly surprising observations can be made. The deviations between measurement and theory seem to follow no clear pattern. For example, one of the two central peaks has extremely good agreement between the two lines, but the other has one of the largest errors. The two troughs to the right both have deviations, but one has the experimental data too low, and the other too high, even though the shape of the curve looks very similar.

If the deviation between the two curves is not dependent upon their shape, the question is what might cause this behaviour. It is clearly not purely random, as the data is acquired through the oscilloscope by averaging a very large number of runs repeating after a short interval, so that any such behaviour would be averaged out to some mean value.

The other plausible explanation for this deviation is to suggest that it is correlated with the phase of the RF. The simplest way to test this hypothesis is to plot the measured deviation from the prescribed power as a function of phase. This is shown in Figure 6.18. It is immediately clear that there is a definite relationship between the measured power deviation and the phase of the radiation. What is more, from Figure 6.17 it can clearly

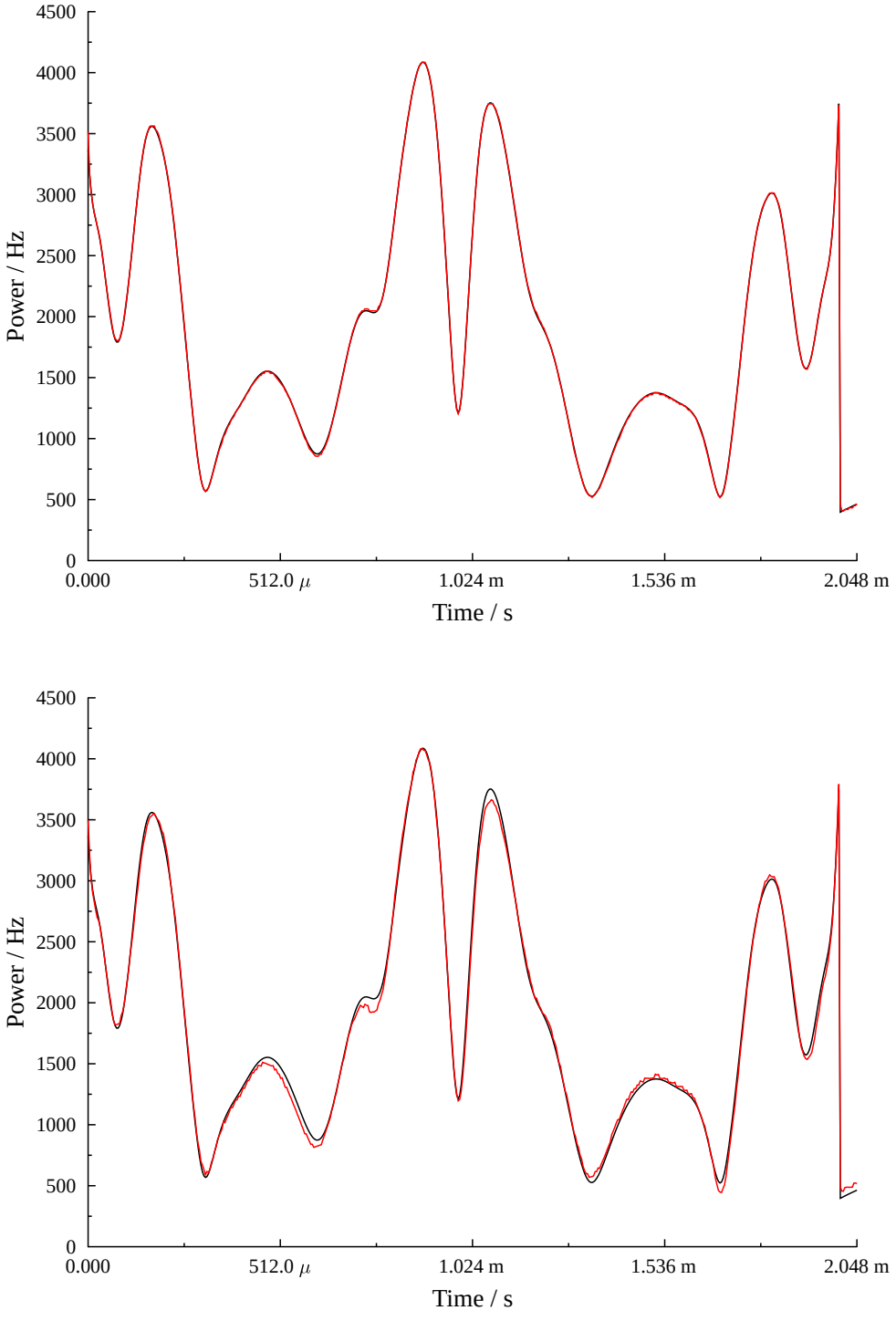


FIGURE 6.16: The sequence data from Figure 6.14 is averaged to give one value per timestep, and the magnitude of the RF power in each case is compared with the power specified in the original sequence file. The sequence file data is shown in black, and the measured data is in red. Once again the top figure shows the results for a shaped pulse with all phase modulation removed, and the bottom figure shows the original GRAPE pulse with phase modulation. The agreement between simulation and experiment is much worse when phase modulation is included.

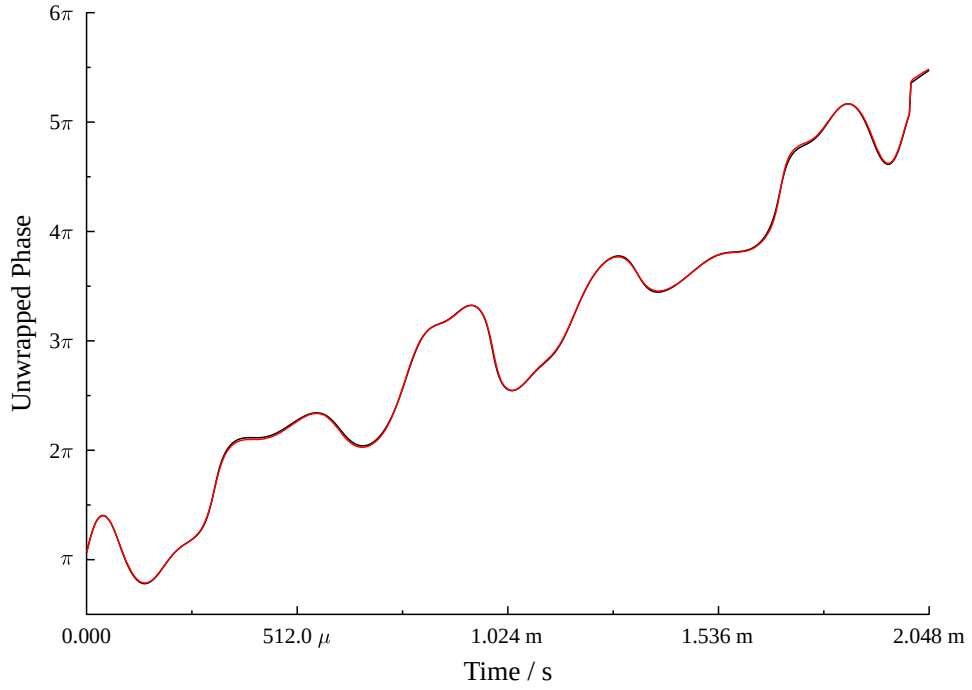


FIGURE 6.17: Here the phase data measured by the oscilloscope for the true pulse sequence is compared to the intended phase values from the sequence file. The phase here is shown "unwrapped" so rather than showing phases bounded between 0 and 2π with sharp jumps at the boundaries, the phase is allowed to increase to take on larger values. Once again the measured data is in red, and the file data in black.

be seen that the range of phases extends over more than 4π, which is implemented in the spectrometer, and plotted in Figure 6.18, as a wrap around of the data points. The two curves overlay one another precisely.

Although Figure 6.18 provides clear evidence of a correlation between phase and power deviation, this approach is not the easiest way to analyse the precise nature of the relationship between the two. Because phase information has to be obtained by measuring two results in quadrature and then recombining them, noise levels are unevenly distributed over the range of angles. Also, the fact that two high frequency signals are being generated and beaten against one another to provide the pulse shape means that errors in the reference signal will also contaminate the measurements.

Instead, the simplest solution is to return to actual NMR experiments. In a simple single qubit system, applying a short pulse of RF to produce a rotation angle of around 10° will produce a signal size which is nearly linear in the applied power. By performing a sequence of such experiments in which the phase angle is varied by 1° each time using the

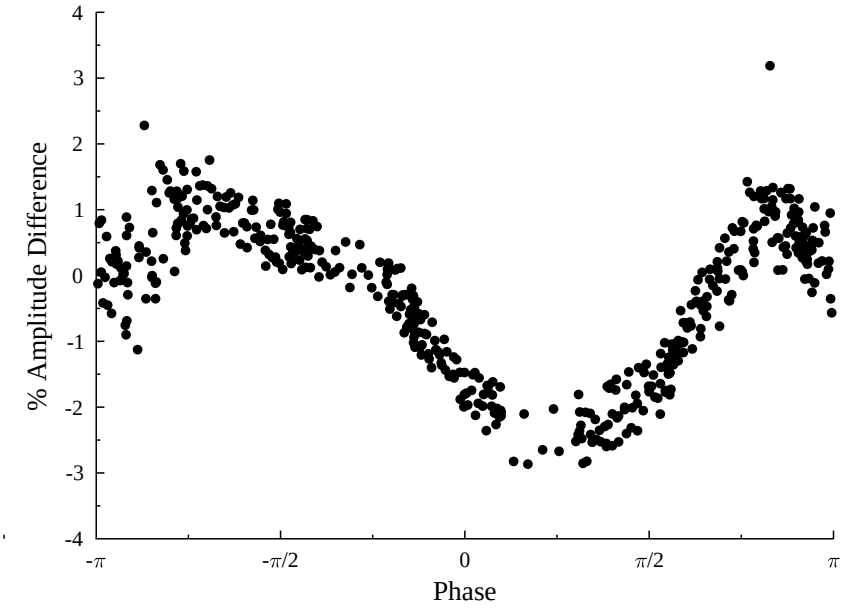


FIGURE 6.18: The percentage difference between measured RF power and theoretical power from the sequence file is plotted against the phase of the RF. It might be expected that the errors in power should be independent of phase, but clear evidence can be seen here of a correlation between the two. There are only significant outlier from the main data set, these both occur in timesteps with very low absolute power so the noise has a larger affect on deviation.

same pulse shaping techniques as used to generate GRAPE sequences, a set of spectra is obtained, which should all be identical when individually phased, but will in fact have some variation in size due to the phase dependent power deviation.

Figure 6.19 shows precisely this variation in signal as a function of phase from what should be identical pulse sequences. Once again there is a clearly repeating pattern showing that the power amplitude can vary by around $\pm 3.5\%$ over 360°. However, the shape of this curve is somewhat different from that of Figure 6.18. In particular, this data distinctly shows discontinuous jumps at the transitions between the four phase quadrants. Apart from that, the data appears to be broadly cosinusoidal but for the much flatter fourth quadrant.

One of the major puzzles of the poor performance of GRAPE sequences on the spectrometer is the relative effectiveness with which conventional shaped pulses can be performed. However, it is important to remember that many of these conventional pulses, although

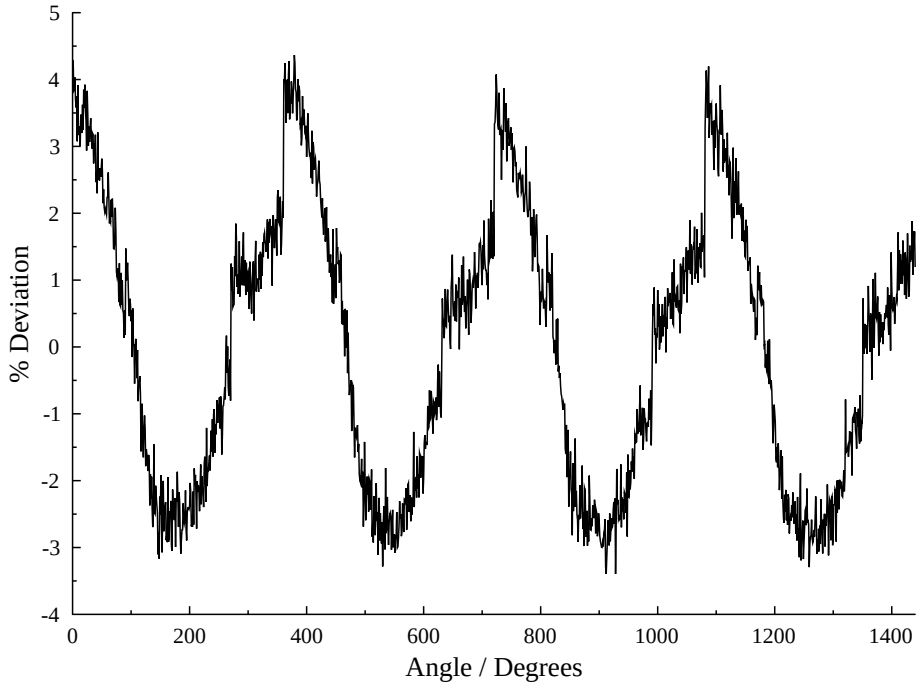


FIGURE 6.19: Percentage deviation of signal from the mean value for a range of phase angles from 0° to 1440° with a step size of 1°. The large range of phase angles was chosen to examine the repeat pattern of the function. The RF generator will ignore multiples of 360° in the phase angle, so each repeat of the pattern should be identical. There are some small differences visible here, these are most probably due to variations in laboratory conditions over the data acquisition period, which is several hours, combined with a certain amount of random noise.

they include very sophisticated amplitude shaping, have little or no phase shifting behaviour. As shown in Figure 6.16, this kind of shaped pulse can be much more accurately reproduced than one with complex phase shifting.

The other common form of conventional shaped pulse uses phase ramping to shift its excitation profile in frequency space. In this case the phase of the pulse tends to shift much more rapidly than in typical GRAPE sequences, so the averaged effect of the pulse amplitude distortion will tend to cancel out. However, by directly moving the transmitter frequency of the spectrometer over the range of 20 Hz, and using phase ramping techniques to attempt to cancel the off-resonance effects, it is possible to see small differences in the effect of pulse sequences as the offset phase of the start of the pulse changes.

It is appropriate at this point to examine more closely the details of how phased RF is

generated on the spectometer, to try and better understand the errors that are visible. On the Varian Unity INOVA spectrometer, phased RF is constructed by combining two raw RF components with phases 90° apart in a suitable ratio.

If the total RF output can be expressed as the weighted sum of a sine and a cosine:

$$\mathrm{RF} = R\sin\omega t + I\cos\omega t \tag{6.1}$$

it is clear that by choosing $R = A\cos\phi$ and $I = A\sin\phi$, the total combined RF signal will take the form

$$\mathrm{RF} = A\cos\phi\sin\omega t + A\sin\phi\cos\omega t = A\sin(\omega t + \phi) \tag{6.2}$$

which is the desired phase shifted RF.

The RF generation circuitry uses a two tiered system to produce its phase shifted RF. Because the most common conventionally required phases of RF are simply the four major axes x, $-x$, y and $-y$, these are generated separately by a phase shifter so they can be used independently of the small angle phase shifting. When a pulse sequence is being performed, for each subpulse the major phase is identified first as the highest phase choice below the target phase, and the phase shifter produces RF at this major phase from the raw input oscillation.

The major phase RF is then supplied to the phase splitter, which converts it into two RF components, one at the original phase, and one ahead by 90°. These components represent the sine and cosine terms of Equation 6.1, and they need to be combined in the correct ratio in the combiner circuit. To calculate the multiplicative factors for each term, the small angle phase circuitry is used.

The spectrometer stores the small angle phase shift using a nine bit register, giving 512 possible values, although in fact only 360 of these are used. Because each major phase covers a range of 90°, the small angle phase shift register specifies the phase to 0.25° of precision. The small angle phase is used as an index into sine and cosine look up tables (this is faster than calculating the values and there are only 360 possible values so is also memory efficient) to calculate the appropriate factors.

Next the sine and cosine factors are passed through a pair of digital to analogue converters to produce an analogue amplitude form for each and these are then passed into the combiner circuitry together with the raw RF components. Finally these four components are mixed together to produce the correctly phased RF output which can be sent to the probe. The entire phase shifting process is represented visually in Figure 6.20.

This form of small angle phase shifting is straightforward to implement because it operates by simply manipulating a source of raw RF oscillating at the resonance frequency, so that it need only make adjustments when the requested phase or power changes, which will be on a much longer timescale than the actual oscillation of the signal. However, precisely because of this design, it requires a whole sequence of operations to be performed on the raw RF to obtain the correct form. Any errors in any of these operations will propagate through to leave errors in the final RF output. It seems most likely that the amplitude deviations observed in Figure 6.19 are caused by errors in this phase shifting process, as Figure 6.14 clearly shows that the amplitude shaping performs very well in the absence of phase shifting.

Unfortunately, the large number of potential sources of error within the small angle phase shifting circuitry mean that is it very difficult to pin-point the exact form of the amplitude deviation curve shown in Figure 6.19. The sharp discontinuities which occur precisely at each Cartesian axis suggest that the phase shifter circuitry may not be correctly producing RF at 0°, 90°, 180° and 270°. However, this in itself is not enough to explain the complete shape of the deviation curve. It is also possible that the phase splitter is splitting the RF incorrectly, leaving the two output signals with different magnitudes or with an incorrect relative phase. One or both of the DACs may also have non-linearities or zero offsets. So far, no simply error model of the phase generation circuitry has adequately accounted for all the features of Figure 6.19, making the application of correction factors rather difficult.

Because of these potential errors, this form of small angle phase shifting is no longer used for state of the art NMR spectrometers. Instead, the prevailing technique is that of "Direct Digital Sythesis" (DDS) [114–116]. This approach does not use any kind of raw oscillator signal, but instead simply maintains a count of the current phase which is increased with each tick of its internal clock according to the oscillator frequency. This phase is then converted into an amplitude through a sine look up table and output

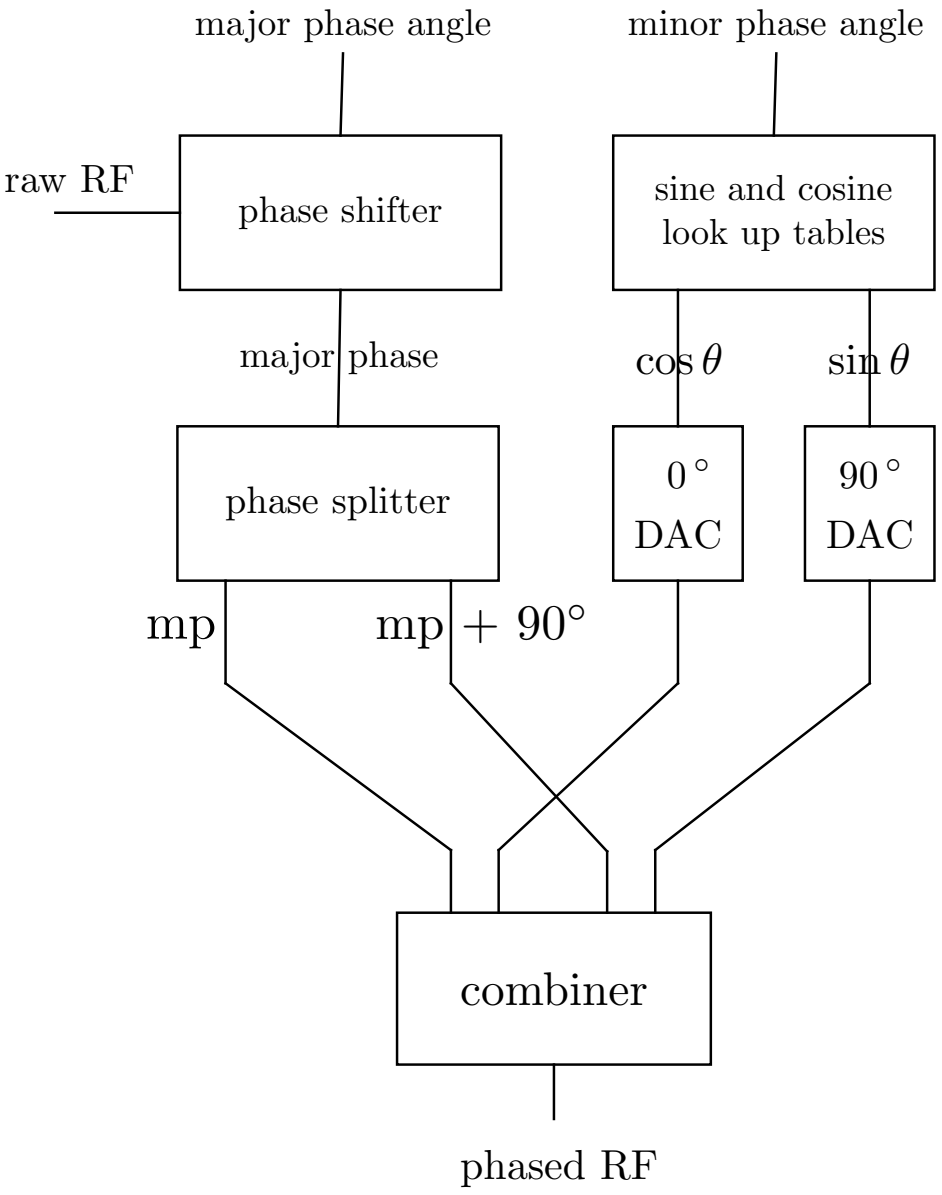


FIGURE 6.20: Phase information for RF is divided into a major angle (0°/90°/180°/270°) and a small angle with 360 possible values giving accuracy of 0.25°. The raw RF, which defines the 0° phase, is fed into the phase shifter, which converts it to the correct major angle specified in the sequence file. This major phase then enters the phase splitter, which converts it into two components, one at the original phase, and one 90° ahead. These components must be combined in the correct ratio to give the correctly phased RF, as described in the main text. To calculate the terms for the ratio, the small angle phase from the pulse sequence is fed into sine and cosine look up tables. These values are converted into analogue magnitudes by a pair of DACs and the resulting signal is combined with the two RF signals in the combiner to give the final phased RF signal.

through a DAC. Thus rather than use an oscillating analogue signal to provide the raw RF, DDS uses a digital signal to generate the oscillating RF directly at the desired frequency. The phase of the RF can be changed simply by adding or subtracting from the reference phase of the DDS system.

DDS means that phase changes can be made on the same kind of timescale as the natural oscillations of the system, and further that performing small angle phase shifts becomes no more complicated than just producing the original oscillating RF in the first place. Therefore GRAPE sequences should perform with the same quality as standard NMR pulses, which is not the case with the Unity INOVA. Furthermore, phase changes in DDS are equivalent amplitude changes, so one might very reasonably expect the transient effects from using DDS to be much less than with the INOVA technology, as shown in Figure 6.14. This would allow the use of shorter timesteps, and improve fidelity and systematic error tolerance.

For all of these reasons, it seems extremely likely that implementing GRAPE sequences on a spectrometer using DDS would offer superior performance of the requested sequence, and correspondingly reduced error signals in the resulting spectra. This is likely to be a priority area of exploration for further research in this area, if access to suitable equipment can be obtained.

6.9 Conclusions

The GRAPE algorithm has become an area of enormous interest throughout the NMR community over the last few years, with many groups making use of it for a wide range of purposes [117–125]. The quality of the experimentally implemented pulse sequences is varies signficantly, but it is clear that the best implementations are somewhat superior to the highest quality sequences achieved in my experiments. As yet the precise reasons for this remain unclear, but it is significant to note that all the other principal groups working with GRAPE at this time are using recent generation Bruker spectrometers equipped with DDS pulse shaping circuitry.

Although at least one significant source of error remains in my experimental implementation of GRAPE, some experimental successes have been demonstrated. Several sources of error have been identified and removed, good quality selective homonuclear 90° rotation

gates have been generated which perform true unitary transformations independent of starting state, and the controlled-transfer gate method of generating pseudo-pure states has been implemented to produce reasonable quality results. The principal remaining error seems to have been identified, and it is to be hoped that by removing it in one way or another, the quality of the GRAPE sequences will increase considerably.

In the meantime, my research has explored some interesting theoretical developments in the GRAPE algorithm, allowing a wider variety of problems to be tackled, and more efficiently, for example by using pulses and delays [126]. I have also demonstrated the enormous potential that can be harnessed from the application of GPUs to the problem. These enhancements will allow the development of a superior quality of GRAPE sequences, and in a much shorter time than is currently possible. Once the physical implementation reaches a quality to match the simulation, or once the simulation has included enough of the errors of the physical implementation, a whole range of new techniques will become possible.

Appendix A

Spectrometer Hardware Specifications

The NMR spectrometer used for all experiments described in this thesis was a Varian UNITY INOVA 600, with a nominal ^{1}H frequency of 600 MHz. The principal specifications of this spectrometer are described in this Appendix in the following table.

Model	UNITY INOVA 600
Vendor	Varian
RF Console	
Phase Resolution	0.25°
Linear Amplitude Modulation	60 dB in 4096 steps
Amplitude Scaling	79 dB in 1 dB steps
RF Channels	
RF Channel Frequency	20-600 MHz
Frequency Resolution	0.1 Hz
Linear RF Amplifier Power	50 W
Acquisition Computer	
Memory	16 MB
Acquisition Timing Resolution	12.5 ns
ADCs	16 bit
Quadrature Sampling Rate	500 kHz
Waveform Generator	
Pulse element timing	50 ns
Minimum Event Time	200 ns
Phase Settling Time	50 ns
Probe	
Probe Model	1H (13C/15N) 5 mm PFG Triple Resonance Probe
Maximum Gradient Strength	z-axis only 6.8 mT cm^{-1}
Gradient Settling Time	50 μs

Bibliography

[1] J. Balchin, *Science: 100 scientists who changed the world.* Enchanted Lion, 2003.

[2] A. Hodges, *Alan Turing: The Enigma.* Vintage, 2006.

[3] G. E. Moore, "Cramming more components onto integrated circuits," *Proceedings of the IEEE*, vol. 86, no. 1, pp. 82–85, Jan. 1998.

[4] D. Deutsch, "Quantum theory, the Church-Turing principle and the universal quantum computer," *Proc. Roy. Soc. Lond. A*, vol. 400, no. 1818, pp. 97–117, 1985.

[5] C. H. Bennett and D. P. DiVincenzo, "Quantum information and computation," *Nature*, vol. 404, no. 6775, pp. 247–255, Mar. 2000.

[6] E. Knill, "Quantum computing," *Nature*, vol. 463, no. 7280, pp. 441–443, Jan. 2010.

[7] R. P. Feynman, "Simulating physics with computers," *Int. J. Theor. Phys.*, vol. 21, no. 6, pp. 467–488, 1982.

[8] L. K. Grover, "Quantum mechanics helps in searching for a needle in a haystack," *Phys. Rev. Lett.*, vol. 79, no. 2, pp. 325–328, 1997.

[9] P. W. Shor, "Proceedings of the 35th Annual Symposium on the Foundations of Computer Science," 1994, pp. 124–134.

[10] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, pp. 120–126, Feb. 1978.

[11] T. Toffoli, "Physics and computation," *Int. J. Theor. Phys.*, vol. 21, no. 3, pp. 165–175, 1982.

[12] J. Shu, X.-B. Zou, Y.-F. Xiao, and G.-C. Guo, "Quantum phase gate of photonic qubits in a cavity QED system," *Phys. Rev. A*, vol. 75, no. 4, Apr. 2007.

[13] W. Rakreungdet, J. H. Lee, K. F. Lee, B. E. Mischuck, E. Montano, and P. S. Jessen, "Accurate Microwave Control and Real-Time Diagnostics of Neutral Atom Qubits," *Phys. Rev. A*, vol. 79, p. 022316, 2009.

[14] J. I. Cirac and P. Zoller, "Quantum Computations with Cold Trapped Ions," *Phys. Rev. Lett.*, vol. 74, no. 20, pp. 4091–4094, May 1995.

[15] H. Haffner, C. F. Roos, and R. Blatt, "Quantum computing with trapped ions," *Phys. Rep.*, vol. 469, pp. 155–203, 2008.

[16] G. Burkard, H.-A. Engel, and D. Loss, "Spintronics and Quantum Dots for Quantum Computing and Quantum Communication," *Fort. der Physik*, vol. 48, no. 9-11, pp. 965–986, 2000.

[17] J. A. Jones, "Quantum computing with NMR," *Progress in Nuclear Magnetic Resonance Spectroscopy*, vol. 59, no. 2, pp. 91–120, 2011.

[18] W. S. Warren, "The Usefulness of NMR Quantum Computing," *Science*, vol. 277, p. 1688, 1997.

[19] D. P. DiVincenzo, "Dogma and heresy in quantum computing," *Quant. Info. Comp.*, vol. 1S, pp. 1–6, 2001.

[20] P. A. M. Dirac, *The Principles of Quantum Mechanics*, 4th ed. Oxford University Press, 1958.

[21] J. A. Jones, "Quantum Computing with NMR," *Prog. NMR Spectrosc*, vol. 59, pp. 91–120, 2011.

[22] R. R. Ernst, G. Bodenhausen, and A. Wokaun, *Principles of Nuclear Magnetic Resonance in One and Two Dimensions.* Oxford University Press, 1987.

[23] O. W. Sørensen, G. W. Eich, M. H. Levitt, G. Bodenhausen, and R. R. Ernst, "Product operator formalism for the description of NMR pulse experiments," *Prog. NMR Spectrosc*, vol. 16, pp. 163–192, 1983.

[24] R. P. Feynman, *Feynman Lectures on Computation*, A. J. G. Hey and R. W. Allen, Eds. Penguin Books, 1999.

[25] N. D. Mermin, *Quantum Computer Science*. Cambridge University Press, 2007.

[26] M. D. Bird, J. E. Crow, and P. Schlottmann, "The National High Magnetic Field Laboratory: Condensed Matter Science in Continuous Magnetic Fields," *J. Low Temp. Phys.*, vol. 133, no. 1, pp. 203–225, 2003.

[27] I. L. Chuang, N. Gershenfeld, M. G. Kubinec, and D. W. Leung, "Bulk Quantum Computation with Nuclear Magnetic Resonance: Theory and Experiment," *Proc. Roy. Soc. Lond. A*, vol. 454, no. 1969, pp. 447–467, 1998.

[28] D. G. Cory, A. F. Fahmy, and T. F. Havel, "Nuclear magnetic resonance spectroscopy: an experimentally accessible paradigm for quantum computing," in *PhysComp96: Proceedings of the fourth workshop on physics and computation*, T. B. M. Toffoli and J. Leão, Eds. New England Complex Systems Institute, 1996, pp. 87–91.

[29] N. A. Gershenfeld and I. L. Chuang, "The Usefulness of NMR Quantum Computing: Response," *Science*, vol. 277, p. 1689, 1997.

[30] W. H. Zurek, "Decoherence and the Transition from Quantum to Classical," *Physics Today*, vol. 44.10, pp. 36–44, 1991.

[31] I. L. Chuang, R. Laflamme, P. W. Shor, and W. H. Zurek, "Quantum computers, factoring, and decoherence," *Science*, vol. 270, no. 5242, pp. 1633–1635, 1995.

[32] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang, "Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance," *Nature*, vol. 414, no. 6866, pp. 883–887, Dec. 2001.

[33] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[34] L. Emsley and G. Bodenhausen, "Phase shifts induced by transient Bloch-Siegert effects in NMR," *Chem. Phys. Lett.*, vol. 168, pp. 297–303, 1990.

[35] I. I. Rabi, "Space Quantization in a Gyrating Magnetic Field," *Phys. Rev.*, vol. 51, pp. 652–654, Apr. 1937.

[36] P. J. Hore, *Nuclear Magnetic Resonance*. Oxford University Press, 1995.

[37] P. J. Hore, J. A. Jones, and S. Wimperis, *NMR: The Toolkit.* Oxford University Press, 2000.

[38] D. P. DiVincenzo, "Quantum gates and circuits," *Proc. Roy. Soc. Lond. A*, vol. 454, no. 1969, pp. 261–276, Jan. 1998.

[39] D. Deutsch, "Quantum computational networks," *Proc. Roy. Soc. Lond. A*, vol. 425, no. 1868, pp. 73–90, 1989.

[40] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, "Elementary gates for quantum computation," *Phys. Rev. A*, vol. 52, no. 5, pp. 3457–3467, 1995.

[41] A. Barenco, "A Universal Two-Bit Gate for Quantum Computation," *Proc. Roy. Soc. Lond. A*, vol. 449, no. 1937, pp. 679–683, 1995.

[42] A. Y. Kitaev, "Quantum computations: algorithms and error correction," *Russian Mathematical Surveys*, vol. 52, no. 6, p. 1191, 1997.

[43] C. M. Dawson and M. A. Nielsen, "The Solovay-Kitaev algorithm," Tech. Rep., 2005.

[44] J. A. Jones, "NMR quantum computation," *Prog. NMR Spectrosc*, vol. 38, no. 4, pp. 325–360, Jun. 2001.

[45] N. Linden, B. Hervé, R. J. Carbajo, and R. Freeman, "Pulse sequences for NMR quantum computers: how to manipulate nuclear spins while freezing the motion of coupled neighbours," *Chem. Phys. Lett.*, vol. 305, no. 1-2, pp. 28–34, May 1999.

[46] M. D. Bowdrey, J. A. Jones, E. Knill, and R. Laflamme, "Compiling gate networks on an Ising quantum computer," *Phys. Rev. A*, vol. 72, no. 3, p. 032315, Sep. 2005.

[47] M. D. Price, T. F. Havel, and D. G. Cory, "Multiqubit logic gates in NMR quantum computing," *New J. Phys.*, vol. 2, pp. 101–109, May 2000.

[48] M. D. Price, E. M. Fortunato, M. A. Pravia, C. Breen, S. Kumaresean, G. Rosenberg, and D. G. Cory, "Information transfer on an NMR quantum information processor," *Concept. Magn. Reson.*, vol. 13, no. 3, pp. 151–158, 2001.

[49] Y. Aharonov and L. Vaidman, "Properties of a quantum system during the time interval between two measurements," *Phys. Rev. A*, vol. 41, no. 1, pp. 11–20, 1990.

[50] J. A. Jones, "Quantum computing and nuclear magnetic resonance," *PhysChemComm*, vol. 11, no. 11, pp. 1–8, 2001.

[51] M. Kawamura, B. Rowland, and J. A. Jones, "Preparing pseudopure states with controlled-transfer gates," *Phys. Rev. A*, vol. 82, no. 3, p. 032315, Sep. 2010.

[52] L. Landau and E. Lifshitz, *Statistical Physics*, 3rd ed. Butterworth-Heinemann, 1996, vol. 1.

[53] L. J. Schulman and U. V. Vazirani, "Molecular Scale Heat Engines and Scalable Quantum Computation," in *Proc. 31st ACM Symp. Theor. Comput.*, J. S. Vitter, L. Larmore, and T. Leighton, Eds. New York: ACM Press, 1999, pp. 322–329.

[54] P. O. Boykin, T. Mor, V. Roychowdhury, F. Vatan, and R. Vrijen, "Algorithmic cooling and scalable NMR quantum computers," *Proc. Natl. Acad. Sci USA*, vol. 99, no. 6, pp. 3388–3393, Mar. 2002.

[55] E. Knill, I. Chuang, and R. Laflamme, "Effective pure states for bulk quantum computation," *Phys. Rev. A*, vol. 57, no. 5, pp. 3348–3363, 1998.

[56] T. S. Mahesh and A. Kumar, "Ensemble quantum-information processing by NMR: Spatially averaged logical labeling technique for creating pseudopure states," *Phys. Rev. A*, vol. 64, no. 1, p. 012307, 2001.

[57] Y. Mori, R. Sawae, M. Kawamura, T. Sakata, and K. Takarabe, "Quantum Circuits for an Effective Pure State in NMR Quantum Computer," *Int. J. Quant. Comp.*, vol. 105, no. 6, p. 758, 2005.

[58] R. Marx, A. F. Fahmy, J. M. Myers, W. Bermel, and S. J. Glaser, "Approaching five-bit NMR quantum computing," *Phys. Rev. A*, vol. 62, no. 1, p. 012310, Jul. 2000.

[59] D. G. Cory, A. F. Fahmy, and T. F. Havel, "Ensemble quantum computing by NMR spectroscopy," *Proc. Natl. Acad. Sci USA*, vol. 94, pp. 1634–1639, 1997.

[60] D. G. Cory, M. D. Price, W. Maas, E. Knill, R. Laflamme, W. H. Zurek, T. F. Havel, and S. S. Somaroo, "Experimental Quantum Error Correction," *Phys. Rev. Lett.*, vol. 81, no. 10, pp. 2152–2155, 1998.

[61] J. A. Jones and M. Mosca, "Approximate quantum counting on an NMR ensemble quantum computer," *Phys. Rev. Lett.*, vol. 83, no. 5, pp. 1050–1053, Aug. 1999.

[62] M. Pravia, E. Fortunato, Y. Weinstein, M. D. Price, G. Teklemariam, R. J. Nelson, Y. Sharf, S. Somaroo, C. H. Tseng, T. F. Havel, and D. G. Cory, "Observations of quantum dynamics by solution-state NMR spectroscopy," *Concept. Magn. Reson.*, vol. 11, no. 4, pp. 225–238, 1999.

[63] D. G. Cory, M. D. Price, and T. F. Havel, "Nuclear magnetic resonance spectroscopy: An experimentally accessible paradigm for quantum computing," *Physica D*, vol. 120, no. 1-2, pp. 82–101, 1998.

[64] J. Du, T. Durt, P. Zou, H. Li, L. C. Kwek, C. H. Lai, C. H. Oh, and A. Ekert, "Experimental Quantum Cloning with Prior Partial Information," *Phys. Rev. Lett.*, vol. 94, no. 4, p. 040505, Feb. 2005.

[65] J. Zhang, X. Peng, N. Rajendran, and D. Suter, "Effect of system level structure and spectral distribution of the environment on the decoherence rate," *Phys. Rev. A*, vol. 75, no. 4, p. 042314, Apr. 2007.

[66] M. S. Anwar, J. A. Jones, and S. B. Duckett, "Sharing polarization within quantum subspaces," *Phys. Rev. A*, vol. 73, no. 2, p. 022322, Feb. 2006.

[67] J. A. Jones, M. Mosca, and R. H. Hansen, "Implementation of a quantum search algorithm on a quantum computer," *Nature*, vol. 393, no. 6683, pp. 344–346, May 1998.

[68] I. L. Chuang, L. M. K. Vandersypen, X. Zhou, D. W. Leung, and S. Lloyd, "Experimental realization of a quantum algorithm," *Nature*, vol. 393, no. 6681, pp. 143–146, 1998.

[69] M. H. Levitt, "Composite pulses," *Prog. NMR Spectrosc*, vol. 18, pp. 61–122, 1986.

[70] H. K. Cummins, G. Llewellyn, and J. A. Jones, "Tackling systematic errors in quantum logic gates with composite rotations," *Phys. Rev. A*, vol. 67, no. 4, p. 042308, 2003.

[71] J. A. Jones, "Robust quantum information processing with techniques from liquid-state NMR," *Phil. Trans. Roy. Soc. A*, vol. 361, no. 1808, pp. 1429–1440, Jul. 2003.

[72] E. Knill, R. Laflamme, R. Martinez, and C. H. Tseng, "An algorithmic benchmark for quantum information processing," *Nature*, vol. 404, no. 6776, pp. 368–370, Mar. 2000.

[73] R. Freeman, "Shaped radiofrequency pulses in high resolution NMR," *Prog. NMR Spectrosc*, vol. 32, no. 1, pp. 59–106, 1998.

[74] J. A. Jones and M. Mosca, "Implementation of a quantum algorithm on a nuclear magnetic resonance quantum computer," *J. Chem. Phys.*, vol. 109, no. 5, pp. 1648–1653, Aug. 1998.

[75] C. Bauer, R. Freeman, T. Frenkiel, J. Keeler, and A. J. Shaka, "Gaussian pulses," *J. Magn. Reson*, vol. 58, no. 3, pp. 442–457, 1984.

[76] W. S. Warren, "Effects of arbitrary laser or NMR pulse shapes on population inversion and coherence," *J. Chem. Phys.*, vol. 81, p. 5437, 1984.

[77] H. Geen and R. Freeman, "Band-selective radiofrequency pulses," *J. Magn. Reson*, vol. 93, no. 1, pp. 93–141, 1991.

[78] L. Viola, E. Knill, and S. Lloyd, "Dynamical decoupling of open quantum systems," *Phys. Rev. Lett.*, vol. 82, no. 12, pp. 2417–2421, 1999.

[79] W. S. Warren, D. P. Weitekamp, and A. Pines, "Theory of selective excitation of multiple-quantum transitions," *J. Chem. Phys.*, vol. 73, pp. 2084–2099, 1980.

[80] C. J. Joachain, *Quantum collision theory.* Amsterdam : North-Holland Pub. Co. ; New York : American Elsevier Pub. Co, 1975.

[81] E. M. Fortunato, M. A. Pravia, N. Boulant, G. Teklemariam, T. F. Havel, and D. G. Cory, "Design of strongly modulating pulses to implement precise effective Hamiltonians for quantum information processing," *J. Chem. Phys.*, vol. 116, no. 17, pp. 7599–7606, May 2002.

[82] N. Boulant, K. Edmonds, J. Yang, M. A. Pravia, and D. G. Cory, "Experimental demonstration of an entanglement swapping operation and improved control in NMR quantum-information processing," *Phys. Rev. A*, vol. 68, no. 3, p. 032305, 2003.

[83] Y. S. Weinstein, T. F. Havel, J. Emerson, N. Boulant, M. Saraceno, S. Lloyd, and D. G. Cory, "Quantum process tomography of the quantum Fourier transform," *J. Chem. Phys.*, vol. 121, p. 6117, 2004.

[84] N. Jacobsen, *NMR spectroscopy explained: simplified theory, applications and examples for organic chemistry and structural biology.* Wiley-Interscience, 2007.

[85] K. G. Murty, *Linear Programming.* John Wiley & Sons, Inc., 1983.

[86] C. A. Ryan, C. Negrevergne, M. Laforest, E. Knill, and R. Laflamme, "Liquid-state nuclear magnetic resonance as a testbed for developing quantum control methods," *Phys. Rev. A*, vol. 78, p. 012328, 2008.

[87] N. Khaneja, T. Reiss, C. Kehlet, T. Schulte-Herbrüggen, and S. J. Glaser, "Optimal control of coupled spin dynamics: design of NMR pulse sequences by gradient ascent algorithms," *J. Magn. Reson*, vol. 172, no. 2, pp. 296–305, 2005.

[88] J. Kiefer, "Sequential minimax search for a maximum," *Proceedings of the American Mathematical Society*, vol. 4, no. 3, pp. 502–506, 1953.

[89] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes – The Art of Scientific Computing.* Cambridge: Cambridge University Press, 1986.

[90] J. R. Shewchuk, "An introduction to the conjugate gradient method without the agonizing pain," Tech. Rep., 1994.

[91] P. de Fouquieres, S. G. Schirma, S. J. Glaser, and I. Kuprov, "Second order gradient ascent pulse engineering," *J. Magn. Reson*, vol. 212, no. 2, pp. 412–417, 2011.

[92] I. R. C. T. Kuprov, "Derivatives of spin dynamics simulations," *Journal of Chemical Physics*, vol. 131, no. 23, 2009.

[93] B. Vogeli, "Comprehensive description of NMR cross-correlated relaxation under anisotropic molecular tumbling and correlated local dynamics on all time scales," *The Journal of Chemical Physics*, vol. 133, no. 1, p. 014501, 2010.

[94] R. Bhattacharyya and A. Kumar, "Use of cross-correlated NMR relaxation for the study of motional anisotropy of liquid crystals," *Chem. Phys. Lett.*, vol. 372, no. 1â"2, pp. 35–44, 2003.

[95] R. Fiala, N. Špačková, S. Foldynová-Trantírková, J. Šponer, V. Sklenář, and L. Trantírek, "NMR Cross-Correlated Relaxation Rates Reveal Ion Coordination Sites in DNA," *JOURNAL OF THE AMERICAN CHEMICAL SOCIETY*, vol. 133, no. 35, pp. 13 790–13 793, 2011.

[96] H. K. Cummins and J. A. Jones, "Use of composite rotations to correct systematic errors in NMR quantum computation," *New J. Phys.*, vol. 2, pp. 1–12, Mar. 2000.

[97] S. Wimperis, "Broadband, narrowband and passband composite pulses for use in advanced NMR experiments," *J. Magn. Reson. Ser. A*, vol. 109, no. 2, pp. 221–231, 1994.

[98] L. Xiao and J. A. Jones, "Robust logic gates and realistic quantum computation," *Phys. Rev. A*, vol. 73, no. 3, p. 032334, Mar. 2006.

[99] K. R. Brown, A. W. Harrow, and I. L. Chuang, "Arbitrarily accurate composite pulse sequences," *Phys. Rev. A*, vol. 70, p. 052318, 2004.

[100] ——, "Erratum: Arbitrarily accurate composite pulse sequences," *Phys. Rev. A*, vol. 72, p. 039905(E), 2005.

[101] P. J. Hore, "Solvent suppression in Fourier transform nuclear magnetic resonance," *J. Magn. Reson*, vol. 55, no. 2, pp. 283–300, 1983.

[102] D. A. Patterson and J. L. Hennessey, *Computer Organization and Design: the Hardware/Software Interface*, 2nd ed. San Francisco: Morgan Kaufmann Publishers, Inc., 1998.

[103] U. Kapasi, S. Rixner, W. Dally, B. Khailany, J. Ahn, P. Mattson, and J. Owens, "Programmable Stream Processors," *IEEE Computer*, vol. 36, no. 8, pp. 54–62, 2003.

[104] nVidia. [Online]. Available: http://www.nvidia.com

[105] CUDA. [Online]. Available: http://www.nvidia.com/object/cuda_home_new.html

[106] DirectCompute on AMD. [Online]. Available: http://blogs.amd.com/play/2009/09/09/directx-11--what-to-expect/

[107] DirectCompute on nVidia. [Online]. Available: http://developer.nvidia.com/directcompute

[108] OpenCL. [Online]. Available: http://www.khronos.org/opencl/

[109] OSC. [Online]. Available: http://www.osc.ox.ac.uk

[110] Tesla Personal Supercomputer. [Online]. Available: http://www.nvidia.com/object/personal-supercomputing.html

[111] R. B. Bradley, E. D. Becker, and H. T. Miles, "Nuclear magnetic resonance studies of methyl derivatives of cytosine," *JOURNAL OF THE AMERICAN CHEMICAL SOCIETY*, vol. 87, no. 24, pp. 5575–5582, 1965.

[112] A. M. Childs, I. L. Chuang, and D. W. Leung, "Realization of quantum process tomography in NMR," *Phys. Rev. A*, vol. 64, no. 1, p. 012314, Jun. 2001.

[113] M. D. Bowdrey and J. A. Jones, "Single qubit gates with jump and return sequences," *Phys. Rev. A*, vol. 74, no. 5, p. 052324, Nov. 2006.

[114] J. Yun, J. Yu, T. Hongyan, and L. Gengying, "A complete digital radio-frequency source for nuclear magnetic resonance spectroscopy," *Review of Scientific Instruments*, vol. 73, no. 9, pp. 3329–3331, Sep. 2002.

[115] C. L. Wilkins, "Digital frequency synthesizers for nuclear magnetic resonance spectroscopy," *Review of Scientific Instruments*, vol. 65, no. 10, pp. 3291–3292, 1994.

[116] W. Mao, "A modularized pulse programmer for NMR spectroscopy," *Measurement Science and Technology*, vol. 22, no. 2, p. 025901, 2011.

[117] J. S. Hodges, J. C. Yang, C. Ramanathan, and D. G. Cory, "Universal control of nuclear spins via anisotropic hyperfine interactions," *Phys. Rev. A*, vol. 78, no. 1, p. 010303(R), 2008.

[118] T. E. Skinner, K. Kobzar, B. Luy, M. R. Bendall, W. Bermel, N. Khaneja, and S. J. Glaser, "Optimal control design of constant amplitude phase-modulated pulses: Application to calibration-free broadband excitation," *J. Magn. Reson*, vol. 179, no. 2, pp. 241–249, 2006.

[119] M. Möttönen, R. de Sousa, J. Zhang, and K. B. Whaley, "High-fidelity one-qubit operations under random telegraph noise," *Phys. Rev. A*, vol. 73, no. 2, p. 022332, 2006.

[120] M. K. Henry, C. Ramanathan, J. S. Hodges, C. A. Ryan, M. J. Ditty, R. Laflamme, and D. G. Cory, "Fidelity Enhancement by Logical Qubit Encoding," *Phys. Rev. Lett.*, vol. 99, no. 22, p. 220501, 2007.

[121] K. Kobzar, B. Luy, N. Khaneja, and S. J. Glaser, "Pattern pulses: design of arbitrary excitation profiles as a function of pulse amplitude and offset," *J. Magn. Reson*, vol. 173, no. 2, pp. 229–235, 2005.

[122] T. Vosegaard, C. T. Kehlet, N. Khaneja, S. J. Glaser, and N. C. Nielsen, "Improved excitation schemes for multiple-quantum magic-angle spinning for quadrupolar nuclei designed using optimal control theory," *J. Am. Chem. Soc.*, vol. 127, pp. 13 768–13 769, 2005.

[123] C. Kehlet, M. Bjerring, A. C. Sivertsen, T. Kristensen, J. J. Enghild, S. J. Glaser, N. Khaneja, and N. C. Nielsen, "Optimal control based NCO and NCA experiments for spectral assignment in biological solid-state NMR spectroscopy," *J. Magn. Reson*, vol. 188, no. 2, pp. 216–230, 2007.

[124] T. Schulte-Herbrüggen, A. Spörl, N. Khaneja, and S. J. Glaser, "Optimal control-based efficient synthesis of building blocks of quantum algorithms: A perspective from network complexity towards time complexity," *Phys. Rev. A*, vol. 72, no. 4, p. 042331, 2005.

[125] Z. Wu, J. Li, W. Zheng, J. Luo, M. Feng, and X. Peng, "Experimental demonstration of the Deutsch–Jozsa algorithm in homonuclear multispin systems," *Phys. Rev. A*, vol. 84, p. 042312, Oct. 2011.

[126] B. Rowland and J. A. Jones, "Implementing quantum logic gates with GRAPE: principles and practicalities," *Phil. Trans. A [Submitted]*.