The implementation of hybrid codes and nuclear magnetic resonance quantum information processing

by

Shayan-Shawn Majidy

A Thesis

Presented to

The University of Guelph

In partial fulfillment of requirements

for the degree of

Master of Science

in

Mathematics

Guelph, Ontario, Canada

©Shayan-Shawn Majidy, October 2017
The development of quantum error correcting techniques is of paramount importance to the ultimate goal of implementing practical quantum computers. The simultaneous transmission of quantum and classical information over a quantum channel was initially investigated in [10], and since been continued by others [16] [17] [27]. It was shown that there is an advantage to transmitting both quantum and classical information simultaneously, compared to independent transmissions. The characterization and construction of codes that allow to transmit both quantum and classical information, which we refer to as hybrid codes, was done from a coding theory perspective in [7] and by using the operator algebra quantum error correction (OAQEC) perspective in [5]. In this work we unify these two perspectives, showing that the coding theory formulation is a specific case of the OAQEC perspective. As a result we generalized the quantum hamming bound to the hybrid case. To date no such hybrid codes have been physically implemented. Nuclear magnetic resonance (NMR) quantum information processors (QIP) have been an excellent test bed for quantum computing [25]. The techniques of NMR had been refined for many years and can now be applied to quantum computing. NMR techniques provides a high degree of qubit control and long decoherence times compared to other QIP. These two properties of NMR QIP make it a prime candidate for implementing hybrid codes. In this work we developed a hybrid code and designed the circuit for encoding it. We used Matlab to simulate the pulse sequence which would be used in an NMR QIP to carry out the encoding of the hybrid circuit. According to those simulations the full encoding, a single qubit Pauli error and a full decoding can be carried out without the effects of decoherence destroying the quantum information beyond retrieval.
Acknowledgment

I would like to thank those have assisted me throughout this Masters degree, in chronological order. First, I would like to thank Monica Cojocaru for giving me the opportunity to pursue graduate studies. You believed in me and opened the door to graduate studies for me, that is something I will always be grateful for.

I would like to thank Jeremy Levick for kindling my interest in quantum computing, his passion for the subject is contagious. I would like to thank Rajesh Pereira for the hours of one on one help he provided, entering this field with no background in pure mathematics would of been impossible without his instruction.

Lastly, I would like to thank my advisors David Kribs and Bei Zeng. Being able to enter a new field under the mentorship of two such experts is an opportunity that not many graduate students are granted. I hope to continue learning from both of them as I continue in my career.
Contents

Abstract ................................................................. ii
Acknowledgments ......................................................... iii
List of Tables ............................................................ vii
List of Figures ............................................................. viii

1 Introduction and Preliminary Material 1

1.1 Quantum Mechanics ................................................. 2
  1.1.1 The postulates of quantum mechanics ......................... 4
  1.1.2 The power of quantum computing .............................. 7

1.2 Quantum Computing ............................................... 9
  1.2.1 The qubit ..................................................... 10

1.3 Quantum Error Correction ........................................ 11
  1.3.1 Error Correcting Codes ...................................... 12
  1.3.2 Abstraction of error correction to subspaces ................. 21
  1.3.3 Error correction .............................................. 25
  1.3.4 Bounds ....................................................... 26
  1.3.5 The stabilizer formalism ..................................... 28

1.4 Designing circuits for encoding .................................. 32
  1.4.1 The 5 step procedure ....................................... 33
  1.4.2 Examples ...................................................... 37

2 Hybrid Codes .......................................................... 42
2.1 Trivial example and simple example ............................................... 42
2.2 A coding theory perspective ....................................................... 45
   2.2.1 Trivial constructions ......................................................... 46
   2.2.2 Hybrid codes with better parameters ................................. 46
2.3 Operator quantum error correction ........................................... 47
   2.3.1 The standard model .......................................................... 47
   2.3.2 Noiseless subsystems ......................................................... 48
   2.3.3 Generalization of noiseless subsystems ............................... 51
   2.3.4 The unified approach ......................................................... 54
2.4 Operator algebra quantum error correction ............................... 55
   2.4.1 The Schrodinger and Heisenberg pictures of noiseless channels 55
   2.4.2 Conservation of algebras of observables .............................. 57
2.5 Unification of the two perspectives for hybrid codes ................... 57
2.6 Hamming Bound on Hybrid Codes ............................................. 59

3 NMR Quantum Information Processing ........................................... 62
3.1 Introduction to NMR QIP ......................................................... 62
   3.1.1 The Qubit ................................................................. 63
   3.1.2 Quantum Gates ......................................................... 66
   3.1.3 Measurement ............................................................ 71
   3.1.4 Initial State ............................................................. 76
3.2 Simulating a NMR QIP in Matlab ............................................. 79
   3.2.1 Read out ................................................................. 80
   3.2.2 Applying gates .......................................................... 81
   3.2.3 Pseudo-pure state preparation ....................................... 83
   3.2.4 Pulse for applying the CNOT ......................................... 87
   3.2.5 Simplifying the pulse sequence .................................... 88
   3.2.6 Simulating with experimental time .................................. 92
3.2.7 Simulating with T2 decoherence ........................................ 99
3.3 Preparing for implementation in an NMR lab .......................... 102
  3.3.1 Decoding circuit .......................................................... 102

4 Conclusions and Future Work .................................................. 115

Bibliography ................................................................................. 118
## List of Tables

1.1 Table of observables and the corresponding quantum operator.......................... 6  
1.2 Generators and Pauli matrices for the [[5,1 ........................................ 33  
1.3 The input for the circuit for the control operator........................................... 36  
1.4 Generators for the [[7,1 ................................................................. 37  
1.5 The generators and Pauli matrices for the 6,1 code......................................... 39  

3.1 Outline of the decoding results for the errors on qubit 4 while transmitting classical  
    bit 0. ........................................................................................................ 104  
3.2 Outline of the decoding results for the errors on qubit 4 while transmitting classical  
    bit 1. ........................................................................................................ 105  
3.3 Outline of the decoding results for the errors on qubit 3 while transmitting classical  
    bit 0. ........................................................................................................ 106  
3.4 Outline of the decoding results for the errors on qubit 3 while transmitting classical  
    bit 1. ........................................................................................................ 107  
3.5 Outline of the decoding results for the errors on qubit 2 while transmitting classical  
    bit 0. ........................................................................................................ 108  
3.6 Outline of the decoding results for the errors on qubit 2 while transmitting classical  
    bit 1. ........................................................................................................ 109  
3.7 Outline of the decoding results for the errors on qubit 1 while transmitting classical  
    bit 0. ........................................................................................................ 110  
3.8 Outline of the decoding results for the errors on qubit 1 while transmitting classical  
    bit 1. ........................................................................................................ 111
### List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Visual of classical physics, quantum physics, relativity and quantum field theory</td>
<td>3</td>
</tr>
<tr>
<td>1.2</td>
<td>A visual comparison of a bit, a probabilistic bit and a quantum bit.</td>
<td>11</td>
</tr>
<tr>
<td>1.3</td>
<td>The bit flip encoding circuit.</td>
<td>15</td>
</tr>
<tr>
<td>1.4</td>
<td>A circuit depicting how ancillary qubits are attached to a three qubit system.</td>
<td>15</td>
</tr>
<tr>
<td>1.5</td>
<td>The phase flip encoding circuit.</td>
<td>17</td>
</tr>
<tr>
<td>1.6</td>
<td>A schematic of how the Shor code is a concatenation of the phase flip and bit flip codes.</td>
<td>18</td>
</tr>
<tr>
<td>1.7</td>
<td>The circuit for encoding the Shor code.</td>
<td>19</td>
</tr>
<tr>
<td>1.8</td>
<td>A schematic of how errors are determined in the Shor code.</td>
<td>19</td>
</tr>
<tr>
<td>1.9</td>
<td>Visualization of the two-dimensional Hilbert space a single qubit exists within.</td>
<td>21</td>
</tr>
<tr>
<td>1.10</td>
<td>Visualization of the space two qubit exists in.</td>
<td>22</td>
</tr>
<tr>
<td>1.11</td>
<td>Visualization of the space a three qubit exists in.</td>
<td>22</td>
</tr>
<tr>
<td>1.12</td>
<td>Using the subspace schematic to depict the action of an $X_2$ error.</td>
<td>22</td>
</tr>
<tr>
<td>1.13</td>
<td>Using the subspace schematic to depict how errors are corrected without determining exactly what error has occurred.</td>
<td>23</td>
</tr>
<tr>
<td>1.14</td>
<td>A depiction of an operator that splits the quantum space into two orthogonal subspaces.</td>
<td>24</td>
</tr>
<tr>
<td>1.15</td>
<td>The generators, error set and encoding circuit for the three qubit bit flip code.</td>
<td>30</td>
</tr>
<tr>
<td>1.16</td>
<td>The generators, error set and encoding circuit for the Shor code.</td>
<td>30</td>
</tr>
<tr>
<td>1.17</td>
<td>The generators, error set and encoding circuit for the five qubit code.</td>
<td>31</td>
</tr>
<tr>
<td>1.18</td>
<td>The generators, error set and encoding circuit for the seven qubit Steane code.</td>
<td>32</td>
</tr>
<tr>
<td>1.19</td>
<td>The standard form structure of the check matrix.</td>
<td>34</td>
</tr>
</tbody>
</table>
3.12 A labeling that demonstrates how peak difference is related to coupling constants.

3.13 Schematic representation of the pseudo-pure preparation for 3 and 4 qubits.

3.14 An example of a measurement readout in Matlab.

3.15 Schematic representation of the pseudo-pure state preparation for 2 qubits.

3.16 The x and y magnetizations of each qubit in crotonic acid when rotated.

3.17 A magnification of individual peaks of crotonic acid.

3.18 3D visualization of the application of the CNOT gate in NMR.

3.19 The control not as pulse sequence.

3.20 Side by side visual of the read out signal with the Bloch sphere representation of a qubit undergoing the CNOT procedure in NMR.

3.21 The resulting encoding circuit by using the method in section 1.4.

3.22 The conversion of the original encoding circuit to one that can be implemented with crotonic acid.

3.23 Resulting pulse sequence from the encoding circuit, before simplification.

3.24 The hybrid pulse sequence once redundant gates are removed.

3.25 The hybrid pulse sequence once Z operators are moved past coupling gates.

3.26 Fully simplified pulse sequence.

3.27 Graph of the control variations of the grape pulse.

3.28 Testing the robustness of the grape pulse.

3.29 Possible pi pulse sequences for 3 qubits.

3.30 Possible pi pulse sequences for 4 qubits.

3.31 Read outs of the encoded qubits along with the fidelity over the GRAPE procedure.

3.32 Example of pulses that can be applied simultaneously.

3.33 An improvement on fidelity compared to figure 3.22, when pulses are grouped together.

3.34 The 2-tuple matrix used for simulating the T2 decoherence.

3.35 The precursor matrix for the decoherence matrix for the four qubit case.

3.36 The measured signal before and after T2 decoherence is applied.
Chapter 1

Introduction and Preliminary Material

Thesis structure

This thesis is organized into four chapters. The first chapter begins with a brief introduction into the axioms of quantum mechanics and the relevant implications of these axioms to quantum computing. From here the development of quantum computing is placed in a historical narrative of the evolution of computing over the past century. This narrative serves to illustrate how classical computers also went through a prolonged period of incremental progress before their sudden acceleration into widespread use. At this point, the most fundamental difference between a classical and quantum computer is discussed; the qubit. In section 1.3 we then introduce field of quantum computing which is relevant to this thesis; quantum error correction. This section places a greater emphasis on the theory of quantum error correction that is necessary to design and implement encoding circuits. Finally, Chapter 1 concludes with a five step procedure for designing encoding circuits when provided with logical codewords.

Having built a foundation in quantum error correction in Chapter 1 we are then able to introduce hybrid codes in Chapter 2. The chapter opens with a brief example of a simple construction and a trivial construction of a hybrid code to give the reader a conceptual understanding of how hybrid codes work. The next portion of the chapter outlines the two characterizations and constructions of hybrid codes that we consider in this thesis; the coding theory perspective and the OAQEC perspective. In the sections regarding the coding theory perspective we look at the notation used
in coding theory, what it means for a code to be trivial and the requirements to have hybrid codes with better parameters. In the following sections, on the OAQEC perspective, we look at the more fundamental model of operator quantum error correction (OQEC), generalize this to OAQEC and then discuss the hybrid code conditions as outlined by OAQEC. After reviewing both perspective we show that the coding theory perspective is a special case of the OAQEC model. We close this chapter with a generalization of the quantum hamming bound to the hybrid case which shows that for hybrid codes to have better parameters than quantum codes they must be degenerate.

In Chapter 3 we now move to simulating the implementation of hybrid codes in a nuclear magnetic resonance (NMR) quantum information processor (QIP) using Matlab. The chapter begins with a background on how NMR can be used to implement a QIP. This is followed by an explanation of how each step of a quantum computation can be simulated using Matlab. This explanation covers how to apply pauli gates, hadamards, CNOT gates, how to prepare a pseudo-pure state, measure resulting states, how to simulate in real time and apply T2-decoherence. The chapter concludes with the conversion of the circuit that we designed to implement a hybrid code into an NMR pulse sequence along with the necessary decoding pulse sequences.

The thesis concludes in Chapter 4 with a review of the results of the thesis and an outline of future work.

1.1 Quantum Mechanics

Over years of living in this world we’ve come to develop an intuition for how the world works. With this intuition we are able to play out scenarios in our mind that we have never experienced ourselves. For example, I can say with certainty that getting hit by a truck will hurt more than getting hit by a child on a tricycle \( F = ma \) even though I have never been hit by either one. I’ve also never been in outer space but if I ever had the good fortune to do so and I happened to throw a ball in some direction I know it won’t abruptly stop moving on its own (the law of inertia). Furthermore, I know that if I were to hold a firearm in front of my face and shoot it then
I would likely knock out a few of my own teeth, but I’ve also never held a firearm (Newton’s 3rd law of motion). These simple examples may seem obvious, but it is useful to think of why they’re obvious. They’re obvious to us because we interact with these laws and live within them every day. These same laws would likely not be so obvious to some alien living in another world which had its own laws of nature. In a sense, we are those aliens, living in a classical world trying to study and understand the quantum world with atoms live in.

The laws of physics start to change when we consider things that are incredibly small (less than a nanometer) incredibly fast (comparable to $3 \times 10^8 m/s$) or both. Quantum physics is the field of study for things which are incredibly small.

A small semantical note to address at the outset is that quantum physics and quantum mechanics are not synonymous. If you were to consider classical physics and its immensity, there are many subfields of classical physics. These classical field include motion, optics, thermal dynamics, electromagnetism and many others. The quantum word is also immense and also includes many topics such as quantization of energy, the wave-particle duality or the uncertainty principal. Quantum mechanics is one such sub-field of the greater field of quantum physics.

This section is not intended to provide a thorough review of quantum mechanics. Instead it aims to introduce a sufficient enough foundation to allow one without a background in physics to under-
stand the basic principals of quantum computing. For a more thorough overview see [15].

1.1.1 The postulates of quantum mechanics

The theory of quantum mechanics is built upon certain axioms. This section will briefly overview each of them.

1. The wave function or state vector

In infinite dimensional quantum mechanics the states of a quantum system are given by the wave function, \( \Psi(x, t) \). Unlike a classical system, all the information that can possibly exist about a quantum system is within a single function. It represents the probability that a particle it at a given point in space at a given time. This probability is given by the wave function multiplied by its complex conjugate \( p(x) = \bar{\Psi} \Psi \). Since the probability the particle exists somewhere is one hundred percent, or one, we impose a normalization condition of \( 1 = \int_{-\infty}^{\infty} \bar{\Psi} \Psi \).

In quantum computing we are concerned with finite dimensional quantum mechanics, where the state vector is used instead of the state function. The state vector consists of the different probability amplitudes of the different states the quantum system can be in. \( \Psi = [\Psi_1, \Psi_2, ..., \Psi_n] \). The probability amplitude squared \( |\Psi_i|^2 \) gives the probability of seeing outcome \( i \), thus, this vector must also be normalized \( |\Psi|^2 = 1 \).

Operator formalism

A quantum state can also be represented as a density matrix. A density matrix can describe a quantum system that is in a mixed state, a statistical ensemble of several quantum states. The single state vector description given in the previous paragraph represents a pure state, one that is exactly one quantum state. The density matrix is the quantum analogue to the probability distribution in classical statistical mechanics.
A pure quantum state is represented as a density matrix by taking the outer product of its vector representation with itself. For example the pure state $|0\rangle$ can be represented as a density matrix:

$$\rho = |0\rangle \langle 0| = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

This can be compared to a mixed state:

$$\rho = \frac{1}{2} |0\rangle \langle 0| + \frac{1}{2} |1\rangle \langle 1| = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix}$$

In this mixed state example there is an equal probability of the state being $|0\rangle$ and $|1\rangle$. In a density matrix the coefficients infront of each pure state represent the probability of the mixed state being in that state, thus these coefficients are necessarily real, between 0 and 1 and sum to 1. Density matrices are necessarily self-adjoint:

$$(\rho)^\dagger = \sum \sigma_i |\alpha_i\rangle \langle \alpha_i| = \sum (\sigma_i)^\dagger (|\alpha_i\rangle \langle \alpha_i|)^\dagger = \sum \sigma_i |\alpha_i\rangle \langle \alpha_i|.$$ 

$\sigma$ is real and positive so $\sigma^\dagger = \sigma$. Density matrices are also positive semi-definite and trace one.

2. Observables

For every observable in classical mechanics there exists a linear hermitian operator $A$ which when acting on the quantum state $\Psi$ returns the value for that observable. The state $\Psi$ acts as the eigenfunction of $A$ and the measure of the observable is the eigenvalue $\lambda$. Thus, the only possible values an observable can take are the eigenvalues of the operator $A$.

$$A |\Psi\rangle = \lambda |\Psi\rangle$$

Table 1.1 contains a few examples of such operators.
<table>
<thead>
<tr>
<th>Observable</th>
<th>Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Momentum</td>
<td>$\frac{\hbar}{i} \frac{\partial}{\partial t}$</td>
</tr>
<tr>
<td>Angular Momentum</td>
<td>$-i\hbar \frac{\partial}{\partial \phi}$</td>
</tr>
<tr>
<td>Energy (time dependent)</td>
<td>$i\hbar \frac{\partial}{\partial t}$</td>
</tr>
<tr>
<td>Energy (time independent)</td>
<td>$-\frac{\hbar^2}{2m} \frac{\partial^2}{\partial x^2} + V(x)$</td>
</tr>
<tr>
<td>Kinetic Energy</td>
<td>$-\frac{\hbar^2}{2m} \frac{\partial^2}{\partial x^2}$</td>
</tr>
</tbody>
</table>

Table 1.1: A few examples of quantum operators and the physical observable they correspond to.

3. The expected value

When taking two separate measurements there is no guarantee of measuring the same value twice, in fact it’s often quite unlikely to do so. Our quantum state has different probabilities of being in different states, and each of these different states will give a different value when an operator acts on it. The expectation value for an operator $A$ is given by.

Wave function $\int_{-\infty}^{\infty} \bar{\Psi} A \Psi dx$

State vector $\langle \Psi | A | \Psi \rangle$

4. The time evolution

The last postulate is also the most brief, it states that every quantum system changes in time according to the same equation, Schrödinger’s equation; $\hat{H} \Psi = i\hbar \frac{\partial \Psi}{\partial t}$.

Summary

We’ll conclude this section with a conceptual summary of the above concepts. First, unlike classical systems which are defined by many different values, all the quantum information of a quantum system can be given by a single vector. To extract the information we desire from these vectors
we apply operators on them that corresponds to the observable we desire. The only possible value
that can be measured from these observables are the eigenvalues of these operators. The overall
expected value for such a measurement is the weighted average of the measurements on each quan-
tum state our quantum system can be in. And finally, every quantum system evolves according to
the same equation.

1.1.2 The power of quantum computing

By simply reading these postulates the inherent power of quantum computing is likely not evident.
The following consequences of the above postulates lends quantum computing some of its power.

Entanglement

Entanglement is a remarkable consequence of quantum mechanics. Entanglement occurs when
two particles interact in such a way that they can no longer be described independently of each
other, even if the two particles are separated. For example imagine two photons that become
entangled such that one must be polarized horizontally and the other vertically. Theoretically,
these two photons could then be moved to opposite ends of the universe. If one were to measure
the polarization of one photon they would immediately knowing the polarization of the other. This
apparent instant communication seems to occur faster than the speed of light, something that
deeply troubled Einstein, prompting him to dub this effect “spooky action at a distance”.

To mathematically define entanglement requires the introduction of the tensor product. Say we
have two quantum systems whose state-spaces are finite dimensional, $\mathcal{H}_A = \mathbb{C}^n$ and $\mathcal{H}_B = \mathbb{C}^m$.
The joint system exists on the tensor product of the two spaces, $\mathcal{H}_A \otimes \mathcal{H}_B$. Let $\{|a_i\rangle\}_{i=1}^n$ be a basis
for $\mathcal{H}_A$ and $\{ |b_j\rangle \}_{j=1}^n$ be a basis for $\mathcal{H}_B$ then we define the basis of $\mathcal{H}_A \otimes \mathcal{H}_B$ as $\{ |a_i\rangle \otimes |b_j\rangle \}_{i=1,j=1}^{n,m}$.

$$|a\rangle \otimes |b\rangle = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} \otimes \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{bmatrix} = \begin{bmatrix} \alpha_1 \beta_1 \\ \alpha_1 \beta_2 \\ \vdots \\ \alpha_1 \beta_m \\ \alpha_2 \beta_1 \\ \vdots \\ \alpha_2 \beta_m \\ \vdots \\ \alpha_n \beta_1 \\ \vdots \\ \alpha_n \beta_m \end{bmatrix} = \begin{bmatrix} a_1 |b\rangle \\ a_2 |b\rangle \\ \vdots \\ a_n |b\rangle \end{bmatrix}$$

For example, the two qubit system of $|0\rangle$ and $|1\rangle$ is written as:

$$|0\rangle \otimes |1\rangle \equiv |01\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \times 0 \\ 1 \times 1 \\ 0 \times 0 \\ 0 \times 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

We can now use the tensor product to give a mathematical definition of entanglement. Not every state in a joint system $\mathcal{H}_A \otimes \mathcal{H}_B$ can be written as a pure tensor product $|\Psi_A\rangle \otimes |\Psi_B\rangle$. When a state can not be written in such a fashion we know that the state is entangled. Consider for example the vector

$$|\Psi_{00}\rangle = \frac{1}{\sqrt{2}} (|00\rangle + |11\rangle) = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Assume there exists $|a\rangle = a_1 |0\rangle + a_2 |1\rangle$ and $|b\rangle = b_1 |0\rangle + b_2 |1\rangle$ such that $|a\rangle \otimes |b\rangle = |\Psi_{00}\rangle$. This would mean that $a_1 b_1 |00\rangle + a_1 b_2 |01\rangle + a_2 b_1 |10\rangle + a_2 b_2 |11\rangle = \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle$. This would give us the system of equations $a_1 b_1 = \frac{1}{\sqrt{2}}$, $a_1 b_2 = 0$, $a_2 b_1 = 0$, $a_2 b_2 = \frac{1}{\sqrt{2}}$ which is inconsistent. For example, multiplying the first and last equations together yields $a_1 a_2 b_1 b_2 = \frac{1}{2}$ yet multiplying the second and third equations together yields $a_1 a_2 b_1 b_2 = 0$. Therefore, the state $|\Psi_{00}\rangle$ cannot be written as a tensor product of single qubit states and is thus an entangled state.

Quantum superpositions
Superposition itself is not an exclusively quantum effect. Most waves are superpositions of simpler waves and the electric field produced from multiple atoms is the superposition of the individual atoms fields. Nonetheless, any quantum states can be added together to form another valid quantum state and the converse is also true. While a classical object exists in one position, a quantum object can exist in a superposition of two different positions. Effectively this object exists in both positions simultaneously, until its position is measured. Unlike classical superposition the amplitudes of quantum superpositions can be complex numbers.

Projective measurements

Projective measurements don’t necessarily give power to quantum computing but create amazing possibilities for quantum algorithms. The very act of observing a quantum system will change it. This was most famously demonstrated in double slit experiment [9] where the act of observing electrons changed whether they behaved as waves or particles. Quantum algorithms can take advantage of this by measuring certain parts of the quantum system which in turn cause other parts of the quantum system to collapse to a certain value.

1.2 Quantum Computing

The brevity of the time between the laying of the foundation of computing theory to the point where computers have pervaded every element of society is astonishing [6]. This short period began in 1936 with Alan Turings proposition of the turing machine. This device was proclaimed as a “universal computing machine” that would be able to perform any conceivable mathematical computation that can be represented as an algorithm. The first general purpose computer would be developed 8 years later[6], consisting of vacuum tubes and taking the space of a large room. In the next 25 years [6] the computing world would be introduced to primitive versions of presently essentials computing tools. The first modem was 75 bits per second and introduced in 1949[6]. This was followed by the first hard drive in 1957[6] which stored 5mB and was larger than a truck.
In the same year the first programming language fortran was born and in 1968 and 1970 RAM [6] and the microprocessor were developed, respectively. This would lead into half a century of rapidly accelerating innovation highlighted by the launching of the world wide web in 1991[6].

Intel co-founder Gordon Moore made an interesting observation in the midst of this computation boom. Moore noticed that the number of transistors per square inch of a circuit nearly doubled every year. Moore’s law, as it was dubbed, predicted that computers will exponentially get smaller and more powerful [21], a law which will soon collide with the laws of quantum physics. In the next decade or two this shrinking of transistors will reach the scale of atoms, at which point the laws of quantum mechanics will come into play.

From one perspective this new interplay of superposition and entanglement can be seen as a stumbling block for computing, from another perspective they can be seen as stepping stone to reach new heights of computing. This later perspective motivates the study of harnessing the power of quantum mechanics to realize the potential of quantum computers.

1.2.1 The qubit

A classical computer stores and processes information using binary digits, “bits”. A bit can be represented by any physical object that can exist in one of two states. A coin for example can be in the state heads or the state tails, so a coin could be used to represent a bit. Most modern computers use current pulses in wires or flip-flop circuits as their bit.

The difference between a classical computer and a quantum computer comes from the difference in their bit. Quantum computers use quantum bits, “qubits”. A qubit is a bit which is physically represented by something that is small enough to obey the laws of quantum mechanics. A photon for example can be polarized horizontally or vertically, a nucleus can have an upward or downward spin, these can be used as quantum bits. Qubits allows quantum computers to capitalize on the aforementioned quantum phenomena like entanglement and superposition. So unlike the classical bit which must be in the states 0 or 1, the qubit can be in any superposition of the states 0 and 1.

Figure 1.2 shows the amount of information which can be stored by different types of bits. The
first is the classical bit, which has two possible states 0 or 1. The next bit depicted is a probabilistic bit, or “pbit”. The pbit has a $p$ chance of being in the state 0 and a $1 - p$ chance of being the state 1. The qubit exists in a superposition of 0 and 1, so a general quantum state can be described as $|\Psi\rangle = \alpha |0\rangle + \beta |1\rangle$. The probability amplitudes of the qubit have the normalization restriction that $|\alpha|^2 + |\beta|^2 = 1$. Clearly then a qubit can take on infinitely more values than the classical bit. Ultimately this is the difference between classical computers and quantum computers, the remarkable power of one qubit.

### 1.3 Quantum Error Correction

A key challenge with building reliable quantum computers is the fragility of qubits. What would be considered negligible noise in a classical computer can completely destroy a quantum state. This challenge of sensitivity can be tackled in different ways. One approach is to provide better physical conditions to protect the qubits information from its environment. Physicists and engineers can be clever in their design of quantum computers, or design more powerful shielding. Another approach is to encode the quantum information in such a way that it can resist different types of errors. Both of these will be necessary if we are to ever have functioning quantum computers. This section will
look at the nature of the later approach, called quantum error correction. Quantum error correction was independently discovered by [23] and by [24].

1.3.1 Error Correcting Codes

Classical error correction

Essentially how error correction works is to encode our quantum state so as to create redundancies, this way if an errors damages a portion of the code the original message can still be decoded. For example, imagine if you are standing in a noisy room and you want to send the message “0” to your friend across the room. You might yell “0” once but not be sure if he heard you. So naturally you repeat yourself, “0, 0, 0”. If your friend knows you will be sending him either the message “0” or “1” and across the noisy room he hears “0, 1, 0” then he can safely assume you were trying to send the message “0”. This is an elementary example of an error-correcting code, the repetition code. In the repetition code the message “0” or “1” is encoded as follows.

\[
\begin{align*}
0 & \rightarrow 000 \\
1 & \rightarrow 111
\end{align*}
\]

These encoded messages are called logical zero and logical one. The encoded message then moves through some channel where errors can occur that flip the value of 0 to 1 or vice-versa. After going through the channel one of eight possible messages can be received. A majority voting system is then used to decode what message was originally meant to be sent.

Decoding

\[
\begin{align*}
000, 001, 010, 100 & \rightarrow 0 \\
111, 110, 101, 011 & \rightarrow 1
\end{align*}
\]

It is intuitive that this process will decrease the risk of our message being corrupted by an error from the channel. If the probability of a single error occurring is \(p\), then before the encoding the
probability the receiver receives the wrong message is $p$. With the above encoding our receiver will get the wrong message only if two of the bits undergo an error or all three bits undergo an error. The probability one of these two outcomes occurring is $3p^2(1-p) + p^3$. This new probability of the receiver receiving the wrong message is less than $p$ provided than $p < 1/2$.

Quantum application

We would now like to apply the above kind of error-correction method for qubits. Unfortunately, the exact same procedure can’t be followed due to three quantum complications. Each of these issues is provided along with how it can be overcome, this will be expanded upon later.

The no-cloning theorem: Their doesn’t exist a quantum operator that can clone any arbitrary quantum state [26]. In other words, there is no such operators $U$ such that $U\ket{\psi}\ket{00} = \ket{\psi\psi\psi}$. Fortunately there does exist a quantum operator to clone basis states. So $\ket{\psi}$ can not be encoded as $\ket{\psi\psi\psi}$ but $\ket{0}$ can be encoded as $\ket{000}$ and $\ket{1}$ as $\ket{111}$. So the state $\ket{\psi} = \alpha \ket{0} + \beta \ket{1}$ can be transformed to $\alpha \ket{000} + \beta \ket{111}$.

Errors are continuous: Classical bits only have two possible states, thus they only have one possible form of error that can occur, a 0 can become a 1 or vice-versa. Quantum bits exists as $\ket{\psi} = \alpha \ket{0} + \beta \ket{1}$ which opens up a host of different errors that can occur. These include the error of a $\ket{0}$ flipping to a $\ket{1}$, the $\alpha$ or $\beta$ changing signs or magnitude and any combination of those three. Thus creating an infinite number of possible errors. Fortunately, this continuous set of errors can all be broken into a combination of a few discrete errors. If those discrete errors can be corrected then so to can any possible error.

Measuring a quantum system destroys information: We need to observe a state before we can decode it, but a quantum state changes when it is observed. This poses two kinds of challenges. One, is that if our system is in a superposition of different states then measuring the system will force it to collapse into one of those states, thus changing the state. The other challenge is that how we measure it to also influences the state, so even if the quantum state is not in a superposition we may measure in an incorrect basis and change the state. For example if we have some mystery state
which is $|0\rangle$ and we measure to see how much of it is in the $|+\rangle |−\rangle$ basis then we'll have equal probability of measuring the state $|+\rangle$ or $|−\rangle$ and no probability of measuring $|0\rangle$. This can be overcome by different techniques, one of which is to essentially only measure enough of the system to determine if an error has occurred and where but with out learning what state the quantum system is.

**Circuit notation**

Before advancing further we will pause and explain the notation used in quantum circuits. A quantum circuit is a model of a quantum computation which represents the computations as a series of quantum gates on an n-bit register of quantum bits. The quantum circuit has n horizontal lines, each representing a qubit. In the circuit quantum gates are represented with different shapes, most commonly a labeled rectangle, which overlaps the n bits which it acts on. Quantum circuits are read chronologically from left to right. Figure 1.3, for example, depicts a 3 qubit quantum circuit. The first qubit begins in the state $|\Psi\rangle$ and the other two begin in the states $|0\rangle$. The first quantum gate depicted overlaps the first and second qubit. This gate represents the control not gate, the closed circle is the control qubit and the open circle with a cross is the qubit being controlled. The circuit continues with a second control not gate which operates on the first and third qubit.

**Quantum bit flip code**

The first quantum error correcting code is the quantum analog to the classical example we looked at. In this example the qubit moves through a channel which can cause one of the following errors, $I, X_1, X_2, X_3$. $I$ is the identity error which is equivalent to no error occurring and $X_i$ represent the bit flip error. This error is given by $X|0\rangle = |1\rangle$ and the index $i$ represents which qubit the error is acting on. The code we use to guard against these errors is the *bit flip code*.

\[
|0\rangle_L \equiv |000\rangle
\]

\[
^{1}|\pm\rangle \equiv \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), |\pm\rangle \equiv \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)
\]
Figure 1.3: The circuit for encoding the quantum bit flip code.

\[ |\psi\rangle \equiv |111\rangle \]

Figure 1.4: This figure depicts three qubits in a circuit which are attached to two ancillary qubits. When attached in such a fashion to a three qubit system these two ancillary qubits can be measured to determine which of the three qubits are the same and which are not.

\[ |1\rangle_L \equiv |111\rangle \]

Fig 1.3 depicts the circuit that encodes the qubit \( |\psi\rangle = \alpha |0\rangle + \beta |1\rangle \) as its logical representation \( |\psi\rangle_L = \alpha |0\rangle_L + \beta |1\rangle_L = \alpha |000\rangle + \beta |111\rangle \). These three qubits then will move through some error channel which can introduce a bit flip on one of the qubits or no bit flip at all. At this point the challenge of measuring what error has occurred is addressed. This is done by attaching the following two ancillary qubits to \( |\psi\rangle_L \) after it goes through the error channel. If the first two qubits are both 0 or both 1, they will apply a bit flip to the second ancillary qubit an even number of times, thus not changing it. If one is a 0 and the other is a 1 then a flip will be applied once and change the qubit. The same is true for the effect of the second and third qubits in \( |\psi\rangle_L \) with the first ancillary qubit. In this way each ancillary qubit is a measure of if two of the qubits are the same or different. If both ancillary qubits are measured to be 0’s then all three qubits are matching and no error has occurred. If the first, second or both qubits are measured to be 1 then an error has occurred on one qubit. Which combination corresponds to what error is the error syndromes and is given below. (q1, q2 and q3 represent the state of the first, second and third qubit respectively.)
• $|00\rangle$: $q_1 = q_2, q_2 = q_3$, and thus no error has occurred

• $|10\rangle$: $q_1 \neq q_2, q_2 = q_3$, an thus an error has occurred on the first qubit

• $|01\rangle$: $q_1 = q_2, q_2 \neq q_3$, an thus an error has occurred on the third qubit

• $|11\rangle$: $q_1 \neq q_2, q_2 \neq q_3$, an thus an error has occurred on the second

This process of error-detection can also be accomplished by using combinations of the $Z^2$ operator. The action of this operator on the basis states is $Z|0\rangle = |0\rangle$ and $Z|1\rangle = -|1\rangle$. Consider the effect of the $Z_1Z_2$ operator on a two qubit system $|xy\rangle$. If $x = y$ then $Z_1Z_2|00\rangle = |00\rangle$ and $Z_1Z_2|11\rangle = (-1)(-1)|11\rangle = |11\rangle$. On the other hand if $x \neq y$ then $|xy\rangle = |01\rangle$ or $|10\rangle$. For either case, one of the Z operators will act on a $|1\rangle$ and the other will act one a $|0\rangle$ pulling out a constant of $-1$ overall. In other words, if the two qubits are the same the $Z_1Z_2$ operator provides an eigenvalue of $+1$ and if the two qubits are different the operators provide an eigenvalue of $-1$. Identically, $Z_2Z_3$ will have the same effect on the second and third qubits. Thus, providing us with another way to determine the error which has occurred.

• Measure $+1$, $+1$. $q_1 = q_2, q_2 = q_3$, and thus no error has occurred

• Measure $-1$, $+1$. $q_1 \neq q_2, q_2 = q_3$, an thus an error has occurred on the first qubit

• Measure $+1$, $-1$. $q_1 = q_2, q_2 \neq q_3$, an thus an error has occurred on the third qubit

• Measure $-1$, $-1$. $q_1 \neq q_2, q_2 \neq q_3$, an thus an error has occurred on the second

Quantum phase flip code

The previous code allowed us to detect and correct bit flip errors, the phase flip code can be used to detect and correct phase errors. The three qubit phase flip channel is a quantum channel that implements one of the following errors, $I, Z_1, Z_2, Z_3$. The code we use to guard against these errors is the phase flip code.

---

2The Pauli matrices are the following $2 \times 2$ matrices: $X = [0, 1; 1, 0], Y = \times[0, -i; i, 0], Z = [1, 0; 0, -1]$
The phase flip code is essentially identical to the bit flip code but acts on a different basis. The $Z$ operators acts on the basis states $|+\rangle \equiv \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|-\rangle \equiv \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ exactly as the $X$ operators acts on the standard basis states, $Z|+\rangle = |-\rangle$ and $Z|-\rangle = |+\rangle$. The Hadamard operator, $H$, changes between these two basis. $H|0\rangle = |+\rangle$ and $H|+\rangle = |0\rangle$. Therefore the circuit for the phase flip code is identical to bit flip code if Hadamard’s are first applied to each qubit as depicted in figure 1.5. The same procedure used for the bit flip code can be used to measure which error has occurred. The only requirement is an operator that acts how $Z_i Z_j$ ($i,j$ are integers, representing which qubit in the system the $Z$ operator is acting on) does on the basis $|0\rangle$ and $|1\rangle$. It is easy to check that $X_1 X_2$ and $X_2 X_3$ do so. The last two channels discussed are unitarily equivalent, meaning their exists a unitary gate which if applied before and after the one channel produces the other.

**The Shor code**

With possession of a code that protects against bit flip errors and one protecting against phase flip errors we can now produce a code that can guard against either, the Shor code [23]. Simpler codes have since been created that are as powerful which makes its significance more historical than practical. Nonetheless, the Shor code is a good entry point into more complex quantum error-correcting codes. This Shor code uses a method of nested encoding, called *concatenation*, where the qubit is first encoded use one encoding then each qubit is encoded using the other. For the Shor
Figure 1.6: A schematic of how the Shor code is a concatenation of the phase flip and bit flip codes.

code the order of concatenation is the phase flip code followed by the bit flip code. Both of these encodings encode 1 qubit to 3 qubits, which means the Shor code ultimately encodes one qubit as nine. This is schematically shown in figure 1.6. The actual circuit that implements this encoding is also provided in figure 1.7. This circuit provides the following logical codewords:

\[ |0\rangle_L \equiv \frac{(|000\rangle+|111\rangle)(|000\rangle+|111\rangle)(|000\rangle+|111\rangle)}{2\sqrt{2}} \]
\[ |1\rangle_L \equiv \frac{(|000\rangle-|111\rangle)(|000\rangle-|111\rangle)(|000\rangle-|111\rangle)}{2\sqrt{2}} \]

After the encoding the qubits enter a channel which may produce a bit and/or phase flip error on a single qubit. A similar process as the bit flip and phase flip codes are used to determine which error occurred. For each of the previous codes we used two operators, one which differentiated the first and second qubit and which differentiated the second and third qubit. Figure 1.8 shows the concatenation depiction of the Shor code with the appropriate operators to distinguish the errors next to each step of the concatenation. One set of the $X_iX_j$ and $X_jX_k$ operators is required to check for phase flip errors and three sets of the $Z_iZ_j$ and $Z_jZ_k$ are required to check for bit flip errors. This is just a consequence of the order of the concatenation. The $X_iX_j$ operators are not comparing individual qubits, since those qubits they are comparing are further encoded, so instead they comparing the entire block of qubits. For $X_iX_j$ and $X_jX_k$, by $i$ is meant qubits 1 to
Figure 1.7: The circuit for encoding the Shor code.

Figure 1.8: A schematic of how errors are determined in the Shor code.
3, \( j \) is qubits 4 to 6 and \( k \) is qubits 7 to 9. This gives the two operators \( X_1X_2X_3X_4X_5X_6 \) and \( X_4X_5X_6X_7X_8X_9 \). \( Z_iZ_j \) and \( Z_jZ_k \) is more simple since there is no further encoding and the operators used to detect the bit flip errors are \( Z_1Z_2, Z_2Z_3, Z_4Z_5Z_6, Z_7Z_8Z_9 \), \( X_1X_2X_3X_4X_5X_6 \) and \( X_4X_5X_6X_7X_8X_9 \). These 8 operators are thus sufficient to detect if errors have occurred.

**Example of Shor Code**

The Shor code can be clarified with an example. If a bit flip were to occur on the 5th qubit then the \( Z_4Z_5 \) and \( Z_5Z_6 \) operator would each return the eigenvalue -1 and all other \( Z_iZ_j \) would return +1. Meaning the 4th and 5th qubit and the 5th and 6th qubit are different. Thus pointing out the location of the error. On the other hand, if a phase flip were to occur on say the second qubit, the first block of qubits would be in the state \(|000⟩+|111⟩⟩\). The \( X_1X_2X_3X_4X_5X_6 \) and \( X_4X_5X_6X_7X_8X_9 \) operators would return the eigenvalues -1 and +1. Meaning the 1st and 2nd block’s of qubits have different phases while the 2nd and 3rd blocks of qubits have the same phase. This points that the error occurred on the first block of qubits. What’s interesting is that which of the three in the block had the error is not important, since correcting any one will correct the error in the block. Since these two sets of operators act independently from one another the Shor code can correct a one qubit error that’s both a phase flip and bit flip error.

**Correcting arbitrary errors**

Of the three potential problems that were outline in the beginning of the section two of the solutions have been discussed, the no-cloning theorem and dealing with measurements. The last one to discuss is dealing with correcting an infinite set of errors.

Two things are important in understanding how this is done. The first is that quantum operations must be reversible, or unitary. This is a consequence of the time evolution postulate discussed earlier. The second is that every possible unitary operator can be written as a product of the Pauli operators \( I, X, Y, Z \). Thus a code which can correct any Pauli error can then correct any unitary
1.3.2 Abstraction of error correction to subspaces

The discussion of hybrid codes in chapter 2 will draw on the perspective of quantum error correction that considers errors as mappings between subspaces. This subsection introduces a basic formulation of this idea.

The detection of errors in the previous examples of error correcting code required the use of multiple operators which compared the state of different qubits. This detection of errors can be consieved of in a more eloquent manor. Recall that each code began with a single qubit, this qubit can be represent as a two-dimensional vector in Hilbert space. One dimension is the amount of the qubit in the state $|0\rangle$ and the other dimension is the amount of the qubit $|1\rangle$. One way to visualize this two-dimensional vector space that generalizes nicely to higher dimensions is depicted in figure 1.9.

A qubit may be entirely in one dimension of the space, $|\psi\rangle = |0\rangle$, or in some combination of the two, $|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. If this one qubit were to now be encoded into two qubits, then our new state will no longer exist in two-dimensions, but in four-dimensions $|00\rangle$, $|01\rangle$, $|10\rangle$ or $|11\rangle$ (figure 1.10). For three qubits it would be some combination of the eight-dimensions $|000\rangle$, $|111\rangle$, $|100\rangle$, $|011\rangle$, $|010\rangle$, $|101\rangle$, $|001\rangle$ or $|110\rangle$ (figure 1.11) and $n$ qubits exist in $2^n$ dimensional Hilbert space.

Figure 1.10 and 1.11 are only schematic but will assist in understanding the notion of qubits being encoded from smaller subspaces to larger subspaces. In the bit flip example the one qubits is encoded in an eight-dimensional space. Two of those subspaces are identified as the error-free
subspaces, $|000\rangle$ and $|111\rangle$, or as the “codewords”. If after moving through the channel the qubit exists in any combination of other subspaces then an error must have occurred. In this sense errors are mappings from the error-free subspace to another subspace. For example, in the bit flip code an $X_2$ error would be depicted as in figure 1.12. After the qubit leaves the channel, some process is required to narrow down which subspace the qubit is in without narrowing down exactly where it is. Using the method described earlier, the operators $Z_1Z_2$ and $Z_2Z_3$ would give an error syndromes that determines that first and second qubits and the second and third qubits aren’t equal, meaning the qubit is in the state $|010\rangle$ or $|101\rangle$. A correction procedure would then be carried out to send the qubit back to $|000\rangle$ or $|111\rangle$. Essentially the error detection and correction procedure acts as
Figure 1.13: Using the subspace schematic to depict how errors are corrected without determining exactly what error has occurred.

A predetermined set of rules that redirect certain spaces back to code word spaces as depicted in figure 1.13.

Figure 1.13 also demonstrates why two $X_i$ errors could lead to a failure of the code. In summary, in quantum error correction a set of qubits is encoded into some higher dimensional space. In these higher dimensions, an error-occurs which moves the encoded qubits to another subspace associated with an error. Depending on which subspace the errors maps the qubits to an appropriate error correction procedure is used to moves the qubits back to the correct subspace. Every example of such detection and correction procedures have a limited range of errors that they can correct. A natural question then, is how is it determined what subspace the error has mapped the qubits to in this conception. This leads to the introduction of projectors.

Projectors

Projectors determine how much of a quantum states is in a particular subspace. Consider an operator that is both hermitian ($A = A^\dagger$) and unitary ($A^\dagger = A^{-1}$). The square of the this operator is consequently the identity operator $A^2 = AA = AA^{-1} = I$. The identity operator has eigenvalues of 1, therefore the non-squared operator must have eigenvalues of ±1. In this respect, two orthogonal subspaces, $A_1$ and $A_{-1}$ can be defined which are spanned by the eigenvectors with eigenvalue +1
Consider the controlled application of such an operator $A$ to a system of four qubits as in figure 1.14. The four qubits $A$ acts on begin in the state $|\psi\rangle$ while the last qubit begins in the state $|0\rangle$. After the first Hadamard transformation the total state of the system is $|\psi\rangle (|0\rangle + |1\rangle) = \frac{|\psi|0\rangle + |\psi|1\rangle}{\sqrt{2}}$. Application of the controlled operator transforms the state to $|\psi\rangle |0\rangle + A|\psi\rangle |1\rangle$. The second hadamard is then applied to give the final state $\frac{|\psi|0\rangle + A|\psi|1\rangle}{\sqrt{2}}$. This state can be expanded in terms of the above defined projection operators.

\[
\frac{1}{\sqrt{2}}(|\psi\rangle |0\rangle + |\psi\rangle |1\rangle + A|\psi\rangle |0\rangle - A|\psi\rangle |1\rangle)
\]

\[
\frac{1}{\sqrt{2}}(|\psi\rangle |0\rangle + |\psi\rangle |1\rangle + A|\psi\rangle |0\rangle - A|\psi\rangle |1\rangle)
\]

\[
\frac{1}{\sqrt{2}}(|\psi\rangle |0\rangle + |\psi\rangle |1\rangle + (|\psi\rangle - A|\psi\rangle) |1\rangle)
\]

\[
\frac{1}{\sqrt{2}}((I + A)|\psi\rangle |0\rangle + (I - A)|\psi\rangle |1\rangle)
\]

\[
\frac{1}{\sqrt{2}}(P_1|\psi\rangle |0\rangle + P_{-1}|\psi\rangle |1\rangle)
\]
The final quantum system given by equation 1.1 is in one of two possible states. If the final qubit is in the state $|0\rangle$ then the first four qubits are in the state $P_1 |\psi\rangle$, if the final qubit is $|1\rangle$ then the first four qubits are in the state $P_{-1} |\psi\rangle$. Thus measuring the last qubit lets us know which half of the space the quantum state is in. Subsequent applications of such controlled operators can be used to narrow what space a quantum state is in. For example for a three qubit system there exists eight orthogonal subspaces. This system would require three such control operators to determine the systems state. The first would narrow it down to four possible spaces, the second to two and ultimately the last one to one. To ensure these hermitian unitary operators are each dividing the space in half in "different directions", as in each operator is giving new information on which half of space the quantum state is in, the operators must commute with one another.

1.3.3 Error correction

Not all useful quantum codes can correct errors. Some codes are limited to being able to only detect whether or not an error has occurred. Error detecting coding will be important in chapter 2 of the thesis.

Quantum evolutions are unitary, meaning they are reversible. Consider a trace preserving error acting on a quantum state, given by the error $E$, $R$ is a quantum operation that corrects the error.

$$(R \circ E) |\psi\rangle = |\psi\rangle$$

This $R$ exists if the Knill-Laflamme condition [18] is satisfied. This condition states: Let $C$ be the space of logical codewords and $P$ be the projectors onto this space. Their exists an error-correction operation $R$ that corrects the error set $E$ if and only if for all elements of $E$.

$$PE_A^\dagger E_B P = \alpha_{AB} P$$

(1.2)

Where $\alpha$ is a complex number. Equation 1.2 can also be equivalently written using codewords instead of projectors as:

$$\langle i_L | E_A^\dagger E_B | j_L \rangle = \alpha_{AB} \delta_{ij}$$

(1.3)
Where $\delta_{ij}$ is the Dirac delta operator and $|i_L\rangle$ and $|j_L\rangle$ are the logical codewords that $P$ maps to. In this form the error correction condition can be read as a set of errors can be correct so long as no error acting on one codeword maps to the space of an error acting on another codeword. This condition holds for non-degenerate codes. A weaker condition holds for error detection without correction.

\[
PE_A P = \alpha_{ij} P \tag{1.4}
\]

\[
\langle i_L | E_A | j_L \rangle = \alpha_A \delta_{ij} \tag{1.5}
\]

Which for non-degenerate codes says that a code can detect a set of error so along as that error doesn’t map one codeword to another.

### 1.3.4 Bounds

What makes a code better or worse is determined by how much information a code can store and the breadth of errors that it can protect against. Bounds on these parameters exist and are useful to know when creating new quantum codes.

**The quantum Hamming bound**

The quantum Hamming bound [13] [22] provides a limit to how much information a non-degenerate code can store. Consider a code encoding $k$ qubits into $n$ qubits with the potential to correct up to $t$ errors. Suppose $j$ errors occur, such that $j \leq t$. These $j$ errors can occur on any of the $n$ qubits, so the total possible combination of errors that can occur are $\binom{n}{j}$. Each $j$ can be one of three possible errors, $X$, $Y$ or $Z$ resulting in $3^j$ possibilities. The total number of errors that can occur are therefore

\[
\sum_{j=0}^{t} \binom{n}{j} 3^j
\]
Each possible error can occur on each possible qubit giving a total of $2^k$ spaces this error can occur on. To correct errors in a non-degenerate code each of these errors acting on each of these codewords must correspond to a unique orthogonal sub-space of the total space. The total number of subspaces can’t exceed the maximum space available to all our encoded qubits, $2^n$. Which provides the quantum Hamming bound:

$$\sum_{j=0}^{t} \binom{n}{j} 3^j 2^k \leq 2^n$$

**Example** We can use this to test if there is a code more effective than Shor code. The Shor code encodes one qubit, $k = 1$ into $n = 9$ qubits, to correct any possible one qubit error, $t = 1$. With using the above values the quantum hamming bound becomes

$$2(1 + 3n) \leq 2^n$$

The smallest $n$ that satisfies the inequality is $n = 5$. Thus, their must exist a five qubit non-degenerate code that is as powerful as the Shor code.

The Knill-Laflamme bound (Quantum Singleton Bound)

This last bound is applicable to degenerate codes. For a code that encodes $k$ qubits to $n$ qubits to protect against $t$ errors it must satisfy the Knill-Laflamme bound [18].

$$n \geq 4t + k$$

This bound can be written in terms of distance with the relationship, $2t = d - 1$, as $n \geq 2(d - 1) + k$. For example for a code correcting any single qubit error on one qubit, $t = 3$ and $k = 1$. The Knill-Laflamme bound states such a code requires at least 5 qubits, which is in agreement with the previous bound.

Quantum Gilbert-Varshamov bound
A final bound to consider is the Gilbert-Varshamov bound [12]. This bound is similar in structure to the quantum hamming bound but works for degenerate codes. Assume we have a code that encodes \( k \) qubits to \( n \) qubits and corrects \( t \) errors. Such a code exists if

\[
\sum_{j=0}^{2t} \binom{n}{j} 3^{j} 2^{k} \leq 2^{n}
\]

### 1.3.5 The stabilizer formalism

The stabilizer formalism is an efficient way to describe most quantum error correcting codes [14] that allows the simulation of some quantum systems on classical computers. Some hybrid codes in literature use this formalism, and the method for encoding circuits used in this thesis also draws upon it. It is easiest to understand the stabilizer formalism by considering an example.

\[
|\psi_{00}\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}
\]  

(1.6)

When an operator acting on a state leaves the state unchanged, we say the state is stabilized by those operators. The state in equation 1.6 is stabilized by the operators \( X_1X_2 \) and \( Z_1Z_2 \) since

\[
X_1X_2 |\psi_{00}\rangle = |\psi_{00}\rangle \quad \text{and} \quad Z_1Z_2 |\psi_{00}\rangle = |\psi_{00}\rangle
\]

\(|\psi\rangle\) is the only quantum state that is stabilized by these operators, so it can be uniquely described by listing its stabilizers. This can provides a more compact representation of a quantum state.

Consider the single qubit Pauli group, which consist of all one qubit Pauli operators along with the multiplicative factors \( \pm 1, \pm i \). \( G_1 = \{ \pm I, \pm X, \pm Y, \pm Z, \pm iI, \pm iX, \pm iY, \pm iZ \} \). This group can be expanded for \( n \) qubit systems by including all \( n \) fold tensor products of Pauli matrices. All stabilizers are a subset of this group. Stabilizers can be further simplified in terms of there generators. Consider a stabilizer which consists of the Pauli terms \( \{ I, Z_1Z_2, Z_2Z_3, Z_1Z_3 \} \). The individual terms stabilize the following quantum states

- \( I \rightarrow \) All three qubit vectors
- \( Z_1Z_2 \rightarrow |000\rangle, |001\rangle, |110\rangle, |111\rangle \)
- \( Z_2Z_3 \rightarrow |000\rangle, |100\rangle, |011\rangle, |111\rangle \)
- \( Z_1Z_3 \rightarrow |000\rangle, |010\rangle, |101\rangle, |111\rangle \)
The vectors then stabilized by the full stabilizer is $|000\rangle$ and $|111\rangle$. The subspace spanned by these two vectors is therefore the vector state stabilized by $S$. This isn’t the only stabilizer that stabilizes this vector space though, for example so to does $S \equiv \{I, Z_1Z_2, Z_2Z_3, Z_1Z_3, Z_1Z_2Z_3\}$. To be able to uniquely describe a quantum state we need to write the stabilizer in terms of its generators. The generators are the smallest set of operators which generate the stabilizer. In the current example, any one of the $Z_iZ_j$ operators can be constructed from the other two, $Z_1Z_3 = Z_1Z_2 * Z_2Z_3$ and the identity can be constructed from the square of any of the operators, $I = (Z_1Z_2)^2$. Thus, one set of generators for this stabilizer can be $S = \langle Z_1Z_2, Z_2Z_3 \rangle$. These generators will play a key role in designing quantum circuits. The convenience of the stabilizer formalism can be seen by writing out the 7 qubit Steane code by its logical codewords

$$|0\rangle_L = \frac{1}{\sqrt{8}}(|0000000\rangle+|1010101\rangle+|1100110\rangle+|0011111\rangle+|1011010\rangle+|0111100\rangle+|0110011\rangle+|1101001\rangle)$$

$$|1\rangle_L = \frac{1}{\sqrt{8}}(|1111111\rangle+|0101010\rangle+|0011001\rangle+|1110000\rangle+|0100101\rangle+|1000011\rangle+|1001100\rangle+|0010110\rangle)$$

Instead of writing both 1.7 and 1.8 one can write the generators \{ $X_4X_5X_6X_7$, $X_2X_3X_6X_7$, $X_1X_3X_5X_7$, $Z_4Z_5Z_6Z_7$, $Z_1Z_2Z_6Z_7$, $Z_1Z_3Z_5Z_7$ \}.

For completeness I feel compelled to mention one of the most remarkable outcomes of the stabilizer formalism is the Gottesman-Knill theorem [14]. The theorem states: A classical computer can efficiently simulate quantum computation which only involves the preparation of qubits in computational basis states, quantum gates from the Clifford group (Hadamard gates, controlled NOT gates, Phase Gate), and measurements in the computational basis. By keeping track of the stabilizers classical computer can simulate highly-entangled states.

Figures 1.15 to 1.18 provide examples of codes written in the stabilizer formalism. These examples also include the logical equivalents of $X$ and $Z$ for the below codewords, denoted as $\bar{X}$ and $\bar{Z}$. These logical operations are included since they will play a role in later circuit design.
The three qubit bit flip code

<table>
<thead>
<tr>
<th>Generators</th>
<th>Pauli Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_1$</td>
<td>$Z_1Z_2$</td>
</tr>
<tr>
<td>$g_2$</td>
<td>$Z_2Z_3$</td>
</tr>
<tr>
<td>$X$</td>
<td>$X_1X_2X_3$</td>
</tr>
<tr>
<td>$Z$</td>
<td>$Z_1Z_2Z_3$</td>
</tr>
</tbody>
</table>

The error set: $\{I, X_i\}$ for all $i = 1, 2, 3$

Encoding Circuit

Figure 1.15: The generators, error set and encoding circuit for the three qubit bit flip code.

The nine qubit Shor code

<table>
<thead>
<tr>
<th>Generators</th>
<th>Pauli Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_1$</td>
<td>$Z_1Z_2$</td>
</tr>
<tr>
<td>$g_2$</td>
<td>$Z_2Z_3$</td>
</tr>
<tr>
<td>$g_3$</td>
<td>$Z_4Z_5$</td>
</tr>
<tr>
<td>$g_4$</td>
<td>$Z_5Z_6$</td>
</tr>
<tr>
<td>$g_5$</td>
<td>$Z_7Z_8$</td>
</tr>
<tr>
<td>$g_6$</td>
<td>$Z_8Z_9$</td>
</tr>
<tr>
<td>$g_7$</td>
<td>$X_1X_2X_3X_4X_5X_6$</td>
</tr>
<tr>
<td>$g_8$</td>
<td>$X_4X_5X_6X_7X_8X_9$</td>
</tr>
<tr>
<td>$X$</td>
<td>$X_1X_2X_3X_4X_5X_6X_7X_8X_9$</td>
</tr>
<tr>
<td>$Z$</td>
<td>$Z_1Z_2Z_3Z_4Z_5Z_6Z_7Z_8Z_9$</td>
</tr>
</tbody>
</table>

The error set: $\{I, X_i, Z_i\}$ for all $i = 1, 2, 3$

Encoding Circuit

Figure 1.16: The generators, error set and encoding circuit for the Shor code.
The five qubit code

<table>
<thead>
<tr>
<th>Generators</th>
<th>Pauli Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>$g_1$</td>
<td>$X_1Z_2Z_3X_4$</td>
</tr>
<tr>
<td>$g_2$</td>
<td>$X_2Z_3Z_4X_5$</td>
</tr>
<tr>
<td>$g_3$</td>
<td>$X_1X_3Z_4Z_5$</td>
</tr>
<tr>
<td>$g_4$</td>
<td>$Z_1X_2X_4Z_5$</td>
</tr>
<tr>
<td>$X$</td>
<td>$X_1X_2X_3X_4X_5$</td>
</tr>
<tr>
<td>$Z$</td>
<td>$Z_1Z_2Z_3Z_4Z_5$</td>
</tr>
</tbody>
</table>

The error set: \( \{I, X_i, Z_i\} \) for all \( i = 1, 2, 3 \)

Encoding Circuit

Figure 1.17: The generators, error set and encoding circuit for the five qubit code.
1.4 Designing circuits for encoding

There currently exists some overviews in the literature for designing circuits for encoding error-correcting codes when given the generators for a code, most notably [14]. What I found during my study of these, is that they tend to provide more than necessary discussion if one simply desires to learn how to design encoding circuits. This section is intended to be written in such a fashion as to allows someone with a basic understand of error-correction to design a circuit for implementing an error correcting code, as a sort of tutorial. Many examples are provided for clarity.
1.4.1 The 5 step procedure

The procedure of designing a circuit for encoding when starting from a set of generators for the code can be broken down into five steps.

1. **Put generators in check matrix**

A set of \( m \) generators of length \( n \) will have a \( m \) by \( 2n \) check matrix. The matrix can be thought of as having two sides, each of dimension \( m \) by \( n \). The left hand size corresponds to \( X \) elements and the right hand size corresponding to \( Z \) elements. In this matrix each row will correspond to one generator. The entries of that row will be a 0 or 1 depending on the following rules

1. If the generator has an \( X \) at location \( i \), place a 1 on the \( i \)th entry
2. If the generator has a \( Z \) at location \( i \), place a 1 on the \( i + n \)th entry
3. if the generator has a \( Y \) at location \( i \), place a 1 on both the \( i \)th and \( i + n \)th entry.

All other entries are zero. When this is done for each of the \( m \) generators then the check matrix is complete. For example consider the \([5,1]\) code generators and Pauli matrices provided in table 1.2.

<table>
<thead>
<tr>
<th>Generators</th>
<th>Pauli Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M_1 )</td>
<td>( XZZXI )</td>
</tr>
<tr>
<td>( M_2 )</td>
<td>( IXZZX )</td>
</tr>
<tr>
<td>( M_3 )</td>
<td>( XIXZZ )</td>
</tr>
<tr>
<td>( M_4 )</td>
<td>( ZXIXZ )</td>
</tr>
</tbody>
</table>

Table 1.2: Generators and Pauli matrices for the \([5,1]\) code

Following the above procedure will produce the following check matrix.

\[
\begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

2. **Put check matrix in standard form**

The standard form for a check matrix takes the general structure in figure 1.19.
n = number of qubits, r = dimensions of identity matrix, k = number of qubits being encoded

Figure 1.19: The standard form structure of the check matrix.

The first step is to carry out elementary row operations to reduce the matrix to the identity matrix. When attempting to do this one of two things can occur. One, is that you will be able to reduce to the matrix to the identity matrix for every row until the last row and have an \( m \) by \( m \) identity sub-matrix on the left hand side of the check matrix, and some other sub-matrix on the right hand size. Meaning in the form outlined above \( r = m \) and \( n - k - r = 0 \). In this case you are done this step and can move on to the next step. The other possibility is that you will reach a row \( i \) where there can be no 1 in the \( i \)th column. If this occurs you stop at that point and will have some identity sub-matrix on the top left hand size of your matrix with all zero entries below it. In either case the dimensions of this identity matrix is your value \( r \), this will become clear after reading the examples. Once this point is reached one must go back to the general structure and now move to the \( r + 1 \) column, and the \( n + r \) row, and continueing doing row reduction to get an identiy matrix. This time you will be able to continue doing row operations until you reached the last row. This will give you a matrix of the general structure provided in figure 1.19. Continuing with the [5,1] example, the check matrix in standard form is given by

\[
\begin{pmatrix}
1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1
\end{pmatrix}
\]

If necessary qubits can also be relabelled, which equates to swapping columns of the check matrix. An example of this will be given later.

3. **Place standard form generators onto circuit**

Each row of the check matrix in standard form corresponds to one controlled operator which is placed on the circuit.
1. If the row of the check matrix has a 1 on the $i$th entry and a 0 on the $i + n$th entry, a $X$ operator is placed on the $i$th qubit.

2. If the row of the check matrix has a 0 on the $i$th entry and a 1 on the $i + n$th entry, a $Z$ operator is placed on the $i$th qubit.

3. If the row of the check matrix has a 1 on the $i$th entry and a 1 on the $i + n$th entry, a $Y$ operator is placed on the $i$th qubit.

The above procedure is repeated for each row. Doing so in the [5,1] code will create the circuit in figure 1.20. The operators corresponding to each row are colour coated for clarity.

![Figure 1.20: Example of transferring standard form check matrix to a circuit.](image)

Each of these sets of operators will have a control qubit. The control qubit for the operator from the first row will be on the first qubit, the control qubit for the operator from the second row will be on the second qubit and so forth. This control qubit replaces whatever operator that would have otherwise been present on the qubit. This control qubit takes one of three forms depending on what operator it is replacing. These are given in the below in table 1.3.
Table 1.3: The corresponding circuit inputs depending on the $i$th qubit.

<table>
<thead>
<tr>
<th>$i$th qubit</th>
<th>Input in circuit</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>Nothing, this will be an input qubit</td>
</tr>
<tr>
<td>X</td>
<td>Regular control</td>
</tr>
<tr>
<td>Y</td>
<td>Control preceded by a $Z$ operator</td>
</tr>
<tr>
<td>Z</td>
<td>Regular control</td>
</tr>
</tbody>
</table>

For the $[5,1]$ code the following controls are inputted as shown in figure 1.21.

![Figure 1.21: The 5,1 encoding circuit with the appropriate controls.](image)

Note that there was no operator on the fifth qubit to replace, thus this will be the location for the input qubit, the qubit that’s being encoded.

4. Place Hadamards immediately before the control qubit on each generator

This step is entirely contained in its name and is shown in figure 1.22 for our $[5,1]$ example.

![Figure 1.22: The 5,1 circuit with the inclusion of hadamard gates.](image)

5. Check to include $\bar{X}$
Finally, if $r < n - k$, the logical $X$ operators, $\bar{X}$, is placed at the start of the circuit, controlled by the input qubit. In our $[5,1]$ example $r < n - k \rightarrow 4 < 5 - 1$. 4 is not less than 5 − 1 so we do not need to include the $\bar{X}$.

This concludes the five step procedure to encode a circuit. The final circuit produced from this procedure can be further synthesized. Many other works exist regarding the simplification and synthesis of circuits.

1.4.2 Examples

[7,1] Code

The first example will be the Steane [7,1] code. First the generators are placed into a check matrix as in Table 1.4.

<table>
<thead>
<tr>
<th>Generators</th>
<th>Pauli Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1$</td>
<td>$XXXXIII$</td>
</tr>
<tr>
<td>$M_2$</td>
<td>$XXIIXXI$</td>
</tr>
<tr>
<td>$M_3$</td>
<td>$IXIXIXI$</td>
</tr>
<tr>
<td>$M_4$</td>
<td>$ZZZZIII$</td>
</tr>
<tr>
<td>$M_5$</td>
<td>$ZZIIZZI$</td>
</tr>
<tr>
<td>$M_6$</td>
<td>$ZIZIZIZ$</td>
</tr>
</tbody>
</table>

Table 1.4: Generators for the [[7,1]] code.

Check Matrix

$$
\begin{bmatrix}
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
\end{bmatrix}
$$

The check matrix then undergoes row operations to enter into the standard form. This $[7,1]$ code
is an example of the second case of step 2 of the procedure. Once the fourth row is reach it is not possible to use row operations to move a 1 into the fourth row fourth column, since there is no 1 in columns 5, 6 or 7 of the fourth row. Thus, the identity matrix is dimension \( r = 3 \) and we move to the \( n + r + 1 = 11 \) entry of the fourth column and continue doing row operations to get the second identity sub-matrix, shown in figure 1.23.

Check Matrix (Standard form)

\[
\begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1
\end{bmatrix}
\]

Figure 1.23: A highlight of the identity matrices in the standard form of the 7,1 check matrix.

In step 3 each row corresponds to a controlled operator. Converting from the check matrix in standard form into the circuit in figure 1.24.

![Figure 1.24: The circuit for the 7,1 code after applying controls.](image)

Lastly, we check for the \( r < n - k \) condition. For this circuit it is satisfied, \( 3 < 7 - 1 \), thus the \( \bar{X} \) is included at the start of the circuit being controlled by the input qubit.
[6,2] Code

In this example qutrits are being used instead of the usual qubit, so the operator $X^2 \neq I$ and $Z^2 \neq I$. We also use a higher dimensional generalization of the Hadamard. At the heart of it, the procedure is essentially the same. First we place our generators in a check matrix and place the check matrix in standard form give in table 1.5

<table>
<thead>
<tr>
<th>Generators</th>
<th>Pauli Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1$</td>
<td>$X \otimes I \otimes X^2Z \otimes Z^2 \otimes Z \otimes X^2Z$</td>
</tr>
<tr>
<td>$M_2$</td>
<td>$Z \otimes I \otimes X \otimes X^2Z^2 \otimes XZ \otimes X$</td>
</tr>
<tr>
<td>$M_3$</td>
<td>$I \otimes X \otimes Z \otimes Z^2 \otimes Z^2 \otimes Z^2$</td>
</tr>
<tr>
<td>$M_4$</td>
<td>$I \otimes Z \otimes XZ \otimes X^2Z^2 \otimes X^2Z^2 \otimes X^2Z^2$</td>
</tr>
</tbody>
</table>

Table 1.5: The generators and Pauli matrices for the 6,1 code.

Check Matrix

$$
\begin{bmatrix}
1 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 1 & 2 & 1 & 1 \\
0 & 0 & 1 & 2 & 1 & 1 & 1 & 0 & 0 & 2 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 2 & 2 \\
0 & 0 & 1 & 2 & 2 & 2 & 0 & 1 & 1 & 2 & 2 & 2 \\
\end{bmatrix}
$$
For the check matrix to be in standard form we need to relabel certain qubits. Qubits 4 and 5 need to be labeled. To do so columns 4 and 5 are swapped as well as columns 4+6 = 10 and 5+6 = 11.

\[
\begin{bmatrix}
1 & 0 & 0 & 0 & 2 & 2 & 0 & 2 & 0 & 1 & 1 & 2 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 2 \\
0 & 0 & 1 & 0 & 2 & 0 & 2 & 2 & 0 & 0 & 2 & 1 \\
0 & 0 & 0 & 1 & 0 & 1 & 2 & 1 & 1 & 0 & 0 & 2
\end{bmatrix}
\]

Hadamard’s are placed immediately before each control qubit which provides the following circuit.

Figure 1.26: The final circuit for the 6,2 code.

The \( r < n - k \) condition is not satisfied, so this circuit is complete.

[8,2] Code

The last example begins from the check matrix in standard form.

\[
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1
\end{bmatrix}
\]

The operators are placed on the circuit with Hadamard’s before each control in figure 1.27.
Figure 1.27: An 8,2 code placed in a circuit with each control.

We now check for the $r < n - k$ condition, which is satisfied for this circuit, $4 < 8 - 2$. So we include the $\bar{X}$ operators for each of the input qubits, as shown in figure 1.28.

Figure 1.28: The final 8,2 circuit.
Chapter 2

Hybrid Codes

The field of quantum error correction can from one perspective be thought of as an optimization problem. In quantum error-correction we exchange a larger number of unprotected qubits for fewer number of qubits which have protection from some form of errors. The goal is to sacrifice as few qubits as possible and protect the qubits from as large of an error set as possible. Hybrid codes add another dimension to this problem, how many classical bits can also be encoded. This is most easily understood by considering one trivial and one simple hybrid code.

2.1 Trivial example and simple example

A two qubit system can be split into two one qubit systems. One qubit is labelled with a 0 and the other with a 1. The qubit which corresponds to the desired classical bit is transmitted. This effectively converts 2 unprotected qubits of information for 1 unprotected qubit and 1 classical bit. By superdense coding [3] we know that 1 qubit is “worth” at least 2 classical bit, which makes such a hybrid construction trivial. This construction is depicted in figure 2.1.
We can now consider a simple example. Consider the four states in a two qubit system, |00⟩, |01⟩, |10⟩ and |11⟩. With these four states we can create two sets of error correcting code words that each correct the $Z_1$ error. These encodings are depicted in figure 2.2.

In this second example the two unprotected qubits are converted into one partially protected qubit and one classical bit. Again, one qubit was exchanged for one classical bit but as opposed to the first example, some protection was added to the second qubit. These types of codes represent a new dimension to quantum error correcting codes. This example can be modified to correct for any single Pauli error by changing the basis of the codewords. The bases I will refer to are defined in equations 2.1 and 2.2.

$$|+⟩ \equiv \frac{1}{\sqrt{2}}(|0⟩ + |1⟩), |−⟩ \equiv \frac{1}{\sqrt{2}}(|0⟩ - |1⟩) \tag{2.1}$$

$$|↑⟩ \equiv \frac{1}{\sqrt{2}}(|0⟩ + i |1⟩), |↓⟩ \equiv \frac{1}{\sqrt{2}}(|0⟩ - i |1⟩) \tag{2.2}$$

Codes to correct $Z_2$
\[ |0\rangle_L = |00\rangle, \quad |1\rangle_L = |10\rangle \]
\[ |0\rangle_L = |01\rangle, \quad |1\rangle_L = |11\rangle \]

Codes to correct \( X_1 \)
\[ |0\rangle_L = |++\rangle, \quad |1\rangle_L = |+-\rangle \]
\[ |0\rangle_L = |--\rangle, \quad |1\rangle_L = |--\rangle \]

Codes to correct \( X_2 \)
\[ |0\rangle_L = |++\rangle, \quad |1\rangle_L = |+-\rangle \]
\[ |0\rangle_L = |+-\rangle, \quad |1\rangle_L = |--\rangle \]

Codes to correct \( Y_1 \)
\[ |0\rangle_L = |\uparrow\uparrow\rangle, \quad |1\rangle_L = |\uparrow\downarrow\rangle \]
\[ |0\rangle_L = |\downarrow\uparrow\rangle, \quad |1\rangle_L = |\downarrow\downarrow\rangle \]

Codes to correct \( Y_2 \)
\[ |0\rangle_L = |\uparrow\uparrow\rangle, \quad |1\rangle_L = |\downarrow\uparrow\rangle \]
\[ |0\rangle_L = |\uparrow\downarrow\rangle, \quad |1\rangle_L = |\downarrow\downarrow\rangle \]

4 qubit example

This theory of hybrid codes can be applied to a 4 qubit example. Figure 2.3 lists the standard 16 basis states and then provides an equivalent basis. In this second basis two quantum codes exists which can detect the error sets \( E = \{ X_i, Y_i, Z_i Z_1 Z_2, Z_3 Z_4 \} \) for \( i = 1, 2, 3, 4 \).

Code Word 1
\[ |0\rangle_L^{(0)} = |0000\rangle + |1111\rangle \]
\[ |1\rangle_L^{(0)} = |0011\rangle - |1100\rangle \]

Code Word 2
\[ |0\rangle_L^{(1)} = |0101\rangle + |1010\rangle \]
\[ |1\rangle_L^{(1)} = |1001\rangle - |0110\rangle \]
It has been shown that in the asymptotic limit there is an advantage in transmitting both quantum and classical information simultaneously [10], compared to using independent channels, from the perspective of channel capacity. In this chapter I review two perspectives on the construction of hybrid codes and show that these two are equivalent. I then use these constructions to develop the quantum hamming bound for hybrid codes which states that non-degenerate unitary quantum codes will strictly have better parameters than their hybrid counterparts.

### 2.2 A coding theory perspective

A coding theory perspective of hybrid codes was developed in [7]. A quantum error-correcting code that encodes \( k \) qubits into \( n \) qubits with a distance \( d \) can be denoted by \( C = [[n, k, d]] \) or equivalently as \( C = ((n, K, d)) \) where \( K \) is the dimensions of the subspace of the \( k \) qubits, \( K = 2^k \). A classical code can be denoted similarly as \( C = [[n, m, d]] \) which encodes \( m \) bits into \( n \) bits, or as \( C = ((n, M, d)) \) where \( M = 2^m \). Using this notation a hybrid code can be denoted as \( C = [[n, k : m, d]] \) or \( C = ((n, K : M, d)) \) for a hybrid code that encodes \( k \) qubits and \( m \) bits or encodes a quantum system of dimension \( K \) and a classical system of dimension \( M \).
2.2.1 Trivial constructions

These hybrid codes can be constructed in different trivial fashions which do not provide an advantage over the independent solutions, three such constructions were outlined in [7]. The first construction is that given any quantum code, \( C = ((n, KM, d)) \), one can factor the code space into two subsystems of dimension \( K \) and \( M \). The \( K \) dimensions can be used transmit quantum information while the \( M \) dimensions are used to transmit classical information, thus creating a hybrid code \( C = ((n, K : M, d)) \) which provides no advantage. The next constructions comes from assuming one has a hybrid code \( C = [[n, k : m, d]] \) and constructing another hybrid code \( C' = [[n, k - 1 : m + 1, d]] \). Since a qubit can always be used to transmit classical information, the inverse obviously not being true, this hybrid code is also trivial. The last example of a trivial construction is to take a quantum code, \( C_q = [[n_1, k, d]] \), and a classical code \( C_c = [[n_2, m, d]] \) and use them independently to form a hybrid code \( C = [[n_1 + n_2, k + m, d]] \). The goal of developing hybrid codes is to find ones that have better parameters than these trivial constructions.

2.2.2 Hybrid codes with better parameters

A hybrid quantum code \( C = ((n, K : M, d)) \) can be described by a collection of \( M \) quantum codes \( \{C^{(\nu)} : \nu = 1, ..., M\} \). Each code has the same length, dimension, distance and corrects the same error set. The classical information \( \nu \) determines which of the \( C^{(\nu)} \) is used. Each code has an orthonormal basis \( \{|c^{(\nu)}_i\rangle : i = 1, ..., K\} \). For each code to correct the linear span of errors \( E_A \) each code must obey the Knill-Laflamme condition [18], given here in the codeword form.

\[
\langle c^{(\nu)}_i | E_A^\dagger E_B | c^{(\nu)}_j \rangle = \alpha^{(\nu)}_{ki} \delta_{ij} \quad (2.3)
\]

Note the difference between equation 2.3 and when the Knill-Laflamme condition was first presented in 1.2. In 2.3 the \( \alpha^{(\nu)}_{ki} \) can depend on the classical information. As well as obeying 2.3 each codeword from each code must be simultaneously distinguishable from all others to be able to retrieve the
classical information, which gives a second condition.

\[ \langle c_i^{(\nu)} | E_A^\dagger E_B | c_j^{(\mu)} \rangle = 0, \quad \text{for } \mu \neq \nu \]  \hspace{1cm} (2.4)

Equations 2.3 and 2.4 can be written as one equation.

\[ \langle c_i^{\nu} | E_A^\dagger E_b | c_j^{\mu} \rangle = \alpha_{ikl} \delta_{ij} \delta_{\nu\mu} \]  \hspace{1cm} (2.5)

The proof for this condition is outlined in [7].

2.3 Operator quantum error correction

In building up to the operator algebra quantum error correction we first explore its predecessor operator quantum error correction [19]. Operator quantum error correction is a unification of two major techniques to error correction, the standard model and the noiseless subsystems model.

2.3.1 The standard model

The standard model for quantum error correction [18] [2] was discussed in the introductory chapter, here it will be expanded on to a language that can be further generalized later. The standard model consists of a 3-tuple \((\mathcal{R}, \mathcal{E}, \mathcal{C})\). In this model \(\mathcal{C}\) is the quantum code which is a subspace of a Hilbert space \(\mathcal{H}\) that all the codewords exist within. \(\mathcal{E}\) is the error channel which is a linear combination of all possible errors that can occur when the channel is applied to a state, \(\mathcal{E}(\rho) = \sum_{i=1}^n E_i \rho E_i^\dagger\). This set of operators \(E_i\) are not unique, but every possible such decomposition is linearly related to one another. As with all quantum channels, \(\mathcal{E}\) is a completely positive trace preserving map, but the operators that compose it are not necessarily. Lastly, \(\mathcal{R}\) is the recovery operation. For an error correcting code \(\mathcal{R}\) reverses the effect of the channel \(\mathcal{E}\). A code \(\mathcal{C}\) is called correctable for a particular \(\mathcal{E}\) if such an \(\mathcal{R}\) exists. Such a recovery operation \(\mathcal{R}\) for an \(\mathcal{E}\) acting on a \(\mathcal{C}\) exists if and only if

\[ P_C E_i^\dagger E_j P_C = \alpha_{ij} P_C \quad \text{for all } i, j \]  \hspace{1cm} (2.6)

Where \(P_C\) is the projection onto the code space.
2.3.2 Noiseless subsystems

In the noiseless subsystem model for error correction a space is identified which is immune to the effects of a set of errors. In the noiseless subsystem model we begin with the error channel $E$ and its errors $\{E_a, E_a^\dagger\}$. This set $\{E_a, E_a^\dagger\}$ generates a C*-algebra $A$ (see [8] for more). The generation of an algebra is different than that of a group in that a generated algebra consists of all linear combinations of the elements as well as the products. Every finite dimensional C*-algebra, $A$, is isomorphic to a direct sum of full matrix algebras $M_{m_1} \oplus \cdots \oplus M_{m_k}$, for some positive integers $m_1, \ldots, m_k$ and where $M_{m_j}$ is the algebra of $m_j \times m_j$ complex matrices. This can be written as a matrix with full matrix sub-algebras $M_{m_j}$ as:

$$
\begin{bmatrix}
M_{m_1} & 0 & 0 & \cdots & 0 \\
0 & M_{m_2} & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & M_{m_k}
\end{bmatrix}
$$

Here $A$ is a subalgebra of $m \times m$ matrices where $m = \sum_j m_j$. Each zero in the above matrix represents a matrix of zeros of the appropriate size. Below is one particular example of an algebra written in such a structure. In fact, it follows from the representation theory for finite-dimensional C*-algebras that any such algebra is unitarily equivalent to a direct sum of tensor products of the form, $A \cong \bigoplus I_{m_j} \otimes M_{n_j}$.

$$
\begin{bmatrix}
M_3 & 0 & 0 & 0 \\
0 & M_3 & 0 & 0 \\
0 & 0 & M_2 & 0 \\
0 & 0 & 0 & M_5
\end{bmatrix}
$$

The above matrix can be written as a direct sum $M_3 \oplus M_3 \oplus M_2 \oplus M_5$. One can also consider algebras of the form $I_m \otimes M_n$, which has matrix representation for instance in the case $m = 2$ as $\{(\begin{smallmatrix} A & 0 \\ 0 & A^t \end{smallmatrix}) : A \in M_n\}$. If we now consider $A$ as an error acting on some state $\rho$ we can see that if $\rho$ commutes with the error then, $E \rho E^\dagger = \rho E E^\dagger$ and if the error is unitary then $\rho E E^\dagger = \rho$ and thus the error has no effect on $\rho$. We can say that in that case $\rho$ exists within a subspace which doesn’t experience the noise of $E$. We can define a group of $\rho$ that commute with $A$ based on the
structure of the algebra that was just determined. We can see that any $\rho$ that exists in the space $\mathcal{A}' \cong \bigoplus M_{m_j} \otimes I_{n_j}$ will commute with the algebra. This is called the noise commutant and any element of it is immune to the noise from the channel $\mathcal{A}$. Note here how it is the “redundancy” in the matrix sub algebras that allowed for $I_n$ to be tensored out and allow for information to pass through $\mathcal{A}$ unharmed.

Essentially the noiseless subsystem method is one of taking the set of errors one wishes to correct against, finding a subspace in which they act as the identity. The information to protect is then encoded in that noiseless subspace. This structure implies a division of the Hilbert space into two subspaces $\mathcal{H} = \bigoplus \mathbb{C}^{m_j} \otimes \mathbb{C}^{n_j}$.

As an aside, note that the definitions for $\mathcal{A}$ and $\mathcal{A}'$ can equivalently be swapped. The difference between the two forms is just a permutation of the ordering of the basis vectors. If one orders the basis vectors as $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ then one will have one definition for $\mathcal{A}$ and one definition for $\mathcal{A}'$ and if one were to order the basis vectors as $|00\rangle, |10\rangle, |01\rangle, |11\rangle$ then those definition would swap, visualized below.

$$A \otimes I = \begin{bmatrix}
    a & 0 & b & 0 \\
    0 & a & 0 & b \\
    c & 0 & d & 0 \\
    0 & c & 0 & d
\end{bmatrix} \quad \text{Basis ordered } |00\rangle, |01\rangle, |10\rangle, |11\rangle$$

$$I \otimes A = \begin{bmatrix}
    a & b & 0 & 0 \\
    0 & a & b & 0 \\
    0 & c & d & 0 \\
    0 & 0 & a & b \\
    0 & 0 & c & d
\end{bmatrix} \quad \text{Basis ordered } |00\rangle, |10\rangle, |01\rangle, |11\rangle$$

With the noiseless subspace defined, we can now create an operator $\Gamma$ which maps any arbitrary state to one that is in the noiseless subspace. The creation of such an operator requires defining a set of projection operators. Assume the $\mathcal{H}^A = \text{span} \{ |\alpha_i\rangle \text{ for } i = 1, \ldots, m \}$ then define a set of projection operators $P_{kl} = |\alpha_k\rangle \langle \alpha_l| \otimes I_n$ for $k, l = 1, \ldots, m$. These operators project $l'th n$ dimensions of a matrix into the $k'th n$ dimensions of the space. These projectors have the following properties.
These projectors generate the map $\Gamma$ from $\mathcal{B}(\mathcal{H})$ to $\mathcal{B}(\mathcal{H})$ which satisfies the following two properties:

$$\Gamma(\rho) = \sum_{k=1}^{m} \sum_{l=1}^{m} P_{kl} \rho P_{kl}^\dagger \in \mathcal{A}'$$

(2.7)

$$\Gamma(\rho^A \otimes \rho^B) \propto I^A \otimes \rho^B$$

(2.8)

Both of these properties can become clear when simple examples are considered. To understand equation 2.7 consider a general 6x6 matrix

$$\rho = \begin{bmatrix}
\rho_{11} & \rho_{12} & \cdots & \rho_{16} \\
\rho_{21} & \rho_{22} & \cdots & \rho_{26} \\
\vdots & \vdots & \ddots & \vdots \\
\rho_{61} & \rho_{62} & \cdots & \rho_{66}
\end{bmatrix}$$

Choosing the normal basis vectors the $\Gamma$ operation maps $\rho$ to a sum of sixteen terms.

$$\Gamma(\rho) = (|00\rangle \langle 00|)\rho(|00\rangle \langle 00|) + (|00\rangle \langle 01|)\rho(|01\rangle \langle 00|) + \cdots + (|11\rangle \langle 11|)\rho(|11\rangle \langle 11|)$$

To write this concisely we can define the following terms

$$\rho_a = \rho_{11} + \rho_{33} + \rho_{55}$$

$$\rho_b = \rho_{12} + \rho_{34} + \rho_{56}$$

$$\rho_c = \rho_{21} + \rho_{43} + \rho_{65}$$

$$\rho_d = \rho_{22} + \rho_{44} + \rho_{66}$$

Illustrating how $\Gamma$ maps from an arbitrary density operator to one in the noise commutant. The next example illustrates equation 2.8. Here we have two matrices $\rho^A$ and $\rho^B$.
\[ \rho^A = \begin{bmatrix} \rho_{11}^A & \rho_{12}^A & \rho_{13}^A \\ \rho_{21}^A & \rho_{22}^A & \rho_{23}^A \\ \rho_{31}^A & \rho_{32}^A & \rho_{33}^A \end{bmatrix} \]
\[ \rho^B = \begin{bmatrix} \rho_{11}^B & \rho_{12}^B \\ \rho_{21}^B & \rho_{22}^B \end{bmatrix} \]

Again for brevity we need to define \( \rho_e = \rho_{11} + \rho_{22} + \rho_{33} \)

\[ \Gamma(\rho^A \otimes \rho^B) = \begin{bmatrix} \rho_{11}^B \rho_e & \rho_{12}^B \rho_e & 0 & 0 & 0 & 0 \\ \rho_{21}^B \rho_e & \rho_{22}^B \rho_e & 0 & 0 & 0 & 0 \\ 0 & 0 & \rho_{11}^B \rho_e & \rho_{12}^B \rho_e & 0 & 0 \\ 0 & 0 & \rho_{21}^B \rho_e & \rho_{22}^B \rho_e & 0 & 0 \\ 0 & 0 & 0 & 0 & \rho_{11}^B \rho_e & \rho_{12}^B \rho_e \\ 0 & 0 & 0 & 0 & \rho_{21}^B \rho_e & \rho_{22}^B \rho_e \end{bmatrix} = \rho_e (I_3 \otimes \begin{bmatrix} \rho_{11}^B & \rho_{12}^B \\ \rho_{21}^B & \rho_{22}^B \end{bmatrix}) \propto I_3 \otimes \begin{bmatrix} \rho_{11}^B & \rho_{12}^B \\ \rho_{21}^B & \rho_{22}^B \end{bmatrix} \]

In conclusion, an error set will generate an algebra \( \mathcal{A} \) which is used to define a noise commutant \( \mathcal{A}' \). Any density operator in the noise commutant will be unaffected by the noise. We introduce projectors that allow us to define an operation \( \Gamma \) which maps any density operator to this noise commutant.

### 2.3.3 Generalization of noiseless subsystems

The noiseless subspace as defined in the previous subsection is actually a more restricted space than is necessary. To protect the information \( \rho^B \) it is not necessary that the entire state \( \bigoplus \rho^A \otimes \rho^B \) commutes with the algebra \( \bigoplus \mathcal{M}_{m_j} \otimes I \) it is only required that \( \rho^B \) commutes with it. In practice we actually do not care what state \( \rho^A \) is in after the channel. For a particular fixed decomposition \( \mathcal{H} = (\mathcal{H}^A \otimes \mathcal{H}^B) \oplus \mathcal{K} \) we can generalize the noiseless subspace from \( \mathcal{A}' = \{ \rho = I \otimes \rho^B \} \) to \( \mathcal{U} = \{ \rho = \rho^A \otimes \rho^B \} \). Obviously \( \mathcal{U} \) contains \( \mathcal{A}' \). Defining the conditions for this new generalized noiseless subsystem will require some new operators.

\[ P_\mathcal{U} = P_1 + P_2 + \cdots + P_m \]
\[ P_\mathcal{U} \mathcal{H} = \mathcal{H}^A \otimes \mathcal{H}^B \]
\[ P_\mathcal{U}^\perp = I - P_\mathcal{U} \]
\[ P_\mathcal{U}^\perp \mathcal{H} = \mathcal{K} \]
\[ \mathcal{P}_U(\rho) = P_U(\rho)P_U \]

With this there are three equivalent defining properties of the generalized noiseless subspace.

\[ \forall \rho^A, \rho^B, \exists \sigma^A : \mathcal{E}(\rho^A \otimes \rho^B) = \sigma^A \otimes \rho^B \]

\[ \forall \rho^B \exists \sigma^A : \mathcal{E}(I^A \otimes \rho^B) = \sigma^A \otimes \rho^B \]

\[ \forall \rho \in \mathcal{U} : (\text{tr}_A \circ P_U \circ \mathcal{E})(\rho) = \text{tr}_A(\rho) \]

The subspace \( \mathcal{H}^B \) is noiseless if it satisfies any, and thus all, of the above conditions. There exists such a semigroup \( \mathcal{U} \) for a channel \( \mathcal{E} \) if and only if:

\[ P_k E_a P_l = \lambda_{akl} P_{kl} \forall a, k, l \]

\[ P_U^\dagger E_a P_U = 0 \text{ } \forall \text{ } a \]

The first condition presents a structural limitation on the Kraus errors \( E \) which is clear when considering an example. Consider the 2x2 error \( E_a \) with the 2 dimensional projectors \( P_1 \) and \( P_2 \):

\[
E_a = \begin{bmatrix}
    e_{11} & e_{12} & e_{13} & e_{14} \\
    e_{21} & e_{22} & e_{23} & e_{24} \\
    e_{31} & e_{32} & e_{33} & e_{34} \\
    e_{41} & e_{42} & e_{43} & e_{44}
\end{bmatrix}
\]

\[
P_1 E_a P_1 = \lambda_{11} P_{11} \rightarrow \begin{bmatrix}
    e_{11} & e_{12} & 0 & 0 \\
    e_{21} & e_{22} & 0 & 0 \\
    0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0
\end{bmatrix} = \begin{bmatrix}
    \lambda_{11} & 0 & 0 & 0 \\
    0 & \lambda_{11} & 0 & 0 \\
    0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
P_1 E_a P_2 = \lambda_{12} P_{12} \rightarrow \begin{bmatrix}
    0 & 0 & e_{13} & e_{14} \\
    0 & 0 & e_{23} & e_{24} \\
    0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0
\end{bmatrix} = \begin{bmatrix}
    0 & 0 & \lambda_{12} & 0 \\
    0 & 0 & 0 & \lambda_{12} \\
    0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
P_2 E_a P_1 = \lambda_{21} P_{21} \rightarrow \begin{bmatrix}
    0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 \\
    e_{31} & e_{32} & 0 & 0 \\
    e_{41} & e_{42} & 0 & 0
\end{bmatrix} = \begin{bmatrix}
    0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 \\
    \lambda_{21} & 0 & 0 & 0 \\
    0 & \lambda_{21} & 0 & 0
\end{bmatrix}
\]

\[
P_2 E_a P_2 = \lambda_{22} P_{22} \rightarrow \begin{bmatrix}
    0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 \\
    0 & 0 & e_{33} & e_{34} \\
    0 & 0 & e_{43} & e_{44}
\end{bmatrix} = \begin{bmatrix}
    0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 \\
    0 & 0 & \lambda_{22} & 0 \\
    0 & 0 & 0 & \lambda_{22}
\end{bmatrix}
\]
Therefore the general structure of $E_a$ must be

$$E_a = \begin{bmatrix} \lambda_{11} & 0 & \lambda_{12} & 0 \\ 0 & \lambda_{11} & 0 & \lambda_{12} \\ \lambda_{21} & 0 & \lambda_{22} & 0 \\ 0 & \lambda_{21} & 0 & \lambda_{22} \end{bmatrix} = \begin{bmatrix} \lambda_{11} & \lambda_{12} \\ \lambda_{21} & \lambda_{22} \end{bmatrix} \otimes I_2$$

Which is in turn something that we already knew must be true, and that is the errors must act on the first subspace $\rho^A$ only and leave the noiseless subspace $\rho^B$ unchanged. The second condition also has an intuitive explanation. If you consider the structure of the operator $\mathcal{P}_U = \mathcal{P}_1 + \cdots + \mathcal{P}_m$ for the standard basis, $\mathcal{P}_U = |0\ldots0\rangle \langle 0\ldots0| \otimes I_n + \cdots + |1\ldots1\rangle \langle 1\ldots1| \otimes I_n$, as an $nm$ dimension matrix takes the shape

$$\begin{bmatrix} I_n & 0 & \ldots & 0 \\ 0 & 0 & \ldots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & \ldots & 0 \\ 0 & I_n & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 \end{bmatrix} + \cdots + \begin{bmatrix} 0 & 0 & \ldots & 0 \\ 0 & 0 & \ldots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & I_n \end{bmatrix} = I_{nm}$$

The entire Hilbert space is not of dimension $mn$ though since recall $\mathcal{H} = \mathcal{H}^m \otimes \mathcal{H}^m \oplus \mathcal{H}^m$, our entire space is of dimension $mn + k$. So $\mathcal{P}_U$ in our entire space takes the form.

$$\begin{bmatrix} I_{nm} & 0 \\ 0 & 0 \end{bmatrix}$$

Which by the definition of $\mathcal{P}_{U\perp} = I_{mn+k} - \mathcal{P}_U$ then $\mathcal{P}_{U\perp}$ has the form.

$$\begin{bmatrix} 0 & 0 \\ 0 & I_k \end{bmatrix}$$

Now with these structures in mind if you rewrite the second condition in terms of its matrix representation.

$$\mathcal{P}_U E \mathcal{P}_{U\perp} \rightarrow \begin{bmatrix} I_{nm} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} E_1 & E_2 \\ E_3 & E_4 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & I_k \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 0 \\ E_3 & 0 \end{bmatrix} = 0$$

So this condition is saying that the bottom left hand $mn$ by $k$ corner of the error must be zero.

What this is saying thus is that the error can not map elements from the $k$ space into the $mn$ space. Since this must be true for each different decomposition of the channel this is saying errors from one subspace can not map to other subspaces which is again a physically intuitive definition.
In conclusion, in the noiseless subspace method we defined a subspace which was much more restricted than necessary. The space was expanded to a semigroup which include elements in which the noisy subspace information could be lost. Two properties were then given that had to be satisfied for such a semigroup that protects a certain set of errors to exist.

2.3.4 The unified approach

With these two spaces defined we can now introduce operation quantum error correction (OQEC). In OQEC we have the 3-tuple \( \{ R, E, U \} \). The noiseless subspace is a specific case of OQEC where there is no need for a recovery operation, \( R = I \). The standard model is a particular case of OQEC where the space that is protected from the errors is just the codewords, \( C = U \). For such a triple \( \{ R, E, U \} \) we say that \( U \) is correctable for \( E \) if

\[
(tr_A \circ P_U \circ R \circ E)(\rho) = tr_A(\rho)
\]

The left hand of the equation can be broken down. An error occurs on \( \rho \) and a recovery operation is applied to that new state. This state is then projected into the \( mn \) dimensions and the other \( k \) dimensions of space \( A \) are traced out. If this is equal to the initial state with the \( k \) dimensions traced out then \( U \) is correctable for \( E \).

A necessary condition for the existence of such a \( U \) that corrects for \( E \) is that if the projectors that \( U \) defines satisfy

\[
P_k E_i^\dagger E_j P_l = \alpha_{ijkl} P_{kl} \text{ for all } i,j,k,l
\]

Since this is a unified approach we should be able to apply the restrictions of the two models and return to their definitions. In the noiseless model we’re in the space \( \rho^A \otimes \rho^B \) such that errors do not occur on \( \rho^B \), thus an error correction operation \( R \) is not necessary and thus \( (tr_A \circ P_U \circ R \circ E)(\rho) = tr_A(\rho) \rightarrow (tr_A \circ P_U \circ E)(\rho) = tr_A(\rho) \). In the noiseless subsystem we don’t need to be concerned with errors on different projectors being orthogonal because the error correction is designed so errors don’t even occur. So we only need one error to not occur but that errors can’t map between these different separate noiseless systems. Thus \( P_k E_i^\dagger E_j P_l = \alpha_{ijkl} P_{kl} \rightarrow P_k E_j P_l = \alpha_{jkl} P_{kl} \) and \( P_U E_j P_{l}^\perp = 0 \).
In the standard model we are not dealing with a direct sum of multiple spaces. This implies we don’t need an operation that maps to the space we’re correcting errors in, \( P_U = I \). This also implies that the dimension of space that we are tracing out, the dimension of \( A \), is zero. 

\[
(\text{tr}_A \circ P_U \circ R \circ E)(\rho) = \text{tr}_A(\rho) \rightarrow (R \circ E)(\rho) = \rho.
\]

This also implies there isn’t multiple projectors defined by \( \mathcal{C} \). So the correction condition simplifies, \( P_k E_i^\dagger E_j P_l = \alpha_{ijkl} P_{kl} \rightarrow P_C E_i^\dagger E_j P_C = \alpha_{jkl} P_C \).

2.4 Operator algebra quantum error correction

2.4.1 The Schrodinger and Heisenberg pictures of noiseless channels

To move from OQEC to operator algebra quantum error correction (OAQEC) will require another generalization [4]. To do so we first consider the Schrodinger and Heisenberg picture of quantum mechanics. When making a measurement in quantum mechanics we have some state \( \rho \) which is being acted on by some observable \( A \). The expectation value of such a measurement is given by \( \text{tr}(A \rho) \).

We know the quantum system that is being observed is changing with time so it is natural to think that the state \( \rho \) is changing with time, this is the Schrodinger picture of quantum mechanics. Another perspective is that the state \( \rho \) is constant and that is it our observable which is changing with time, this is the Heisenberg picture of quantum mechanics. How this becomes interesting for error correction is when you consider the action of a channel \( \mathcal{E} \) on the expected measurement. Consider the action of a particular Kraus operator. \( \text{tr}(AE_i \rho E_i^\dagger) \). Elements in the trace operator can undergo cyclic permutations without changing value, thus the previous equation is equal to \( \text{tr}(E_i^\dagger A E_i \rho) \).

Notice how even though these two equations are equal it appears as though the error that was once acting on \( \rho \) now is a different error acting on \( A \). This motivates the following theorem. For every error channel \( \mathcal{E} (\rho) = \sum E_i \rho E_i^\dagger \) acting on a state \( \rho \) in the Schrodinger picture there exists a corresponding dual map \( \mathcal{E}^\dagger (A) = \sum E_i^\dagger A E_i \) acting on the observable in the Heisenberg picture. An interesting relationship between the two is that if one is unital the other is trace preserving. In this case we know any quantum channel is trace preserving so the dual map is necessarily unitary. This is easy to show, first since \( \mathcal{E} \) is trace preserving \( \text{tr}(\rho) = \text{tr}(\sum E_i \rho E_i^\dagger) = \text{tr}(\sum E_i^\dagger E_i \rho) \). Therefore \( \sum E_i^\dagger E_i = I \) which means the dual map is unital.
Recall in the Schrodinger picture a subspace was noiseless for a channel $\mathcal{E}$ if $\mathcal{E}(\rho^A \otimes \rho^B) = \rho^A \otimes \sigma^B$. Therefore there exists an equivalent formulation of this definition for the Heisenberg picture, where we consider the effect of the dual channel on the evolution of the observables. A space is noiseless for an error channel $\mathcal{E}^\dagger$ if and only if $P\mathcal{E}^\dagger(X \otimes I)P = X \otimes I$ for all $X$ which are observables. Here $P$ is the projector of the Hilbert space onto the subspace $A \otimes B$. This is also not difficult to prove.

First assume $A$ is a noiseless subsystem for $\mathcal{E}$ and show that the equation must be true. The expected value of the operator on the density state $\rho$ is given by $\text{tr}(P\mathcal{E}^\dagger(X \otimes I)P(\rho^A \otimes \rho^B))$. The state is already in the space $A \otimes B$ so the projectors don’t change the state, $\text{tr}(\mathcal{E}^\dagger(X \otimes I)(\rho^A \otimes \rho^B))$. As we saw before the error can shift from the operators to the state $\rho$, $\text{tr}((X \otimes I)\mathcal{E}(\rho^A \otimes \rho^B))$. Since the state is in the noiseless subspace the state $\rho^A$ is unchanged, $\rho$, $\text{tr}((X \otimes I)(\rho^A \otimes \sigma^B))$. From here we make use of the fact that any quantum channel will have trace 1, $\text{tr}(X\rho^A \otimes \sigma^B) = \text{tr}(X\rho^A)\text{tr}(\sigma^B) = \text{tr}(X\rho^A) = \text{tr}(X\rho^A)\text{tr}(\rho^B) = \text{tr}((X \otimes I)(\rho^A \otimes \rho^B))$. This thus shows that the equation holds.

Looking at the proof from the other direction, we consider an operator $X$ acting on some density matrix being acted on by $\mathcal{E}$. This channel is projected onto the subspace $A \otimes B$, $\text{tr}(X \text{tr}_B(P\mathcal{E}(\gamma)P))$. Here the partial traces out the $B$ dimensions of the system and the operator $X$ acts on what’s remaining. This is equivalent to $X$ acting on the first $A$ dimensions and the identity acting on the second $B$ dimensions. $\text{tr}((X \otimes I)(P\mathcal{E}(\gamma)P))$. The projectors can cycle through the trace and the error can shift to the operators, $\text{tr}(P(\mathcal{E}^\dagger(X \otimes I))P(\mathcal{E}(\gamma)))$. We assume the equation holds and thus is equal to $\text{tr}((X \otimes I)\gamma)$. We can apply the opposite transformation of when we removed the partial trace to reintroduce it $\text{tr}(X \text{tr}_B(\gamma))$. Which means the error doesn’t effect the first $A$ dimensions of the state and is a noiseless subsystem.

This gives two equivalent definitions for a noiseless subsystem, when one is satisfied the other one is as well.
2.4.2 Conservation of algebras of observables

Another approach to these noiseless subsystems is to pick a subspace and ask the question of what operators are conserved in such a subspace, that is which elements satisfy $P\mathcal{E}^\dagger(X_a)P = PX_aP$. We say a set of operators $S$ on $\mathcal{H}$ are conserved by $\mathcal{E}$ for states on some subspace $\mathcal{H}_S$ if every element of $S$ is conserved.

These observables can generate an algebra that we wish to protect from errors. This can also be done via another theorem [5], which states: Let $\mathcal{A}$ be a subalgebra of $\mathcal{L}(\mathcal{H}_S)$ $\mathcal{A}$ is conserved by $\mathcal{E}$ if and only if $E_aP$ commutes with every element of the algebra. This can be expanded to the subalgebra $\mathcal{A}$ being correctable for $\mathcal{E}$ if and only if $PE_a^\dagger E_bP$ commutes with every element of the algebra for every combination of errors.

This generalization considered from the perspective of the Schrodinger picture gives the following result. The algebra $\mathcal{A}$ is correctable for $\mathcal{E}$ for subspaces of the Hilbert space $\mathcal{H}_S$ if there exists a recovery operation $\mathcal{R}$ such that for any density operator which can be separated into a sum of tensor products of operators in the separate spaces, $\rho = \sum_k \alpha_k (\rho_k \otimes \tau_k)$ for $\sum_k \alpha_k = 1$ the following equation holds.

\[
(\mathcal{R} \circ \mathcal{E})(\rho) = \sum_k \alpha_k \mathcal{R}(\mathcal{E}(\rho_k \otimes \tau_k)) = \sum_k \alpha_k (\rho_k \otimes \tau'_k) \tag{2.9}
\]

Not that for $\alpha_1 = 1$ this reduces to the OQEC condition. $(\mathcal{R} \circ \mathcal{E})(\rho_k \otimes \tau_k) = \rho_k \otimes \tau'_k$. There exists such a correction operation if and only if for all $a, b$ there are operators $X_{abk} \in \mathcal{L}(B_k)$ such that

\[
PE_a^\dagger E_bP = \sum_k I_{Ak} \otimes X_{abk} \tag{2.10}
\]

2.5 Unification of the two perspectives for hybrid codes

It can be shown that the coding theory perspective on hybrid codes can be viewed as a special case of the OAQEC formulation. First consider the hybrid code condition from the coding theory construction of hybrid codes. The two necessary conditions, that each individual quantum code obeys the Knill-Laflamme condition and that these codewords from each quantum code are distin-
guishable from all other codewords, was captured in equation 2.5.

\[ \langle c_i^\nu | E_a^\dagger E_b | c_j^\mu \rangle = \alpha_{k \ell}^{\nu} \delta_{ij} \delta_{\nu \mu} \]

This construction consists of three restrictions that do not exist in the OAQEC model of hybrid codes. The first is that \( \mathcal{E} \) is restricted to unitary errors, particularly the Pauli channel. The second is that each quantum code in this construction is viewed as a subspace where the OAQEC model uses subsystems. Lastly in this construction each quantum channel is restricted to be of equal dimensions. In summary:

- \( \mathcal{E} \subseteq P_n \)
- \( C^\nu \subseteq \mathcal{H}, \forall 1 \leq \nu \leq M \) subspaces
- \( \dim C^\nu = K \forall \nu, C^\nu = \text{span}\{\langle c_i^\nu | : 1 \leq i \leq K\} \)

The coding theory condition uses codewords in Hilbert space while the OAQEC models is dealing with operators on Hilbert space. To unify the two it will be necessary to rewrite the coding theory condition in terms of operators. This can be done by considering the two equivalent forms of the regular Knill-Laflamme condition.

\[ \langle c_i | E_a^\dagger E_b | c_j \rangle = \alpha_{k \ell} \delta_{ij} \iff PE_a^\dagger E_b P = \alpha_{k \ell} P \]

One can use either form. Inspired by this we can define the projector onto the hybrid codeword space as:

\[ P = \sum_i \sum_\nu \langle c_i^\nu | c_i^\nu \rangle \]

This projector can then be used to rewrite the coding theory perspective condition in the following form.

\[ PE_a^\dagger E_b P = \sum_i \sum_\nu \alpha_{k \ell}^{\nu} \langle c_i^\nu | c_i^\nu \rangle \]  \hspace{1cm} (2.11)
Note that unlike the general Knill-Laflamme condition, $\alpha$ depends on the codewords so it must be included 'with in' the projector on the right hand side of equation 2.5. To better understand why this is, consider substituting in our definition of the projectors on the left hand side of equation 2.11.

\[
PE_a^\dagger E_b P = \sum_i \sum_j \sum_\nu \sum_\mu |c^\nu_i\rangle \langle c^\nu_i| E_a^\dagger E_b |c^\mu_j\rangle \langle c^\mu_j|
\]

\[
PE_a^\dagger E_b P = \sum_i \sum_j \sum_\nu \sum_\mu \alpha^\nu_{ab} \delta_{ij} \delta_{\nu\mu} |c^\nu_i\rangle \langle c^\mu_j|
\]

\[
PE_a^\dagger E_b P = \sum_i \sum_\nu \alpha^\nu_{ab} |c^\nu_i\rangle \langle c^\nu_i|
\]

So starting from the left hand side of our modified version of the equation, if we follow the two requirements outlined in having a hybrid quantum code we arrive at the equation we had with $\alpha$ contained in the summations.

Now we can shift our attention to the OAQEC condition for a hybrid quantum code. From this perspective we simply need to consider the case where the noisy subspace is just 1, in other words the entire space is correctable and the OAQEC condition simplifies to:

\[
PE_a^\dagger E_b P = \sum_k^M \alpha^\nu_{k1} P_{Ak}
\]

Which gives us the same condition. Thus unifying the two perspectives.

### 2.6 Hamming Bound on Hybrid Codes

An important question in the discussion of hybrid codes is when these code will have some advantage over codes which transmit quantum and classical information separately. What is meant by advantageous depends on the application of the code. One way in which advantageous can be described is by comparing the code parameters. Currently no discussions exists on this topic in the literature for hybrid codes.

The hybrid quantum hamming bound
The quantum hamming bounds can be extended to be applied to hybrid codes, which would then allow for a comparison between the parameters of a hybrid code and a quantum code. The quantum hamming bound applies to non-degenerate codes with the error set being the Pauli matrices. The bound is given by the following equation.

\[ \sum_{j=0}^{t} \binom{n}{j} 3^j 2^k \leq 2^n \]

The bound can be applied to hybrid codes by deconstructing the hamming bound and then reconstructing it for hybrid codes. Firstly, the quantum hamming bound is essentially a packing argument, the amount of space available to the qubits must be greater than the total space the errors can map codewords to combined with the amount of space taken by the codewords themselves. A code which encodes \( k \) qubits to \( n \) qubits will have \( 2^n \) spaces to map errors to. Some of this space will need to be taken by the logical codewords themselves, a code encoding \( k \) qubits will require \( 2^k \) codewords. This leave \( 2^n - 2^k \) space for the errors. If \( j \) errors occur then there are \( \binom{n}{j} \) sets of location where an error can occur. At each set of locations there are three possible errors that can occur, the three pauli errors, giving \( 3^j \) possible errors for each set of locations. All of these errors can occur on each of the codewords which gives a total of \( \sum_{j=1}^{t} \binom{n}{j} 3^j 2^k \) possible errors. Finally, the total number of possible errors and the total number of codewords must be less than or equal to the total space of the system.

\[ \sum_{j=1}^{t} \binom{n}{j} 3^j 2^k + 2^k \leq 2^n \]
\[ \sum_{j=0}^{t} \binom{n}{j} 3^j 2^k \leq 2^n \]

Which is the quantum hamming bound. This same construction can be done for hybrid codes. Similarly, the total space of the system is \( 2^n \). For a hybrid code with \( M \) codes, there will be \( M2^k \) logical codewords. Since each quantum code that makes up a hybrid code must have the same size and correct the same error set, the number of locations an error can occur and the number of possible errors does not change. The number of codewords this error can occur on has changed.
though to $M2^k$ thus the total number of errors which can occur is $\sum_{j=1}^{t} \binom{n}{j}3^j M2^k$. Therefore the quantum hamming bound for hybrid codes becomes, or the \textit{hybrid hamming bound} is given by

$$\sum_{j=1}^{t} \binom{n}{j}3^j M2^k + M2^k \leq 2^n$$

$$M \sum_{j=0}^{t} \binom{n}{j}3^j 2^k \leq 2^n$$

For the simplest case where $M = 2$ the hybrid hamming bound can be written as $\sum_{j=0}^{t} \binom{n}{j}3^j 2^k \leq 2^{n-1}$. Therefore, for a quantum code with parameters $(n, k, d)$ the best hybrid code which encodes the same quantum information can encode 1 classical bit and have parameters $(n+1, k : 1, d)$. For any quantum code an additional qubit can be added to the channel which is used to submit classical information, just a system does not provide an advantage over separately transmitting quantum and classical information. This situation is precisely what is happening with the best equivalent hybrid codes. Therefore, for the non-degenerate case where the error set consists of the Pauli operators the hybrid code provides no advantage over an equivalent quantum code and a separate classical channel.

Thus, as a consequence of the hybrid hamming bound, degeneracy is necessary for constructing non-trivial hybrid codes.
Chapter 3

NMR Quantum Information Processing

In Chapter 3 we begin by reviewing how NMR can be used as a QIP. Once this established we move into simulating NMR QIP experiments with Matlab. The entirety of the work done in this chapter is referring to simulations done in Matlab, none of the work in this chapter is done in a NMR lab.

3.1 Introduction to NMR QIP

As the field of quantum information continues to grow so do the proposals for physical implementations of quantum information processors (QIPs). These proposals include quantum optics based devices (cavity QED, circuit QED), ultra-cold atoms (trapped ions), superconducting quantum computers and spin-based quantum computers. In this chapter we explore one of the later types, particularly NMR quantum computing.

D.P. DiVincenzo identified five criteria that must be satisfied in order for a system to be used as a quantum information processor [11].

1. A scalable physical system with a well-define qubit.

2. A set of quantum gates that can implement any quantum computation.

3. Being initializable to a simple fiducial state.

4. Permit qubit-specific measurements
5. Have a sufficiently long decoherence time

These criteria will guide the general structure of section 3.1. For a more thorough introduction to NMR QIP see [20]

3.1.1 The Qubit

The first step to having a QIP is having a well-defined qubit. One natural consideration for a qubit, and one that is used by other quantum information processors is an isolated spin $\frac{1}{2}$ nuclei. Nuclear spins do not interact significantly with the systems other degrees of freedom making them a good candidate for qubits [1]. When such a nuclei is placed in a strong magnetic field $B_o$ two things of interest occur. One is that the nuclei’s spin will either align or anti-align with the magnetic field, similar to how bar magnets would align themselves. The other is that the nuclei will be quantized into two states, spin up or spin down, whose energy difference is given by the Zeeman effect

$$\Delta E = \hbar \gamma B$$

This gives us two distinct quantum states which can be encoded as the two logical states for a qubit. Namely, the spin up and the spin down states are encoded as the basis state $|0\rangle$ and $|1\rangle$ respectively. This model of a single ion qubit is useful in its own right and is used in quantum information processors such as ion trap quantum computers. A primary drawback of this model is that the qubit-qubit coupling interaction is quite weak and highly susceptible to decoherence. This
principal drawback can be overcome by storing the nuclei in molecules. Within the structure of a molecule the neighbouring nuclei’s magnetic dipole and electron mediated interactions provide a strong coupling and the effects of the environment are small enough to increase decoherence time. In this new framework the molecule is a tiny quantum computer and each spin $\frac{1}{2}$ nuclei is one qubit of the computer.

Unfortunately single molecules cannot be used as quantum information processors due to technological limitations. This limitation arises when attempting to measure the state of the nuclei. Measurement of the states of the nuclei in NMR are done through use of Faraday’s law of induction, which states that changing magnetic fields induce electric signals. Each nuclei has its own magnetic dipole moment and thus its own magnetic field, but to be able to detect these nuclei the magnetic field needs to be changing. To do so the nuclei is exposed to a magnetic pulse $B_{rf}$ that is perpendicular to $B_o$, that tips the nuclei off the z-axis. Since the nuclei is now at some angle with the z-axis and has an angular momentum the presence of the stronger magnetic field $B_o$ will induce a precession of the nuclei around the z-axis. This precession results in an oscillating magnetic field produced by the nuclei which provides the necessary induced current. The frequency of this precession is called the *larmor frequency* and is given by the equation $\omega = \mu B$, where $\mu$ is the magnetic moment of the nuclei.

Unfortunately the signal induced from a single nuclei is much too weak to be measured with current technology so an ensemble of many nuclei, on the order of $10^{10}$, are required to produce
a detectable signal. So measurements are not done of single quantum computers but an average value of many quantum computers acting together.

With the qubit now defined we will consider how such measurements are done in principal by considering a common example of a molecule used as a NMR QIP, $^{13}\text{C}$-labeled trichloroethylene (TCE).

TCE contains three spin $\frac{1}{2}$ nuclei, one $^1\text{H}$ and two $^{13}\text{C}$, which gives TCE a total computational power of three qubits. We are able to detect each of the three spin $\frac{1}{2}$ nuclei separately since each has a unique Larmor frequency and thus induces a unique signal. The difference in Larmor frequencies can arise from either the nuclei having different magnetic dipole moments or from the nuclei experiencing different effective magnetic fields. The earlier reason is why $^1\text{H}$ and the two $^{13}\text{C}$ have different magnetic fields and the later reason is why the two $^{13}\text{C}$ have different Larmor frequencies. The two $^{13}\text{C}$ nuclei experience different magnetic fields due to having different chemical environments. The $^{13}\text{C}$ molecule are bonded to different nuclei which slightly alter the magnetic field experience in their immediate environment. This effect is known as the chemical shift. For example, at 11.7T, the precession frequency for $^1\text{H}$ for $^{13}\text{C}$ are approximately 500Mhz and 125Mhz respectively. The system Hamiltonian for TCE is $H = 2(2989.17\pi Z_1 + 25459.09\pi Z_2 + 21592.45\pi Z_3 + 29341.79\pi Z_4 + 41.62\pi Z_1 Z_2 + 1.46\pi Z_1 Z_3 + 7.02\pi Z_1 Z_4 + 69.66\pi Z_2 Z_3 + 1.18\pi Z_2 Z_4 + 72.16\pi Z_3 Z_4)$.

Coils tuned to the different precession frequencies of the nuclei are wrapped around the sample to detect the signal in the x-y plane. These coils would detect the average magnetization produced by all $10^{10}$ molecules of TCE. This signal is then used to determine the state of our three qubits.
Lastly, with a bit of foresight we can see that the application of gates will be quite complicated if each of our qubits is constantly precessing. To avoid this we consider each qubit from its own rotating frame of reference. If from the lab frame of reference each qubits is precessing at different frequencies \( \omega_1, \omega_2, \) and \( \omega_3 \) then from the reference frame of qubit one, it itself is stationary and the other qubits are rotating with a frequency of \( \omega_1 + \omega_2 \) and \( \omega_1 + \omega_3 \).

### 3.1.2 Quantum Gates

With a well defined qubit established we not turn our attention to the goal of universal computation. Achieving universal computation is most easily done by achieving a finite set of quantum transformations which can be combined in different ways to achieve any possible quantum computation. Such a set of transformations or gates is called universal. For an NMR QIP our universal gates will be composed of the single qubit Pauli gates and the two qubit control not gate. To understand how Pauli gates are applied in an NMR QIP a requisite understanding of the Bloch sphere representation of a qubit is required.

#### The Bloch Sphere

The Bloch sphere is a purely mathematical construct which is used represent the state of a qubit as a three dimensional vector in a sphere. Generally a qubit \( |\psi\rangle = \alpha |0\rangle + \beta |1\rangle \) can be written in terms of the density matrix \( |\psi\rangle \langle \psi| \) which can in turn be expanded in to the form \( |\psi\rangle \langle \psi| = \frac{1}{2} (I + \alpha_x X + \alpha_y Y + \alpha_z Z) \). Each density matrix can then be represented by the unique vector \( \vec{v} = (\alpha_x, \alpha_y, \alpha_z) \). When plotted on a three dimensional unit sphere this vector serves as the Bloch sphere representation of the qubit \( |\psi\rangle \).

In this representation the vector aligned with the positive and negative z-axis is \( |0\rangle \) and \( |1\rangle \) respectively. This is can be shown mathematically.

\[
|\psi\rangle = |0\rangle \\
|\psi\rangle \langle \psi| = |0\rangle \langle 0| = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}
\]
Similarly it can be shown that every point on the surface of the sphere corresponds to a unique pure state and every point in the interior of the sphere represents a unique mixed state.

Before discussing how one qubit gates are applied in this model it is important to step back and think of the application of quantum gates more broadly. The application of a quantum operation transforms one quantum state to another. How this operation is represented depends on how we have chosen to represent the quantum state. For example, if the quantum state is represented as a vector then the quantum operation would be given by a single matrix $|\psi\rangle = U |\psi'\rangle$. Contrarily, if our quantum state was represented as a density matrix then our quantum operation could not be represented by a single matrix, but as an operation on the density matrix $|\psi\rangle \langle \psi | = U |\psi'\rangle \langle \psi' | U'$. More over, if our quantum state is represented by a three-dimensional vector on the Bloch sphere, the effect of a unitary operation can be represented as rotation around an axis of the sphere.
In an NMR QIP individual gates are applied as rotations around the Bloch sphere. The rotations are physically implemented by applying a magnetic pulse $B_{RF}$ in the radio frequency spectrum to our spin. The effect of such a pulse is to cause the spin to rotate around the axis that the pulse is applied. Control of these transformations is given by the direction, strength and duration of the applied pulse. The stronger $B_{RF}$ is the faster the spin will rotate in the chosen direction. The longer $B_{RF}$ is applied the further the spin will rotate in the chosen direction.

Since the effect of the pulse depends on the direction it is applied from we must apply the pulse consistently from one direction to get the desired transformation. Recall though that are nuclei are constantly precessing. To address this we apply $B_{RF}$ from nuclei’s frame of reference, so in the lab the $B_{RF}$ is oscillating at the same frequency as the Larmor frequency of our target qubit.

A natural question that arises at this point is how these pulses can target one qubit without effecting the others when the pulse is necessarily applied to the entire molecule. As we said earlier the pulse is oscillating at the same frequency as the target nuclei so for non-target nuclei which have significantly different rotational frequencies, such as qubits of different nuclei isotopes, the effect of the magnetic pulse will be experienced by the non-target qubit in all directions as seen in figure 3.5. The net effect then is no change in the direction of the spin for the qubits which have different nuclear isotopes.

For nuclei of the same isotope and different chemical shifts the difference in precession frequency is quite small and so the $B_{rf}$ will stay close enough in phase with the non-target nuclei of the same isotope to noticeably effect it. To counter this we instead weaken the strength of the applied pulse and apply it for a much longer duration. This effectively gives the $B_{rf}$ enough time to lag behind the phase of non-target nuclei so as to not effect its orientation.

Two qubit gates

Implementation of two-qubit gates in NMR is made possible by the effect of j-coupling. The nuclei in the molecule interact with one another through their magnetic fields, as we would expect any 'bar magnet' to. This interaction is first made more complex than simple bar magnets due to the
Figure 3.5: The first figure shows the effect of an applied magnetic field precessing at the same frequency as the nuclei and the second figure shows the effect when the applied magnetic field is precessing at a different frequency.

presence of the moving electrons who each shield and produce their own magnetic field. This in turn is made even more complex when we consider that these 'bar magnets' are also rotating and thus have an angular momentum. This complex interaction is conveniently summarized as j-coupling.

To understand the effect that j-coupling has on qubits we will use the qubits Hamiltonian to see how the system evolves for two examples. First, the Hamiltonian for a two qubit system in NMR is provided by

\[
H = \omega_1 Z_1 + \omega_2 Z_2 + \frac{J_\pi}{2} Z_1 Z_2
\]

The first two terms are contributed from the rotational energy of the first and second qubit on their own and the third term is a result of the j-coupling. The \( Z \) operators in the Hamiltonian are applied on each qubit with respect to their own rotating frame of reference.

Consider the evolution of the system if the first qubit is in the spin up state \( \langle 0 \rangle \). \( Z_1 \) applied to such a state returns the eigenvalue of +1.

\[
H_{\langle 0 \rangle} = \omega_1 (+1) + \omega_2 Z_2 + \frac{J_\pi}{2} (+1) Z_2
\]
The first term then reduces to an overall constant which we can effectively ignore with out loss of generality and the Hamiltonian simplifies to

\[ H_{|0\rangle} = (\omega_2 + \frac{J\pi}{2})Z_2 \]

Which is identical to the evolution of the second qubit on its own had the j-coupling not been there, except there exists an extra \( \frac{J\pi}{2} \) term. This translates to when the first qubit is in the spin up state the second qubit precess \( \frac{J\pi}{2} \) faster. If we carry out the same logic for when the first qubit is in the spin down state we get the same result but the second qubit precess \( \frac{J\pi}{2} \) slower.

\[ H_{|1\rangle} = (\omega_2 - \frac{J\pi}{2})Z_2 \]

This is therefore effectively a control phase operation, or a control-\( Z_{\theta(t)} \). By sandwiching the j-coupling effect with other 90° Pauli operators and allowing the j-coupling to run for a time that results in a 90° rotation we can create a CNOT gate. This is provided in figure 3.6 schematically where the red arrow depicting the evolution of the second qubit when the first qubit is in the state \(|0\rangle\) and the blue arrow is depicting the evolution of the second qubit when the first qubit is in the state \(|1\rangle\). Unfortunately, since j-coupling is a natural effect of the Hamiltonians time evolution we are not able to turn it "on and off" as we desire. To be able to have proper control of the CNOT gate we need to make use of a technique called refocusing.

The nuclei are constantly spreading apart due to the effect of the j-coupling. Once the nuclei have spread out to a certain degree a 180° rotation is applied in the x-direction. The nuclei then continue spreading in the direction they were, but now due to the 180° rotation the nuclei are moving together. After a period of time equal to the amount of time they were spreading out for
the nuclei will all realign. At this point a $-180^\circ$ rotation is applied that reverses the nuclei back to their starting positions, effectively muting the effect of the j-coupling. How this looks in the x-y plane is shown schematically in figure 3.7.

3.1.3 Measurement

As mentioned early, an NMR QIP differs from other quantum computers in that measurements occur on an ensemble of spins as opposed to a single one. The readout from an NMR quantum computer then does not give the definite state of a single qubit but instead gives an average of all possible states. This poses a challenge since most existing quantum algorithms depend on taking definite measurements. Fortunately, most of these algorithms can be modified for use in NMR. Another consequence of NMR measurements is that the coupling between the measuring instrument and the nuclei is not strong, so measurements in NMR do not collapse the state of the system. As a result we can think of NMR measurements as classical in nature.

Physically the state of the qubits is measured by coils wrapped around the NMR sample transverse to the external field $B_0$. Ideally all of the nuclei corresponding to one qubit are precessing at the same frequency such that each of their individual magnetic fields add to form a detectable magnetic field. This bulk magnetization is oscillating with the nuclei Larmor frequency and thus induces a current in the coil that is tuned to that frequency. This current is then sent to a classical computer to be analyzed. The signal is measured solely in the xy plane and due to the effects of relaxation is constantly decaying. We refer to this signal as the free induction decay (FID). This decaying
\[ \rho_o = XI, \text{x magnetization (left) and y magnetization (right)} \]

\[ \rho_o = IY, \text{x magnetization (left) and y magnetization (right)} \]

Figure 3.8: The measured magnetization in the x plane and y plane of the two carbon nuclei in TCE.

Planar magnetization has x and y components, \( M_x(t) \) and \( M_y(t) \), given by the equations

\[
M_x(t) = m \text{tr}[(U(t)\rho_oU(t)^\dagger)(\sum_{i=1}^{n} X_i)]
\]

\[
M_y(t) = m \text{tr}[(U(t)\rho_oU(t)^\dagger)(\sum_{i=1}^{n} Y_i)]
\]

where \( n \) is the number of qubits, \( m \) is a scalar value that depends on the number of nuclei and their magnetic moment, \( U \) is the operator that represents our initial states time evolution \( U = e^{-iHt} \) and \( \rho_o \) is our initial state. These two magnetizations are combined to form a complex signal \( M(t) \).

\[
M(t) = M_x(t) + iM_y(t) = \text{tr}[\rho(t)(\sigma_1 + \sigma_2)]
\]

Where \( \sigma_j = X_j + iY_j \). For example, the time evolution of the two carbon nuclei in TCE is given by the operation \( U = e^{-i(\omega_1 Z_1 + \omega_2 Z_2 + J_{12} Z_1 Z_2)t} \). Using this as the \( U(t) \) and setting \( m = 1 \) the magnetization in the x and y plane for different initial states \( \rho_o \) will produce a signal like the one graphed in figure 3.8. To see how the magnetization is changing in the xy plane refer to figure 3.9. The graphs of the form in figure 3.9 are not used in NMR measurements but are useful for visualizing how the magnetization is evolving over time.

The measured signal, like in figure 3.8, is actually a composition of many different frequencies. For example if we ignore the coupling term in the above TCE Hamiltonian, then the measured magnetization would simply result from the combined signal from the first and second carbon. The
Fourier transform can be used to break the signal down into the different frequencies that compose it. Figure 3.10 shows an example of what the measurement from 3.8 would look like after the fourier transform has been applied. If the coupling term is now included then the measured signal will be a combination of four different frequencies. To be explicit, the four signals come from the first qubit’s frequency when the second qubit is in the state |0⟩, the first qubit’s frequency when the second qubit is in the state |1⟩, the second qubit’s frequency when the first qubit is in the state |0⟩ and the second qubit’s frequency when the second qubit is in the state |1⟩. Therefore when the Fourier transform is applied on the signal the resulting read out will have four distinct peaks. This in effect each of the two peaks seen in figure 3.10 into two peaks. In this example these four peaks can be labelled |σ_i0⟩, |σ_i1⟩, |0σ_i⟩ and |1σ_i⟩ where i represent the x or y magnetization depending on which signal is being observed. Two example are provided in figure 3.11 of this peak splitting phenomenon.

The two peaks from each qubit will have only slight differences in phase so the peaks will be close together, even seemingly overlapping, after the Fourier transform. The distance between these two
\[ \rho_0 = X I \]

\[ \rho_0 = I Y \]

Figure 3.10: The Fourier transform is applied to the read out signal to distinguish the individual frequencies that compose it.

x magnetization, qubit 1

\[ |\sigma_x 0\rangle \]
\[ |\sigma_x 1\rangle \]

y magnetization, qubit 2

\[ |0_{\sigma_y}\rangle \]
\[ |1_{\sigma_y}\rangle \]

Figure 3.11: Labeling of distinct peaks in an NMR read out.
close peak is proportional to the strength of our coupling constant as can be seen in figure 3.12. For two qubits the effect of j-coupling was essentially that if one qubit is in the state 0 or 1 the other qubits frequency increases or decreases respectively. In the Fourier transform this is shown by the two peak splitting. This effect naturally extends to when there are multiple other qubits, the peaks will split further because there will be many more combinations of states the other qubits can be in to shift that qubits frequency. For example if there exists three qubits, the two qubits effecting the third qubit can be in the states |00⟩, |01⟩, |10⟩ or |11⟩, so the peak for the third qubit would split into four other peaks. In general for \( n \) total nuclei there will be \( 2^{n-1} \) split peaks, each corresponding to the state of the other qubits.
3.1.4 Initial State

Most quantum computations require a pure initial state, often this desired state is the set of fully polarized spins in the states $|00\ldots0\rangle$. Unfortunately, the energy difference between the state $|0\rangle$ and $|1\rangle$ is much smaller than the energy in the environment in liquid state NMR experiments conducted at room temperature. As a result, when a sample is left in room temperature outside of a magnetic field it enters a nearly random state in approximately 10-40s. This equilibrium state density matrix is proportional to $e^{H/kT}$, according to the Boltzmann distribution. We can approximate the equilibrium state density matrix of a system with $n$ spins as

$$
\rho_{eq} = e^{-H/kT} \frac{\operatorname{tr}(e^{-H/kT})}{\operatorname{tr}(e^{-H/kT})} \approx I - \frac{1}{kT}(\epsilon_1 Z_1 + \epsilon_2 Z_2 + \ldots + \epsilon_n Z_n)
$$

To circumvent this challenge certain properties of NMR quantum computations can be used to create initial states that effectively behave as if they are pure states. These states are called pseudo-pure states and take the form.

$$
\rho = I - \epsilon \frac{1}{2^n} I + \epsilon |00\ldots0\rangle \langle 00\ldots0|
$$

Where $\epsilon$ is on the order of $10^{-5}$. This is a combination of the maximally mixed state and the desired initial state.

These pseudo pure states behave as desired pure states would due to three properties of NMR quantum computations. First, the measured magnetization depends only on the traceless portion of the initial states density matrix. To see this consider two density matrices, $\rho$ and $\delta = \rho + \alpha I$. The magnetization is proportional to the trace of the product of the density matrix and some traceless operator, $\vec{m}$ (depending on which direction we’re measuring in).

$$
M_\delta \propto \operatorname{tr} (\delta \vec{m}) = \operatorname{tr} (\rho \vec{m} + I \vec{m}) = \operatorname{tr} (\rho \vec{m})
$$

Equivalently, since all NMR observables are traceless the maximally mixed state provides no observable signal meaning the signal depends only on the deviation matrix. Secondly, since the maximally mixed state does not evolve under any unitary transformation it will thus be unchanged by any quantum computation. For example.
\[ U \rho U^\dagger = U (\frac{1 - \epsilon}{2n} I + \epsilon \langle 00 \ldots 0 \rangle \langle 00 \ldots 0 \rangle) U^\dagger = U (\frac{1 - \epsilon}{2n} I) U^\dagger + U (\epsilon \langle 00 \ldots 0 \rangle \langle 00 \ldots 0 \rangle) U^\dagger = \frac{1 - \epsilon}{2n} I + U (\epsilon \langle 00 \ldots 0 \rangle \langle 00 \ldots 0 \rangle) U^\dagger \]

Therefore, any final measurement made after a quantum operation is performed depend only on the initial deviation. Lastly, measurements made in NMR quantum computation are all relative. When analyzing the read-out measurements it was never considered what the actual value of the magnetization measured was, only the shape of the induced signal was considered. These shapes can all be scaled up or down with out changing the read out signal, as long as the signal remains sufficiently greater than the background noise.

There exists three methods for preparing effective pure states starting from thermal equilibrium, these are logical labeling, temporal averaging and spatial averaging \[28\]. All of these methods incurs an exponential cost in the number of experiments required or in the strength of the signal. None the less, these methods are essential to having an NMR quantum computer.

The methodology of the averaging techniques is conceptually simple, this can be illustrated with a simple example. Consider a 2-spin system given by the density matrix.

\[ \rho_0 = \begin{bmatrix} \alpha & 0 & 0 & 0 \\ 0 & \beta & 0 & 0 \\ 0 & 0 & \gamma & 0 \\ 0 & 0 & 0 & \delta \end{bmatrix} \]

Permutations matrices can be simulated with series of CNOT gates to create the density matrices

\[ \rho_1 = \begin{bmatrix} \alpha & 0 & 0 & 0 \\ 0 & \delta & 0 & 0 \\ 0 & 0 & \beta & 0 \\ 0 & 0 & 0 & \gamma \end{bmatrix}, \quad \rho_2 = \begin{bmatrix} \alpha & 0 & 0 & 0 \\ 0 & \gamma & 0 & 0 \\ 0 & 0 & \delta & 0 \\ 0 & 0 & 0 & \beta \end{bmatrix} \]

The average of these three density matrices creates the states

\[ \frac{1}{3} \begin{bmatrix} 3\alpha & 0 & 0 & 0 \\ 0 & \beta + \delta + \gamma & 0 & 0 \\ 0 & 0 & \beta + \delta + \gamma & 0 \\ 0 & 0 & 0 & \beta + \delta + \gamma \end{bmatrix} = \frac{\beta + \delta + \gamma}{3} I + \frac{3\alpha - \beta - \delta - \gamma}{3} |00\rangle \langle 00| \]

Which is in the form of a pseudo pure state. Various methods have been created to achieve the effect of averaging. With the above example, we would run the same quantum operation three
times once with each of the three different initial states above. Once we have our three outcomes we then average the three and the result is equivalent to having run the quantum operation on our pseudo pure state. This is known as \textit{temporal averaging}.

Alternatively, instead of running three separate experiments we can also use a gradient field to create different layers of our sample which are in different initial states and then run the experiment. Then in the same way we average the results of these different layers to get the same effect. This is known as \textit{spatial averaging}.

In practise both these techniques are quite complex. For example for spatial averaging a sequences of gates and gradients is required to set up the initial pseudo-pure state. Examples of such sequences for three and four qubit gates is provided in figure 3.13, two are provided to make obvious the symmetry in the sequence.

With both methods the number of qubits and pulses required increases rapidly as the number of
qubits increases. A conceptual way to understand what is happening is to consider system of $n$ spin-half nuclei. This system will have $2^n$ energy levels with nearly equal populations except for the ground state which will have a slightly higher population. This is different than our desired pure state since the higher energy levels do not all have the same population. We then apply some operation that equalizes the population of the non-ground state energy levels, to a population less than our ground state. We can then imagine the signals from one of each molecule ”cancelling out” until we’re left with only molecules in our ground state. This is essentially what happens with our pseudo pure state.

### 3.2 Simulating a NMR QIP in Matlab

Implementing NMR QIP experiments can be expensive and time consuming. To save time and resources NMR experiments can be simulated before hand to correct mistakes before entering the lab. In this thesis Matlab was used to simulate the the encoding of the hybrid code, the effects of the errors on the codewords and the decoding of the hybrid code. Experimental values were used to simulate the decoherence experienced in the lab. This section will outline a procedure simulating an NMR QIP on Matlab. The following code uses the following defined matrices

**Matlab input**

\[
\begin{align*}
I &= \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; \\
X &= \frac{1}{2}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}; \\
Y &= \frac{1}{2}\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}; \\
Z &= \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix};
\end{align*}
\]

For convenience all tensor products of operators were defined at the start of the code, for example.

**Matlab input**

\[
ZI = \text{kron}(Z,I); \\
II = \text{kron}(I,I); \\
Z1 = \text{kron}(ZI,II);
\]

Then for convenience of input into functions corresponding matrices were stored in single variables, for example.

**Matlab input**

\[
\begin{align*}
Iz(:,:,1) &= Z1; \\
Iz(:,:,2) &= Z2; \\
Iz(:,:,3) &= Z3; \\
Iz(:,:,4) &= Z4;
\end{align*}
\]
This was done for each tensor product and is the form they will be provided in this section.

### 3.2.1 Read out

The first element of the experiment to be simulated is the read out signal. Simulating a read out signal requires the defined Hamiltonian for the molecule being used. Crotonic Acid is a four qubit molecule in NMR QIP and has strong coupling between neighbouring nuclei, making it a good choice of molecule for this experiment. The Hamiltonian for crotonic acid consists of two terms, one corresponding to the chemical shift and one to the coupling. Note, that we are dealing with four qubits and thus four qubit operations, so for example $Z_3 \equiv IZI$.

\[
H_{cs} = 2989.17\pi Z_1 + 25459.09\pi Z_2 + 21592.45\pi Z_3 + 29341.79\pi Z_4
\]

\[
H_{coupling} = 41.62\pi Z_1 Z_2 + 1.46\pi Z_1 Z_3 + 7.02\pi Z_1 Z_4 + 69.66\pi Z_2 Z_3 + 1.18\pi Z_2 Z_4 + 72.16\pi Z_3 Z_4
\]

\[
H = 2H_{cs} + 2H_{coupling}
\]

**Matlab input**

```matlab
Hcs = v(1)*pi*Iz(:,:,1) + v(2)*pi*Iz(:,:,2) + v(3)*pi*Iz(:,:,3) + v(4)*pi*Iz(:,:,4);
Hcoup = J(1,2)*pi*Iz(:,:,1)*Iz(:,:,2) + J(1,3)*pi*Iz(:,:,1)*Iz(:,:,3) + J(1,4)*pi*Iz(:,:,1)*Iz(:,:,4) + J(2,3)*pi*Iz(:,:,2)*Iz(:,:,3) + J(2,4)*pi*Iz(:,:,2)*Iz(:,:,4) + J(3,4)*pi*Iz(:,:,3)*Iz(:,:,4);
H = 2*Hcs + 2*Hcoup
```

Recall each read out signal requires two measurements, the x-magnetization and the y-magnetization. For the four qubit example these magnetizations for some state $\rho$ evolving with $U$ are given by the trace of the state multiplied by the corresponding measurement operator.

\[
M_x(t) = tr(U(t)\rho U(t)\dagger(X_1 + X_2 + X_3 + X_4))
\]

\[
M_x(t) = tr(U(t)\rho U(t)\dagger(Y_1 + Y_2 + Y_3 + Y_4))
\]

Matlab operates in discrete steps, so acquiring these magnetizations will require a loop that takes the magnetization at discrete intervals. Such a loop is provided below in intervals of 30 microseconds for a total time of 90000 microseconds. Recall in NMR quantum states are described as
density matrices, in the below example rho is in the input matrix.

**Matlab input**
for t = 0 : 30e-6 : 3000*30e-6  
U = expm(-1i*H*t);  
measure_X = sum(Ix,3);  
x(j) = trace((U*rho*U')*measure_X)*(exp(-t/.2));  
j = j+1;  
end

The exponential term is added for $T_1$ relaxation of the signal. The more involved $T_2$ relaxation will be dealt with later. The x-component of the signal can be retrieved by graphing the real component of the magnetization over time. The information of interest though is the frequency of the different signals, so before plotting a Fourier transform is taken.

**Matlab input**
plot(real(fft(x)))

The same procedure was used except with a different measurement operators to find the magnetization in the y component. To be able to measure both simultaneously, the second measurement operator can be multiplied by the imaginary value $i$ and two plots can be taken, one of the real component of the signal and one of the imaginary. This is done in Matlab by changing the measurement operator and the plot command

**Matlab input**
Measure = sum(Ix,3)+1i*sum(Iy,3);  
...  
subplot(2,1,1),plot(real(fft(x)));  
subplot(2,1,2),plot(imag(fft(x)));

An examples of suchs plots result in Matlab for the $X_1$ state is given in figure 3.14.

### 3.2.2 Applying gates

In NMR gates are applied with pulses which cause rotations around the axis of application. The transformation that describes such an application is given by $e^{-i\sigma\theta}$. Where $\sigma$ is the axis we're
Figure 3.14: An example of a measurement readout in Matlab.
rotating around and $\theta$ is the angle of rotation. The angle of rotation depends on the strength of the pulse and duration of time it is applied for. These rotations can be implemented in Matlab as such

```
Matlab input
X1_90 = expm(-1i*IX(:,:,1)*(pi/2))
Y3_45 = expm(-1i*IY(:,:,4)*(pi/4))
X1_90*rho*X1_90'
```

This can also be done for two qubit evolutions, particularly the effect of j-coupling which can be simulated as

```
Matlab input
Coup12_90 = expm(-1i*Iz(:,:,1)*Iz(:,:,2)*(pi/2));
```

Another gate that will be necessary for the encoding is the Hadamard. In NMR QIP the Hadamard gate can be represented as a sum of two pulses.

```
Matlab input
Y1_90 = expm(-1i*kron(I,Y)*(pi/2))
X1_180 = expm(-1i*kron(X,I)*(pi))
H = X1_180*Y1_90
```

### 3.2.3 Pseudo-pure state preparation

In NMR experiments samples begin in the thermal state and need to be prepared to a pseudo-pure state as described in the last section. The pseudo-pure preparation outlined in figure 3.13 is modified for a 2 qubit system in figure 3.15.

The pink bars represent the application of gradients. Application of the gradient in Matlab takes some consideration of what is physically happening when a gradient is applied. The spatial averaging method is essentially slicing the sample into thin slices that are experiencing slightly different magnetic field and thus have slightly different precession frequency. These differences in precession frequencies average out over time such that they cancel out their net magnetization in the x and y direction.

Constructing an operator that does this requires two concepts, *transition* and *coherence*. A tran-
position is the location of an entry in a matrix, for a two qubit examples this is given by $|ab\rangle\langle cd|$ where $a, b, c$ and $d$ can be 0 or 1. Each transition has a coherence which is given by the equation $\Delta(ab, cd) = (a - c) + (b - d)$, for example $|01\rangle\langle 00|$ has a coherence of 1. These coherences describe how that transition in a density matrix evolves under the presence of a gradient field. The coherences for each possible transition can form a matrix. For the two qubit case this matrix is

$$
\begin{bmatrix}
0 & -1 & -1 & -2 \\
1 & 0 & 0 & -1 \\
1 & 0 & 0 & -1 \\
2 & 1 & 1 & 0 \\
\end{bmatrix}
$$

Each coherence matrix has a corresponding matrix that implements the gradient. The gradient matrix is composed entirely of 0’s and 1’s, with a 0 everywhere the coherence matrix is non-zero and 1 everywhere the coherence matrix is zero. For the two qubit case the gradient matrix is.

$$
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 \\
0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
$$

The application of gradients in Matlab is done by entry wise matrix multiplication by the gradient matrix, applied to one side of the density matrix.

Matlab input

\begin{verbatim}
G = [1 0 0 0 ; 0 1 1 0 ; 0 1 1 0 ; 0 0 0 1]
G.*rho
\end{verbatim}

The gradient matrix used in the crotonic acid simulation
The only operation from the pseudo-pure preparation that has not been described is the cos operation. This can be done with respect to any axis, we will arbitrarily choose the Y axis for our example. The operator is $e^{-i(IY)\cos^{-1}(\theta)}$. The Matlab input for the $\cos^{-1}(1/2)$ is provided below and is also applied in conjugation.

**Matlab input**

```matlab
Cos1 = expm(-1i*Iy(:,:,2)*(acos(1/2))
```

The entire psuedo-pure preparation procedure can now be simulated. The thermal state is given as the sum of all $Z_i$, this will be our starting state for the pseudo-pure preparation. If the pseudo-pure preparation is done correctly the measurement operator should produce no signal. This is since in the pseudo-pure state the magnetization is entirely in the z-axis. To detect the pseudo pure state a rotation is required on the final state to rotate the signal into the xy plane to be able to detect the signal. What the signal looks like depends on which qubit was rotated into the xy plane. Figure 3.16 shows what the pseudo pure state will look like for crotonic acid depending on which qubit was rotated.

The signal from each qubit is a single peak and a slightly different frequency. The code for the pseudo-pure preparation for 4 qubits is provided below.

**Matlab input**

```matlab
% Define operators for pseudopure state preparation
Cos2 = expm(-1i*Iy(:,:,2)*(acos(1/2)))
Cos3 = expm(-1i*Iy(:,:,3)*(acos(1/4)))
```
Figure 3.16: The x and y magnetizations of each qubit in crotonic acid when rotated.

\[
\begin{align*}
\text{Cos4} & = \exp(-1i * \text{Iy}(::,4) \ast (\text{acos}(1/8))) \\
\text{Z1Z2} & = \exp(-1i * (\text{Iz}(::,1) + \text{Iz}(::,2)) \ast (\pi/2)) \\
\text{Z2Z3} & = \exp(-1i * (\text{Iz}(::,2) + \text{Iz}(::,3)) \ast (\pi/2)) \\
\text{Z3Z4} & = \exp(-1i * (\text{Iz}(::,3) + \text{Iz}(::,4)) \ast (\pi/2)) \\
\text{X1} & = \exp(-1i * \text{Ix}(::,1) \ast (\pi/2)) \\
\text{X2} & = \exp(-1i * \text{Ix}(::,2) \ast (\pi/2)) \\
\text{Y1neg} & = \exp(-1i * \text{Iy}(::,1) \ast (\pi/4)) \\
\text{Y2neg} & = \exp(-1i * \text{Iy}(::,2) \ast (\pi/4)) \\
\text{X1b4} & = \exp(-1i * \text{Ix}(::,1) \ast (\pi/4)) \\
\text{X2b4} & = \exp(-1i * \text{Ix}(::,2) \ast (\pi/4)) \\
\text{X3b4} & = \exp(-1i * \text{Ix}(::,3) \ast (\pi/4)) \\
\text{Y1b4neg} & = \exp(-1i * \text{Iy}(::,1) \ast (\pi/4)) \\
\text{Y2b4neg} & = \exp(-1i * \text{Iy}(::,2) \ast (\pi/4)) \\
\text{Y3b4neg} & = \exp(-1i * \text{Iy}(::,3) \ast (\pi/4)) \\
\text{G} & = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \end{bmatrix} \\
\text{rho1} & = \text{G} \ast (\text{Cos4} \ast (\text{Cos3} \ast (\text{Cos2} \ast \text{rho} \ast \text{Cos2'}) \ast \text{Cos3'}) \ast \text{Cos4'}) \\
\end{align*}
\]
rho2 = G.*((Y1neg*(Z1Z2*(X1*rho1*X1')*Z1Z2')*Y1neg');
rho3 = G.*((Y2neg*(Z2Z3*(X2*rho2*X2')*Z2Z3')*Y2neg');
rho4 = G.*((Y3b4neg*(Z3Z4*(X3b4*rho3*X3b4')*Z3Z4')*Y3b4neg');
rho5 = G.*((Y2b4neg*(Z2Z3*(X2b4*rho4*X2b4')*Z2Z3')*Y2b4neg');
rho6 = G.*((Y1b4neg*(Z1Z2*(X1b4*rho5*X1b4')*Z1Z2')*Y1b4neg');

Figure 3.17 blows up the individual peaks to show the shape of each of the peaks before and after the procedure.

### 3.2.4 Pulse for applying the CNOT

Recall from the introductory chapter that the CNOT in NMR is implemented by regulating the natural evolution of two qubits due to j-coupling. Figure 3.18 illustrates this process. This process that simulates the CNOT is represented by the pulse sequence in figure 3.19. (In this sequences the block connecting two qubits represents the coupling).
The effect of the CNOT can be seen in each step of the sequence from the perspective of the bloch sphere and the output signal, provided in figure 3.20.

As we would expect, the first and last read out show no signal (note, the scale on the last signal is 10^{-10}) since the qubit is entirely in the z-axis).

### 3.2.5 Simplifying the pulse sequence

Circuit synthesis is a key step before implementing any quantum circuit. With NMR even a maximally synthesized circuit can be simplified once written as a pulse sequence. This is exemplified with the circuit that is being implemented in simulation in this thesis.

The technique outlined in section 1.4 for encoding circuits produces the circuits shown in figure 3.21. To be able to use crotonic acid the circuit needs to be reorganized such that the couplings only exist between neighbouring qubits, since crotonic acid only has strong coupling between neighbouring nuclei. By reordering qubits and making use of a switch gate this can be achieved with the addition of two more CNOT gates, this is demonstrated in figure 3.22.
Figure 3.20: Side by side visual of the read out signal with the bloch sphere representation of a qubit undergoing the CNOT procedure in NMR.

Figure 3.21: The resulting encoding circuit by using the method in section 1.4.
The qubits are first reordered, $q_1 \rightarrow q_2 \rightarrow q_4 \rightarrow q_3 \rightarrow q_1$. The control control gate is broken into one gate which only communicates with neighbouring qubits and another which interacts with a distant one. In the third step a swap gate is inserted to swap qubits 2 and 3, which moves all control gates to be between neighbouring qubits. Finally the redundant sequence of CNOT gates is removed ($\text{CNOT}^2 = I$). This can now be implemented as an NMR pulse sequence and then simplified further.

First, redundant sequences of gates, such as Y90 followed by Y-90, can be removed from throughout the circuit.

Then all Z operators can be eliminated by moving the Z operators to the beginning of the circuit by using commuting relationships. Z operators at the start of a circuit can be ignored since the Z
operator acting on an initial pure state will have no effect. The Z operator also commutes with the j-coupling operator.

\[ Y(\theta_1)Z(\theta_2) = Z(\theta_2)Z(-\theta_2)Y(\theta_1)Z(\theta_2) \] (3.1)

\[ Z(-\theta_2)Y(\theta_1)Z(\theta_2) = (\alpha X + \beta Y)(\theta_1) \] (3.2)

Using this the initial gate on qubits 1 and 3, \( Y(90)Z(-180) \) can be rewritten as \( Z(-180)Y(-90) \). Since the Z is now at the beginning of the circuit it can be dropped.

By continuing to use a combination of moving Z gates over j-couplings and using commuting relationships to reorder gates the circuit can ultimately be simplified to the final form. This simplified pulse sequence can now be simulated. In the above simulations angles of rotations were manually entered. In the lab setting these pulse sequences are implemented by adjusting the duration and strength of the pulse sequence. Also in the above simulations the pulse sequences were constantly

- [Diagram of pulse sequence]

Figure 3.25: The hybrid pulse sequence once Z operators are moved past coupling gates.

Some Z operators can be removed immediately, for example on the 2nd and 4th qubit in the above circuit they Z operators cancel one another. We can move the remaining Z operators across other pulses by using equations 3.2 and 3.1 For \( \alpha = \cos -\theta_2, \beta = \sin -\theta_2 \)

\[ Y(\theta_1)Z(\theta_2) = Z(\theta_2)Z(-\theta_2)Y(\theta_1)Z(\theta_2) \] (3.1)

\[ Z(-\theta_2)Y(\theta_1)Z(\theta_2) = (\alpha X + \beta Y)(\theta_1) \] (3.2)

Using this the initial gate on qubits 1 and 3, \( Y(90)Z(-180) \) can be rewritten as \( Z(-180)Y(-90) \). Since the Z is now at the beginning of the circuit it can be dropped.

- [Diagram of simplified pulse sequence]

Figure 3.26: Fully simplified pulse sequence.
applied, in the LAB setting the applied pulses will have a shape which can be optimized. These factors are addressed in the following subsection.

3.2.6 Simulating with experimental time

The next layer to add to the Matlab code to better simulate the actual lab experience is the simulate the circuit in experimental time. The first step in doing so is implementing the gradient ascent pulse engineering (GRAPE) method.

**GRAPE**

The pulses that are implemented in an NMR have unique shapes. The shape of the pulse will influence the coherence transfer between coupled spins and the relaxation effects in a given coherence transfer step. So far the MATLAB gates that have been used have been highly idealized. In the actual lab there is never complete coherence transfer in a pulse application and there is never no relaxation between pulses. The GRAPE procedure optimizes the pulse shape to maximize coherence transfer and minimize the relaxation effects for a pulse that can actually be implemented in the lab.

Many codes already exists for implementing the GRAPE procedure. This paper uses one developed by Hemant Katiyar. The code requires an input file describing the molecule being used and the pulse being simulated. For this experiment one preexisting input file was used for crotonic acid and the gate was edited between each run of the code. Execution of the code will provide the following readout when completed.

```
Matlab input
rungrape('Croton_Acid') ans =
GRAPEnname: 'Crotonic_90x1'
Date: '26-Jul-2017'
spinlist: 4
nspins: 4
spinNumbers: 0.5000
```
m: 2
N: 200
del_t: 2.0000e-06
T: 4.0000e-04
Chem_shift: [-2.6353e+04 -3.8827e+03 -7.7493e+03 0]
Couplings: [4x4 double]
Hint: [16x16 double]
Hrf: [16x16 double] [16x16 double]
initdelay: 4.0000e-06
Utarg: [16x16 double]
Usim: [16x16 double]
rflNHrange: [3x1 double]
rflNHiwt: [3x1 double]
threshold: 1.0000e-15
u: [200x2 double]
IDEALfidelity: 0.9981
RFfidelity: 0.9921

Two things to test with the GRAPE procedure are the control variations and the robustness.
The control variations need to avoid having any sudden changes.

Matlab input
plotGRAPE(GRinfo)

Figure 3.27: Graph of the control variations of the grape pulse.

The above figure is an example of one that is sufficiently smooth. The green graph is the phase
and the red graph is the power. The other factor to check that the change in fidelity with respect to the homogeneity is parabolic. An example is provided in figure 3.28.

Matlab input
testRobust(GRinfo)

![Robustness Plot II Avg. Fidelity: 98.7%](image)

Figure 3.28: Testing the robustness of the grape pulse.

These grape pulses now take the place of the original pulses that were implemented in the matlab code. Each previous gate.

Matlab input
Y1 = expm(-1i*Iy(:,:,1)*(pi/2));

Is replaced with

Matlab input
load save_structure/Crotonic_90y1.mat; Y1 = GRinfo.Usim;

The GRAPE code needs to be run for each pulse that needed to be applied for the circuit and then implemented in the code for each pulse as shown above. Running the circuit with these GRAPE pulses will introduce simulated noise into the read out. The above procedure allows each single qubit pulse to be simulated in realtime in Matlab. For complete real time simulation the effect of the $Z_iZ_j$ coupling also needs to be simulated in real time. As outline in 1.1 the j-coupling evolution is not an applied pulse but instead is a natural
evolution of the system which is regulated to achieve the effect of a CNOT gate. This regulation is done by the timely application of 180 degree pulses, called π pulses. Essentially the process entails dividing the application time of the CNOT gate into equal partitions. Between each partition one or more π pulse is applied which changes the sign of some terms in the Hamiltonian. This is done in such a fashion such that by the end of the applied time all the Hamiltonian terms besides the one desired j-coupling term desired cancel with one another. Figure 3.13 shows examples for the 3 and 4 qubit case. Figure 3.29 and 3.30 depict each pulse sequence and a table showing the signs of each term during each partition of time.

To implement this pulse sequence a modified time evolution operator is required which simulates the evolution over the smaller partitions of time between pi pulses.

**Matlab input**

```
Time12 = expm(-1i*H/(J(1,2)/16));
```

`J(1,2)` was defined earlier as the coupling constant between qubits 1 and 2, and is an experimentally found value. This time delay gate is used for the j-coupling between qubits 1 and 2, and is applied between each pi pulse. Since the full duration of time, 1/2J, is being broken into 8 pieces
Figure 3.30: Possible pi pulse sequences for 4 qubits.
the gate is applied for $1/J_{16}$. The π pulses are loaded exactly as in the single qubits pulses using GRAPE and the appropriate $Z_i Z_j$ gate can be constructed.

```
Matlab input
load save_structure/Crotonic_180x12.mat; Pi12 = GRinfo.Usim;
load save_structure/Crotonic_180x3.mat; Pi3 = GRinfo.Usim;
load save_structure/Crotonic_180x4.mat; Pi4 = GRinfo.Usim;
load save_structure/Crotonic_180x12neg.mat; Pi12neg = GRinfo.Usim;
load save_structure/Crotonic_180x3neg.mat; Pi3neg = GRinfo.Usim;
load save_structure/Crotonic_180x4neg.mat; Pi4neg = GRinfo.Usim;
Z1Z2 = Pi4neg*Time12*Pi12neg*Time12*Pi3neg*Time12*Pi12*Time12*Pi4*Time12
*Pi12neg*Time12*Pi3*Time12*Pi12*Time12;
```

Figure 3.31 shows each encoded qubit being rotated into the xy plane. The top figure is before the GRAPE procedure and below it is the read out after. The fidelity between the signal before and after the GRAPE procedure is provided below the signals.

The fidelity of the final sequence can improved by reducing the time of experiment by reducing the number of pulses applied. To do so we can group together pulses that can be applied simultaneously. For example, the Y $- 90$ pulse on the third qubit and the X $- 90$ pulse on the fourth qubit can be applied simultaneously. To implement this the GRAPE procedure is run again with the input pulse being the product of the two pulses. The improvement in fidelity after grouping like pulses can be seen in figure 3.33.

The only operator that does not need to be converted to real time is the gradient operator. In the lab the application of the gradient takes a relatively negligible amount of time and for the purposes of simulations can be left in the previous matrix form. With the pulse sequence now being simulated in real time a final consideration is decoherence. The T1 decoherence, the relaxation of the nuclei to the z-axis, has been included in the read outs from the very beginning. The more challenging decoherence to implement is the T2 decoherence.
Figure 3.31: Read outs of the encoded qubits along with the fidelity over the GRAPE procedure.

Figure 3.32: Example of pulses that can be applied simultaneously.
3.2.7 Simulating with T2 decoherence

To simulate the effect of T2 decoherence requires a relaxation matrix. First consider the two qubit example, \( n = 2 \), in figure 3.34.

The matrix in figure 3.34 is a \( 2^n \) square matrix where each value is an \( n \)-tuple. The value of the ordered pair is the difference between the labels at the top and side of the matrix. These labels are the logical states written sequentially. This matrix is then used to create the relaxation matrix. If the ordered pair has a 1 in the first element and/or second element then that element in the corresponding relaxation matrix will have the term \( e^{-t/T_1} \) and/or \( e^{-t/T_2} \) respectively. Here \( T_1^1 \) and \( T_2^2 \) are experimentally determined constants. Additionally all the terms on the diagonal are \( e^{-t/T_1} \). Thus, for the two qubit case the relaxation matrix \( T_2 \) is.
For the experiment in this thesis the four qubit decoherence matrix is required. The full 16x16 matrix with up to 4 terms in each cell is too large to fit onto a page so instead the precursor matrix is provided.

![Matrix](image)

Figure 3.35: The precursor matrix for the decoherence matrix for the four qubit case.

The decoherence matrix is applied to each pulse at each time iteration. The strength of the pulse in the x and y direction at each time interval can be loaded from the GRAPE file. These values are used to add to the energy from the pulse to the internal Hamiltonian to create the overall Hamiltonian of the system and the pulse. This new Hamiltonian is used to determine the systems evolution and it is this pulse which the decoherence is applied to. This is done for each time interval and saved as one final evolution.

Matlab input

```matlab
Ux = GRinfo.u(:,1);
Uy = GRinfo.u(:,2);
Ufin = eye(16);
for k = 1:GRinfo.N
    % Matlab code...
end
```
Figure 3.36: The measured signal before and after T2 decoherence is applied.

\[ H = \text{GRinfo.Hinit} + \text{Ux}(k) \times \text{sum}(I_x,3) + \text{Uy}(k) \times \text{sum}(I_y,3); \]
\[ U = \expm(-1i \times H \times \text{GRinfo.del_t}); \]
\[ U_{90} = T2 \times U; \]
\[ U_{\text{fin}} = U_{90} \times U_{\text{fin}}; \]
end

Another adjustment to include at this point is that there is a 4 microsecond delay before and after every applied pulse in the lab. This can be applied after the T2 decoherence.

**Matlab input**

\[ U_{4s} = \expm(-1i \times H \times \text{GRinfo.initdelay}); \]
\[ U_{\text{fin}} = U_{4s} \times U_{\text{fin}} \times U_{4s}; \]

*Using the same pulse shape for different rotations*

Finally, in the lab setting different pulse shapes are not required to apply an a Y90 and X90 pulse for example. Both of these are the same pulse just applied in different direction. So the code can be modified to only use X pulses which are rotated using a code that rotates the operator. Codes like this for NMR already exist and were included in the code.
3.3 Preparing for implementation in an NMR lab

The hybrid code being simulated consists of the following pair of codewords, the quantum bit $q$ and classical bit $m$ are labeled in the code word as $|q|^m$.

**Code Word 1**

$|0\rangle_L^{(0)} = |0000\rangle + |1111\rangle$

$|1\rangle_L^{(0)} = |0011\rangle - |1100\rangle$

**Code Word 2**

$|0\rangle_L^{(1)} = |0101\rangle + |1010\rangle$

$|1\rangle_L^{(1)} = |1001\rangle - |0110\rangle$

3.3.1 Decoding circuit

Each codeword will be encoded, each error applied and then the appropriate decoding will take place. For the decoding we assume we know the position of the single qubit errors (1,2,3 or 4) but not the error type (X, Y or Z). There should then be four different decoding circuit corresponding to each error position. The four different decoding circuits are provided below.

For errors on qubit 1, circuit D1

![Diagram for qubit 1 error]

For errors on qubit 2, circuit D2

![Diagram for qubit 2 error]

For errors on qubit 3, circuit D3

![Diagram for qubit 3 error]
The decoding circuits were determined by first running the encoding circuit in reverse and seeing what state the second and fourth qubits are in. The first and third qubits correspond, respectively, to the classical and quantum bit of information being transmitted. From here we used control not and control phase gates, with the control attached to the ancillary qubits, to correct for errors. Different combinations of gates were tested until one was found which corrected for any Pauli error on a single qubit. The following tables provide a detailed summary of the results. In each case the qubit being transmitted is $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$. 
Errors on Qubit 4
Classical Bit = 0

| After encoding: $|\psi(0)\rangle = \alpha(|0000\rangle + |1111\rangle) + \beta(|0011\rangle - |1100\rangle)$ |
| After $X_4$: $\alpha(|0001\rangle + |1110\rangle) + \beta(|0010\rangle - |1101\rangle)$ |
| After encoding in reverse: $\alpha|0001\rangle + \beta|0011\rangle$ |
| After full decoding: $\alpha|0001\rangle + \beta|0011\rangle$ |
| State of qubit 1: | 0 |
| State of qubit 3: $\alpha|0\rangle + \beta|1\rangle$ |

| After $Y_4$: $i\alpha(|0001\rangle - |1110\rangle) + i\beta(-|0010\rangle - |1101\rangle)$ |
| After encoding in reverse: $-i\alpha|0100\rangle + i\beta|0110\rangle$ |
| After full decoding: $i\alpha|0100\rangle + i\beta|0110\rangle$ |
| State of qubit 1: | 0 |
| State of qubit 3: $\alpha|0\rangle + \beta|1\rangle$ |

| After $Z_4$: $\alpha(|0000\rangle - |1111\rangle) + \beta(-|0011\rangle - |1100\rangle)$ |
| After encoding in reverse: $-\alpha|0101\rangle + \beta|0111\rangle$ |
| After full decoding: $\alpha|0101\rangle + \beta|0111\rangle$ |
| State of qubit 1: | 0 |
| State of qubit 3: $\alpha|0\rangle + \beta|1\rangle$ |

Table 3.1: Outline of the decoding results for the errors on qubit 4 while transmitting classical bit 0.
Errors on Qubit 4  
Classical Bit = 1

<table>
<thead>
<tr>
<th></th>
<th>State of qubit 1:</th>
<th>State of qubit 3:</th>
</tr>
</thead>
<tbody>
<tr>
<td>After encoding:</td>
<td>$</td>
<td>\psi^{(1)}\rangle = \alpha(</td>
</tr>
<tr>
<td>After $X_4$:</td>
<td>$\alpha(</td>
<td>0100\rangle +</td>
</tr>
<tr>
<td>After encoding in reverse:</td>
<td>$\alpha</td>
<td>1001\rangle + \beta</td>
</tr>
<tr>
<td>After full decoding:</td>
<td>$\alpha</td>
<td>1101\rangle + \beta</td>
</tr>
<tr>
<td>State of qubit 1:</td>
<td>$</td>
<td>1\rangle$</td>
</tr>
<tr>
<td>State of qubit 3:</td>
<td>$\alpha</td>
<td>0\rangle + \beta</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>State of qubit 1:</th>
<th>State of qubit 3:</th>
</tr>
</thead>
<tbody>
<tr>
<td>After $Y_4$:</td>
<td>$i\alpha(-</td>
<td>0100\rangle +</td>
</tr>
<tr>
<td>After encoding in reverse:</td>
<td>$-i\alpha</td>
<td>1100\rangle + i\beta</td>
</tr>
<tr>
<td>After full decoding:</td>
<td>$-i\alpha</td>
<td>1100\rangle - i\beta</td>
</tr>
<tr>
<td>State of qubit 1:</td>
<td>$</td>
<td>1\rangle$</td>
</tr>
<tr>
<td>State of qubit 3:</td>
<td>$\alpha</td>
<td>0\rangle + \beta</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>State of qubit 1:</th>
<th>State of qubit 3:</th>
</tr>
</thead>
<tbody>
<tr>
<td>After $Z_4$:</td>
<td>$\alpha(-</td>
<td>0101\rangle +</td>
</tr>
<tr>
<td>After encoding in reverse:</td>
<td>$-\alpha</td>
<td>1101\rangle + \beta</td>
</tr>
<tr>
<td>After full decoding:</td>
<td>$\alpha</td>
<td>1101\rangle + \beta</td>
</tr>
<tr>
<td>State of qubit 1:</td>
<td>$</td>
<td>1\rangle$</td>
</tr>
<tr>
<td>State of qubit 3:</td>
<td>$\alpha</td>
<td>0\rangle + \beta</td>
</tr>
</tbody>
</table>

Table 3.2: Outline of the decoding results for the errors on qubit 4 while transmitting classical bit 1.
### Errors on Qubit 3

**Classical Bit = 0**

| After encoding: | $|\psi(0)^{(0)}\rangle = \alpha(|0000\rangle + |1111\rangle) + \beta(|0011\rangle - |1100\rangle)$ |
|----------------|----------------------------------------------------------------------------------|
| After $X_3$:   | $\alpha(|0010\rangle + |1101\rangle) + \beta(|0001\rangle - |1110\rangle)$ |
| After encoding in reverse: | $\alpha |0100\rangle + \beta |0110\rangle$ |
| After full decoding: | $\alpha |0100\rangle + \beta |0110\rangle$ |
| State of qubit 1: | $|0\rangle$ |
| State of qubit 3: | $\alpha |0\rangle + \beta |1\rangle$ |

| After $Y_3$: | $i\alpha(|0010\rangle - |1101\rangle) + i\beta(-|0001\rangle - |1110\rangle)$ |
| After encoding in reverse: | $-i\alpha |0001\rangle + i\beta |0011\rangle$ |
| After full decoding: | $i\alpha |0001\rangle + i\beta |0011\rangle$ |
| State of qubit 1: | $|0\rangle$ |
| State of qubit 3: | $\alpha |0\rangle + \beta |1\rangle$ |

| After $Z_3$: | $\alpha(|0000\rangle - |1111\rangle) + \beta(-|0011\rangle - |1100\rangle)$ |
| After encoding in reverse: | $-\alpha |0101\rangle + \beta |0111\rangle$ |
| After full decoding: | $\alpha |0101\rangle + \beta |0111\rangle$ |
| State of qubit 1: | $|0\rangle$ |
| State of qubit 3: | $\alpha |0\rangle + \beta |1\rangle$ |

Table 3.3: Outline of the decoding results for the errors on qubit 3 while transmitting classical bit 0.
### Errors on Qubit 3

**Classical Bit = 1**

<table>
<thead>
<tr>
<th>Step</th>
<th>State After Encoding</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>After encoding:</strong> $</td>
<td>\psi\rangle^{(1)} = \alpha(</td>
</tr>
<tr>
<td><strong>After $X_3$:</strong></td>
<td>$\alpha(</td>
</tr>
<tr>
<td><strong>After encoding in reverse:</strong></td>
<td>$\alpha</td>
</tr>
<tr>
<td><strong>After full decoding:</strong></td>
<td>$\alpha</td>
</tr>
<tr>
<td><strong>State of qubit 1:</strong></td>
<td>$</td>
</tr>
<tr>
<td><strong>State of qubit 3:</strong></td>
<td>$\alpha</td>
</tr>
<tr>
<td><strong>After $Y_3$:</strong></td>
<td>$i\alpha(</td>
</tr>
<tr>
<td><strong>After encoding in reverse:</strong></td>
<td>$i\alpha</td>
</tr>
<tr>
<td><strong>After full decoding:</strong></td>
<td>$i\alpha</td>
</tr>
<tr>
<td><strong>State of qubit 1:</strong></td>
<td>$</td>
</tr>
<tr>
<td><strong>State of qubit 3:</strong></td>
<td>$\alpha</td>
</tr>
<tr>
<td><strong>After $Z_3$:</strong></td>
<td>$\alpha(</td>
</tr>
<tr>
<td><strong>After encoding in reverse:</strong></td>
<td>$\alpha</td>
</tr>
<tr>
<td><strong>After full decoding:</strong></td>
<td>$\alpha</td>
</tr>
<tr>
<td><strong>State of qubit 1:</strong></td>
<td>$</td>
</tr>
<tr>
<td><strong>State of qubit 3:</strong></td>
<td>$\alpha</td>
</tr>
</tbody>
</table>

Table 3.4: Outline of the decoding results for the errors on qubit 3 while transmitting classical bit 1.
Errors on Qubit 2
Classical Bit = 0

| After encoding: | \( |\psi\rangle^{(0)} = \alpha(|0000\rangle + |1111\rangle) + \beta(|0011\rangle - |1100\rangle) \) |
|-----------------|----------------------------------------------------------------------------------|
| After \( X_2 \):| \( \alpha(|0100\rangle + |1011\rangle) + \beta(|0111\rangle - |1000\rangle) \)                              |
| After encoding in reverse: | \( \alpha|1001\rangle - \beta|1011\rangle \)                                               |
| After full decoding: | \( \alpha|0101\rangle + \beta|0111\rangle \)                                            |
| State of qubit 1: | \( |0\rangle \)                                                              |
| State of qubit 3: | \( \alpha|0\rangle + \beta|1\rangle \)                                                |

| After \( Y_2 \): | \( i\alpha(|0100\rangle - |1011\rangle) + i\beta(|0111\rangle + |1000\rangle) \) |
|-----------------|----------------------------------------------------------------------------------|
| After encoding in reverse: | \( -i\alpha|1100\rangle - i\beta|1110\rangle \)                                                |
| After full decoding: | \( -i\alpha|0000\rangle - i\beta|0010\rangle \)                                            |
| State of qubit 1: | \( |0\rangle \)                                                              |
| State of qubit 3: | \( \alpha|0\rangle + \beta|1\rangle \)                                                |

| After \( Z_2 \): | \( \alpha(|0000\rangle - |1111\rangle) + \beta(|0011\rangle + |1100\rangle) \) |
|-----------------|----------------------------------------------------------------------------------|
| After encoding in reverse: | \( \alpha|0101\rangle + \beta|0111\rangle \)                                            |
| After full decoding: | \( \alpha|0001\rangle + \beta|0011\rangle \)                                            |
| State of qubit 1: | \( |0\rangle \)                                                              |
| State of qubit 3: | \( \alpha|0\rangle + \beta|1\rangle \)                                                |

Table 3.5: Outline of the decoding results for the errors on qubit 2 while transmitting classical bit 0.
### Errors on Qubit 2

Classical Bit = 1

| After encoding: | $|\psi^{(1)}\rangle = \alpha(|0101\rangle + |1010\rangle) + \beta(|1001\rangle - |0110\rangle)$ |
|----------------|--------------------------------------------------------------------------------------------------|
| After $X_2$:   | $\alpha(|0001\rangle + |1110\rangle) + \beta(|1101\rangle - |0010\rangle)$ |
| After encoding in reverse: | $\alpha |0001\rangle - \beta |0011\rangle$ |
| After full decoding: | $\alpha |1101\rangle + \beta |1111\rangle$ |
| State of qubit 1: | $|1\rangle$ |
| State of qubit 3: | $\alpha |0\rangle + \beta |1\rangle$ |

| After $Y_2$: | $i\alpha(- |0001\rangle + |1110\rangle) + i\beta(|1101\rangle + |0010\rangle)$ |
| After encoding in reverse: | $i\alpha |0100\rangle + i\beta |0110\rangle$ |
| After full decoding: | $-i\alpha |1000\rangle - i\beta |1010\rangle$ |
| State of qubit 1: | $|1\rangle$ |
| State of qubit 3: | $\alpha |0\rangle + \beta |1\rangle$ |

| After $Z_2$: | $\alpha(- |0101\rangle + |1010\rangle) + \beta(|1001\rangle + |0110\rangle)$ |
| After encoding in reverse: | $\alpha |1101\rangle + \beta |1111\rangle$ |
| After full decoding: | $\alpha |1101\rangle + \beta |1111\rangle$ |
| State of qubit 1: | $|1\rangle$ |
| State of qubit 3: | $\alpha |0\rangle + \beta |1\rangle$ |

Table 3.6: Outline of the decoding results for the errors on qubit 2 while transmitting classical bit 1.
Errors on Qubit 1
Classical Bit = 0

<table>
<thead>
<tr>
<th>Operation</th>
<th>State Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>After encoding: $</td>
<td>\psi^{(0)}\rangle = \alpha(</td>
</tr>
<tr>
<td>After $X_1$:</td>
<td>$\alpha(</td>
</tr>
<tr>
<td>After encoding in reverse:</td>
<td>$\alpha</td>
</tr>
<tr>
<td>After full decoding:</td>
<td>$\alpha</td>
</tr>
<tr>
<td>State of qubit 1:</td>
<td>$</td>
</tr>
<tr>
<td>State of qubit 3:</td>
<td>$\alpha</td>
</tr>
<tr>
<td>After $Y_1$:</td>
<td>$i\alpha(</td>
</tr>
<tr>
<td>After encoding in reverse:</td>
<td>$i\alpha</td>
</tr>
<tr>
<td>After full decoding:</td>
<td>$i\alpha</td>
</tr>
<tr>
<td>State of qubit 1:</td>
<td>$</td>
</tr>
<tr>
<td>State of qubit 3:</td>
<td>$\alpha</td>
</tr>
<tr>
<td>After $Z_1$:</td>
<td>$\alpha(</td>
</tr>
<tr>
<td>After encoding in reverse:</td>
<td>$\alpha</td>
</tr>
<tr>
<td>After full decoding:</td>
<td>$\alpha</td>
</tr>
<tr>
<td>State of qubit 1:</td>
<td>$</td>
</tr>
<tr>
<td>State of qubit 3:</td>
<td>$\alpha</td>
</tr>
</tbody>
</table>

Table 3.7: Outline of the decoding results for the errors on qubit 1 while transmitting classical bit 0.
Errors on Qubit 1
Classical Bit = 1

| After encoding: | \( |\psi\rangle^{(1)} = \alpha(|0101\rangle + |1010\rangle) + \beta(|1001\rangle - |0110\rangle) \) |
|-----------------|----------------------------------------------------------------------------------|
| After \( X_1 \): | \( \alpha(|1101\rangle + |0010\rangle) + \beta(|0001\rangle - |1110\rangle) \) |
| After encoding in reverse: | \( \alpha |0100\rangle + \beta |0110\rangle \) |
| After full decoding: | \( \alpha |1100\rangle + \beta |1110\rangle \) |
| State of qubit 1: | |1\rangle |
| State of qubit 3: | \( \alpha |0\rangle + \beta |1\rangle \) |

| After \( Y_1 \): | \( i\alpha(|1101\rangle + |0010\rangle) + i\beta(-|0001\rangle - |1110\rangle) \) |
| After encoding in reverse: | \( -i\alpha |0001\rangle - i\beta |0011\rangle \) |
| After full decoding: | \( -i\alpha |1001\rangle - i\beta |1011\rangle \) |
| State of qubit 1: | |1\rangle |
| State of qubit 3: | \( \alpha |0\rangle + \beta |1\rangle \) |

| After \( Z_1 \): | \( \alpha(|0101\rangle - |1010\rangle) + \beta(-|1001\rangle - |0110\rangle) \) |
| After encoding in reverse: | \( -\alpha |1101\rangle - \beta |1111\rangle \) |
| After full decoding: | \( -\alpha |1101\rangle - \beta |1111\rangle \) |
| State of qubit 1: | |1\rangle |
| State of qubit 3: | \( \alpha |0\rangle + \beta |1\rangle \) |

Table 3.8: Outline of the decoding results for the errors on qubit 1 while transmitting classical bit 1.

As mentioned before to implement the circuit in NMR using crotonic acid will require control gates to only exist between neighbouring qubits since crotonic acid only has strong coupling between neighbouring nuclei. This means that the decoding circuits for errors on qubit 1, 2 and 3 will require the implementation of swap gates. These modified decoding circuits are provided below.

For errors on qubit 1, circuit D1
For errors on qubit 2, circuit D2

For errors on qubit 3, circuit D3

As was mentioned before to improve the decoherence the circuit is synthesized after its converted to a pulse sequence. Since this is done for the circuit as a whole, we can not design the encoding, errors and decoding separately but will need to design the system as a whole and synthesize it. This will require designing and synthesizing 12 (3 errors x 4 different locations) different pulse sequences.

First the encoding circuit is converted to a pulse sequence.

This pulse sequence is then subsequently synthesized.

From here each decoding circuit is converted into a pulse sequence. The full procedure is shown for one of the decoding circuits, the decoding circuit for errors on qubit 1. The decoding circuit for errors on qubit 1 is given below as a pulse sequence. The pulse sequence has been split into 2 images to fit onto a single page.
Once synthesized the decoding circuit for errors on qubit 1 is given by the following pulse sequence:

The following table summarizes the fidelity of the encoding circuit, each decoding circuit and each error after applying the GRAPE procedure.
Further procedures will need to be performed to improve the overall fidelity of the experiment before the pulse sequence can be implemented in NMR.
Chapter 4

Conclusions and Future Work

Throughout this thesis we outlined the characterization and construction of hybrid codes from a coding theory perspective and from the OAQEC perspective. We then showed that the coding theory construction is a special case of the broader OAQEC model. In the coding theory construction there exists three practical restrictions. These restrictions are that the errors are elements of the Pauli group, each codeword is a subspace of a Hilbert space as opposed to being a subsystem and that the dimension of each of the quantum codes in a hybrid code are equal. In this worked we presented a detailed review of NMR QIP and how it can be used to implement hybrid quantum codes, along with a method for simulating NMR QIP experiments in Matlab. We designed a hybrid code that transmits one qubit and one classical bit that detects any single Pauli error. We created a procedure for encoding quantum circuits, and used this procedure to create an encoding circuit for the codeword we designed. This encoding circuit, along with its corresponding decoding circuit was converted to a pulse sequence which was simulated in an NMR QIP simulation. With this work complete, the foundation has been laid to implement the hybrid code in a physical NMR QIP, which is an immediate next step in this research. From a mathematical perspective the discussion in this thesis has opened up other questions. One is build of the development of the hybrid quantum hamming bound to determining the hybrid equivalents of other quantum bounds, particularly the quantum singleton bound and the quantum Gilbert-Varshamov bound. As well, this work introduced one example of a hybrid code, future work can investigate the development of other hybrid codes which may potentially have better parameters than their quantum counterparts.
Bibliography


