EFFECTS OF ERRORS IN THE HAMILTONIAN PROGRAMMING PROCESS OF ADIABATIC QUANTUM COMPUTING

by

Aaron Whiteway

Submitted in partial fulfillment of the
requirements for the degree of
Master of Science

at

Dalhousie University
Halifax, Nova Scotia
August 2014

Table of Contents

List of Tables

List of Figures

Abstract

In order to measure the effects of these errors we measure of the likelihood an erroneous instance of a Hamiltonian shares the same ground state as the intended Hamiltonian (resilience). The effects of errors on the spectrum of the instantaneous Hamiltonian through the evolution of the system is studied through the instantaneous gap as well as the minimum.

For this work a simplistic model is used. An Ising Spin Glass to model the adiabatic quantum computer. The structure of the computers spins and allowable couplings are taken to be in accordance with the D-Wave architecture. The errors are modeled with Gaussian distributions.

The model allowed for a simple scaling relation to be extracted for resilience. It was observed that resilience drops quickly as system size grows; although, computational complexity limited the study of larger systems. A minimal effect on the evolution process itself was observed through the simulations.

List of Abbreviations and Symbols Used

GS_{H_i}	Ground state of Hamiltonian labelled by i
H_i	Hamiltonian labelled by i
J_{ij}	Coupling term between i and j
$N(\sigma)$	Random number from Gaussian distribution of width σ
N_{spins}	Number of spins in system
N_{terms}	Number of field and coupling terms in Hamiltonian
σ	Width of Gaussian distribution
σ_i	Pauli spin matrix for particle i
b_i	Field term on i
d	Distance between Hamiltonians given by a metric
AQC	Adiabatic Quantum Computing
c	Cross over
NP	Class of problems not easily solved, but easily verified
P	Class of problems easily solved, and easily verified
R	Resolution
r	Resilience
S	Eigenstate

Acknowledgements

I would like to thank Dr. Jordan Kyriakidis for his excellent support throughout this project. I also would like to thank my mom and family for their support over these past few years.

Chapter 1

Introduction

Computers are ubiquitous in today's society existing in nearly every home and in most pockets in the form of smart phones. Tasks we assign to computers vary vastly. Examples of everyday uses are email and social media, while less common, more fantastic uses include simulations of the universe [9] and simulations of quantum systems [33]; in this world, the world of large computations we find ourselves classifying problems based on their difficulty. The field devoted to the difficulty of problems is Computational Complexity Theory. Computational Complexity Theory is concerned with the difficulty of computation which is not the same as algorithmic difficulty. The difficulty of an algorithm is quantified in terms of the number of how the number basic operates scales as we increase the size of the input. Computation difficulty is different in that it does not pertain to the particular instance of the problem but the problem as a whole; to clarify, "What is 15 * 3?" is not in the realm of complexity theory however "Is 15 * 3 = 45?" is. By restricting the types of problems to decision problems that involve a binary answer computational complexity theory circumvents the need to choose a specific algorithm allowing the best possible algorithm to be chosen [5]. There are essentially two classes of problems that are of concern for us. Both are a subset of deterministic algorithms. A deterministic algorithm is one that must execute the exact same given the same input. The two classes of interest are:

P The class of problems that are efficiently solvable and verifiable. The number of basic operations scales polynomially with input size for any (known) algorithm. An example would be integer multiplication.

NP The class of problems that are efficiently verifiable but not efficiently solvable. The number of basic operations scales exponentially with input size for any (known) algorithm; however, to verify the solution the number of basic operations scales polynomially with input size for any (known) algorithm. An example would be integer factorization.

For more on computational complexity theory refer to [5].

Computers have been growing in power at an exponential rate in accordance to Moore's law; however, this trend cannot last forever [21] as thermal and quantum effects will begin to dominate as the size shrinks; therefore, growing computer power cannot save us from the immensity of NP problems. The limitation is caused by leakage errors inside the transistor. Leakage means that some of the charge used to stop current with the transistor leaks inside the transistor causing it to malfunction. Since many problems of interest fall into the NP category we are stuck with the problem of developing a computer capable of doing these calculations. Many believe the answer to this problem may lie in Quantum Computing. The first example of an algorithm being able to solve an NP problem in sub NP time is Shor's algorithm for factorization and discrete logarithms [47]. Shor's algorithm is designed for a gate model quantum computer, discussed in Section 1.3, which was experimentally realized on a small scale [49]. Problems, such as short decoherence time of the quantum dots, have limited the scale of these machines; although, new techniques are being developed to extend the decoherence time [52] these computers are still mainly theoretic with many obstacles to over come. Another form of quantum computing known as Adiabatic Quantum Computing, discussed in Section 1.4, offers an alternative to gate model quantum computing with its own set of challenges.

1.1 Fundamentals

In quantum computing the fundamental unit of the computer, the bit, is modified to take on a key property of quantum mechanics, superposition. In classical computers a bit is either 1 or 0 at anytime throughout the computation; however, in quantum mechanics until you measure the bit, which we will now call qubit, it is in a superposition of both states given by

$$|\Psi_{qubit}\rangle = \alpha|0\rangle + \beta|1\rangle \tag{1.1}$$

where as usual in quantum mechanics α and β are complex numbers obeying the normalization relation imposed by $|\alpha|^2$ and $|\beta|^2$ being probabilities:

$$|\alpha|^2 + |\beta|^2 = 1 \tag{1.2}$$

Qubits can become entangle when their combined state is not separable into multiplication of states just involving individual qubits and in Equation 1.3.

$$|\Psi_{entangled}\rangle = \alpha|01\rangle - \beta|10\rangle \tag{1.3}$$

In quantum computing calculation requires the qubits to be in a superposition and entangled throughout the calculation. This requirement can allow for all possible branches of the calculation to be considered simultaneously such as in Shor's algorithm [47] for a gate model quantum computer.

1.2 Decoherence

Coherence is an ideal property of waves that enables stationary interference. Decoherence is the loss of these interference properties due to various causes. Thermal energy being introduced or removed through stochastic processes bringing the system to thermal equilibrium over time scale T_1 (average time between energy level occupation changes) can cause the system to decohere; furthermore, energy can be borrowed from the environment causing the phases to change and the system to decohere over time scale T_2 (average time between phase changes)[22]. In most cases $T_1 \gg T_2$ (decoherence caused by a single source) which means that T_2 is more important in quantum computing [22].

1.3 Gate Model Quantum Computing

In the gate model of quantum computing quantum equivalents are made of the classical components of a computer, such as the processor, memory, and bus. The processor now works by making modifications on quantum states in place of electrical voltages. The modifications are made through operators as one would expect and can be written down in matrix form. One example of a quantum gate is the Pauli-X gate which acts on a single spin and is the quantum equivalent of a not gate:

$$\text{Not} \equiv \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \tag{1.4}$$

which you may recognize as the σ_x Pauli spin matrix. The remaining two Pauli matrices act as rotations by π about their respective axis. An example of a two spin gate would be the controlled not gate:

$$\text{CNot} \equiv \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{1.5}$$

These transformations on the state constitutes the processor of the quantum computer. Unlike classical logic gate, the nature of quantum mechanics require that each gate be reversible, meaning that each set of inputs uniquely maps to a set of outputs. Gates with these properties are constantly in development and recently a full adder with overflow detection was developed [15]. In classical computing an arbitrary classical function can be described by a set of And, Or, and Not gates properly strung together. For quantum computing a similar set exists which consists of CNot, phase, $\frac{\pi}{8}$, and Hadamard gates [37].

$$\text{Hadamard} \equiv \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{1.6}$$

$$\text{Phase} \equiv \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \tag{1.7}$$

$$\frac{\pi}{8} \equiv \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix} \tag{1.8}$$

By most definitions in order to have a computer two more aspects are required, a memory to store information before and after it is processed and a bus to transfer information throughout the computer. Quantum memory is typically comprised of atoms, quantum dots, or superconducting junctions such as Josephson junctions [22]. A typical quantum bus involves photons or phonons [33]. Problems with scaling the gate model computer architecture are caused by speed limitations and decoherence [33]. There are several approaches to this style of quantum computing that each use a different unit to be their base for computation.

1.3.1 Photons

The polarization state of photons can be used as a qubit and photons have the advantage that they are relatively free from the decoherence that exists in the other techniques [22]. The one-qubit gates such as those of Equations 1.6, 1.7, and 1.8 can be achieved through the use of waveplates comprised of birefringent material. A technique known as coherent photon conversion based on interacting bosonic fields promises to be able to allow for deterministic multiqubit entanglement gates, high-quality single and multi photon states free from higher-order imperfections, as well as high-efficiency detection [23].

1.3.2 Trapped Atoms

Due to the excellent coherence time of certain energy levels within atom one can construct reliable qubits with coherence times on the order of seconds or longer; furthermore, interactions between atoms can provide entangling quantum gates. qubits can be initialized through optical pumping at nearly 100% efficiency. [22]

The trapped atomic ions can be spatially manipulated with nanometre precision using electric fields [6, 41]. Trapped ions can be entangled through laser-induced coupling of spins [24]. Experimentally up to 8 ions have been entangled with this method [6]. Scaling the trapped ion system becomes difficult when large numbers of ions undergo collective motion. This can be caused by inefficient laser-cooling, increased susceptibility to noisy electric fields, and decoherence of the motional modes [22]. One approach to avoid these issues is to shuttle individual atoms through the system so that entangling gates only need to operate with a small number of ions [19]. Alternatively one can couple the qubits via photonic interactions; in-fact, atomic ions have been entangled over macroscopic distances with this method [38].

Neutral atoms can also be used as qubits similar to ionized atoms. The neutral atoms are confined in space through a set of lasers forming an optical lattice [35]. Interactions between atoms can be achieved through the large electric dipole moment of Rydberg states or bringing the atoms closer together. The main difficulties of this approach is the initialization, interaction, and measurement of the atomic qubits [22].

1.3.3 Nuclear Magnetic Resonance

Nuclear spins molecules in liquid solutions have a rapid molecular motion that acts as a gyroscope causing increased stability which extends their coherence times to seconds, comparable with trapped atoms [22]. When immersed in a strong magnetic field nuclear spins can be identified by their Larmor frequencies. Atoms in a molecule have different Larmor frequencies depending on their local molecular bonds. Single qubit gates can be obtained through resonant radio-frequency pulses; while two qubit interactions come from indirect coupling controlled by molecular electrons [22]. A large coil surrounding the sample allows for measurement. Manipulation of a system of a dozen qubits has been experimentally demonstrated [36].

1.3.4 Quantum Dots

Quantum dots remove the need to trap atoms because they are integrated into a solid state host. Quantum dots are artificial atoms that occur when a small semiconductor nanostructure binds an electron or whole into a localized potential with discrete energy levels [22, 50]. Quantum dots can be manufactured in various ways. One method is through controlled voltages on lithographically defined metallic gates; alternatively, quantum dots can self assemble through a stochastic process on a semiconductor that creates the potential for confining electrons and holes. The former quantum dot can operate at temperatures below 1 Kelvin and electronically controlled while the latter operate around 4 Kelvin and optically controlled [22].

It is now possible to make quantum dots with no variation in size, shape, and orientation by assembling the dots one atom at a time [14]. This is achieved by using a scanning tunneling microscope to move Indium atoms to precise locations on an InAs substrate. Although problems with decoherence, atomic-scale gate control, ultrafast read-out and cost efficient scalability still remain [50]. The stochastic self assembly process does not share this benefit as the random locations and optical properties cause them to be inherently difficult to work with [22]; however, they can be controlled optically. Decoherence time of dots are typically on the order of milliseconds.

1.3.5 Superconductors

A superconducting qubit is formed of an inductor, capacitor, and Josephson junction. A Josephson junction is a thin insulating layer separating sections of a superconductor. The behaviour of the qubit is defined by the ratio $\frac{E_J}{E_C}$, where $E_C = \frac{e^2}{2C}$ is the single electron charging energy of the capacitor and E_J is the Josephson energy which is the the energy accumulated in a Josephson junction when a supercurrent flows through it [22]. When the charging energy exceeds the Josephson energy ($\frac{E_J}{E_C} \ll 1$) the quantum states of the device approach charge states, the resulting qubit is known as a charge qubit [39]. When the charging energy is kept small you can either have a flux or phase qubit. A flux qubit has a persistent current and a double well potential, the two minima correspond to the current going in opposite directions [11]. Each direction corresponds to a state, either 1 or 0. The depths of each well can be altered using magnetic flux to bias current flow in one direction or another. In a phase qubit the potential of the wells is biased so that the qubit uses the two lowest lying eigenstates in a single well [28]. The excitation energy of these qubits is typically designed between 5-10 GHz to avoid thermal effects at their operating temperature of around 10 mK [22]. Theoretically the decoherence time of superconducting qubits was calculated to be on the order of milliseconds [39]; however, initially their decoherence time was on the order of nanoseconds which has grown to microseconds over recent years [22, 53].

Neighbouring flux qubits inductively couple naturally [27]; however, to control the coupling between flux qubits one must introduce a time-dependant magnetic flux (TDMF) [26]. The approach works by initially detuning the qubits to create a non-resonant interaction. When a coupling is desired a time-dependant magnetic flux (TDMF) is applied to one of the flux qubits to counteract the initial detuning and couple the flux qubits [26]. Evidence was also presented for entanglement between coupled flux qubits [20]. It is possible to manufacture flux qubits that are scalable and robust against against fabrication variations in the Josephson junctions [17]. The flux qubits used by DWave are rf-SQUID qubits which means they are comprised of a single Josephson junction [7, 17]. Another form of flux qubit that uses 3 or more Josephson junctions is known as as PC qubit [34, 40].

1.4 Adiabatic Quantum Computing

Adiabatic Quantum Computing relies on the quantum adiabatic theorem which states that if a Hamiltonian changes sufficiently gradually from an initial Hamiltonian, H_i, to a final Hamiltonian, H_f, and the system was in the n^{th} state of $H_i \equiv H_0$ it will also be in the n^{th} state of $H_f \equiv H_p$ [16], provided there are no crossing energy levels. The theorem was developed in 1928 by Max Born and Vladimir Fock. In 2004 the validity of the adiabatic theorem was called into question [29]. The issue was not corrected until 2008 when the inconsistency was explained as a relic of the difference in the time derivatives of the solutions obtained from the two approaches [51]. It was argued that the state vector was physically relevant and not its time derivative, validating the quantum adiabatic theorem. A proof of the quantum adiabatic theorem is provided in Section 3.1.

An adiabatic computer works by using a collection of spin particles, typically spin-$\frac{1}{2}$. The particles are then subject to a Hamiltonian that is easily prepared, such as all spins in the +x direction. The Hamiltonian is then evolved adiabatically to a final Hamiltonian that encodes the desired logic in its ground state(s) and H_0 and H_p do not commute.

$$H(s) = f(s)H_0 + [1 - f(s)]H_p \tag{1.9}$$

where s is reduced time and $f(s)$ ranges from 0 to 1.

H_p can be constructed from smaller Hamiltonian's gluing their logic together [30]. For instance an AND gate, $A \wedge B \to C$, could be made with the Hamiltonian

$$H_{AND} = -A - B + 2C + AB - 2AC - 2BC \tag{1.10}$$

which has the ground states of Table 1.1. By fixing A and B the value of C becomes

A	B	C
1	1	1
1	0	0
0	1	0
0	0	0

Table 1.1: Ground states of H_{AND}

limited. Chaining these operations together becomes simple, two ANDs have the Hamitlonian

$$H_{2AND} = -A - B + C - D + 2E + AB - 2AC - 2BC + CD - 2CE - 2DE \quad (1.11)$$

the ground states then becomes that of Table 1.2 by the gluing theorem [30]. Table 1.2 refers to the equation $A \wedge B \wedge D \rightarrow E$ so it logically reduces to Table 1.3. This is because C is an auxiliary spin, meaning it is not an input or output spin.

A	B	C	D	E
1	1	1	1	1
1	0	0	1	0
0	1	0	0	0
0	0	0	0	0

Table 1.2: Ground states of two H_{AND} combined

A	B	D	E
1	1	1	1
1	0	1	0
0	1	0	0
0	0	0	0

Table 1.3: Ground states of two H_{AND} combined with the carrier spin removed.

The potential of adiabatic quantum computers has been a topic of debate. It was shown that adiabatic quantum computing is equivilent to the gate model of quantum computing within a polynomial factor [1, 32]. In order to have a sparse matrix we require that the Hamiltonian be limited to n-body interactions where $n < N$ where N is the number of spins in the network. It has been shown that an adiabatic quantum computer could speed up for searching algorithms by a quadratic factor; furthermore, one cannot rule out polynomial complexity algorithms for traditionally non-polynomial complexity problems using the usual query complexity algorithms [48]. The reason for this is because there are many possible paths, $f(s)$, to transition between the initial and final Hamiltonian, each resulting in a their own difficulty [44, 48]. The result of having many choices could be that the search space requires non-polynomial time to find a function $f(s)$ whose execution will provide polynomial complexity on an adiabatic quantum computer [12]. It has further been argued that proving that you cannot find a suitable Hamiltonian for a given problem in polynomial time is infeasible [12].

Adiabatic quantum computing is naturally robust against certain unitary perturbations and their perturbations were shown to sometimes even increase the success rate of calculations [3, 10]. A successful calculation returns a ground state state of the Hamiltonian, an implication of this is that an algorithm could remain scalable as long as the natural frequencies of the Hamiltonian are much larger than the frequencies contained in the noise [45]. One feature required to further increase the success rate of adiabatic quantum computing is fault tolerance. Lidar introduced a scheme using dynamical decoupling, subsystems, stabalizer codes, and the energy gaps to move towards fault tolerance in 2008 [25]. In 2014 Mizel presented a hands off approach to fault tolerance that is capable of scalable universal quantum computation in a non-degenerate ground state, as well as fault-tolerant against a variety of noise [31].

The nature of adiabatic quantum computation, the smooth transition and finite energy gap between the ground state and the first excited state, allows the ground state to maintain its coherence properties much longer than the individual qubits [4]; furthermore, it has been shown that AQC maintains its properties even in for general non-Markovian noise larger than the minimum gap [2]. This increased coherence makes AQC easier than the gate model of quantum computing; furthermore, evidence has been presented to suggest that during a critical portion of quantum annealing the qubits on D-Wave's chip becoming entangled and maintain entanglement as the system reaches equilibrium [24]. Testing a quantum computer to ensure it behaves quantum mechanically is no easy task, though the concept was explored [57] and recently there has been evidence to suggest that D-Waves chip is in fact a quantum annealing chip [8]. The D-Wave chip uses flux qubits, which were discusses in 1.3.5.

1.5 Project Description

This study is focused on taking the theoretic Hamiltonian from an idea to a physical entity on the hardware. This act will be referred to as programming the AQC. The particular aspect of programming that will be focussed on is how imprecisions caused by errors or uncertainties in the field and coupling terms effect the systems likelihood to obtaining the correct answer. We will also look at how these errors change the spectrum throughout the adiabatic evolution process to determine if there are any issues introduced by the errors.

For this endeavour a simplistic model was used; by doing so, the study is limited to gaining insight into scaling relations where the exact values of constants are not significant. Important terms for this work are:

Intended Hamiltonian is the Hamiltonian that was desired to be put onto the machine.

Erroneous Hamiltonian is a version of the intended Hamiltonian that has errors in it according to the model discussed in Chapter 2.

Also errors and noise will be used synonymously in this work. Using these terms we can heuristically describe the key quantities of interest in this study:

Resilience is a measure of the probability that the ground state of the intended Hamiltonian will be the ground state of erroneous Hamiltonian that gets programmed into the system.

Crossover is an indicator of where the ground state of the intended Hamiltonian will lie in the spectrum of the erroneous Hamiltonian.

End Gap measures the change in energy between the ground state of the theoretic Hamiltonian.

These quantities combine to give a complete insight into how the lowest lying eigenstates of the spectrum of a Hamiltonian are affected by errors in the programming process. For an ideal AQC only the resilience is relevant as there is no chance that the computer will return an excited state. For a non-ideal AQC there is a probability related to the gap and cross over that the correct eigenstate will be returned erroneously, which is not explored in this work.

Chapter 2 is concerned with the models used throughout this work. Section 2.1 deals with how the hardware of the adiabatic quantum computer is modeled and Section 2.2 deals with how the errors were modeled. In Chapter 3 we look at the adiabatic theorem of quantum mechanics. In Chapter 4 the results are presented and discussed. Sections 4.1 and 4.2 deal with errors in the field terms while Sections 4.3 and 4.4 deal with errors in the coupling terms only. In Section 4.5 errors are considered in both terms simultaneously. Finally in Section 4.6 the effects of these errors on the adiabatic evolution is explored. In

Appendix A degeneracy is introduced into the Hamiltonians and the definition of resilience split into two cases. In Appendix C the programs used to gather the results are outlined. Appendix B pertains to how Hamiltonians with the same ground state are related.

Chapter 2

Model

In this thesis a simple model was used in order to gain insight into scaling relations of quantities interest, as mentioned in Section 1.5. In Section 2.1 the model for the hardware is examined and in Section 2.2 the model of the programming errors is discussed.

2.1 Ising Spin Glass

The fundamental model for this thesis is an Ising Spin Glass, which models a collection of spin particles that can take a spin-up or spin-down configuration and have arbitrary couplings(J) and field (b) terms. The Hamiltonian of an Ising Spin Glass can be written as:

$$H = -\sum_{i} b_i \sigma_i + \sum_{i<j} J_{ij} \sigma_i \sigma_j \tag{2.1}$$

where we are assuming the interactions involve no more than 2 bodies otherwise more terms would haveto be added to account for 3 body interactions and so forth. For this thesis we will further restrict the Hamiltonian to be classical and only include σ^z terms:

$$H = \sum_{i} b_i \sigma_i^z + \sum_{i<j} J_{ij} \sigma_i^z \sigma_j^z \tag{2.2}$$

This model was chosen because it can sufficiently capture the behavior of current adiabatic quantum computers. It is also possible to use a harmonic oscillator model for an adiabatic quantum computer [56]; however, this model gives a series of evenly distributed energy levels; meaning, in order to insert errors one would have-to either cause random shifts in the energy levels or add terms to the oscillator Hamiltonian. Neither option is ideal for this study as the errors are in programming the Hamiltonian. This means that the errors come before those perturbations would have-to be made. Furthermore, the Ising Spin Glass model has been used in other studies of AQCs [8, 24, 42, 46, 54, 55].

In an effort to map closer the experiment the Hamiltonians used in this study use the

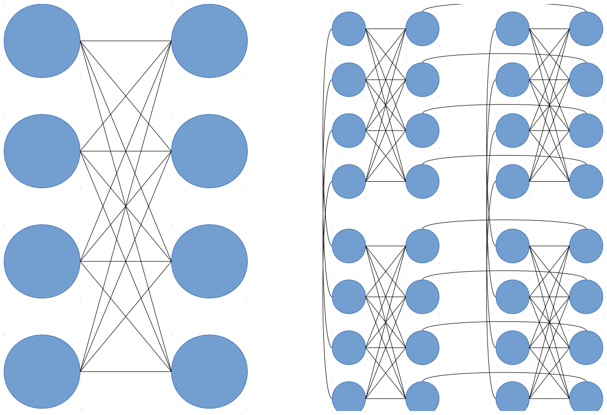

Figure 2.1: Schematic of the $K_{4,4}$ set-up which we use as a template for all Hamiltonians considered in this study. These were chosen due to their appropriateness in size for computational difficulty as well as their real world relevance as the fundamental block of the D-Wave computers. On the right we show how the $K_{4,4}$'s are connected.

template of a $K_{4,4}$ (shown in Figure 2.1), which is the fundamental block of spins used in the D-Wave's quantum hardware [18]. A $K_{4,4}$ consists of two columns of four spins. Each spin on one column is connected with all the spins on the opposing column, but none of the spins in its own column. $K_{4,4}$'s are strung together using the spins in the left column vertically or the right column for horizontal connections. These were chosen because their size allows efficient simulations to be run and they also have real world significance. Each of the couplings and field terms were chosen at random from a set of values defined by the Hamiltonian's resolution. The values of the fields and coupling terms fall in the range $[-1, 1]$ and a Hamiltonian's resolution specifies how many available evenly spaced values lie in the interval; for instance, a Hamiltonian of resolution 7 has possible field and couple term values of

$$\pm\left\{\frac{3}{3}, \frac{2}{3}, \frac{1}{3}, \frac{0}{3}\right\}$$

where a Hamiltonian of resolution 15 would have values ranging from

$$\pm\left\{\frac{7}{7}, \frac{6}{7}, \frac{5}{7}, \frac{4}{7}, \frac{3}{7}, \frac{2}{7}, \frac{1}{7}, \frac{0}{7}\right\}$$

The restriction to the interval $[-1, 1]$ is again to be closer to the D-Wave machine which has a fixed interval over which the values of field and coupling terms can be drawn. To achieve different resolutions they splice this interval accordingly. For generality, we assume that each value is equally likely to be in the Hamiltonian and thus the distribution that

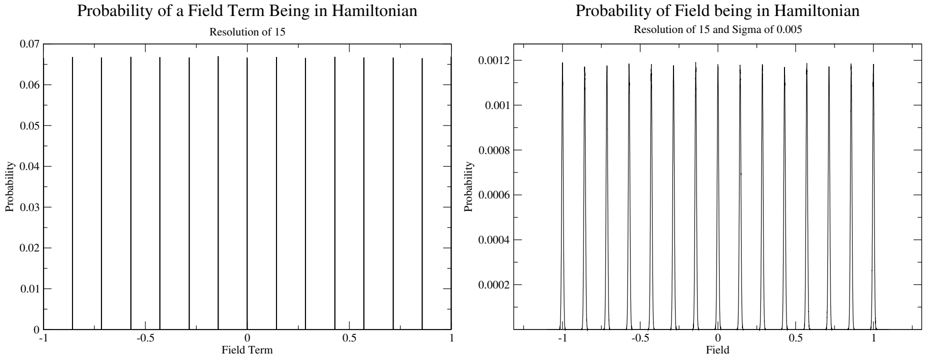

Figure 2.2: Left: Distribution that the field and coupling values were chosen from in the noise free model of the Hamiltonian. The distribution is simply delta spikes at each of the possible values. Right: Distribution that the field and coupling values were chosen from in the noisy model of the Hamiltonian. What were once delta spikes have become Gaussian distributions.

the terms are sampled from simply has delta spikes at each of the possible values, shown on the left of Figure 2.2. In actuality this may not be the case for a given Hamiltonian or class of Hamiltonians; however, Hamiltonians from any set that does not draw evenly from the set of allowed values for field and coupling terms can also be drawn from the set of Hamiltonians that draw with an even probability (in fact provided there are no 0 probabilities and Hamiltonian can be drawn from any set). In this way our results are not skewed to value a particular kind of Hamiltonian.

2.2 Errors or Uncertainties

The errors in this thesis took on the form of a normal distribution given by

$$p(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.3}$$

to obtain random numbers according to this distribution the *std::normal_distribution* class was used in C++, analogues of this class were used in the other languages. As an input these functions take μ and σ and return an x with the proper probability. In C++ the *std::normal_distribution* was chosen as it was faster than the function offered in the boost library. Though the details of the method to which the random numbers are generated are not specified in the C++ standard, the algorithm is likely using the Box-Muller transformation, or a variant of it. The Box-Muller transformation works by generating two random numbers, x_1 and x_2 between 0 and 1 from a uniform distribution. Then z_1

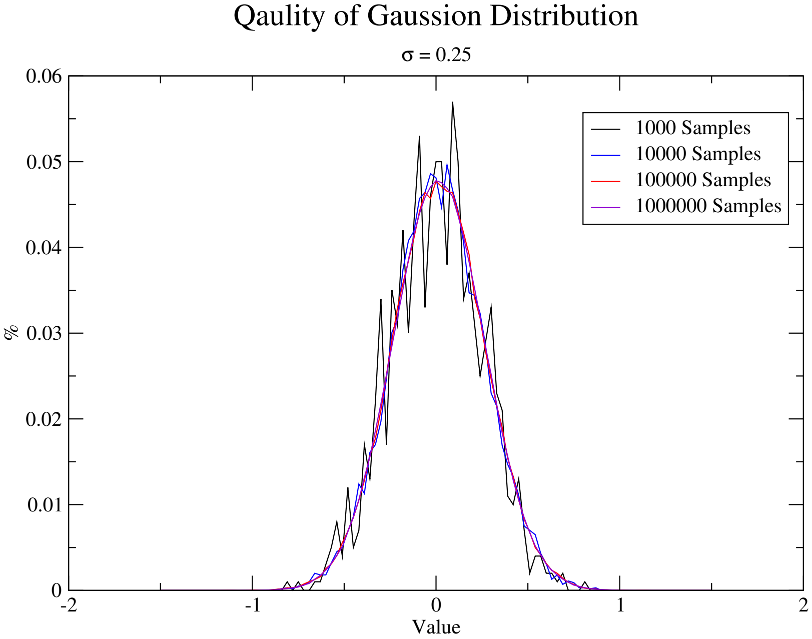

Figure 2.3: The shape of the approximate Gaussian distribution obtained by sampling various many times. As can be seen as you increase the number of samples you increase the smoothness of the distribution. This is important as to get meaningful statistics it is important to properly sample each Gaussian distribution. As you can see 10,000 samples appears to be sufficient, but not ideal. The typical data point in this study uses 10,000,000 samples.

and z_2, defined below, will have a normal distribution with $\mu = 0$ and $\sigma = 1$.

$$z_1 = \sqrt{-2 \ln x_1} \cos(2\pi x_2) \tag{2.4}$$

$$z_2 = \sqrt{-2 \ln x_1} \sin(2\pi x_2) \tag{2.5}$$

A normal distribution is a reasonable model for there errors as they are likely to be caused by things such as unbiased fluctuations or uncertainty in electromagnetic fields. The number of samples required to accurately sample a normal distribution is atleast 10,000 as illustrated in Figure 2.3. Once the errors are introduced the effect is to turn each of the delta spikes for the field and coupling terms into Gaussian distributions as shown on the right of Figure 2.2.

Chapter 3

Theory

3.1 Quantum Adiabatic Theorem

Here we provide a proof of the quantum adiabatic theorem taken from [16]. The proof assumes a non-degenerate Hamiltonian. When the system becomes degenerate the complexity increases and there is a set of conditions, one necessary and two sufficient in order for the theorem to hold, for details on these conditions see [43].

Given a time dependent Hamiltonian $H(t)$ we can write Schödingers equation as such:

$$i\hbar\frac{\delta\Psi(t)}{\delta t} = H(t)\Psi(t) \tag{3.1}$$

where $\Psi(t)$ is a linear superposition of all the $\psi_n(t)$ terms

$$\Psi(t) = \sum_n c_n(t)\psi_n(t)e^{i\theta_n(t)} \tag{3.2}$$

where

$$\theta_n(t) \equiv -\frac{1}{\hbar}\int_0^t E_n(t')dt'. \tag{3.3}$$

And $H(t)$ is such that

$$H(t)\psi_n(t) = E_n(t)\psi_n(t) \tag{3.4}$$

where the eigenfunctions $\psi_n(t)$ constitute an orthonormal set

$$\langle\psi_n(t)|\psi_m(t)\rangle = \delta_{nm} \tag{3.5}$$

Now substituting Equation 3.2 into Equation 3.1 and using the chain rule we obtain

$$i\hbar\sum_n\left(\frac{\delta c_n}{\delta t}\psi_n + c_n\frac{\delta\psi_n}{\delta t} + ic_n\psi_n\frac{\delta\theta_n}{\delta t}\right)e^{i\theta_n} = \sum_n c_n(H\psi_n)e^{i\theta_n} \tag{3.6}$$

now taking into consideration Equation 3.4 and Equation 3.3 Equation 3.6 reduces to

$$\sum_n \frac{\delta c_n}{\delta t}\psi_n e^{i\theta_n} = -\sum_n c_n \frac{\delta \psi_n}{\delta t} e^{i\theta_n} \tag{3.7}$$

Taking the inner product with ψ_m and using Equation 3.5 Equation 3.7 becomes

$$\sum_n \frac{\delta c_n}{\delta t}\delta_{nm} e^{i\theta_n} = -\sum_n c_n \langle \psi_m | \frac{\delta \psi_n}{\delta t}\rangle e^{i\theta_n} \tag{3.8}$$

which can be written

$$\frac{\delta c_m(t)}{\delta t} = -\sum_n c_n \langle \psi_m | \frac{\delta \psi_n}{\delta t}\rangle e^{i(\theta_n - \theta_m)}. \tag{3.9}$$

Now differentiating Equation 3.4 with respect to time gives

$$\frac{\delta H(t)}{\delta t}\psi_n(t) + H(t)\frac{\delta \psi_n(t)}{\delta t} = \frac{\delta E_n(t)}{\delta t}\psi_n(t) + E_n(t)\frac{\delta \psi_n(t)}{\delta t} \tag{3.10}$$

once again taking the inner product with ψ_m

$$\langle \psi_m | \frac{\delta H}{\delta t} | \psi_n \rangle + \langle \psi_m | H | \frac{\delta \psi_n}{\delta t}\rangle = \frac{\delta E_n}{\delta t}\delta_{nm} + E_n \langle \psi_m | \frac{\delta \psi_n}{\delta t}\rangle \tag{3.11}$$

Since H is Hermitian we can write $\langle \psi_m | H | \frac{\delta \psi_n}{\delta t}\rangle + E_m \langle \psi_m | \frac{\delta \psi_n}{\delta t}\rangle$ and it follows that (for $n \neq m$)

$$\langle \psi_m | \frac{\delta H(t)}{\delta t} | \psi_n \rangle = (E_n - E_m)\langle \psi_m | \frac{\delta \psi_n(t)}{\delta t}\rangle \tag{3.12}$$

substituting this into Equation 3.9 and making an assumption that the energy levels are non-degenerate it follows that

$$\frac{\delta c_m(t)}{\delta t} = -c_m \langle \psi_m | \frac{\delta \psi_m}{\delta t}\rangle - \sum_{n \neq m} c_n \frac{\langle \psi_m | \frac{\delta H}{\delta t} | \psi_n \rangle}{E_n - E_m} e^{\frac{1}{i\hbar}\int\limits_0^t (E_n(t') - E_m(t'))dt'} \tag{3.13}$$

Now making the adiabatic approximation $\frac{\delta H}{\delta t} \rightarrow 0$ Equation 3.13 simplifies to

$$\frac{\delta c_m(t)}{\delta t} = -c_m \langle \psi_m | \frac{\delta \psi_m}{\delta t}\rangle \tag{3.14}$$

which has the solution

$$c_m(t) = c_m(0)e^{i\gamma_m(t)} \tag{3.15}$$

where

$$\gamma_m(t) \equiv i\int\limits_0^t \langle\psi_m(t')|\frac{\delta\psi_m(t')}{\delta t'}\rangle dt' \tag{3.16}$$

This means we can write the eigenstates as

$$\psi_n(t) = e^{i\theta_n(t)}e^{i\gamma_n(t)}\psi_n(0) \tag{3.17}$$

where the coefficients only pick up phase factors and thus a particle that starts out in the n^{th} eigenstate will remain in the n^{th} eigenstate. The smaller the gap between the two states the slower the rate $\frac{\delta H}{\delta t}$ must be.

Chapter 4

Results and Discussion

The Hamiltonian of an Ising spin glass, such as the ones used in Adiabatic Quantum Computing, have the form:

$$H_p = \sum_i b_i \sigma_i^z + \sum_{i<j} J_{ij} \sigma_i^z \sigma_j^z \tag{4.1}$$

where b_i represent the field terms and the J_{ij}s represent the coupling terms. We will look at three different quantities to determine the effect of programming errors, i.e. uncertainties, on the resulting Hamiltonian: resilience, cross over, and final gap. These quantities are described heuristically in Section 1.5 and quantitatively below. For the next few sections we will restrict our attention to single $K_{4,4}$ Hamiltonians (see Section 2.1) that are non-degenerate. Degenerate Hamiltonians are considered in Appendix A.

In Section 4.1 errors are only in the field terms and are independent of the magnitude of the individual coupling. Section 4.2 the errors in the field term are made dependant on the magnitude of the errors. Sections 4.3 and 4.4 do the same for errors only in the coupling terms. Section 4.5 introduces errors into both terms simultaneously. Finally, Section 4.6 explores the effects of the errors on the Hamiltonian through its evolution.

4.1 Absolute Field Noise

In this section we will focus on errors in the field terms, b_i. The Hamiltonian will thus take the form:

$$H_p = \sum_i (b_i + N(\mu, \sigma^2)_i) \sigma_i^z + \sum_{i<j} J_{ij} \sigma_i^z \sigma_j^z \tag{4.2}$$

Where $N(\mu, \sigma^2)$ is a random number drawn from a normal distribution centred at μ, with width σ. See Figure 2.3. Throughout this work $\mu = 0$ while σ is varied. Note that σ is the width of the distribution and is not the same as σ_i which is a Pauli spin matrix. Since $\mu = 0$ for the entirety of the work here we can write $N(\mu, \sigma^2) \to N(\sigma)$.

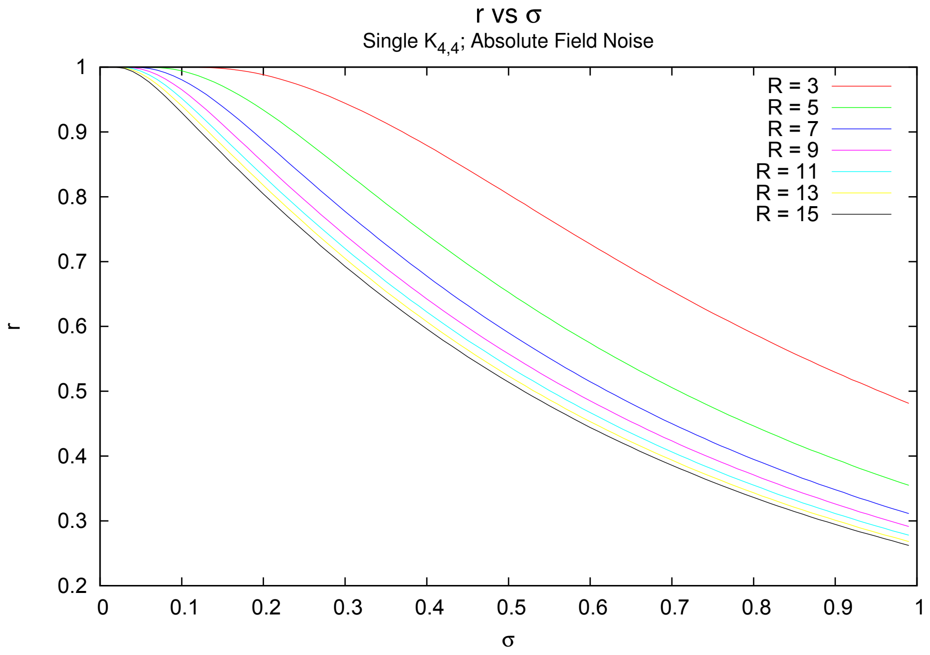

Figure 4.1: Resilience (r) shown as a function of σ for various resolutions (R). We see that the resilience drops with increasing resolution as expected. As σ increases the errors cause the field and coupling terms to displace, on average, further from their intended values which causes the resilience to drop. Increasing the resolution makes the spacing between the field and coupling terms smaller and thus it takes less displacement to cause the resilience to drop.

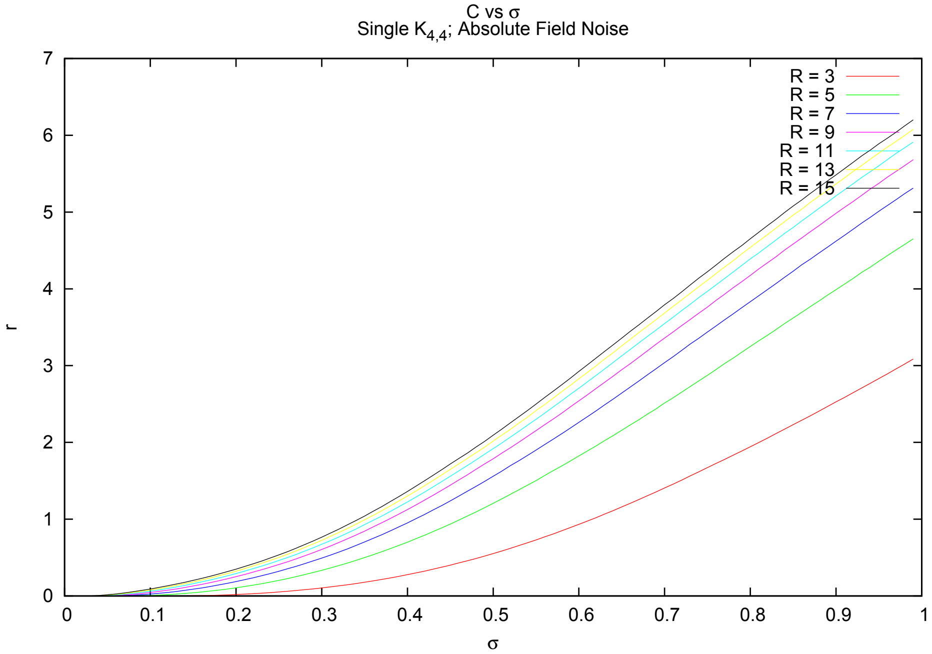

Figure 4.2: Cross over as a function of σ shown for various resolutions. It is easy to see that there is a correlation between cross over and resilience, as the resilience drops the cross over climbs. The increase will plateau as the erroneous Hamiltonians approach randomness.

4.1.1 Resilience

Resilience is a measure of the probability that the ground state of the erroneous Hamiltonian remains the same as the intended (error-free) Hamiltonian. It is calculated computationally by averaging over a large number of instances of erroneous Hamiltonians drawn from the noise distribution given by $N(\sigma)$. Let

$$Equiv(H_1, H_2) = \begin{cases} 1 & GS_{H_1} = GS_{H_2} \\ 0 & otherwise \end{cases} \tag{4.3}$$

where GS indicates the ground state of a Hamiltonian. We determine the resilience, r, of a Hamiltonian by

$$r = \frac{1}{N} \sum_{i}^{N} Equiv(H_p, H_p^i) \tag{4.4}$$

using N instances of erroneous Hamiltonians denoted by H_p^i. Here H_p is an error free Hamiltonian and each H_p^i is an erroneous instance of that Hamiltonian. It is desired that r be independent of any particular Hamiltonian instance. We further average across different randomly generated noise-free Hamiltonians. The resilience then becomes

$$r_{avg} = \frac{1}{N_{ham} N_{inst}} \sum_{p}^{N_{ham}} \sum_{i}^{N_{inst}} Equiv(H_p, H_p^i) \tag{4.5}$$

with typical values of N_{inst} and N_{ham} being 10,000 and 1,000 respectively; thus, each data point is an average of 10 million Hamiltonians.

Computationally the results are calculated by first selecting a resolution. A Hamiltonian is then generated according to the $K_{4,4}$ topology, and its ground state computed. Then 10,000 independent instances of erroneous versions of the original Hamiltonian are generated and their ground states calculated and compared via Equation 4.3 for each σ value. The averaging of these runs is done in accordance to Equation 4.4. This is repeated for 1,000 error-free Hamiltonians and averaged via Equation 4.5. This process is repeated for each resolution.

The results of these computations for various resolutions are shown in Figure 4.1. The behaviour of the resilience is as expected. Increasing σ causes more error in the system resulting in the field and coupling terms to be displaced further from their intended values, on average. Increasing the resolution causes the field and coupling terms to be

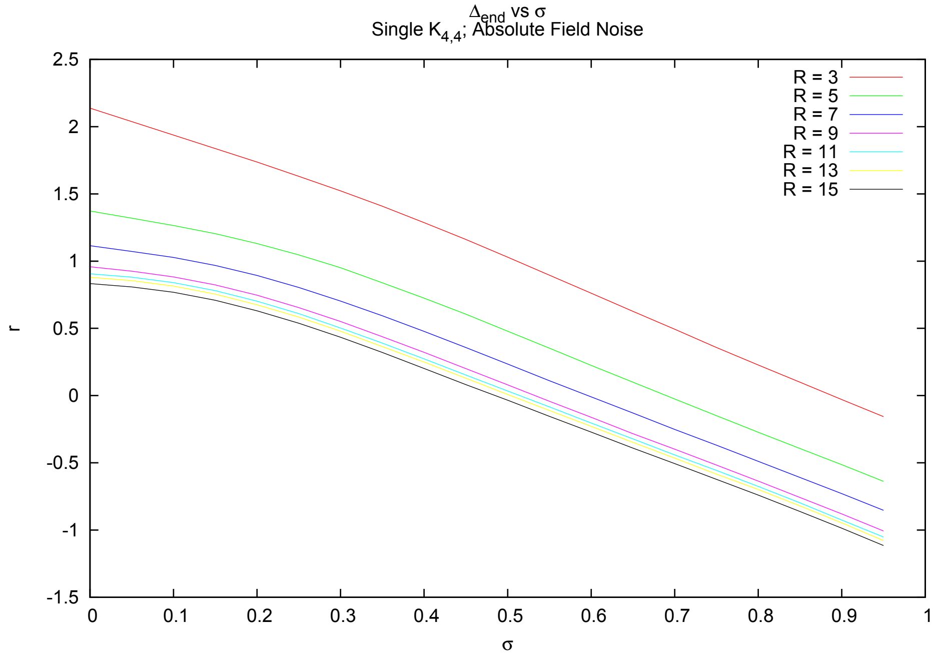

Figure 4.3: The gap versus σ for various resolutions. We again see that this is, as expected, correlated to the resilience, Figure 4.1, and the cross over, Figure 4.2. As the resilience drops, the gap goes more negative and exhibits the same plateau behaviour.

closer together (reduces ΔJ & Δb) making the Hamiltonian more susceptible to errors. Fitting of resilience curves is done in Section 4.5.1.

4.1.2 Cross over

Cross over, C, counts the number of eigenstates that cross below the ground state of the intended Hamiltonian. Quantitatively it is measured by determining where the original ground state lies in the spectrum of the erroneous Hamiltonian. The averaging for this quantity is done the same as it was for resilience. A cross over of 1 means that the original ground state is now the first excited of the erroneous Hamiltonian. Figure 4.2 shows the behaviour of C versus σ for various resolutions with errors occurring in the field terms. Comparing Figure 4.2 with Figure 4.1 we see that there is a correlation between resilience and cross over, as would be expected. As resilience drops the cross over climbs with similar relations between the resolutions.

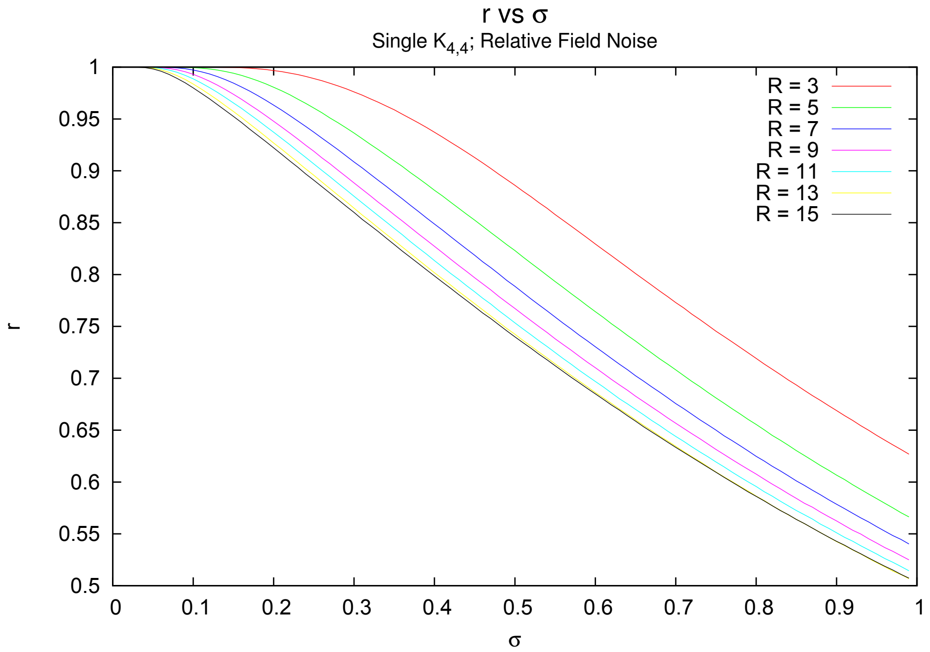

Figure 4.4: Resilience as a function of σ for various resolutions. We can see that the same behaviour is seen as was in Figure 4.1 that describes a system of absolute noise.

4.1.3 End Gap

End gap is a measure of the spectral properties of the erroneous Hamiltonians in relation to the intended Hamiltonian. Quantitatively this can be described by defining S_i^H as the spin configuration of the i^{th} energy level of Hamiltonian H. We will call the problem Hamiltonian H_p and the erroneous instance of the problem Hamiltonian H_n. We will define functions

$$Lowest(H, S) = \text{Lowest lying eigenstate of } H \text{ that is not } S \tag{4.6}$$

$$Energy(H, S) = \text{Energy of eigenstate } S \text{ according to } H \tag{4.7}$$

which then allows us to define the gap as

$$Gap(H_p, H_n) = energy(H_n, GS_{H_p}) - energy(H_n, Lowest(H_n, GS_{H_p})). \tag{4.8}$$

If $H_p = H_n$, Equation 4.8 gives the usual definition of the gap as the energy separation between the ground and first excites states. If $H_p \neq H_n$, and if the ground state of H_p

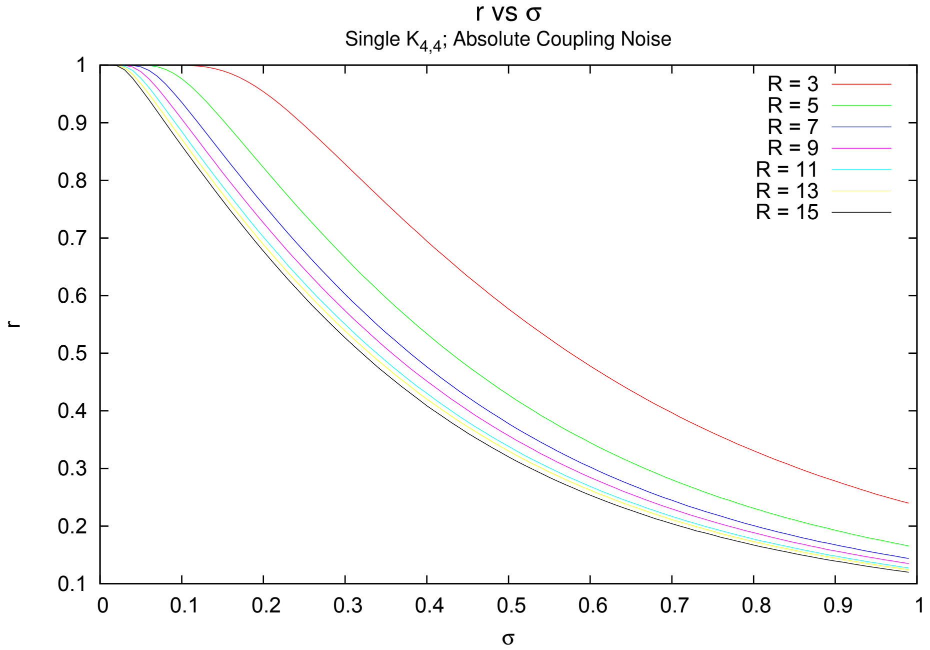

Figure 4.5: A graph of resilience versus σ for various resolutions where there is noise only in the coupling terms that is independent of the strength of the individual terms. Comparing with Figure 4.1 we see that the resilience is lower when there is noise in the coupling terms. This is partially because there are 16 coupling terms versus just 8 field terms; furthermore, coupling terms effect the relative orientation of the spins.

has been moved to an excited state of the erroneous Hamiltonian (H_n) this definition of Gap will give a negative result. Figure 4.3 shows the behaviour of the gap versus σ for various resolutions. Note that an average gap of 0 does not mean the gap closed, it is an effect of the averaging. The averaging was done as it was in the previous sections. We see that the gap exhibits a correlation between the resilience and cross over which are shown in Figure 4.1 and Figure 4.2 respectively. Since there is a strong correlation between these three quantities from this point on we will restrict our attention to just resilience, knowing we can infer the behaviour of the other quantities from the resilience.

4.2 Relative Field Noise

In the last section the magnitude of errors was independent of the magnitude of the field strength. Here we look at the behaviour of the system when the Hamiltonian takes the

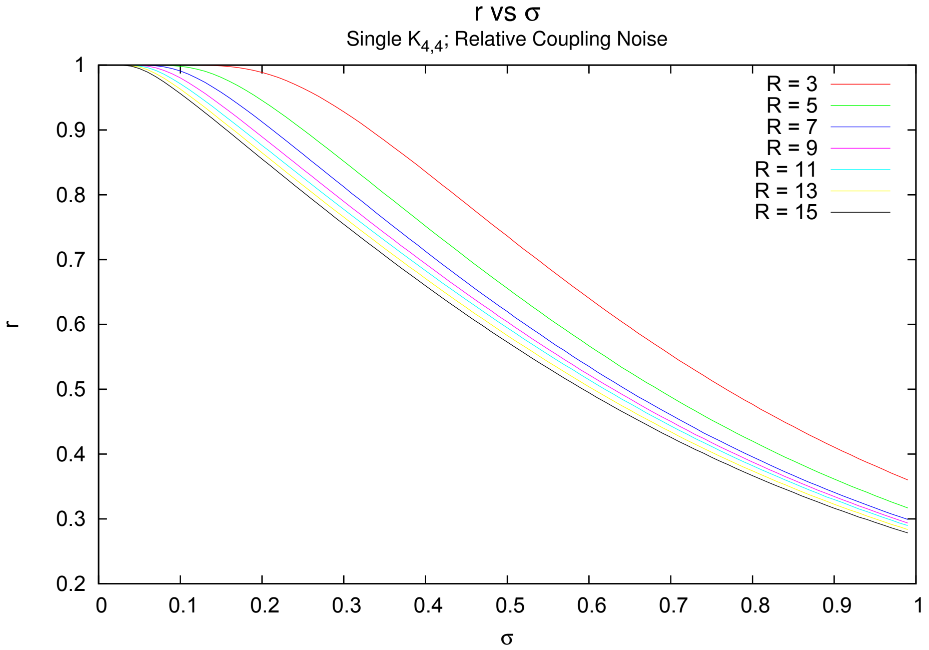

Figure 4.6: Graph of resilience versus σ for various resolutions where the noise is only in the coupling terms and proportional to each coupling term.

form

$$H_p = \sum_i b_i(1 + N(\sigma)_i)\sigma_i^z + \sum_{i<j} J_{ij}\sigma_i^z\sigma_j^z. \tag{4.9}$$

For brevity, as previously mentioned, we will restrict our attention to resilience knowing that there is a correlation between resilience and gap and cross over. Figure 4.4 shows the behaviour of the resilience when the noise is as in Equation 4.9. We can see that the general behaviour matches that of the absolute noise seen in Figure 4.1. The resilience drops more slowly as the magnitudes of errors is in general lessened by making the errors relative. For absolute errors sigma describes the percentage of the error relative to the maximum value and for relative errors sigma describes the errors relative to the instantaneous magnitude, and the instantaneous magnitude of the field terms is likely to be less than one.

4.3 Absolute Coupling Noise

Focusing now on the terms J_{ij} we will explore how inaccuracies or noise effect the resilience of a Hamiltonian. The Hamiltonians now take the form:

$$H_p = \sum_i b_i \sigma_i^z + \sum_{i<j} (J_{ij} + N(\sigma)_{ij}) \sigma_i^z \sigma_j^z \tag{4.10}$$

Figure 4.5 shows the resilience versus σ for various resolutions for absolute errors in the coupling terms. Comparing Figure 4.5 with Figure 4.1 we see that the resolution is less when there is coupling noise. This is partially because, for a single $K_{4,4}$, there are 8 field terms and 16 coupling terms. The increased number of terms leads to more noise in the system. Note the introduction of an apparent lower plateau (high σ, low r) in the resilience (this is reflected in the other quantities). In Appendix B histograms are presented showing the probability of Hamiltonians of a given difference having the same ground state. The lower plateau is believed to be a relic of the Gaussian nature of these distributions coupled with the probability of obtaining the correct ground state with a randomly chosen Hamiltonian. The suspected reason for the appearance of the plateau here and not in the previous section is the increased number of coupling terms increases the amount of errors and thus increases the difference between the intended and erroneous Hamiltonians, see Appendix B for definitions of difference and further details.

4.4 Relative Coupling Noise

Making the noise relative to the strength of the coupling terms as we did for field terms in Section 4.2 we obtain a Hamiltonian of the form:

$$H_p = \sum_i b_i \sigma_i^z + \sum_{i<j} J_{ij} (1 + N(\sigma)_{ij}) \sigma_i^z \sigma_j^z \tag{4.11}$$

Here the strength of the coupling noise depends on each coupling term itself. In Figure 4.6 we see the same qualitative behaviour as before in Figure 4.5 for coupling noise that is independent of each coupling's strength; however, the decline in resilience is lessened, as it did for relative field noise, and the plateau effect appears to be displaced further right. The displacement of the plateau makes sense as the relative errors decrease the magnitude

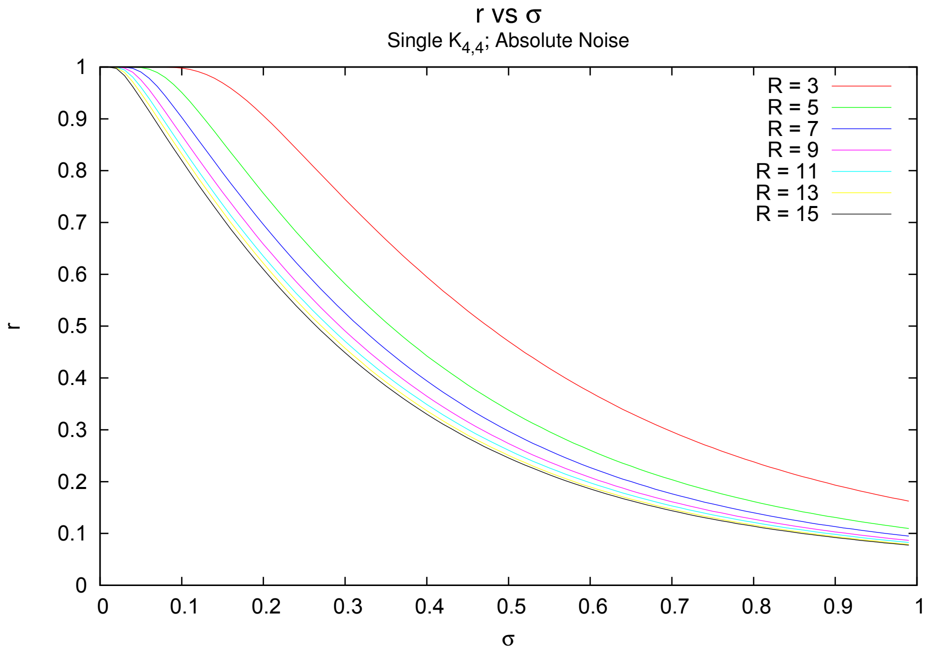

Figure 4.7: Resilience versus σ for various resolutions when there is noise in both the field and coupling terms that is of equal strength and independent of the strength of the field or coupling term.

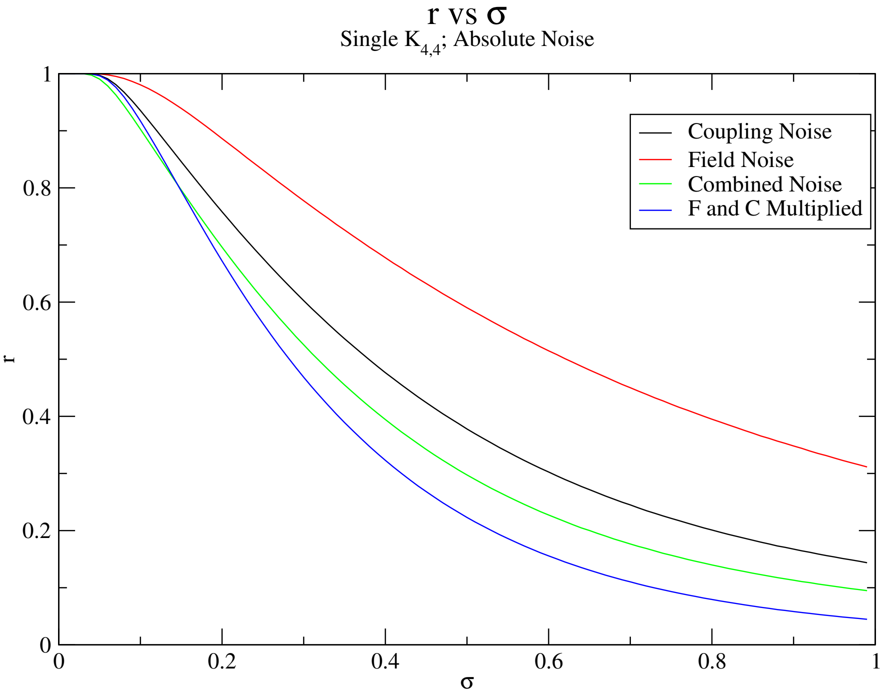

Figure 4.8: This figure compares the resilience of the combined noise system (Equation 4.13) with the resilience of the Hamiltonians with just field and coupling noises alone (Equations 4.2 and 4.10 respectively) and their multiple.

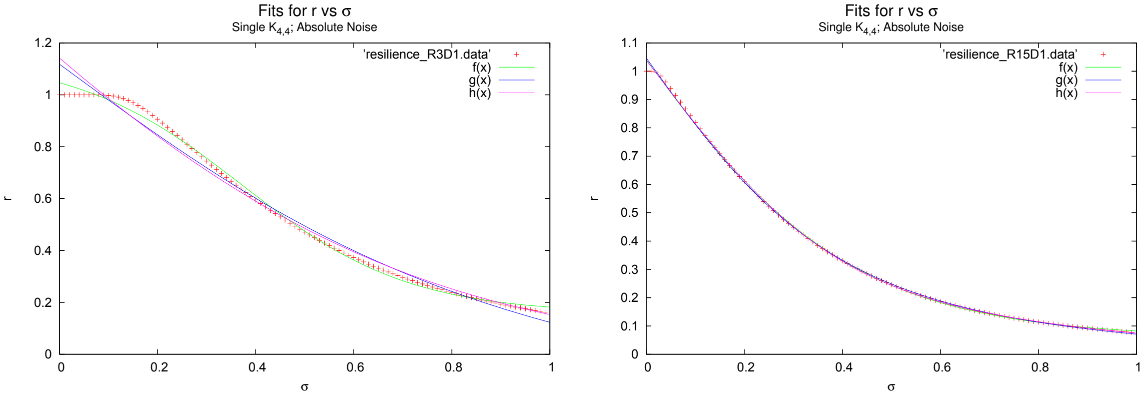

Figure 4.9: Three different attempt to fit the resilience curves, the graph on the left is of resolution 3 while the one on the right is of resolution 15. The three fits are given in Equations 4.14, 4.15 & 4.16. All three first work well away from the resilience = 1 plateau or when it is small as in the figure on the right.

of the errors by a factor of

$$\langle J \rangle = \langle b \rangle = \frac{R+1}{2R} \tag{4.12}$$

on average, which is the average field or coupling value when the terms are chosen from a uniform distribution. This reduction in the average magnitude of the errors means that the average difference between the intended and erroneous Hamiltonian is lessened and thus the plateau moved to larger σ's.

4.5 Combined Field and Coupling Noise

To this point errors only in either the field or coupling terms have been considered. This section considers errors in both terms simultaneously.

4.5.1 Absolute Combined Errors

The Hamiltonian will now take the form:

$$H_p = \sum_i (b_i + N(\sigma)_i)\sigma_i^z + \sum_{i<j} (J_{ij} + N(\sigma)_{ij})\sigma_i^z \sigma_j^z \tag{4.13}$$

Where the error in the field terms and coupling terms is of identical magnitude, in Section 4.5.3 we will consider what happens when the errors are independent. Considering Figure 4.7 and comparing it with Figures 4.1 and 4.5 it is apparent that the resilience of

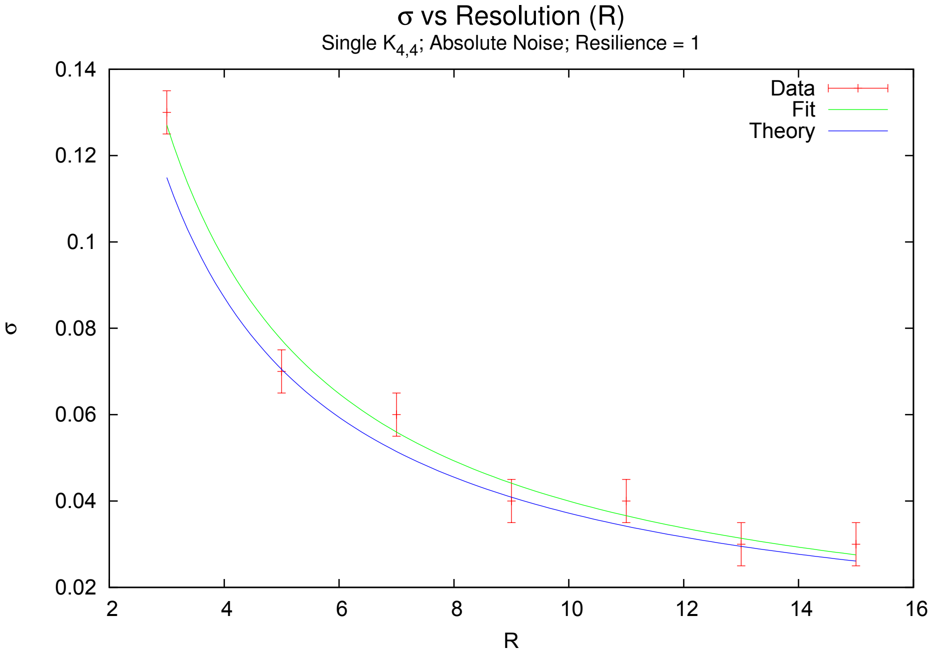

Figure 4.10: Length of the resilience = 1 plateau versus resolution for absolute errors. There appears to be a $\frac{1}{x}$ relation. The roughness is due to a 0.01 step in sigma for the simulations from which the data was collected. The equation for the fit is given by Equation 4.21. The theory line refers to an estimate on the lower bound which is given by Equation 4.25.

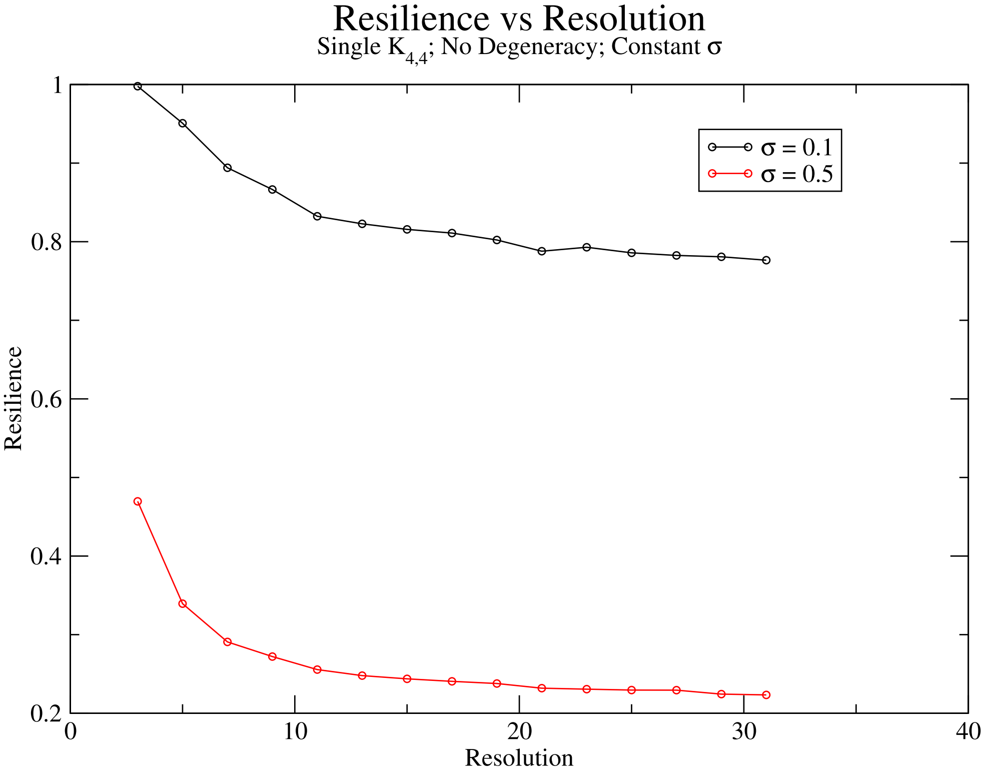

Figure 4.11: Resilience versus Resolution for $\sigma = 0.1$ and $\sigma = 0.5$. We see a plateau style behaviour characteristic of a $y = \frac{1}{x}$ relationship. The $\frac{1}{x}$ relationship comes from the inverse relationship between resolution and the spacing between the field and coupling terms. There are only odd points because for any resolution there must be an equal number of terms on both sides of 0, plus 0 in the allowed values for the field and coupling terms. Alternate definitions are allowed, but not considered here.

the combined errors is not simply the multiplication of the two resiliences of the previous sections. This fact is emphasized in Figure 4.8 where we show the difference between the multiple of the two resiliences from the individual noises compared to the measured resilience. The lower plateau has shifted to lower values of σ due to the increased number of terms for errors to enter increasing the difference between the Hamiltonians on average. Attempting to fit the resilience curves is difficult due to the plateau at resilience = 1.

Figure 4.9 shows attempts to fit the curves at resolution 3 (left) and resolution 15 (right) to three different functions:

$$f(x) = \frac{a}{b + e^{-cx}} + d \tag{4.14}$$

$$g(x) = \frac{ax}{\sqrt{b + cx^2}} + d \tag{4.15}$$

$$h(x) = \sum_{i}^{6} N(\mu_i, \sigma_i^2) \tag{4.16}$$

All three fits work well away from the $r = 1$ plateau. Focusing on the value of final value of σ that $r > 0.99 \approx 1$ an inverse relationship appears, which is shown in Figure 4.10. The fit is given by

$$\sigma(R) = \frac{0.513316}{1.3755R} + 0.00266851. \tag{4.17}$$

To improve the accuracy of the data one would need to write a simulation to precisely find the location of the end of the plateau. Methods used here were only accurate to within ± 0.005 in sigma. It is possible to rewrite this to be a relation between the spacing between field and coupling terms, Δb and ΔJ respectively. Note that, for simplicity, throughout this study

$$\Delta J = \Delta b, \tag{4.18}$$

because

$$R_J = R_b. \tag{4.19}$$

This is not a requirement of adiabatic quantum computing, and in-fact it is probable that there are benefits to having these values independent of each other; however, that is left for future studies to determine. The relationship between R and ΔJ is

$$R = \frac{2}{\Delta J} + 1. \tag{4.20}$$

Thus we can rewrite Equation 4.17 as

$$\sigma(\Delta J) = \frac{0.373185\Delta J}{\Delta J + 2} + 0.00266851. \tag{4.21}$$

A lower bound on the maximum magnitude of errors that are permitted in the system (while it maintains $r = 1$) can be approximated through

$$\sigma(\Delta J) = \frac{N_{spins}}{N_{terms}} \frac{\Delta J}{\Delta J + 2}. \tag{4.22}$$

Which for a single $K_{4,4}$ gives

$$\sigma(\Delta J) = \frac{1}{3} \frac{\Delta J}{\Delta J + 2}. \tag{4.23}$$

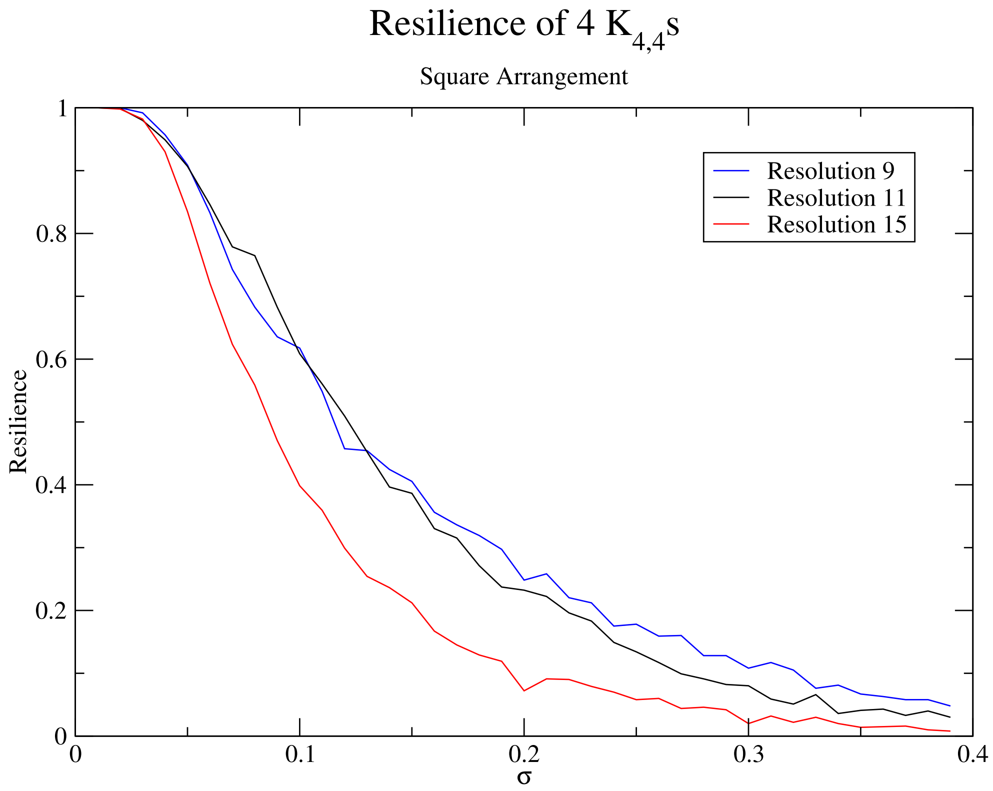

Figure 4.12: Resilience versus σ graph for $4K_{4,4}$'s and various resolutions. The statistics here are not nearly as good as before at the problem contains 32 spins which is roughly $2^{32-8} \approx 17,000,000$ times more computationally demanding. Comparing with Figure 4.7 we see that the resilience drops off much quicker. This is partially because there are 112 terms for noise to effect which is much more than a single $K_{4,4}$ and also the increased number of spins allows for more frustration which brings the energy levels closer together.

Equation 4.23 is reasonably close to Equation 4.21; furthermore, the values from Equation 4.23 are always less than Equation 4.21 so there will be no overestimating. As for the constant term, it seems reasonable to assume that it is the chance that the ground state will remain the same out of random luck. In Appendix B the probabilities of this are measured and all ground states are found to be equally likely. Equation 4.22 thus becomes

$$\sigma(\Delta J) = \frac{N_{spins}}{N_{terms}} \frac{\Delta J}{\Delta J + 2} + \frac{1}{2^{N_{spins}}}. \tag{4.24}$$

Which for a single $K_{4,4}$ gives

$$\sigma(\Delta J) = \frac{1}{3} \frac{\Delta J}{\Delta J + 2} + 0.0039. \tag{4.25}$$

In Figure 4.10 shows how Equation 4.25 fits the data. We see that is it an good lower bound for the fit, the quantitative values are shown in the table below; however, it is also a good fit for high resolutions.

ΔJ	Equation 4.21	Equation 4.24
1/2	0.0773055	0.0610491
1/4	0.0441335	0.0356523
1/5	0.0365944	0.0298803
1/7	0.0275475	0.0229539

Table 4.1: Comparing values from Equation 4.21 with those from the theoretical lower bound model Equation 4.24.

When one applies Equation 4.24 to the $4K_{4,4}$ system the relation is

$$\sigma(\Delta J) = \frac{32}{112} \frac{\Delta J}{\Delta J + 2} + \frac{1}{2^{32}} \approx \frac{32}{112} \frac{\Delta J}{\Delta J + 2}. \tag{4.26}$$

This gives values of

$$\sigma(\frac{1}{4}) \approx 0.032 \tag{4.27}$$

$$\sigma(\frac{1}{5}) \approx 0.026 \tag{4.28}$$

$$\sigma(\frac{1}{7}) \approx 0.019 \tag{4.29}$$

which is within reason of the values observed in Figure 4.12; noting, the statistics in

Figure 4.12 are lower quality than the single $K_{4,4}$ graphs, so the uncertainties are greater. If this approximation holds to larger systems like the Vesuvius chip from DWave then the equation will take the form

$$\sigma(\Delta J) = \frac{512}{1984}\frac{\Delta J}{\Delta J + 2} + \frac{1}{2^{512}} \approx \frac{512}{1984}\frac{\Delta J}{\Delta J + 2}. \tag{4.30}$$

Thus the estimated error tolerance is

$$\sigma(\frac{1}{4}) \approx 0.029 \tag{4.31}$$

$$\sigma(\frac{1}{5}) \approx 0.023 \tag{4.32}$$

$$\sigma(\frac{1}{7}) \approx 0.017 \tag{4.33}$$

Further studies are required to determine the accuracy of these estimates.

A similar inverse behaviour is seen in the resilience of a system when sigma is fixed and the resolution increased. See Figure 4.11. The relationship between resilience and resolution appears to be of the form of $\frac{1}{x}$ which makes sense as the spacing between the field and coupling terms has an inverse relationship to the resolution. In Appendix C.5 the relationship between Hamiltonians with the same ground state is explored; however, a link between the behaviour in Figure 4.11 was not found. This suggested the behaviour stems from the change in spacing of the field and coupling terms as the resolution is increased. For $\sigma = 0.5$ the curve can be well approximated by:

$$r = \frac{4957}{6339R} + 0.19281 \tag{4.34}$$

When the problem size is increased to $4K_{4,4}$'s the resilience drops off much quicker, as shown in Figure 4.12. Comparing with Figure 4.7 we see that the resilience drops off much quicker. This is partially because there are 112 terms for noise to effect which is much more than a single $K_{4,4}$ and also the increased number of spins allows for more frustration which brings the energy levels closer together. Limited computational resources bounded the amount of work that could be done on this system size. The algorithms required to exactly solve for the ground state scale exponentially, which makes $4K_{4,4}$'s roughly 17 million times more computationally demanding.

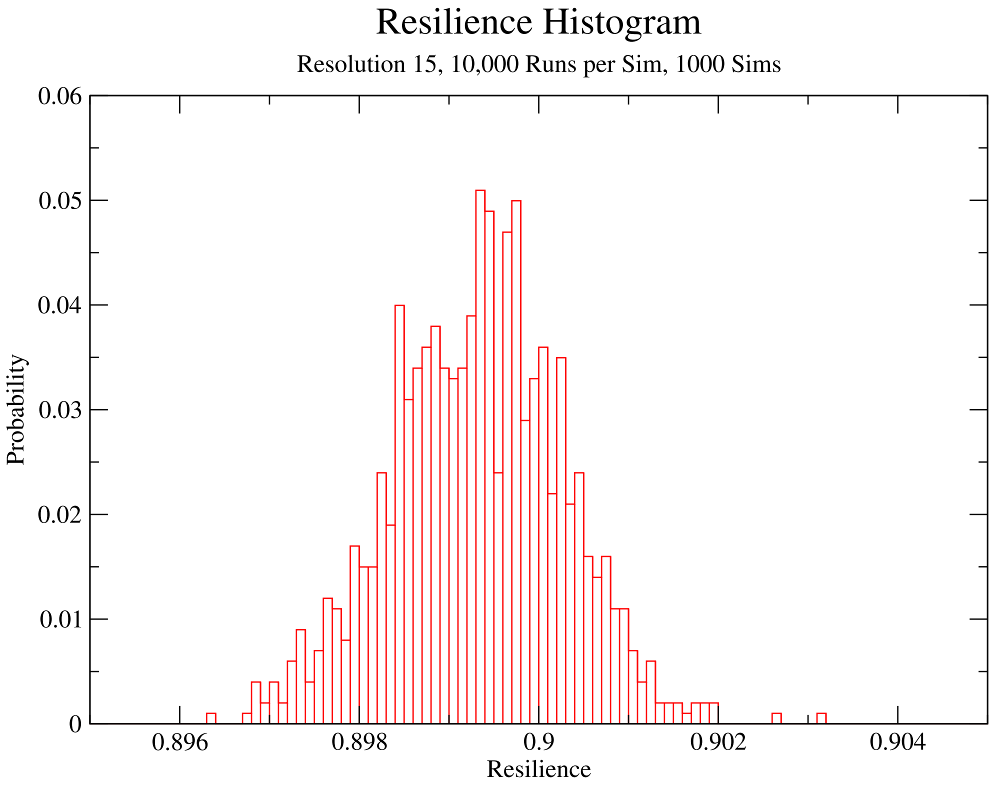

Figure 4.13: The distribution of resiliences from independent simulations. The distribution appears to be a skewed Gaussian with a small width.

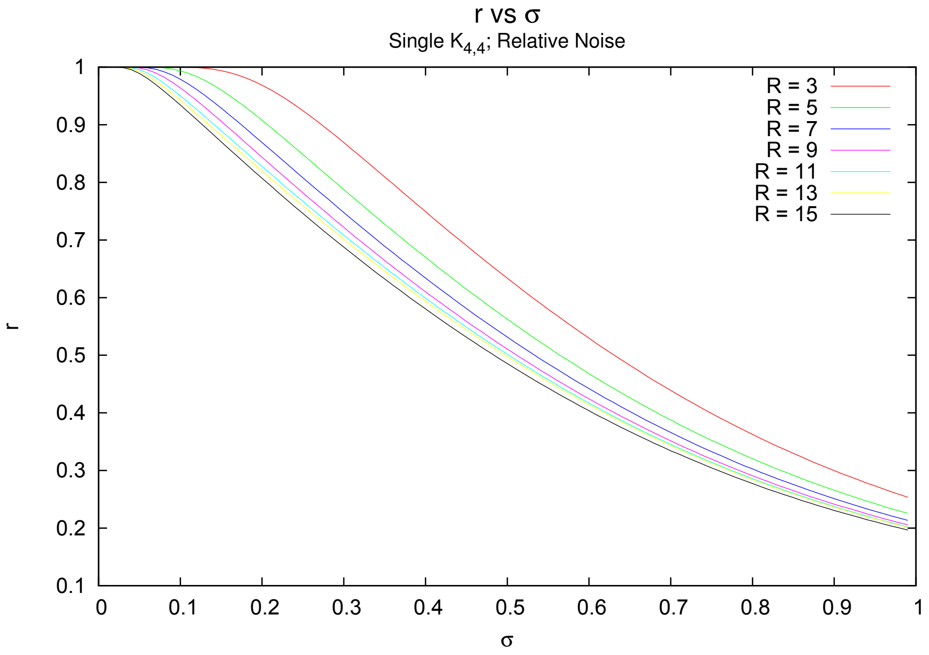

Figure 4.14: Resilience versus σ for various resolutions when there is noise in both the field and coupling terms that is of equal strength and are dependant on the strength of the field or coupling term.

Uncertainty

The statistics gathered for the graphs for single $K_{4,4}$'s were gathered by running simulations a large number of times to maximize their statistical significance. Figure 4.13 shows a distribution of resiliences calculated from 1000 independent simulations, the spread is minimal and therefore error bars have been omitted from this thesis. Uncertainty introduced from the simplistic model is likely more significant than the spread from statistical error. Similar graphs can be produced for other resolutions, lower resolutions yield a tighter distribution.

4.5.2 Relative Combined Errors

For completeness we look at a Hamiltonian of the form:

$$H_p = \sum_i b_i(1 + N(\sigma)_i)\sigma_i^z + \sum_{i<j} J_{ij}(1 + N(\sigma)_{ij})\sigma_i^z\sigma_j^z \tag{4.35}$$

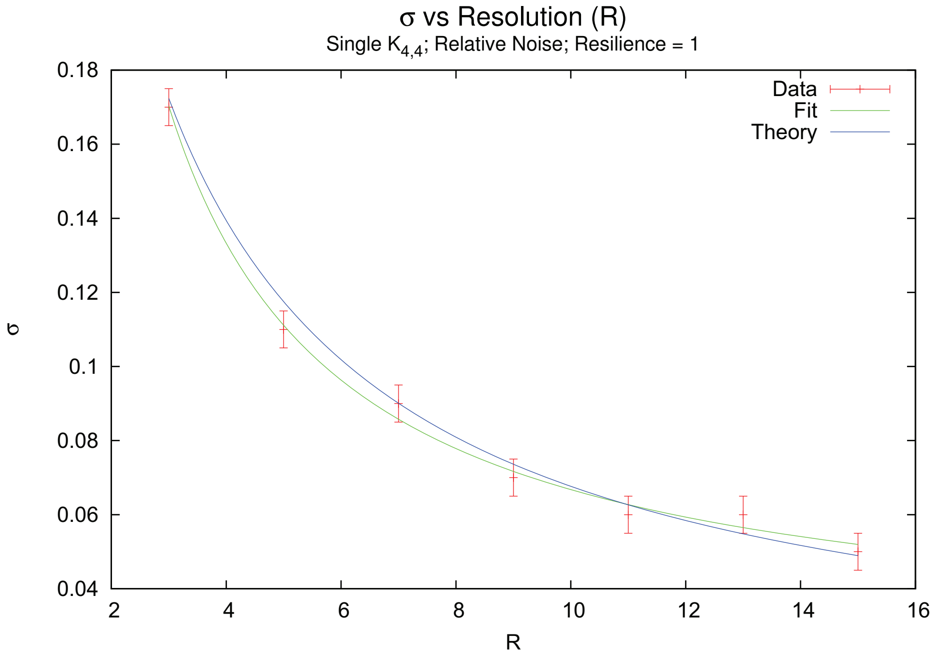

Figure 4.15: Length of the resilience = 1 plateau versus resolution for relative errors. There appears to be a $\frac{1}{x}$ relation. The roughness is again due to the data being mined from other simulations that were not designed to measure this quantity.

to see what the resilience looks like with relative errors in both terms. We see the same general shape of the curves as we did in Figure 4.13 but with a slower decline and postponed plateau as would be expected from results in previous sections. This suggests the noise in Hamiltonians will always have the same decay properties with different limits and rates.

Focusing again on the length of the plateau at $r = 1$, Figure 4.15, a familiar inverse relationship appears, which reworked for ΔJ looks like

$$\sigma(\Delta J) = \frac{0.443633\Delta J}{\Delta J + 2} + 0.0223849. \tag{4.36}$$

The difference between Equation 4.21 and Equation 4.36 stems from the fact that the relative errors are smaller on average. This can be adjusted by realizing that the average field and coupling value for a Hamiltonian of resolution R is

$$\frac{R+1}{2R} = \frac{\Delta J + 1}{\Delta J + 2}. \tag{4.37}$$

Taking this into account Equation 4.24 for absolute errors becomes

$$\sigma(\Delta J) = \frac{\Delta J + 2}{\Delta J + 1}\left(\frac{N_{spins}}{N_{terms}}\frac{\Delta J}{\Delta J + 2} + \frac{1}{2^{N_{spins}}}\right). \tag{4.38}$$

Equation 4.38 has good agreement with Equation 4.36 as illustrated in Table 4.2 and Figure 4.15.

ΔJ	Equation 4.36	Equation 4.38
1/2	0.1111120	0.117622
1/4	0.0716775	0.073698
1/5	0.0627152	0.062717
1/7	0.0519604	0.048991

Table 4.2: Comparing values from Equation 4.36 with those from the theoretical lower bound model Equation 4.38.

4.5.3 Independent Absolute

Now we will consider what happens when the strength of the errors in field and coupling terms independent of one another. Restricting our attention to noise that is independent of the individual field and coupling values the Hamiltonian will take the form:

$$H_p = \sum_i (b_i + N(\mu, \sigma_b^2)_i)\sigma_i^z + \sum_{i<j}(J_{ij} + N(\mu, \sigma_J^2)_{ij})\sigma_i^z\sigma_j^z \tag{4.39}$$

The graphs will now take the form of heat maps where the x and y values represent σ_b and σ_J respectively and the z value represents the resilience or other quantity of interest. Figure 4.16 shows a set of heat maps of the resilience versus σ_b and σ_J each with different resolutions. We can see that there is more tolerance to noise in the field terms than there is in the coupling terms, which is in line with previous observations. Similar to in Section 4.1 we can look at the other quantities of interest, namely the cross over and end gap, and see the same type of correlations as before, see Figure 4.18.

Measuring the length of the plateau of resilience 1 leads to Figure 4.17. The errors in precisely finding the value there the drop begins becomes more apparent and leads to difficulty in fitting the curves, which are most likely elliptical curves; however, attempts to fit them as elliptical curves leads to large errors.

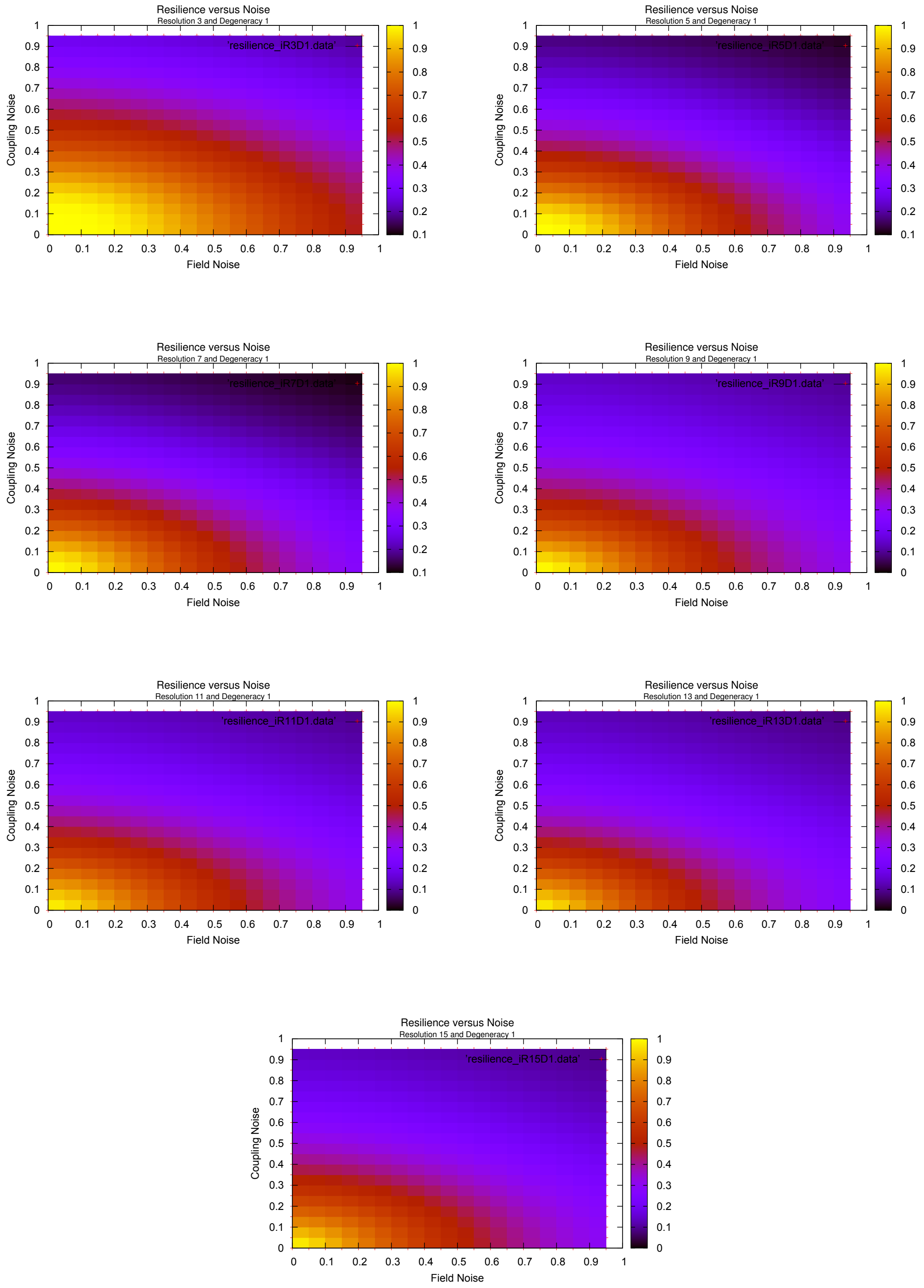

Figure 4.16: Set of heat maps of the resilience versus σ_b and σ_J each with different resolutions. We can see that there is more tolerance to noise in the field terms than there is in the coupling terms. This asymmetry can partially be explained by the relative number of field and coupling terms, 8 and 16 respectively, along with the fact that the field terms effect single spin orientation while coupling terms effect their relative orientations.

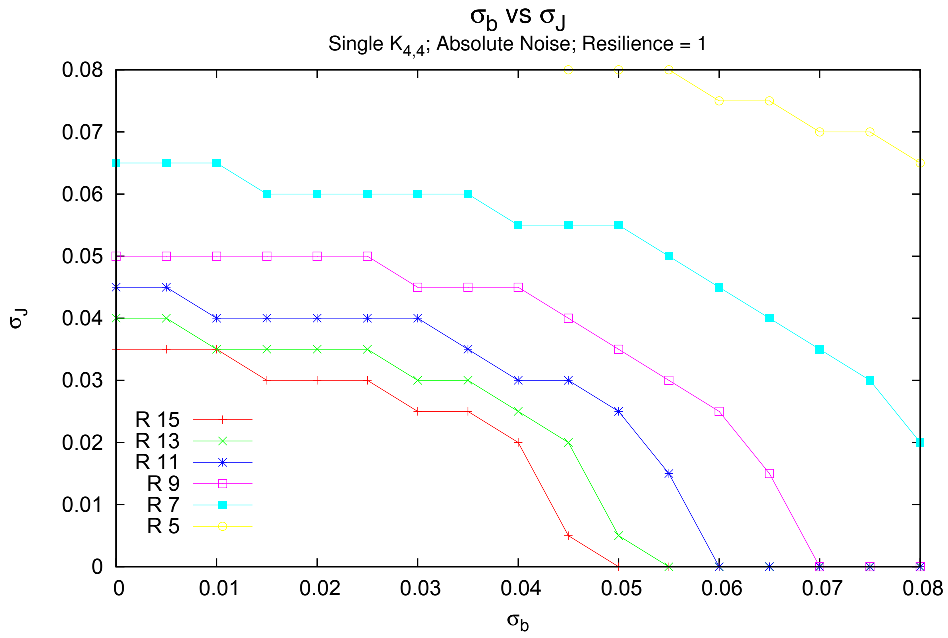

Figure 4.17: Length of the resilience = 1 plateau versus resolution for independent absolute errors. Each line appears to be following an elliptical path.

4.6 Adiabatic Spectrum

This section will focus at how errors effect the spectrum of the instantaneous Hamiltonian throughout the evolution. This is accomplished by the measuring instantaneous gap and its minimum. The gap here will be defined in the traditional way as the spacing between the instantaneous ground and first excited states. To model the adiabatic evolution we will use a linear evolution path and thus the instantaneous Hamiltonian can be written:

$$H(s) = (1 - s)H_0 + sH_p \tag{4.40}$$

Where H_p takes the form of Equation 4.13 and H_0 is a Hamiltonian with a ground state that is easy to prepare and does not commute with the problem Hamiltonian.

4.6.1 Gap Versus s

Figure 4.19 shows how, on average, the gap behaves for various σ's. There is a general trend that increasing the noise raises the gap and makes the well-like feature more shallow. Examining Figure 4.20 shows that these changes in path have essentially no effect on the

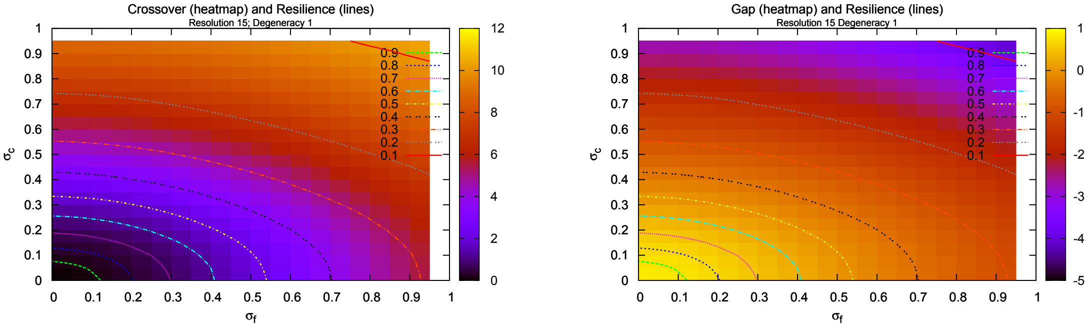

Figure 4.18: Comparative of the resilience, cross over, and end gap when the field and coupling noises are independent. The resilience is overlaid on each graph in the form of contour lines. We observe the same type of correlations that were seen in Section 4.1.

probability of the computer obtain the ground state of the programmed Hamiltonian.

4.6.2 Minimum Gap

Now we consider the minimum gap alone. The gap is averaged over 1,000 instances of intended and erroneous Hamiltonians. The computationally demands of these simulations were also quite great which limited the amount of data that could be obtained. For small amounts of noise the system begins to close the minimum gap, on average, before the noise start to cause the energy levels to spread, see Figure 4.21. By the time the spreading takes effect the erroneous Hamiltonian is unlikely to share the same ground state as the intended Hamiltonian.

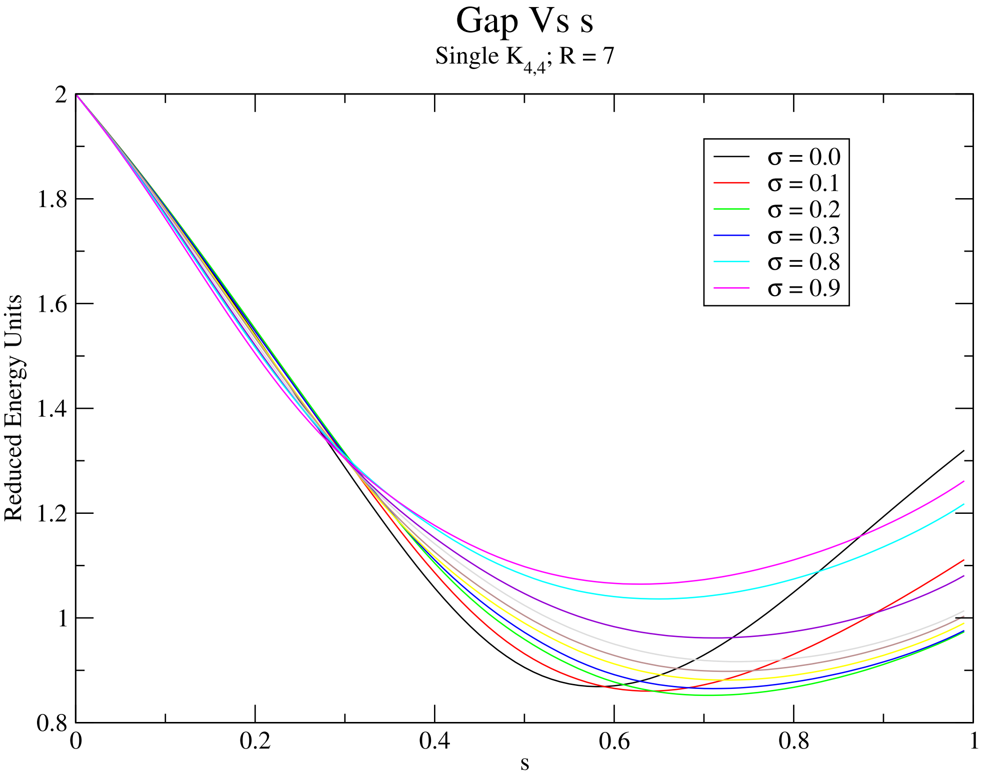

Figure 4.19: The gap between the ground state and first excited state through out the evolution path for various values of σ. H_0 is Pauli spin-x matrix expanded to 8 spins. H_p is a randomly generate Hamiltonian according to the $K_{4,4}$ topology with resolution 7. The time-step $\Delta s = 0.01$. A linear evolution trajectory was used, $(1-s)H_0 + sH_p$. Each The computational intensity of this simulation restricted the number of Hamiltonians averaged over to 3, each with 10,000 erroneous instances.

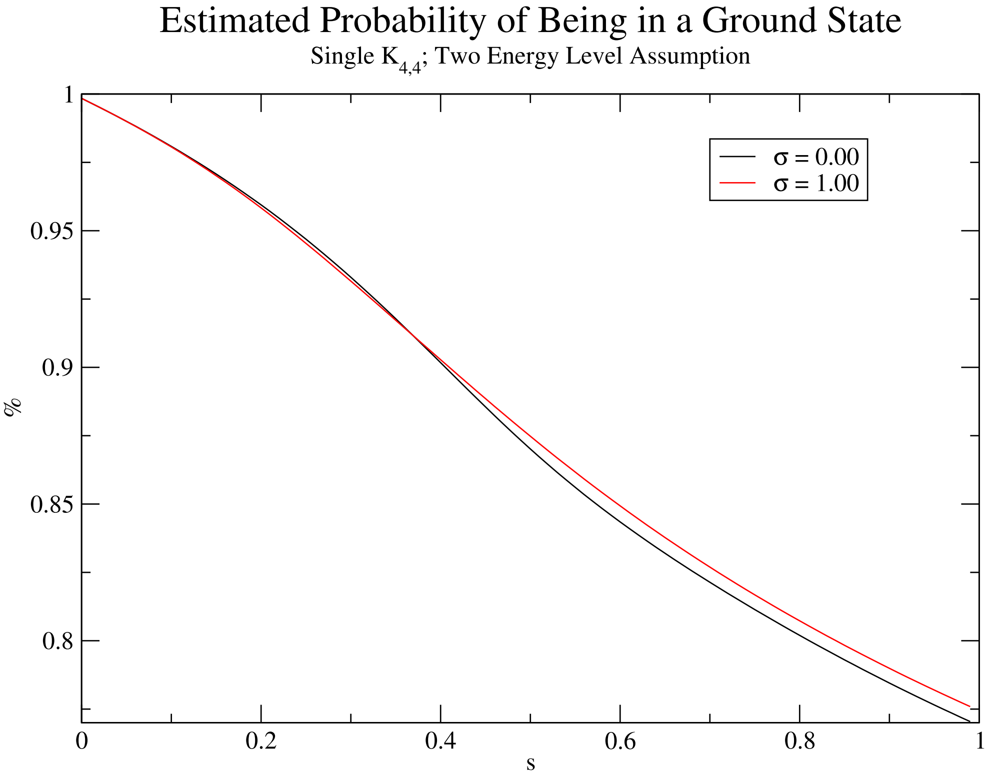

Figure 4.20: Estimated probability of being in the ground state versus s using a two level approximation obtained from reference [56]. The probability of being in the ground state is estimate via $1 - \int\limits_0^1 \frac{\exp\left(\frac{-cs}{g^2}\right)}{cg^2} ds$. Due to the reduced units c must be estimated. It is set such that the probability of being in the ground state at $s = 0$ is 1. The integral is estimated using $\Delta s = 0.01$. The effect of errors is to make the evolution path have a more shallow minimum (see Figure 4.19), the result is that there is less of a dip in the probability curve seen in this figure. This may be miss leading as it is very unlikely the correct Hamiltonian was programmed into the computer at $\sigma = 1$, but the effect is negligible on the computers ability to obtain the systems ground state, as expected.

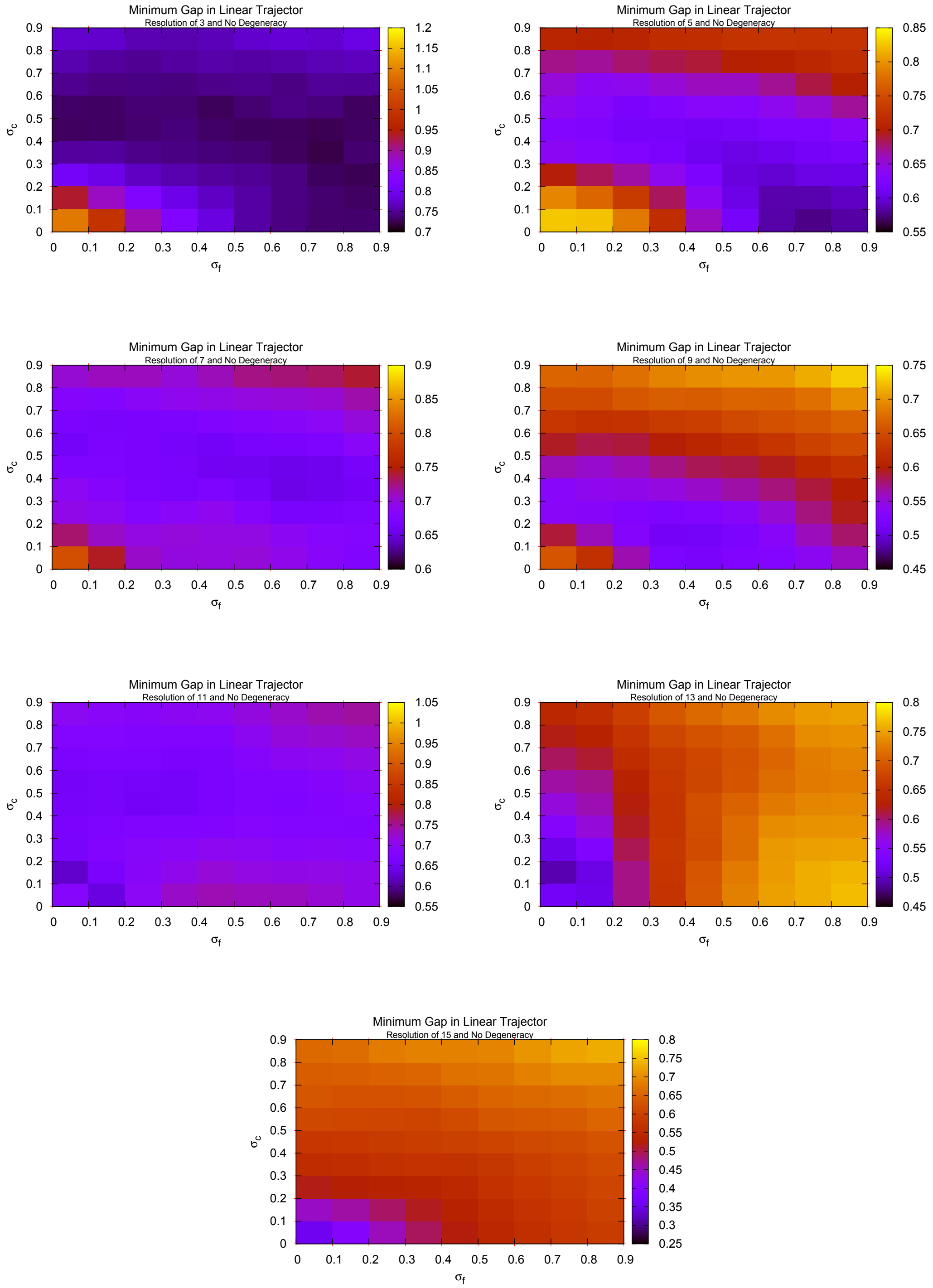

Figure 4.21: Minimum Gap versus σ for various resolutions. The errors are now independent of one another as in Equation 4.39. The results shown in this figure are obtain by averaging 10 Hamiltonians with 1,000 erroneous instances each. The minimum is calculated by going through the evolution and generating curves similar to those seen in Figure 4.19. The minimum is then found of each of these curves and averaged.

Chapter 5

Conclusion

We have studied how errors in setting the field and coupling terms can effect the success of adiabatic quantum computing. Equations 4.21 and 4.36 give an upper bound on the amount of error aloud in the system before the resilience drops off from 100% for a single $K_{4,4}$, where the errors in the field and coupling terms are of equal magnitude. Increasing the size of the network to $4K_{4,4}$'s shows a much more rapid decline in resilience. Unfortunately the statistics gathered for the larger system were not sufficient to extract a reliable scaling relation; though, one would expect the same form as Equations 4.21 and 4.36 with different coefficients. We put forth Equation 4.24 and 4.38 to estimate the maximum amount of error a system of arbitrary size and resolution can have; though, more study is required to determine the validity of these equations. They are consistent with results in this paper. Attempts to fit the curves for independent errors resulted in large errors that prohibited a relation from being extracted.

To connect the values of σ to experiment details of Vesuvius become important. Vesuvius has a resolution of 15 ($\Delta J = \Delta b = \frac{1}{7}$). A field or coupling term of 1 corresponds to 33.8 GHz; therefore, according to Equation 4.33 errors up to $0.5746 GHz$ can be tolerated. The spacing $\Delta J = \Delta b \approx 4.8286 GHz$.

When considering how these errors effect the evolution process it is evident that for errors small enough to retain a high resilience, there is minimum change in the evolution path and essentially no change in the computer's probability of obtaining the ground state of the programmed Hamiltonian; though, more a detailed simulation may be required to accurately measure the effects on the evolution.

Future studies could use more sophisticated techniques to find the ground state of larger networks. A problem with doing a study on large systems would be that these techniques would have-to be probabilistic so the ground state would have to be easily confirmed.

Appendices

Appendix A

Degeneracy

Until this point we restricted our attention to Hamiltonians that were non-degenerate. We will now lift this restriction and consider general Hamiltonians. Our previous definition of resilience is no longer sufficient, nor our definitions of cross over and end gap. We therefore, for simplicity, will look solely at resilience and provide two new definitions of resilience:

Hard Resilience is defined as the probability that all the original ground states will remain the lowest lying eigenstates of the new Hamiltonian.

Soft Resilience is defined as the probability that the new ground state will be a ground state of the original Hamiltonian.

Hard Resilience is of importance when one it is required to obtain all the ground states of the Hamiltonian while *Soft Resilience* is appropriate when only a single ground state is required.

A.1 Hard Resilience

Figure A.1 shows how hard resilience changes versus σ with increasing resolution for a doubly degenerate Hamiltonian. The drop in resilience is much faster than that of a non-degenerate Hamiltonian. This is expected as the condition is much more stringent. In Figure A.2 the degeneracy is varied for a Hamiltonian of resolution 7; as expected, as degeneracy increases hard resilience drops as the condition becomes more difficult to full-fill.

When the field and coupling errors are independent of one another the picture is much the same as it was in the non-degenerate case as shown in Figures A.3 and A.4.

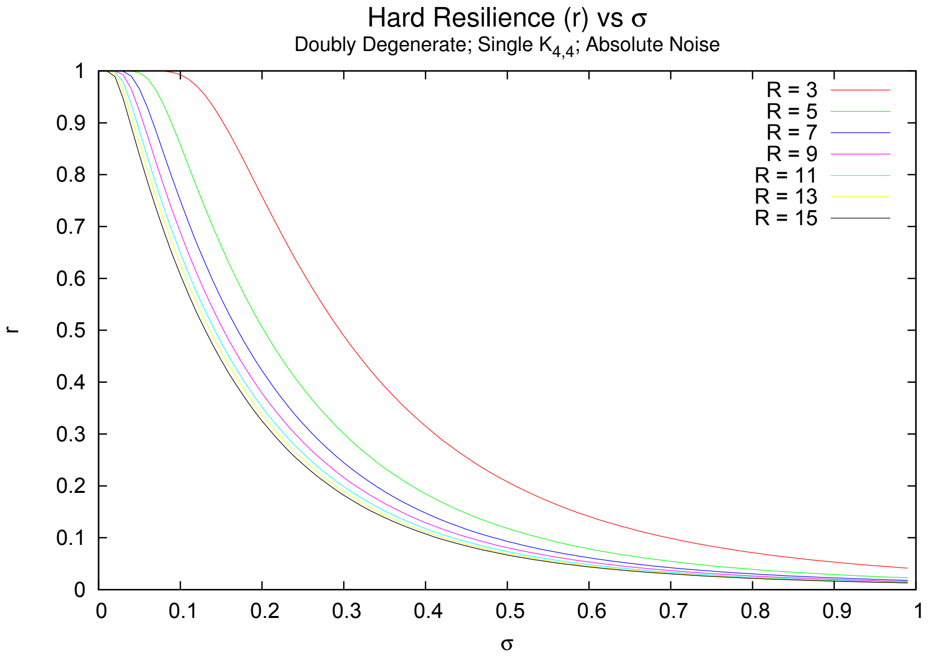

Figure A.1: Hard Resilience versus σ for various resolutions for a doubly degenerate Hamiltonian. Compared to resilience non-degenerate Hamiltonians hard resilience drops off significantly faster.

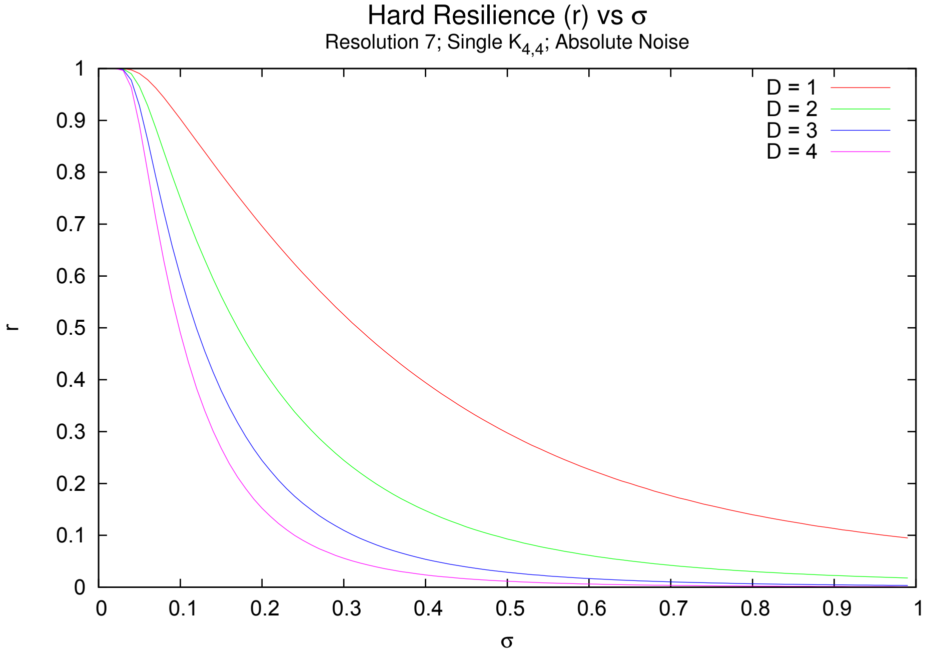

Figure A.2: Hard Resilience versus σ for various degeneracies of a Hamiltonian with resolution 7.

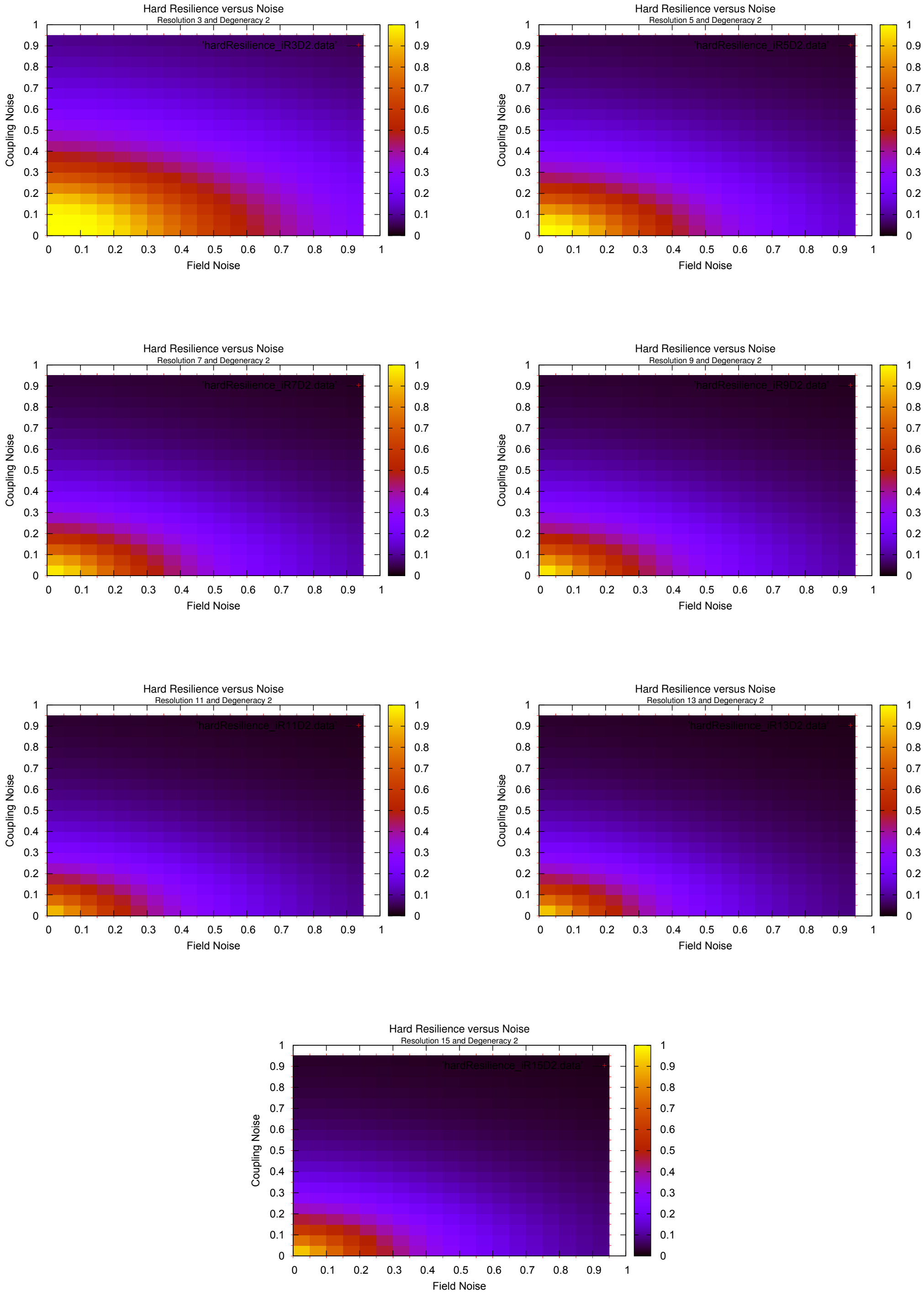

Figure A.3: Hard Resilience versus independent σ for various resolutions.

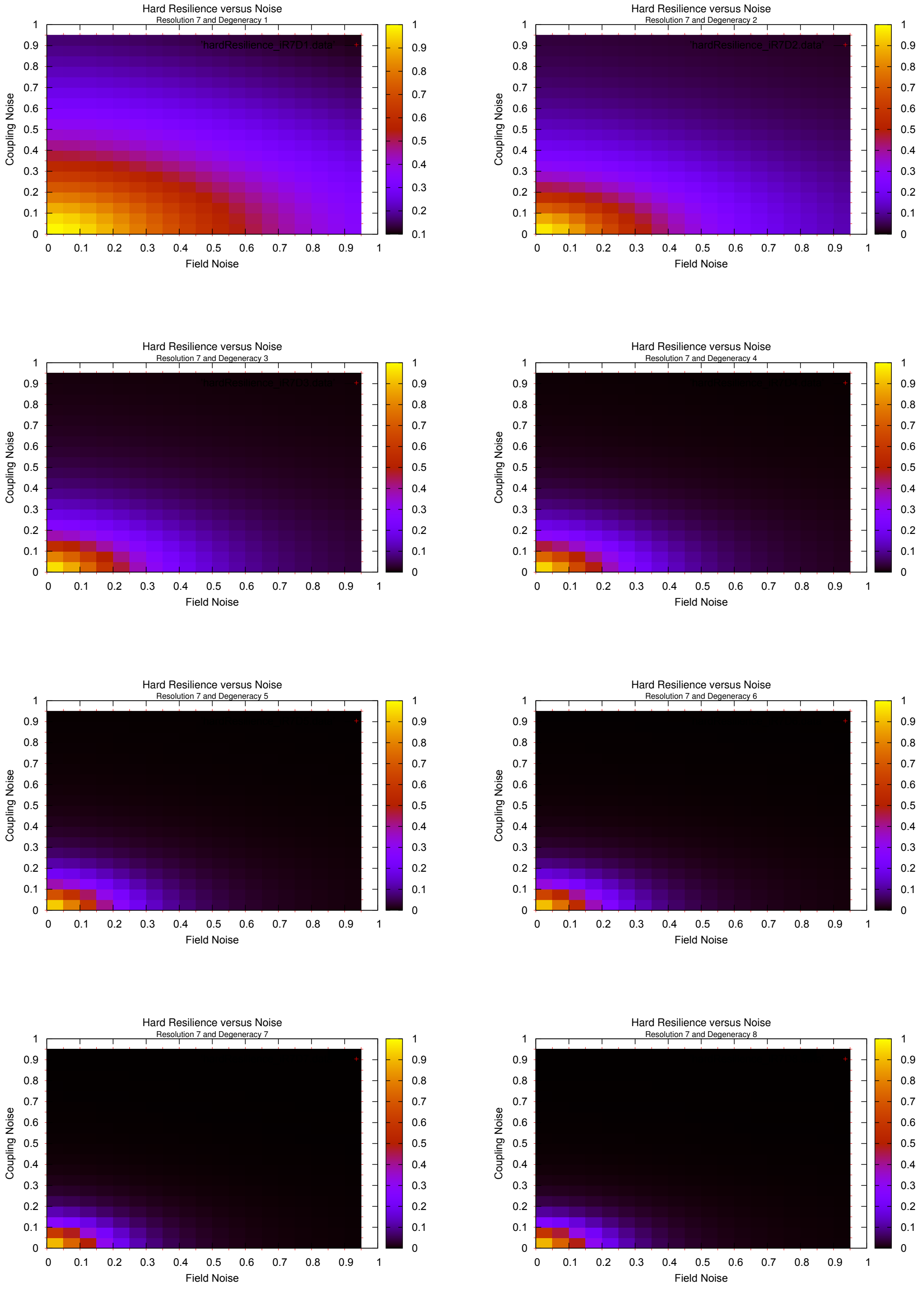

Figure A.4: Hard Resilience versus independent σ for various degeneracies.

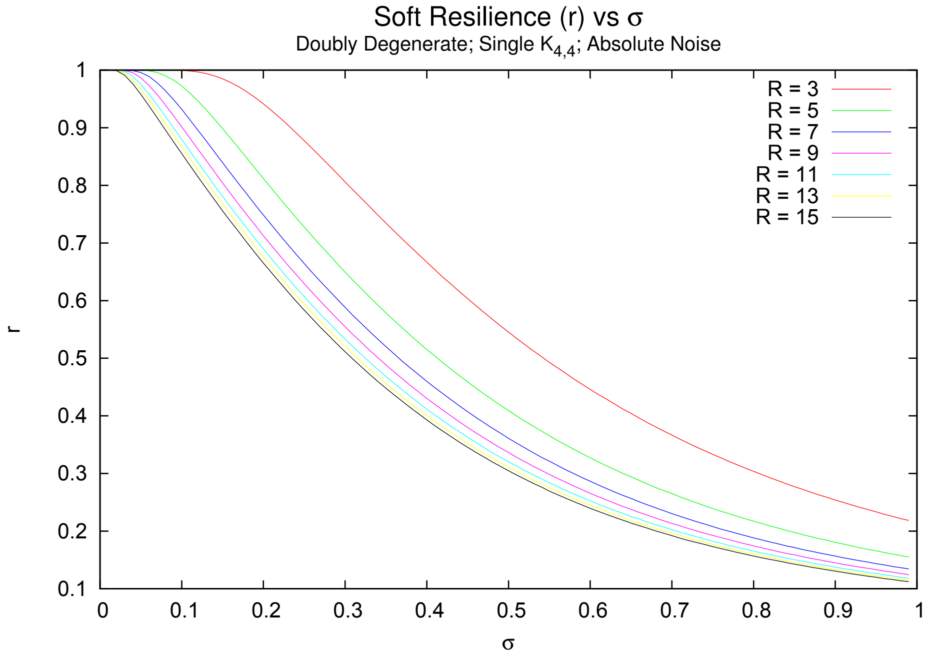

Figure A.5: Soft Resilience versus σ for various resolutions for a doubly degenerate Hamiltonian. Compared to resilience non-degenerate Hamiltonians hard resilience drops off significantly slower.

A.2 Soft Resilience

Figure A.5 shows how soft resilience changes versus σ with increasing resolution for a doubly degenerate Hamiltonian. The drop in resilience is much slower than that of a non-degenerate Hamiltonian. This is expected as the condition is easier to full-fill. In Figure A.6 the degeneracy is varied between curves; as expected, as degeneracy increases soft resilience increases as the condition becomes easier.

When the field and coupling errors are independent of one another the picture is much the same as it was in the non-degenerate case as shown in Figures A.7 and A.8.

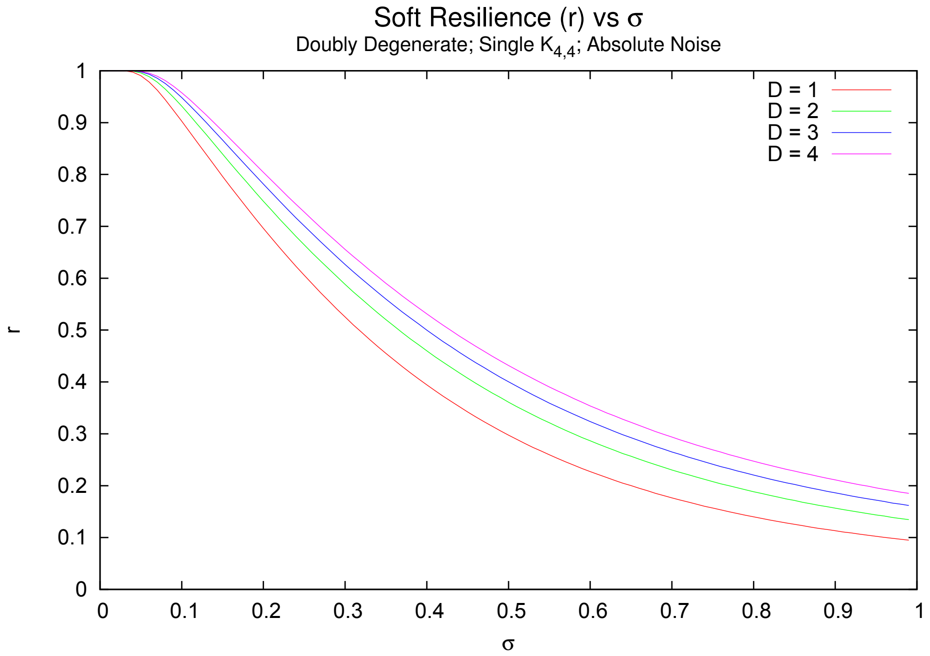

Figure A.6: Soft Resilience versus σ for various degeneracies.

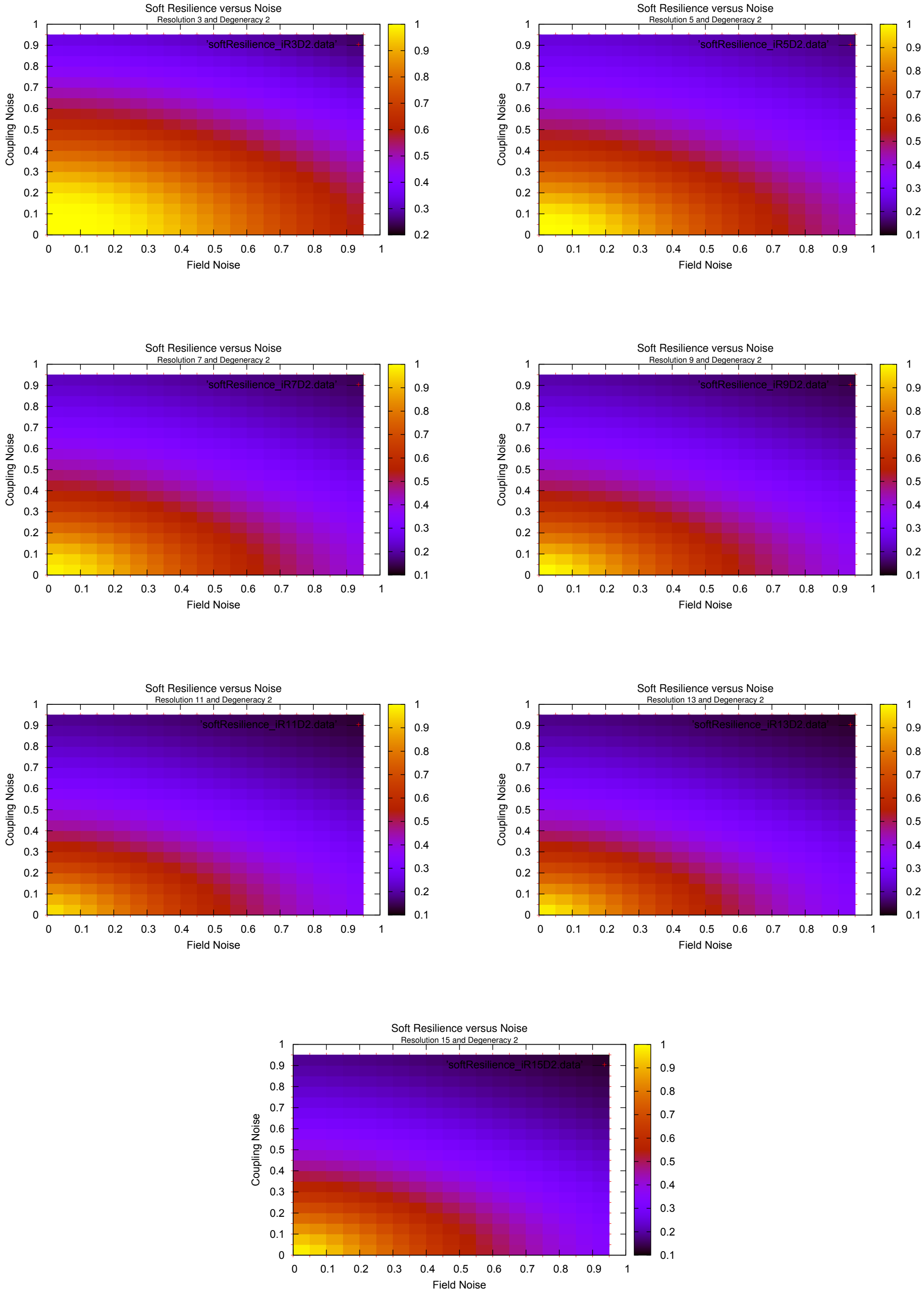

Figure A.7: Soft Resilience versus independent σ for various resolutions.

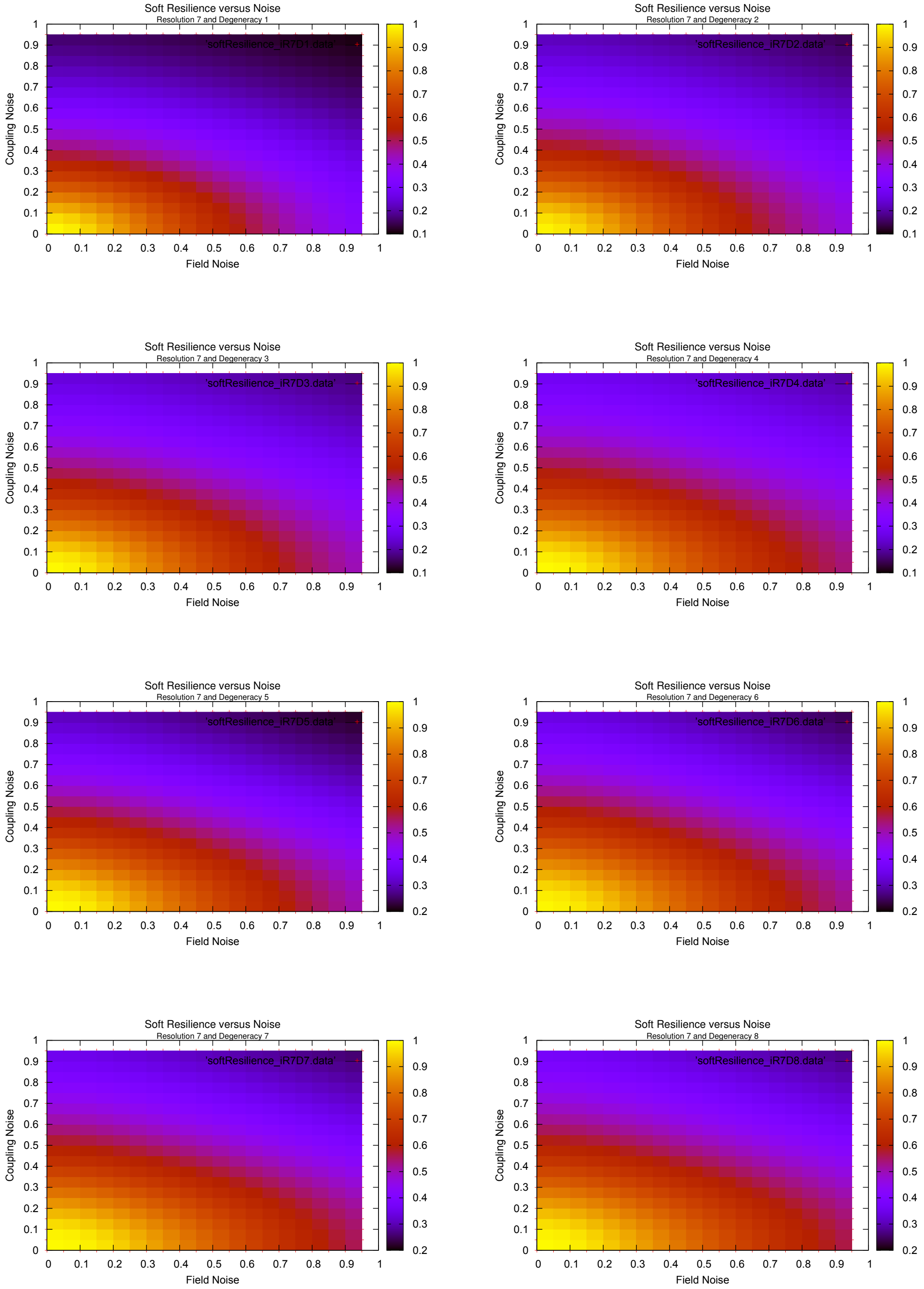

Figure A.8: Soft Resilience versus independent σ for various degeneracies.

Appendix B

Hamiltonian and State Space

In this section we look at how Hamiltonians with the same ground state are related to one another. To do this a metric must be defined to assess their similarity. There are many ways to define such a metric but we will define it

$$d = |H_a - H_b| \tag{B.1}$$

where H_a and H_b are in the form of Equation 4.13, and

$$|H_a - H_b| = \sum_i |b_i^a - b_i^b| + \sum_{ij} |J_{ij}^a - J_{ij}^b| \tag{B.2}$$

For an N-spin system, there are 2^N possible states. We will label each state by G_i where i ranges from $[0, 2^N)$. Each state, G_i, is the ground state of a set of Hamiltonians, denoted by $\{G_i\}$. The graphs here are done by first generating a large number of Hamiltonians (between 100,000 and 10,000,000) and classifying them by their ground state, making the sets $\{G_i\}$. Then for each $\{G_i\}$ draw an even larger number of randomly chosen pairs (between 1,000,000 and 100,000,000) and measure their distance, d. A histogram of the distances is then made for all $\{G_i\}$.

We see that within a given resolution each ground state has an identical distribution, see Figure B.1; furthermore, the distributions across multiple resolutions has the same relative shape, see Figure B.2. The shape of the curves in Figures B.1 and B.2 indicate that when shifting the Hamiltonian by a small amount it is unlikely to maintain the same ground state. When considering Hamiltonians of differing resolutions the structure of the curve becomes more difficult, but the general shape remains the same, as seen in Figure B.3. In Figure B.3 there are 5 distinct overlapping Gaussians, each of which likely correspond to one of the resolutions available between the resolutions 7 and 5 ($R \epsilon \{7, 9, 11, 13, 15\}$).

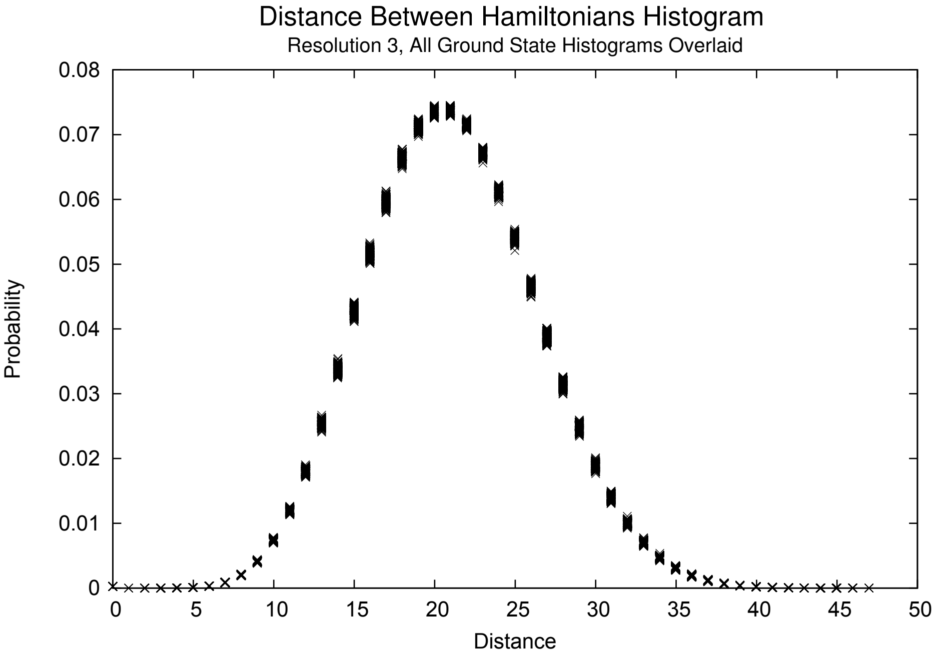

Figure B.1: A histogram of the distance between Hamiltonians in set $\{G_i\}$ for all i. 100,000 Hamiltonians comprise the set of $\{G_i\}$s. Each set was sampled from 1,000,000 times, making the total number of distances sampled to build this graph 256,000,000.

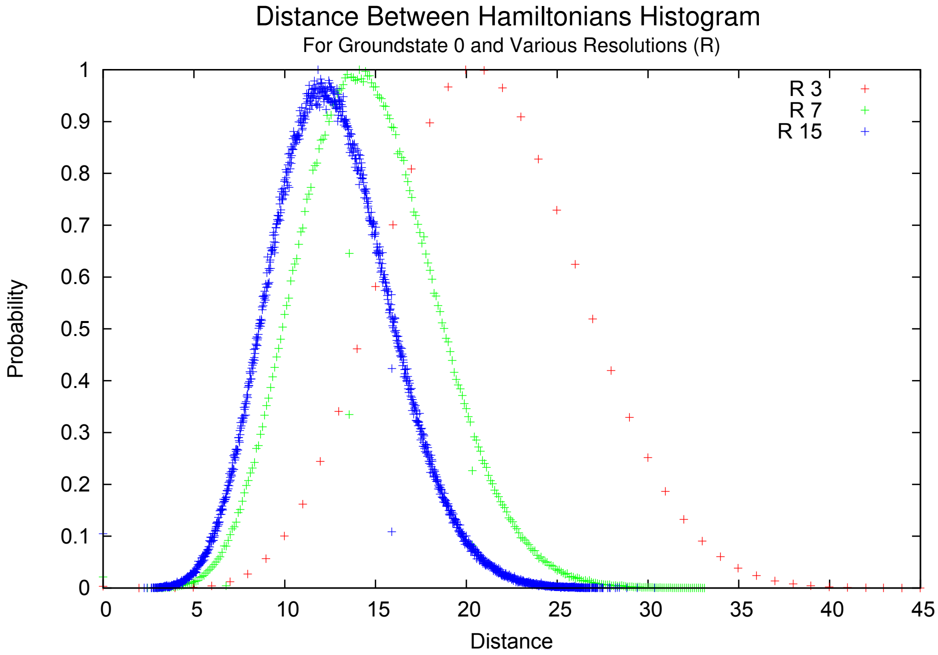

Figure B.2: A histogram of the distance between Hamiltonians in set $\{G_i\}$ for all $i = 0$ for various resolutions. The points inside the skewed Gaussian are a relic of the Monte Carlo simulation used to create the Gaussian not using a large enough dataset to properly sample the entire set. The histograms have been normalized to have the same amplitude. For each resolution 100,000 Hamiltonians were developed, and 1,000,000 distance measurements from each set G_0.

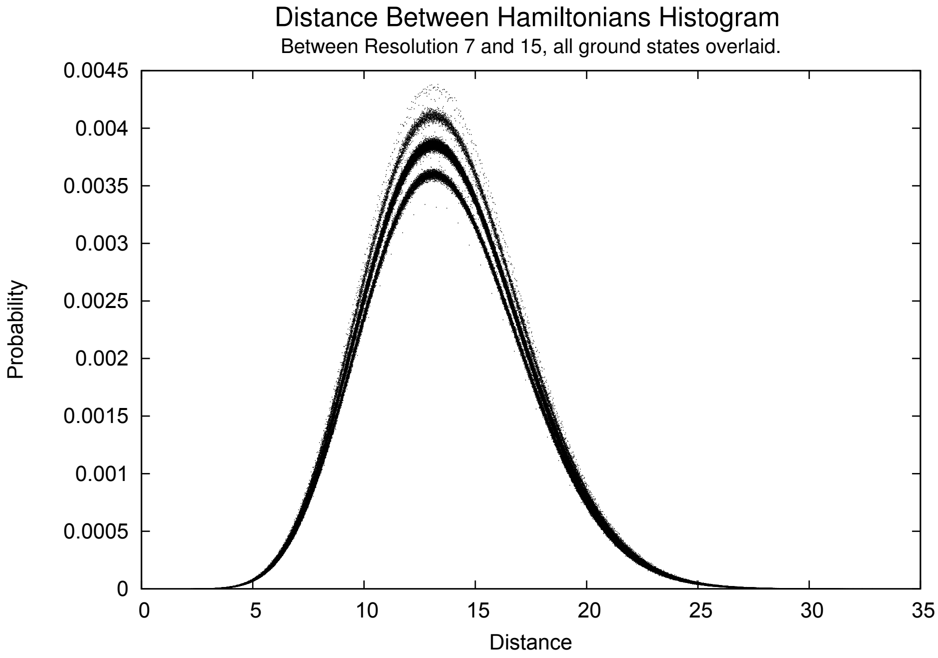

Figure B.3: A histogram of the distance between Hamiltonians with the same ground state where the Hamiltonians have different resolutions. The resolutions used in this calculation are 7 and 15. There appears to be multiple Gaussians overlaid on one another. This is not an effect of overlaying all the groundstates as individual groundstate histograms also appear to have multiple Gaussians. Interestingly there are 5 distinct Gaussians overlaid on top of one another. This is the same number of available resolutions within this range: 7, 9, 11, 13, 15. It seems reasonable to draw the conclusion that there is one Gaussian per resolution available. To build this 10,000,000 Hamiltonians were developed to build the set of $\{G_i\}$s. Then 100,000,000 distances measurements made and a histogram made of the results.

Appendix C

Programs

The algorithms described in Sections C.1-C.3 are all different engines for essentially the same program. Here the general program is laid out. Note that the eigenstates function is unique to each implementation.

```
Data: Number of Hamiltonians (nHam), Number of Runs per Hamiltonian
      (nRuns), Resolution, Degeneracy, Sigma
Result: Resilience, Crossover, Gap
for $h$ in nHam do
    Fields $\leftarrow$ genFields(Resolution);
    Spectrum $\leftarrow$ eigenstates(Fields);
    for $r$ in nRuns do
        ErroneousFields $\leftarrow$ addErrors(Fields, Sigma);
        ErroneousSpectrum $\leftarrow$ eigenstates(ErroneousFields);
        Resilience += isResilient(Spectrum, ErroneousSpectrum);
        CrossOver += whichEigenstate(Spectrum.groundState,
        ErroneousSpectrum);
        Gap += gap(Spectrum, ErroneousSpectrum);
    end
end
Resilience /= nRuns * nHam ;
CrossOver /= nRuns * nHam ;
Gap /= nRuns * nHam ;
return Resilience, CrossOver, Gap;
```

Algorithm 1: General Program Layout for Sections C.1-C.3.

Algorithm 1 was originally written in Python; however, when the time came for massive data collection on a single $K_{4,4}$ Python's speed limitations became evident. The program was then ported to Java where a speed up of more than a factor of 600 was observed. After the port to Java the runs times of the program was still too high for efficient data collection so the program was ported to C++ where a factor of 2 speed increase was observed. Later parallel implementations observed even more speed up.

The genFields(Resolution) function is given by

```
Data: Resolution
Result: Set of Field and Coupling terms
for $b$ in NumberOfSpins do
    Fields.add(randomField(Resolution));
end
for $j$ in NumberOfCouplings do
    Fields.add(randomField(Resolution));
end
return Fields;
```

Algorithm 2: genFields: Generate set of random field and coupling terms

Where randomField(Resolution) is

```
Data: Resolution
Result: Random number from set of allowable numbers, see Section 2.1
split = (Integer) Resolution / 2;
N = Random() % Resolution ;
N -= split;
N /= (float)split;
return N;
```

Algorithm 3: randomField: Generate a random field or coupling term

The addErrors function is simply

```
Data: Fields, Sigma
Result: Add errors according to a Gaussian distribution to field and coupling
        terms
for f in Fields do
    if Counter greater than NumberOfSpins then
        ErroneousFields.add(f + randomGaus(Sigma.J));
        Continue;
    end
    ErroneousFields.add(f + randomGaus(Sigma.b));
    Counter++;
end
return Fields;
```

Algorithm 4: addErrors: Add errors to the field and coupling terms according to a Gaussian distribution

The resilience is calculated by adding 0 and 1's dependant on whether the ground state of the erroneous Hamiltonian is the same as the intended.

```
Data: Spectrum, ErroneousSpectrum
Result: 0 or 1 dependent on equivalence of groundstates.
if Spectrum.groundstate ≡ ErroneousSpectrum.groundstate then
    return 1;
end
return 0;
```

Algorithm 5: isResilient: return 0 or 1 dependent on whether the ground states are the same.

The cross over:

```
Data: Spectrum, ErroneousSpectrum
Result: Position of the intended groundstate in the erroneous Hamiltonian's
        spectrum.
n = 0;
for $S$ in ErroneousSpectrum ascending do
    if $S \equiv \textit{Spectrum.groundstate}$ then
        return $n$;
    end
    n++;
end
```

Algorithm 6: whichEigenstate: return the position of the groundstate of the intended Hamiltonian in the erroneous Hamiltonian's spectrum.

The gap function:

```
Data: Spectrum, ErroneousSpectrum
Result: Gap between the intended groundstate in the erroneous Hamiltonian's
        spectrum first state other than the intended groundstate.
if Spectrum.groundstate $\neq$ ErroneousSpectrum.groundstate then
    for $S$ in ErroneousSpectrum ascending do
        if $S \equiv$ Spectrum.groundstate then
            State1 = S;
            break;
        end
    end
    State2 = ErroneousSpectrum.groundstate;
end
else
    State1 = ErroneousSpectrum.groundstate;
    State2 = ErroneousSpectrum.firstExcitedState;
end
return State1.energy - State2.energy;
```

Algorithm 7: gap: return the gap between the intended groundstate in the erroneous Hamiltonian's spectrum first state other than the intended groundstate.

Below is the energy calculation function for a single $K_{4,4}$. Only slight modifications are required for multiple $K_{4,4}$'s:

```
Data: Fields, Spins
Result: Eigenvalue
H = 0;
for s in Spins do
    H -= s * Fields.nextField();
end
for s1 in Spins[0:4) do
    for s2 in Spins[4:8) do
        H -= s1 * s2 * Fields.nextField();
    end
end
return H
```

Algorithm 8: eigenvalue: Returns the eigenvalue of a given state subject to given fields. Assumes $K_{4,4}$ structure. Spin values are ±1.

C.1 Sequential Exhaustive Search

This algorithm works by enumerating all the possible configurations of the spins and iterating through them to create a full eigenvalue, eigenvector pair list.

```
Data: Fields
Result: Eigenstates, Eigenvalues
# N is the number of spins;
for i in 0..($2^N$-1) do
    state ← bitVector(i);
    states.add(eigenvalue(Fields, state),i);
end
sort(states);
return states;
```

Algorithm 9: eigenstates: uses sequential exhaustive search on CPU. Scales as $O(2^N)$.

Where the sort algorithm in C++ was a modified Tim sort to take on key value pairs. The other languages had built in functions.

C.2 Parallel Exhaustive Search

At this point the details of programming GPUs becomes central to a detailed understanding of the algorithms. The interested reader is referred to [13]. The algorithms will be presented in pseudo-code that is as readable as possible with minimal discussion on why they are crafted the manner they are. Details of the algorithms that are not central to the understanding of the general function are omitted.

```
Data: Fields
Result: Eigenstates, Eigenvalues
# N is the number of spins;
# P is the number of Processes (1 per GPU);
# pId is process id which is unique and in the range $[0, P-1]$;
pStart = $2^N$ / P * pId;
states ← cudaEigenstates(pStart, $2^N$/P, Fields);
states ← cudaSortStates(states);
Gather(states);
sort(states);
return states;
```

Algorithm 10: eigenstates: uses MPI to divide up the task to P processors and each processors uses a GPU to calculate the eigenstates and eigenvalues of its branch. The results are then gathered back and sorted using a modified version of Tim sort.

```
Data: start, length, Fields
Result: Eigenstates, Eigenvalues
# gId is threads global id on the GPU.;
blockLoadToShared(Fields);
state ← bitVector(gId + start);
eigenval[gId] = eigenvalue(state);
eigenvec[gId] = gId + start;
```

Algorithm 11: cudaEigenstates: cuda kernel for determining the eigenstates and eigenvalues.

Speed up compared the sequentially algorithm on C++ was approximately 1000x.

C.2.1 Sorting With Cuda

Due to the large number of memory accesses required to sort a list, efficient sorting is difficult on a GPU. Because we are only concerned with the lowest lying energy eigenstates efficient sorting is possible through reduction. The general idea is to take a list of length N and reduce it to a short list, say of length 16 that is sorted. One can guarantee to get the bottom n states globally by finding the bottom n on each group of states. To do the sorting of the smaller list a modified insertion sort algorithm is used.

```
Data: States
Result: bottom16States
bottom16 ← states[0:15];
insertionSort(bottom16);
for s in States[16:N-1] do
    if s.energy greater than  bottom16[15].energy then
        bottom16[15] = s;
        insertionSort(bottom16);
    end
end
return bottom16
```

Algorithm 12: cudaSortStates

Speed up versus the same algorithm sequentially executed in C++ was around 20x.

C.3 Parallel Branch and Bound

Branch and bound is faster way to find the lowest lying energy eigenstate, or states if degenerate. This is useful for calculations that are only concerned with resilience. Branch and bound algorithms work by progressively traversing through the possible configurations in an order and checking to see if it is possible to reach the desired state from this point, if not the algorithm prunes that tree and moves over to another branch. Branch and bound is a simplification of the A* search algorithm used ubiquitously in artificial intelligence.

There are two types of branch and bound algorithms classified by their search technique: breadth first and depth first. Breadth first expands all nodes at a given depth simultaneously and prunes all branches possible before expanding. This is a good approach when one is concerned with all possible answers; however, it comes with a high memory cost of storing all the states at each level which typically grow exponentially. Depth first search traverses completely down one branch before moving onto the next (given that the branch isn't pruned). This is a good approach when one only requires one state or if there are memory limitations. It also has the benefit of updating its own heuristics every time it reaches the end of a branch.

Depth first branch and bound is not well suited for the GPU due to its recursive style and limitations on programming on GPUs memory access wise. The breadth first algorithm naturally works on a GPU; however, one looses the benefits of memory flexibility and self-updating heuristics of the depth first algorithm.

A solution to this is to bundle groups of branches together. Each bundle is then considered an independent problem that is subject to a breadth first search on the GPU. The group of bundles is then another network that is searched depth first (by nature of finishing one bundle without interference from the others) and communicates heuristic updates.

This algorithm as well as the parallel exhaustive search were used to gain statistics on $4K_{4,4}$'s. This algorithm was roughly 30 times faster than the parallel exhaustive search but could only be used to gain insight into resilience. For simplicity in Algorithm 13 it is assumed that the number of bundles equals the number of processors and GPUs. This is rarely the case and a cuing system must be implemented.

The bound function can be implemented in many ways, including through the use of the cudaSort algorithm; however, it will suffice to say here that it simply iterates through all the states pruning any states that cannot reach the ground state energy estimate. The remaining possible energy in the network is estimated using a heuristic that always overestimates, this way no branches are prematurely pruned. For this application the remaining energy is estimated as the negative sum of the magnitudes of the remaining coupling and field terms to be assigned.

```
Data: Fields
Result: Ground state(s)
# N is the number of spins;
# P is the number of Processes (1 per GPU);
# pId is process id which is unique and in the range $[0, P - 1]$;
initialDepth = P-1;
pStart = $2^N$ / P * pId;
states ← expand12Levels(pStart, Fields);
currentDepth = initialDepth + 12;
for currentDepth less than  N do
    states ← cudaBranch(states, Fields);
    states ← bound(states, minEnergyGuess);
    if is empty  states then
        break;
    end
    N++;
end
Gather(states);
sort(states);
return states;
```

Algorithm 13: eigenstates: Return the ground state(s) of the given Hamiltonian.

cudaBranch works by expanding all the nodes in the set of states. Each state then has its energy updated by calculating the field and coupling terms introduced by the newly assigned spin.

C.4 Adiabatic Evolution

The program to gain insight into the evolution spectrum was only written in C++ with MPI parallelism. It used the Eigen 3.0 C++ framework to do the required linear algebra. The algorithm works by discretizing s in

$$H(s) = (1 - s)H_0 + sH_p \tag{C.1}$$

into 100 time steps. The Hamiltonian at that time step is calculated and its eigenstates found. The bottom N eigenstates of interest recorded before moving onto the next time step. A 3 time-step correlation function is used to track the eigenstates evolution trajectory around crossings.

```
Data: Fields, Sigma
Result: A set of histories of the eigenstates of the instantaneous Hamiltonians.
$H_0 \leftarrow$ SpinXHamiltonian();
$H_P \leftarrow$ K44Hamiltonian(Fields);
s = 0;
for $s$ less than 1 do
    H = (1 - s)$H_0$ + s $H_P$;
    eigenstates $\leftarrow$ eigenSolve(H);
    insertionSort(eigenstates);
    s += 0.1;
    history.add(eigenstates);
end
nextHistory();
for $i$ less than N do
    $H_P \leftarrow$ K44Hamiltonian(addErrors(Fields));
    s = 0;
    for $s$ less than 1 do
        H = (1 - s)$H_0$ + s $H_P$;
        eigenstates $\leftarrow$ eigenSolve(H);
        insertionSort(eigenstates);
        s += 0.1;
        history.add(eigenstates);
    end
    nextHistory();
end
```

Algorithm 14: Program generates a set of histories of the eigenstates of the instantaneous Hamiltonians through the evolution. The set of histories can be used to calculate quantities such as the average gap at each value of s and the minimum gap.

C.5 Hamiltonian Distance

This program was designed to get data for Section C.5. The program first generates a large number of Hamiltonians and classifies them according to their ground state. After a sufficient number of Hamiltonians have been classified the algorithm then goes through each set of Hamiltonians (classified by their ground state) and draws n random pairs of Hamiltonians and measures their distance. After n pairs have been measured a histogram is made of the results.

```
Data: Resolution
Result: Set of histrograms of the distance between Hamiltonians with the same
        ground state.
for $i$ less than N do
    Fields ← genFields(Resolution);
    states ← eigenstates(Fields);
    hamiltonians[states.groundstate()].add(Fields)];
    i++;
end
for $g$ less than $2^8$ do
    for $p$ less than n do
        distances.add(distance(hamiltonians[g].getRandom(),
        hamiltonians[g].getRandom());
        p++;
    end
    g++;
    histograms.add(histrogram(distances);
end
```

Algorithm 15: Algorithm generates a set of histograms of the distance between Hamiltonians with the same ground state.

Bibliography

[1] Dorit Aharonov, Wim Van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM review*, 50(4):755–787, 2008.

[2] MHS Amin, Dmitri V Averin, and James A Nesteroff. Decoherence in adiabatic quantum computation. *Physical Review A*, 79(2):022107, 2009.

[3] MHS Amin, Peter J Love, and CJS Truncik. Thermally assisted adiabatic quantum computation. *Physical review letters*, 100(6):060503, 2008.

[4] MHS Amin, CJS Truncik, and DV Averin. Role of single-qubit decoherence time in adiabatic quantum computation. *Physical Review A*, 80(2):022303, 2009.

[5] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach.* Cambridge University Press, 2009.

[6] Rainer Blatt and David Wineland. Entangled states of trapped atomic ions. *Nature*, 453(7198):1008–1015, 2008.

[7] Mark F Bocko, Andrea M Herr, and Marc J Feldman. Prospects for quantum coherent computation using superconducting electronics. *IEEE Transactions on Applied Superconductivity*, 7(2):3638–3641, 1997.

[8] Sergio Boixo, Troels F Rønnow, Sergei V Isakov, Zhihui Wang, David Wecker, Daniel A Lidar, John M Martinis, and Matthias Troyer. Evidence for quantum annealing with more than one hundred qubits. *Nature Physics*, 10(3):218–224, 2014.

[9] Michael Boylan-Kolchin. Cosmology: A virtual universe. *Nature*, 509(7499):170–171, 2014.

[10] Andrew M Childs, Edward Farhi, and John Preskill. Robustness of adiabatic quantum computation. *arXiv preprint quant-ph/0108048*, 2001.

[11] I Chiorescu, Y Nakamura, CJP Ma Harmans, and JE Mooij. Coherent quantum dynamics of a superconducting flux qubit. *Science*, 299(5614):1869–1871, 2003.

[12] Neil G Dickson and MHS Amin. Does adiabatic quantum optimization fail for np-complete problems? *Physical Review Letters*, 106(5):050502, 2011.

[13] Rob Farber. *CUDA application design and development.* Elsevier, 2011.

[14] Stefan Fölsch, Jesús Martínez-Blanco, Jianshu Yang, Kiyoshi Kanisawa, and Steven C Erwin. Quantum dots with single-atom precision. *Nature nanotechnology*, 9(7):505–508, 2014.

[15] S Manjula Gandhi, J Devishree, and S Sathish Mohan. A new reversible smg gate and its application for designing twos complement adder/subtractor with overflow detection logic for quantum computer-based systems. In *Computational Intelligence, Cyber Security and Computational Models*, pages 259–266. Springer, 2014.

[16] David J. Griffiths. *Introduction to Quantum Mechanics*, chapter 10. Pearson, 2 edition, 2005.

[17] R Harris, J Johansson, AJ Berkley, MW Johnson, T Lanting, Siyuan Han, P Bunyk, E Ladizinsky, T Oh, I Perminov, et al. Experimental demonstration of a robust and scalable flux qubit. *Physical Review B*, 81(13):134510, 2010.

[18] Richard Harris, MW Johnson, T Lanting, AJ Berkley, J Johansson, P Bunyk, E Tolkacheva, E Ladizinsky, N Ladizinsky, T Oh, et al. Experimental investigation of an eight-qubit unit cell in a superconducting optimization processor. *Physical Review B*, 82(2):024511, 2010.

[19] Jonathan P Home, David Hanneke, John D Jost, Jason M Amini, Dietrich Leibfried, and David J Wineland. Complete methods set for scalable ion trap quantum information processing. *Science*, 325(5945):1227–1230, 2009.

[20] A Izmalkov, M Grajcar, E Ilichev, Th Wagner, H-G Meyer, A Yu Smirnov, MHS Amin, Alec Maassen van den Brink, and AM Zagoskin. Evidence for entangled states of two coupled flux qubits. *Physical review letters*, 93(3):037003, 2004.

[21] Laszlo B Kish. End of moore's law: thermal (noise) death of integration in micro and nano electronics. *Physics Letters A*, 305(3):144–149, 2002.

[22] Thaddeus D Ladd, Fedor Jelezko, Raymond Laflamme, Yasunobu Nakamura, Christopher Monroe, and Jeremy L OBrien. Quantum computers. *Nature*, 464(7285):45–53, 2010.

[23] Nathan K Langford, Sven Ramelow, Robert Prevedel, William J Munro, Gerard J Milburn, and Anton Zeilinger. Efficient quantum computing using coherent photon conversion. *Nature*, 478(7369):360–363, 2011.

[24] T Lanting, AJ Przybysz, A Yu Smirnov, FM Spedalieri, MH Amin, AJ Berkley, R Harris, F Altomare, S Boixo, P Bunyk, et al. Entanglement in a quantum annealing processor. *arXiv preprint arXiv:1401.3500*, 2014.

[25] Daniel A Lidar. Towards fault tolerant adiabatic quantum computation. *arXiv preprint arXiv:0707.0021*, 2007.

[26] Yu-xi Liu, LF Wei, JS Tsai, and Franco Nori. Controllable coupling between flux qubits. *Physical review letters*, 96(6):067003, 2006.

[27] JB Majer, FG Paauw, ACJ Ter Haar, CJPM Harmans, and JE Mooij. Spectroscopy on two coupled superconducting flux qubits. *Physical review letters*, 94(9):090501, 2005.

[28] John M Martinis, S Nam, J Aumentado, and C Urbina. Rabi oscillations in a large josephson-junction qubit. *Physical Review Letters*, 89(11):117901, 2002.

[29] Karl-Peter Marzlin and Barry C Sanders. Inconsistency in the application of the adiabatic theorem. *Physical review letters*, 93(16):160408, 2004.

[30] Micah Blake McCurdy, Jeffrey Egger, and Jordan Kyriakidis. Decomposition and gluing for adiabatic quantum optimization. *arXiv preprint arXiv:1312.5169*, 2013.

[31] Ari Mizel. Fault-tolerant, universal adiabatic quantum computation. *arXiv preprint arXiv:1403.7694*, 2014.

[32] Ari Mizel, Daniel A Lidar, and Morgan Mitchell. Simple proof of equivalence between adiabatic quantum computation and the circuit model. *Physical review letters*, 99(7):070502, 2007.

[33] C Monroe, R Raussendorf, A Ruthven, KR Brown, P Maunz, L-M Duan, and J Kim. Large-scale modular quantum-computer architecture with atomic memory and photonic interconnects. *Physical Review A*, 89(2):022317, 2014.

[34] JE Mooij, TP Orlando, L Levitov, Lin Tian, Caspar H Van der Wal, and Seth Lloyd. Josephson persistent-current qubit. *Science*, 285(5430):1036–1039, 1999.

[35] Oliver Morsch and Markus Oberthaler. Dynamics of bose-einstein condensates in optical lattices. *Reviews of modern physics*, 78(1):179, 2006.

[36] C Negrevergne, TS Mahesh, CA Ryan, M Ditty, F Cyr-Racine, W Power, N Boulant, T Havel, DG Cory, and R Laflamme. Benchmarking quantum control methods on a 12-qubit system. *Physical Review Letters*, 96(17):170501, 2006.

[37] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

[38] S Olmschenk, DN Matsukevich, P Maunz, D Hayes, L-M Duan, and C Monroe. Quantum teleportation between distant matter qubits. *Science*, 323(5913):486–489, 2009.

[39] TP Orlando, S Lloyd, LS Levitov, KK Berggren, MJ Feldman, MF Bocko, JE Mooij, CJP Harmans, and CH Van der Wal. Flux-based superconducting qubits for quantum computation. *Physica C: Superconductivity*, 372:194–200, 2002.

[40] TP Orlando, JE Mooij, Lin Tian, Caspar H van der Wal, LS Levitov, Seth Lloyd, and JJ Mazo. Superconducting persistent-current qubit. *Physical Review B*, 60(22):15398, 1999.

[41] Christian Ospelkaus, Christopher E Langer, Jason M Amini, Kenton R Brown, Dietrich Leibfried, and David J Wineland. Trapped-ion quantum logic gates based on oscillating magnetic fields. *Physical review letters*, 101(9):090502, 2008.

[42] Kristen L Pudenz, Tameem Albash, and Daniel A Lidar. Error-corrected quantum annealing with hundreds of qubits. *Nature communications*, 5, 2014.

[43] Gustavo Rigolin and Gerardo Ortiz. Adiabatic theorem for quantum systems with spectral degeneracy. *Physical Review A*, 85(6):062111, 2012.

[44] Jérémie Roland and Nicolas J Cerf. Quantum search by local adiabatic evolution. *Physical Review A*, 65(4):042308, 2002.

[45] Jérémie Roland and Nicolas J Cerf. Noise resistance of adiabatic quantum computation using random matrix theory. *Physical Review A*, 71(3):032330, 2005.

[46] Giuseppe E Santoro, Roman Martoňák, Erio Tosatti, and Roberto Car. Theory of quantum annealing of an ising spin glass. *Science*, 295(5564):2427–2430, 2002.

[47] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM journal on computing*, 26(5):1484–1509, 1997.

[48] Wim Van Dam, Michele Mosca, and Umesh Vazirani. How powerful is adiabatic quantum computation? In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 279–287. IEEE, 2001.

[49] Lieven MK Vandersypen, Matthias Steffen, Gregory Breyta, Costantino S Yannoni, Mark H Sherwood, and Isaac L Chuang. Experimental realization of shor's quantum factoring algorithm using nuclear magnetic resonance. *Nature*, 414(6866):883–887, 2001.

[50] Hanno H Weitering. Quantum dots: One atom at a time. *Nature nanotechnology*, 2014.

[51] Zhaoyan Wu and Hui Yang. Validity of the quantum adiabatic theorem. *Physical Review A*, 72(1):012114, 2005.

[52] Xiaodong Xu, Wang Yao, Bo Sun, Duncan G Steel, Allan S Bracker, Daniel Gammon, and LJ Sham. Optically controlled locking of the nuclear field via coherent dark-state spectroscopy. *Nature*, 459(7250):1105–1109, 2009.

[53] F Yoshihara, K Harrabi, AO Niskanen, Y Nakamura, and JS Tsai. Decoherence of flux qubits due to 1/f flux noise. *Physical review letters*, 97(16):167001, 2006.

[54] AP Young, S Knysh, and VN Smelyanskiy. First-order phase transition in the quantum adiabatic algorithm. *Physical review letters*, 104(2):020502, 2010.

[55] Navid Yousefabadi. Optimal annealing paths for adiabatic quantum computation. 2011.

[56] Man-Hong Yung. Thermal noise on adiabatic quantum computation. *arXiv preprint arXiv:0807.4819*, 2008.

[57] Alexandre M Zagoskin, Evgeni Ilichev, Miroslav Grajcar, Joseph J Betouras, and Franco Nori. How to test the" quantumness" of a quantum computer? *arXiv preprint arXiv:1401.2870*, 2014.